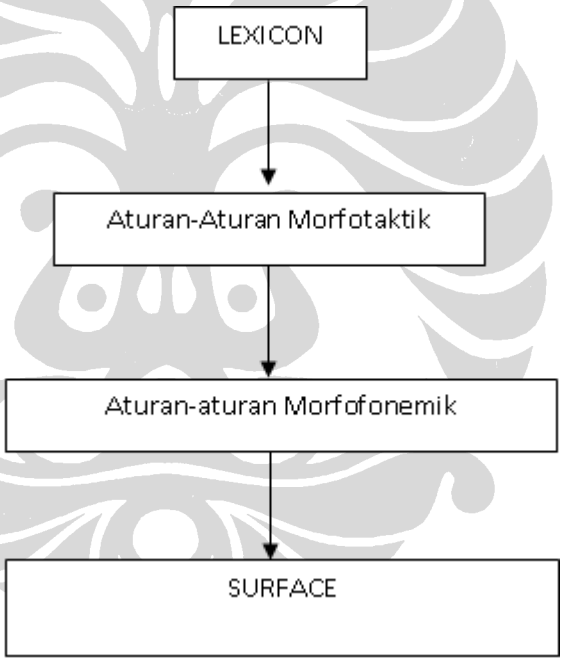
BAB 3 RANCANGAN PENGURAI MORFOLOGI BAHASA INDONESIA

Dalam bab ini akan dijelaskan semua rancangan yang dikerjakan selama penelitian ini. Rancangan-rancangan tersebut mencakup rancangan *lexicon*, *tags*, aturan-aturan morfotaktik, hingga aturan-aturan morfofonemik.

3.1 Rancangan *Two-Level Morphology*

Seperti yang sudah dijelaskan sebelumnya, pengurai morfologi ini akan memanfaatkan prinsip *two-level morphology* yang sudah dibahas pada BAB 2. Untuk rancangannya, kira-kira bisa digambarkan seperti pada gambar 3-1.

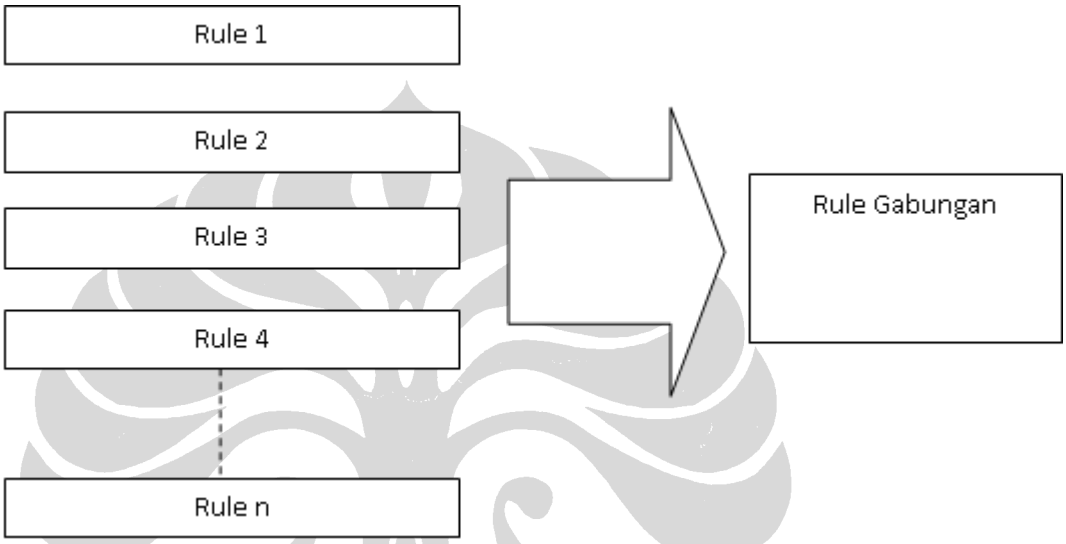


Gambar 3-1. Rancangan Pengurai Morfologi

Pada level paling atas ada *lexicon* yang berisi daftar semua kata dasar. Kemudian dari *lexicon* akan masuk ke dalam aturan morfotaktik, yang mengatur penggabungan morfem yang sah. Hasil olahan dari aturan morfotaktik akan masuk ke dalam aturan morfofonemik, yang menangani variasi ejaan, kemudian baru akan ditampilkan ke dalam *surface*. *Surface* akan menampilkan kata beserta

tags yang dirancang pada pengurai morfologi ini. Adapun penjelasan mengenai *tags* akan diberikan pada subbab 3.3.

Di dalam aturan-aturan morfotaktik dan morfofonemik itu sendiri terdapat banyak sekali *rule*, di mana *rules* tersebut akan bergabung dengan beberapa variasi penggabungan yang nantinya akan dijelaskan pada BAB 4. Untuk saat ini bisa digambarkan *rules* tersebut akan bergabung seperti pada gambar 3-2.



Gambar 3-2. Skema Penggabungan *Rules*

Setealh dilakukan penggabungan aturan-aturan tersebut, didapat satu kesatuan aturan morfofonemik maupun satu kesatuan aturan morfotaktik yang akan digunakan dalam pengurai morfologi ini. Aturan-aturan morfotaktik akan digabungkan menjadi satu aturan gabungan morfotaktik, sedangkan aturan-aturan morfofonemik akan digabungkan menjadi satu aturan morfofonemik.

3.2 Rancangan *Lexicon*

Lexicon merupakan daftar semua morfem yang ada baik kata dasar maupun imbuhan, namun pada penelitian ini *lexicon* hanya berisi kata dasar, sebab untuk masing-masing imbuhan akan dikerjakan di bagian morfofonemik dan morfotaktik. Daftar kata dasar diperoleh dari KBBI versi elektronis yang dikembangkan di LAB IR Fasilkom UI sekitar tahun 1990-an.

Lexicon ini sendiri nantinya akan berdiri pada tingkatan di atas aturan morfofonemik dan morfotaktik yang akan berfungsi untuk memeriksa apakah suatu kata dasar itu sah atau tidak dengan melihat ke dalam isi *lexicon* (ini berhubungan dengan fungsi pengurai morfologi ini sebagai *morphological analyzer*). Sedangkan untuk fungsi pengurai morfologi sebagai *morphological generator*, *lexicon* ini akan berfungsi untuk melihat kata dasar yang akan dibangun.

Untuk keperluan pengurai morfologi ini, kami perlu menggolongkan *lexicon* kata dasar dibagi menjadi 4 kelas kata, di antaranya:

3.2.1 Kelas Kata Verba

Verba atau sering juga disebut kata kerja adalah kata yang menggambarkan proses perbuatan atau keadaan **[KBBD08]**. Kelas kata verba ini nantinya akan dibagi menjadi subkelas kata verba transitif dan subkelas kata verba intransitif. Subkelas kata verba transitif berisi kata-kata kerja yang memiliki objek, sedangkan subkelas kata verba intransitif berisi kata kerja yang tidak menggunakan objek.

Adapun contoh dari kelas kata verba ini antara lain “minum”, “main”, “baca”, “beri”, “lari”, “pergi”, “jalan”, dan masih banyak contoh kata kerja lainnya.

3.2.2 Kelas Kata Nomina

Nomina atau kata benda adalah kata di bahasa Indonesia yang ditandai oleh tidak dapatnya bergabung dengan kata “tidak” **[KBBD08]**.

Adapun contoh dari kelas kata nomina ini antara lain “batu”, “bingkai”, “buku”, “sekolah”, “tanda”, “pagar”, dan masih banyak contoh kata benda lainnya.

3.2.3 Kelas Kata Adjektiva

Adjektiva atau kata sifat adalah kata yang menerangkan nomina (kata benda) dan secara umum dapat bergabung dengan kata “lebih” dan “sangat” **[KBBD08]**.

Adapun contoh dari kelas kata adjektiva ini antara lain “hitam”, “dingin”, “keras”, “sepi”, “biru”, dan masih banyak contoh kata sifat lainnya.

3.2.4 Kelas Kata Dll

Kelas kata dll ini dibuat untuk kata-kata yang tidak termasuk tiga kelas kata di atas, namun tetap perlu didaftarkan dan digunakan dalam proses perancangan pengurai morfologi ini. Biasanya kata-kata tersebut dijabarkan lebih rinci sebagai pronomina, adverbia, numeralia, dan partikel, akan tetapi pada penelitian ini disederhanakan pemodelannya.

Adapun contoh dari kelas kata dll ini antara lain “satu”, “dua”, “atau”, “hendak”, dan masih banyak contoh kata dll lainnya.

Alasan penggolongan kelas kata tersebut didasarkan oleh kebutuhannya di tingkatan aturan morfofonemik dan morfotaktik yang akan dilihat setelah ini.

3.3. Rancangan *Tags*

Dalam membangun suatu pengurai morfologi, rancangan suatu *tag* bisa dikatakan memiliki peranan yang sangat penting. Dari perancangan *tag* inilah informasi tentang morfologi kata tersebut didapatkan. Selain itu apa yang ditampilkan pada akhirnya merupakan kata dasar dan *tags* tersebut sehingga dari *tags* ini terlihat hasil pembangunan pengurai morfologi ini.

Dalam merancang *tags* pada pengurai morfologi perlu diperhatikan seberapa jauh yang diinginkan untuk ditampilkan sebagai hasil analisa morfologi tersebut. Karena makin dalam analisis morfologi yang ingin didapatkan, makin banyak *tags* yang harus dirancang. *Tags* yang dirancang ini sekaligus sebagai salah satu batasan yang ditentukan untuk pembangunan pengurai morfologi tersebut.

Dalam pengurai morfologi ini, ada beberapa *tags* yang dirancang. Adapun *tags* tersebut bisa dilihat pada tabel 3-1.

Tabel 3-1. Rancangan *Tags*

Nama Tag	Arti Penggunaan Tag	Contoh
+Verb	Menunjukkan bahwa kata tersebut merupakan kelas kata Verba	menulis→ tulis+Verb+AV
+Noun	Menunjukkan bahwa kata tersebut merupakan kelas kata Nomina	makanan→ makan+Noun
+Adjective	Menunjukkan bahwa kata tersebut merupakan kelas kata Adjektiva	terpandai→ pandai+Adjective
+BareVerb	Menunjukkan bahwa kata tersebut merupakan kata dasar dari kelas kata Verba	tulis→ tulis+BareVerb+UV
+BareNoun	Menunjukkan bahwa kata tersebut merupakan kata dasar dari kelas kata Nomina	buku→ buku+BareNoun
+BareAdjective	Menunjukkan bahwa kata tersebut merupakan kata dasar dari kelas kata Adjektiva	dingin→ dingin+BareAdjective
+BareEtc	Menunjukkan bahwa kata tersebut merupakan kata dasar dari kelas kata Dll	dua→ dua+BareEtc
+AV	AV merupakan singkatan dari Active Voice, menunjukkan bahwa kata tersebut merupakan bentuk kata aktif	memakan→ makan+Verb+AV
+PASS	PASS merupakan singkatan dari Passive Voice, menunjukkan bahwa kata tersebut merupakan bentuk kata pasif	dimakan→ makan+Verb+PASS
+UV	UV merupakan singkatan dari Undegoer Voice, merupakan suatu verba transitif yang bisa bergabung dengan proklitik	makan→ makan+BareVerb+UV
+Redup	Menunjukkan bahwa kata tersebut merupakan kata ulang	makan-makan→ makan+BareVerb+Redup
+Caus_kan	Menunjukkan bahwa kata tersebut berimbuhan -kan dan merupakan bentuk *Causative* dimana memerlukan pelaku berupa seseorang yang menyebabkan terjadinya kata tersebut	jatuhkan→ jatuh+Verb+Caus_kan
+Appl_kan	Menunjukkan bahwa kata tersebut berimbuhan -kan dan merupakan bentuk *Applicative* dimana tidak memerlukan pelaku berupa seseorang yang menyebabkan terjadinya kata tersebut. Pada kasus ini justru lokasi, instrumen, atau hal-hal lain yang terlibat didalamnya	belikan→ beli+Verb+Appl_kan

+Caus_i	Menunjukkan bahwa kata tersebut berimbuhan -i dan merupakan bentuk *Causative* dimana memerlukan pelaku berupa seseorang yang menyebabkan terjadinya kata tersebut	panasi→ panas+Verb+Caus_i
+Appl_i	Menunjukkan bahwa kata tersebut berimbuhan -i dan merupakan bentuk *Applicative* dimana tidak memerlukan pelaku berupa seseorang yang menyebabkan terjadinya kata tersebut. Pada kasus ini justru lokasi, instrumen, atau hal-hal lain yang terlibat didalamnya	duduki→ duduk+Verb+Appl_i
+Actor	Menunjukkan kata tersebut seseorang yang melakukan suatu perilaku atau pekerjaan dari kata dasarnya	penari→ tari+Noun+Actor
+Instrument	Menunjukkan kata tersebut merupakan sesuatu alat atau instrumen yang digunakan untuk melakukan kata dasarnya	penggaris→ garis+Noun+Instrument

Enam *tags* terakhir merupakan *optional tags* atau *tags* khusus di mana *tag* tersebut akan muncul apabila ada tanda khusus yang terdapat di dalam *lexicon*. Hal ini dikarenakan memerlukan lebih dari *lexicon* biasa untuk memunculkan *tag* tersebut. Untuk pembahasan selanjutnya enam *tags* ini akan disebut sebagai *tags* khusus.

3.4. Aturan-aturan Morfotaktik

Proses morfotaktik adalah suatu proses mempelajari dan memodelkan susunan kata yang sah **[BEES03]**. Pada rancangan morfologi untuk pengurai morfologi ini, perlu dipelajari terlebih dahulu bagaimana perilaku-perilaku morfotaktik pada suatu kata, yaitu bagaimana suatu kata dasar tersebut bergabung dengan imbuhan apa saja. Pada subbab ini akan diberikan aturan-aturan proses morfotaktik yang mengacu pada tata bahasa baku bahasa Indonesia **[ALWI03]**.

3.4.1 Morfotaktik Imbuhan meN-

Tabel yang berisi aturan-aturan morfotaktik untuk imbuhan meN- dapat dilihat pada tabel 3-2.

Tabel 3-2. Aturan Morfotaktik Imbuhan meN-

Variasi Imbuhan	Proses Pembentukan	Kelas Kata Stem	Kelas Kata Berimbuhan	Contoh	*Counter Example*
meN + stem	meN + (stem)	Verba transitif	Verba	menulis memutar	
		Nomina	Verba	membatu membingkai	meN + tanah meN + pigura
		Adjektiva	Verba	mengeras membiru	meN + rajin meN + kotor
		DLL	Verba	menyatu mendua	
meN + stem + kan	meN + (stem) + kan atau meN + ((stem) + kan)	Verba	Verba	melarikan membacakan	meN + pergi + kan meN + simpan + kan
		Nomina	Verba	membukukan menyekolahkan	meN + kertas + kan meN + gedung + kan
		Adjektiva	Verba	membersihkan menyuburkan	
		DLL	Verba	menyatukan menduakan	
meN + stem + i	meN + (stem) + i	Verba yang fonem terakhir selain /i/	Verba	menyirami	meN + dengar + i

	atau meN + ((stem) + i)			menduduki	meN + bicara + i
		Nomina yang fonem terakhir selain /i/	Verba	menandai mengabari	meN + bola + i meN + berita + i
		Adjektiva yang fonem terakhir selain /i/	Verba	menerangi mengotori	meN + cantik + i meN + rendah + i
meN + per + stem	meN + (per + (stem))	• (per + stem) merupakan Verba. • (per + stem) yang bisa digabung dengan imbuhan meng- adalah per- yang berubah morfem menjadi per- • Kelas kata stem dibahas pada tabel imbuhan per-	Verba	memperalat mempererat memperburuk	
meN + per + stem + kan	meN + (per + (stem) + kan)	• (per + stem + kan) merupakan Verba. • Kelas kata stem dibahas pada tabel imbuhan per-	Verba	mempersatukan mempermasalahkan mempercakapkan	
meN + per + stem + i	meN + (per + (stem) + i)	• (per + stem + i) merupakan Verba. • Kelas kata stem dibahas pada tabel imbuhan per-	Verba	memperdayai mempersenjatai	
meN + ke + stem + kan	meN + (ke + (stem) + kan)	• (ke + stem + kan) merupakan Verba. • Kelas kata stem dibahas pada	Verba	mengedepankan	

		tabel imbuhan ke-			
meN + ber + stem + kan	meN + (ber + (stem)) + kan atau meN + (ber + (stem) + kan)	• (ber + stem) merupakan Verba. • Kelas kata stem dibahas pada tabel imbuhan ber-	Verba	member lakukan member hentikan	
meN + ter + stem + kan	meN + (ter + (stem)) + kan atau meN + (ter + (stem) + kan)	• (ter + stem) merupakan Verba. • Kelas kata stem dibahas pada tabel imbuhan ter-	Verba	menerta wakan	

Seperti kita lihat di atas, afiks meN- dan meN-kan bisa bergabung dengan kata dasar dari kelas kata verba, nomina, adjektiva, maupun dll, sedangkan afiks meN-i hanya dapat bergabung dengan kelas kata verba, nomina, dan adjektiva, dengan syarat tidak boleh dengan kata dasar yang berakhiran fonem –i.

Selain tiga macam afiks tersebut, dalam kategori imbuhan meng- ini juga terdapat gabungan imbuhan (konfiks) yaitu gabungan meN- dengan per-, per-kan, per-i, ke-kan, ber-kan, dan ter-kan. Penggabungan tersebut bisa dilihat cukup jelas pada tabel di atas.

Hal yang menarik dari tabel di atas bisa dilihat bahwa kata dasar dari kelas kata apapun jika digabungkan dengan imbuhan meN- akan menghasilkan kata dengan kelas kata verba. Bisa disebut juga meN- merupakan imbuhan pembuat kata verba.

3.4.2 Morfotaktik Imbuhan di-

Tabel yang berisi aturan-aturan morfotaktik untuk imbuhan di- dapat dilihat pada tabel 3-3.

Tabel 3-3. Aturan Morfotaktik Imbuhan di-

Variasi Imbuhan	Proses Pembentukan	Kelas Kata Stem	Kelas Kata Berimbuhan	Contoh	*Counter Example*
di + stem	meng + (stem)	Verba transitif	Verba	dibuka dilipat ditukar	
di + stem + kan	di + (stem) + kan atau di + ((stem) + kan)	Verba	Verba	dimainkan diberikan	di + pergi + kan di + simpan + kan
		Nomina	Verba	dikalengkan diceritakan	di + pintu + kan di + tangan + kan
		Adjektiva	Verba	dimiringkan disalahkan	
		DLL	Verba	disatukan diduakan	
di + stem + i	di + (stem) + i atau di + ((stem) + i)	Verba yang fonem terakhir selain /i/	Verba	disirami diduduki	di + dengar + i di + bicara + i
		Nomina yang fonem terakhir selain /i/	Verba	dibuahi diselimuti dipagari	di + rumah + i di + kertas + i
		Adjektiva yang fonem terakhir selain /i/	Verba	diterangi dikotori	di + bersih + i di + tajam + i

di + per + stem	di + (per + (stem))	• (per + stem) merupakan Verba. • (per + stem) yang bisa digabung dengan imbuhan meng- adalah per- yang berubah morfem menjadi per- • Kelas kata stem dibahas pada tabel imbuhan per-	Verba	diperberat dipermanis dipercepat	
di + per + stem + kan	di + (per + (stem) + kan)	• (per + stem + kan) merupakan Verba. • Kelas kata stem dibahas pada tabel imbuhan per-	Verba	dipertemukan dipertontonkan diperjuangkan	
di + per + stem + i	di + (per + (stem) + i)	• (per + stem + i) merupakan Verba. • Kelas kata stem dibahas pada tabel imbuhan per-	Verba	diperdayai dipersenjatai	
di + ke + stem + kan	di + (ke + (stem) + kan)	• (ke + stem + kan) merupakan Verba. • Kelas kata stem dibahas pada tabel imbuhan ke-	Verba	dikedepankan	
di + ber + stem + kan	di + (ber + (stem)) + kan atau di + (ber + (stem) + kan)	• (ber + stem) merupakan Verba. • Kelas kata stem dibahas pada tabel imbuhan ber-	Verba	diberlakukan diberhentikan	
di + ter + stem + kan	di + (ter + (stem)) + kan atau di + (ter + (stem) + kan)	• (ter + stem) merupakan Verba. • Kelas kata stem dibahas pada tabel imbuhan ter-	Verba	ditertawakan	

Bisa dilihat pada tabel di atas imbuhan di- memiliki banyak kemiripan dengan imbuhan meN-. Keduanya sama-sama menghasilkan kelas kata verba sebagai hasilnya tidak peduli apapun kelas kata dasar yang digabungkan. Perbedaan imbuhan meN- dengan di- terletak pada sifat kata hasil imbuhannya. Imbuhan meN- akan menghasilkan verba aktif, sedangkan imbuhan di- akan menghasilkan verba pasif. Hal ini akan dibahas lebih lanjut nantinya di BAB 4.

Afiks di-kan dan di-i mirip dengan afiks meN-kan dan meN-i pada kategori imbuhan meN-, namun berbeda dengan kategori imbuhan meN-, pada kategori imbuhan di- tepatnya pada afiks di- hanya dapat bergabung dengan kelas kata verba atau lebih rincinya dari subkelas kata verba transitif.

Untuk gabungan imbuhan (konfiks) kurang lebih mirip dengan imbuhan meN-.

3.4.3 Morfotaktik Imbuhan peN-

Tabel yang berisi aturan-aturan morfotaktik untuk imbuhan peN- dapat dilihat pada tabel 3-4.

Tabel 3-4. Aturan Morfotaktik Imbuhan peN-

Variasi Imbuhan	Proses Pembentukan	Kelas Kata Stem	Kelas Kata Berimbuhan	Contoh	*Counter Example*
peN + stem	peN + (stem)	Verba	Nomina	pengaduk pembungkus penulis	peN + mandi peN + lihat
		Nomina	Nomina	penyapu pengecat	peN + botol peN + sabun
		Adjektiva	Nomina	pembersih	peN + adil peN +

				pengrajin	malas
peN + stem + an	peN + (stem) + an	Verba	Nomina	penulisan pencatatan	
		Nomina	Nomina	pengalengan pengasapan	peN + lemari + an peN + kabut + an
		Adjektiva	Nomina	penghematan pemecahan	peN + malas + an peN + cantik + an
peN + ber + stem + an	peng + (ber + (stem)) + an	• (ber + stem) merupakan Verba. • Kelas kata stem dibahas pada tabel imbuhan ber-	Nomina	pemberlakuan pembelajaran	peN + bernyanyi + an peN + berlibur + an

Dari tabel di atas bisa dilihat bahwa afiks peN- dapat bergabung dengan kata dasar dari kelas kata verba, nomina, dan adjektiva. Afiks peN-an dapat bergabung dengan kata dasar dari kelas kata verba, nomina, adjektiva, dll.

Selain dua afiks tersebut, di kategori imbuhan peN- juga terdapat gabungan antara afiks peng-an dengan ber- yang menghasilkan konfiks peN-ber-an.

Konstruksi penggabungan kata dengan imbuhan peN- akan selalu menghasilkan suatu kata dari kelas kata nomina.

3.4.4 Morfotaktik Imbuhan per-

Tabel yang berisi aturan-aturan morfotaktik untuk imbuhan per- dapat dilihat pada tabel 3-5.

Tabel 3-5. Aturan Morfotaktik Imbuhan per-

Variasi Imbuhan	Proses Pembentukan	Kelas Kata Stem	Kelas Kata Berimbuhan	Contoh	*Counter Example*
per + stem	per + (stem)	Nomina	Verba	peralat	per + botol per + sabun
		Adjektiva	Verba	pertinggi perhalus	per + adil per + rajin
per + stem + an	per + (stem) + an	Verba	Nomina	perputaran permintaan	
		Nomina	Nomina	perindustrian perkebunan	
		DLL	Nomina	persatuan	
per + stem + kan	per+ (stem) + kan atau per + ((stem) + kan)	Verba	Verba	pertontonkan perdengarkan	per + baca + an per + raba + kan
		Nomina	Verba	permasalahkan	peng + solusi + kan
		DLL	Verba	persatukan	
per + stem + i	per + (stem) + i	Nomina	Verba	persenjatai perdayai	peng + pisau + i peng + usaha + i

Pada tabel imbuhan per- di atas bisa kita lihat afiks per-, per-an, per-kan, dan per-i memiliki proses penggabungan dengan kelas kata yang cukup bervariasi. Kata yang dihasilkanpun dapat berupa verba atau nomina.

3.4.5 Morfotaktik Imbuhan ke-

Tabel yang berisi aturan-aturan morfotaktik untuk imbuhan ke- dapat dilihat pada tabel 3-6.

Tabel 3-6. Aturan Morfotaktik Imbuhan ke-

Variasi Imbuhan	Proses Pembentukan	Kelas Kata Stem	Kelas Kata Berimbuhan	Contoh	*Counter Example*
ke + stem	ke + (stem)	Nomina	Nomina	kekasih	ke + cinta
		Adjektiva	Nomina	ketua	ke + muda
		DLL	Nomina	ketiga kehendak	
ke + stem + an	ke + (stem) + an	Verba	Nomina	kedudukan	ke + jalan + an
		Adjektiva	Nomina	kecerdikan kerugian	
		Adjektiva	Adjektiva	kemerahan kekecilan	
		DLL	Nomina	kesebelasan kesatuan	
ke + stem + kan	ke + (stem) + kan	Nomina	Verba	kedepankan	
ke + ber + stem + an	ke + (ber + (stem)) + an	• (ber + stem) merupakan Verba. • Kelas kata stem dibahas pada tabel imbuhan ber-	Nomina	keberhasilan keberlanjutan	ke + berlaku + an ke + bertangan + an
ke + peN + stem + an	ke + (peN + (stem)) + an	• (peN + stem) merupakan Nomina. • Kelas kata stem dibahas pada tabel imbuhan peN-	Nomina	kependudukan	ke + pendapat + an

Pada tabel di atas bisa dilihat imbuhan ke- bisa bergabung dengan kata dasar dari kelas kata nomina, adjektiva, dan dll. Imbuhan ke-an bisa bergabung dengan kata dasar dari kelas kata verba, adjektiva, dan dll. Dari imbuhan ke- juga bisa dihasilkan 2 macam konfiks yaitu ke-ber-an (gabungan afiks ke-an dengan afiks ber-) dan ke-peN-an (gabungan afiks ke-an dengan afiks peN-). Keempat macam imbuhan tersebut bila digabungkan dengan kata dasar yang tertera di atas akan menghasilkan sebuah kata dari kelas kata nomina.

Sedikit perbedaan dari kebanyakan afiks pada kategori ke- terjadi pada afiks ke-kan. Afiks ke-kan bila digabungkan dengan kata dasar dari kelas kata dll justru akan menghasilkan suatu kata dengan kelas kata verba.

3.4.6 Morfotaktik Imbuhan ber-

Tabel yang berisi aturan-aturan morfotaktik untuk imbuhan ber- dapat dilihat pada tabel 3-7.

Tabel 3-7. Aturan Morfotaktik Imbuhan ber-

Variasi Imbuhan	Proses Pembentukan	Kelas Kata Stem	Kelas Kata Berimbuhan	Contoh	*Counter Example*
ber + stem	ber + (stem)	Verba intransitif	Verba	berjalan berbicara	
		Nomina	Verba	bertopi berkaki berbuah	ber + sayur
		Adjektiva	Verba	berlaku bersedih	ber + laris ber + lengkap
		DLL	Verba	bersatu berlima	
ber + stem + kan	ber + (stem) + kan atau (ber + (stem))	Verba	Verba	berhentikan berdirikan	

	+ kan				
		Nomina	Nomina	beratapkan berselimutkan	ber + buah + kan
		Adjektiva	Verba	berlakukan	ber + nyata + kan
ber + stem + an	ber + (stem) + an	Verba	Verba	berlarian bersalaman	ber + pengaduan ber + perikanan
ber + peN + stem + an	ber + (peN + (stem) + an)	• (peN + stem + an) merupakan Nomina. • Kelas kata stem dibahas pada tabel imbuhan peN-	Verba	berpendidikan berpenghasilan	ber + pengaduan ber + perikanan
ber + ke + stem + an	ber + (ke + (stem) + an)	• (ke + stem + an) merupakan Nomina. • Kelas kata stem dibahas pada tabel imbuhan ke-	Verba	berkedudukan berkekurangan	

Pada imbuhan ber- seperti ditunjukkan di atas ada tiga macam variasi afiks yang dapat dihasilkan yaitu ber-, ber-kan, dan ber-an. Ketiganya memiliki aturan yang berbeda dalam bergabungnya dengan suatu kata dasar dari kelas kata yang bersangkutan.

Selain tiga macam afiks tersebut, imbuhan ber- juga dapat menghasilkan 2 macam variasi konfiks yaitu ber-peN-an (gabungan afiks ber- dengan afiks peN-an) dan ber-ke-an (gabungan afiks ber- dengan ke-an). Untuk lebih lengkapnya tentang keenam variasi tersebut bisa dilihat langsung pada tabel di atas.

Kalau diperhatikan, seperti halnya imbuhan di- dan meN-, imbuhan ber- juga akan selalu menghasilkan suatu kata dengan kelas kata verba tidak peduli bergabung dengan kata dasar dari kelas kata apa.

3.4.7 Morfotaktik Imbuhan ter-

Tabel yang berisi aturan-aturan morfotaktik untuk imbuhan ter- dapat dilihat pada tabel 3-8.

Tabel 3-8. Aturan Morfotaktik Imbuhan ter-

Variasi Imbuhan	Proses Pembentukan	Kelas Kata Stem	Kelas Kata Berimbuhan	Contoh	*Counter Example*
ter + stem	ter + (stem)	Verba Transitif	Verba	terangkat terinjak	
		Adjektiva	Adjektiva	terpandai terbaru	
ter + stem + kan	ter + (stem) + kan atau ter + ((stem) + kan)	Verba	Verba	tertinggalkan terlupakan	ter + pergi + kan
		Nomina	Verba	terpinggirkan terkelompokkan	ter + tengah + kan

Imbuhan ter- memiliki dua macam variasi imbuhan yaitu ter- dan ter-kan yang untuk lebih jelasnya dapat dilihat pada tabel di atas.

Kata yang dihasilkan akan memiliki kelas kata verba kecuali untuk afiks ter- yang bergabung dengan adjektiva. Khusus untuk itu, akan menghasilkan suatu kata dengan kelas kata adjektiva.

3.4.8 Morfotaktik Imbuhan -an

Tabel yang berisi aturan-aturan morfotaktik untuk imbuhan -an dapat dilihat pada tabel 3-9.

Tabel 3-9. Aturan Morfotaktik Imbuhan -an

Variasi Imbuhan	Proses Pembentukan	Kelas Kata Stem	Kelas Kata Berimbuhan	Contoh	*Counter Example*
stem + an	(stem) + an	Verba	Nomina	tontonan dorongan makanan	lihat + an mandi + an
		Nomina	Nomina	sayuran piringan	ikan + an teko + an buku + an
		Adjektiva	Nomina	asinan manisan	pahit + an gemuk + an
		DLL	Nomina	satuan	sepuluh + an

Imbuhan –an seperti ditunjukkan pada tabel di atas dapat bergabung dengan kata dasar dari semua kelas kata yang ada. Hasil dari penggabungan tersebut akan menjadi suatu kata dengan kelas kata nomina.

3.4.9 Morfotaktik Imbuhan -kan

Tabel yang berisi aturan-aturan morfotaktik untuk imbuhan kan- dapat dilihat pada tabel 3-10.

Tabel 3-10. Aturan Morfotaktik Imbuhan -kan

Variasi Imbuhan	Proses Pembentukan	Kelas Kata Stem	Kelas Kata Berimbuhan	Contoh	*Counter Example*
stem + kan	(stem) + kan	Verba	Verba	bawakan belikan	
		Nomina	Verba	bunyikan kategorikan	
		Adjektiva	Verba	hitamkan	cantik +

				dinginkan	kan rajin + kan
		DLL	Verba	satukan	

Imbuhan –kan seperti ditunjukkan pada tabel di atas dapat bergabung dengan kata dasar dari semua kelas kata yang ada. Hasil dari penggabungan tersebut akan menjadi suatu kata dengan kelas kata verba.

3.4.10 Morfotaktik Imbuhan -i

Tabel yang berisi aturan-aturan morfotaktik untuk imbuhan -i dapat dilihat pada tabel 3-11.

Tabel 3-11. Aturan Morfotaktik Imbuhan -i

Variasi Imbuhan	Proses Pembentukan	Kelas Kata Stem	Kelas Kata Berimbuhan	Contoh	*Counter Example*
stem + i	(stem) + i	Verba	Verba	kendarai pukuli	dengar + i jual + i
		Nomina	Verba	kelilingi minyaki	luas + i kelapa + i
		Adjektiva	Verba	panasi terangi	bagus + i gelap + i

Imbuhan –i seperti ditunjukkan pada tabel di atas dapat bergabung dengan kata dasar dari verba, nomina, dan adjektiva asalkan tidak diakhiri fonem /i/. Hasil dari penggabungan tersebut akan menjadi suatu kata dengan kelas kata verba.

3.4.11 Morfotaktik Kata Ulang Sejati

Tabel yang berisi aturan-aturan morfotaktik untuk kata ulang sejati dapat dilihat pada tabel 3-12.

Tabel 3-12. Aturan Morfotaktik Kata Ulang Sejati

Imbuhan	Kelas Kata Stem	Kelas Kata Setelah Reduplikasi	Contoh	*Counter Example*
Tanpa imbuhan	Nomina	Nomina	buku-buku meja-meja	
	Verba	Verba	makan-makan jalan-jalan	
peN(-)	Nomina	Nomina	penyapu-penyapu	(peN + sabun) (-)
	Verba	Nomina	penulis-penulis pembungkus-pembungkus	(peN + lihat) (-) (peN + mandi) (-)
	Adjektiva	Nomina	pengrajin-pengrajin pembesar-pembesar	(peN + adil) (-)
peN-an(-)	Verba	Nomina	penulisan-penulisan pencatatan-pencatatan	
	Nomina	Nomina	pengalengan-pengalengan pengasapan-pengasapan	(peN + lemari + an) (-) (peN + kabut + an) (-)
	Adjektiva	Nomina	penghematan-penghematan pemecahan-pemecahan	(peN + malas + an) (-) (peN + cantik + an) (-)
per-an(-)	Nomina	Nomina	perindustrian-peindustrian perkebunan-perkebunan	
	Verba	Nomina	perputaran-perputaran permintaan-permintaan	(per + dengar + an) (-)

ke(-)	Nomina	Nomina	kekasih-kekasih	(ke + cinta) (-)
	Adjektiva	Nomina	ketua-ketua	(ke + muda) (-)
ke-an(-)	Verba	Nomina	kedudukan-kedudukan	(ke + jalan + an) (-)
	Adjektiva	Nomina	kekayaan-kekayaan kerugian-kerugian	
	DLL	Nomina	kesebelasan-kesebelasan	
-an(-)	Nomina	Nomina	piringan-piringan	(ikan + an) (-) (teko + an) (-)
	Verba	Nomina	tontonan-tontonan makanan-makanan	(lihat + an) (-) (mandi + an) (-)
	Adjektiva	Nomina	asinan-asinan manisan-manisan	(pahit + an) (-) (gemuk + an) (-)
	DLL	Nomina	satuan-satuan	(sepuluh + an) (-)

Di dalam kata ulang sejati terdapat tujuh macam imbuhan yang mungkin terjadi. Penggabungannya dengan kelas kata yang berkesesuaian dapat dilihat pada tabel di atas.

3.4.12 Morfotaktik Kata Ulang Sebagian

Tabel yang berisi aturan-aturan morfotaktik untuk kata ulang sebagian dapat dilihat pada tabel 3-13.

Tabel 3-13. Aturan Morfotaktik Kata Ulang Sebagian

Imbuhan	Kelas Kata Stem	Kelas Kata Setelah Reduplikasi	Contoh	*Counter Example*
meN- *(-)	Verba	Verba	memasak-masak memukul-mukul	meN + simpan(-) meN + tukar(-)

di- *(-)	Verba	Verba	diputar-putar	di + ikat(-) di + masak(-)
ber- *(-)	Verba	Verba	berlari-lari berbincang-bincang	ber + bicara(-) ber + tutup(-)
ter- *(-)	Verba	Verba	teringat-ingat	ter + angkat(-)

Di dalam kata ulang berimbuhan terdapat empat macam imbuhan yang mungkin terjadi. Penggabungannya dengan kelas kata yang berkesesuaian dapat dilihat pada tabel di atas.

3.4.13 Morfotaktik Kata Ulang Berimbuhan

Tabel yang berisi aturan-aturan morfotaktik untuk Kata Ulang Berimbuhan dapat dilihat pada tabel 3-14.

Tabel 3-14. Aturan Morfotaktik Kata Ulang Berimbuhan

Imbuhan	Kelas Kata Stem	Kelas Kata Setelah Reduplikasi	Contoh	*Counter Example*
*(-) meN-	Verba	Verba	tanam-menanam kejar-mengejar	
meN *(-) kan	Verba	Verba	membagi-bagikan	meN + baca(-) + kan meN + jalan(-) + kan
di *(-) kan	Verba	Verba	dimain-mainkan	di + pergi(-) + kan di + simpan(-) + kan
ke *(-) an	Adjektiva	Adjektiva	kekanak-kanakan kehijau-hijauan	ke + bersih(-) + an ke + rajin(-) + an
ber *(-) an	Verba	Verba	bersalam-salaman berlari-larian	
*(-) an	Nomina	Nomina	kuda-kudaan	buku(-) + an

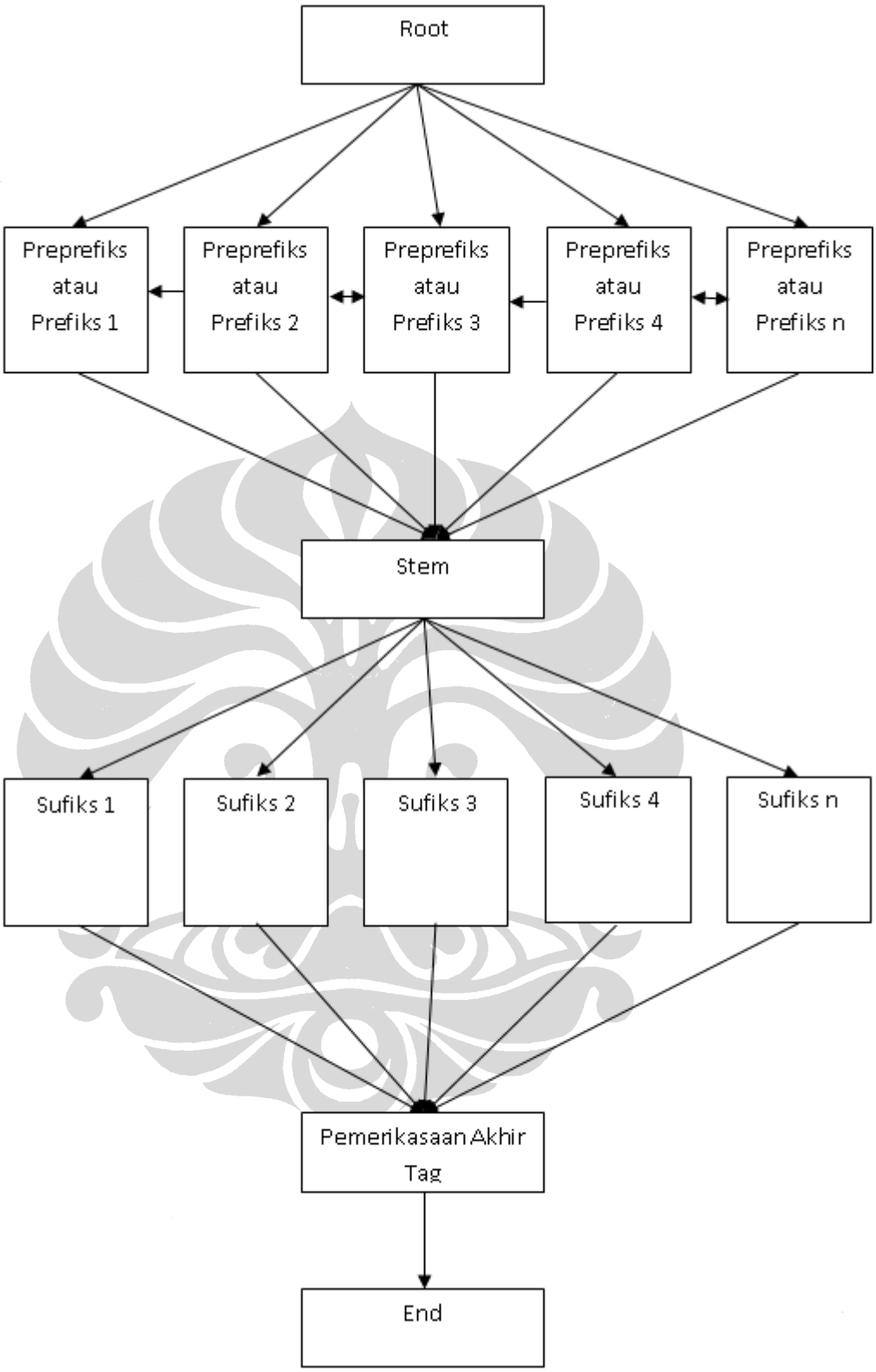
			buah-buahan	bola(-) + an
	Verba	Verba	tidur-tiduran	duduk(-) + an

Di dalam kata ulang berimbuhan terdapat enam macam imbuhan yang mungkin terjadi. Penggabungannya dengan kelas kata yang berkesesuaian dapat dilihat pada tabel di atas.

3.5 Rangkuman Aturan-aturan Morfotaktik

Setelah melihat semua tabel-tabel di atas, bisa dilihat bahwa untuk aturan morfotaktik hal utama yang perlu diamati adalah bisa atau tidaknya suatu imbuhan bergabung dengan bentuk kata dasar penyusunnya dan apa yang dihasilkan bila imbuhan tersebut dapat bergabung dengan kata dasar penyusunnya tersebut serta *tags* apa saja yang harus dimunculkan. Kecenderungan hasil banyak tergantung dari prefiks atau preprefiks yang mendahuluinya.

Untuk menggabungkan dan merangkum semua aturan-aturan morfotaktik tersebut pertama perlu dimulai dari *root* yang kemudian bergerak ke pemberian preprefiks dan prefiks-nya. Kemudian barulah bergerak ke dalam kata dasar penyusunnya dan kemudian bergerak ke variasi imbuhannya. Ilustrasi rangkuman rancangan morfotaktik dapat dilihat pada gambar 3-3.



Gambar 3-3. Rangkuman Rancangan Morfotaktik

Pada gambar 3.3 bisa kita lihat runtutan proses yang terjadi. Pada tahapan preprefiks atau prefiks, bisa bergerak ke prefiks lainnya jika suatu prefiks bisa menjadi preprefiks untuk prefiks tersebut. Hal ini ditunjukkan dengan panah kesamping pada ilustrasi di atas.

Lexicon diakses pada tahapan *Stem*. Pada bagian itu kita mendapatkan kata dasar penyusunnya. Kemudian dari tahapan itu akan menuju ke seluruh variasi imbuhan. Pada tahapan itulah akan diberikan *tags* yang sesuai.

Kemungkinan-kemungkinan variasi kata dasar, imbuhan, dan kata ulang yang akan diterima dalam rancangan morfotaktik ini di antaranya dapat dilihat pada tabel 3-15.

Tabel 3-15. Variasi yang Diterima dalam Rancangan Morfotaktik

No.	Variasi yang Diterima	Preprefiks	Prefiks	Sufiks Awal	Sufiks Akhir
1.	Kata Dasar	Tidak ada	Null	Null	Tidak Ada
2.	Imbuhan meN-	Tidak ada	meN-	Null	Tidak Ada
3.	Imbuhan meN-kan	Tidak ada	meN-	-kan	Tidak Ada
4.	Imbuhan meN-i	Tidak ada	meN-	-i	Tidak Ada
5.	Imbuhan meN-per	meN-	per-	Null	Tidak Ada
6.	Imbuhan meN-per-kan	meN-	per-	-kan	Tidak Ada
7.	Imbuhan meN-per-i	meN-	per-	-i	Tidak Ada
8.	Imbuhan meN-ke-kan	meN-	ke-	-kan	Tidak Ada
9.	Imbuhan meN-ber-kan	meN-	ber-	-kan	Tidak Ada
10.	Imbuhan meN-ter-kan	meN-	ter-	-kan	Tidak Ada
11.	Imbuhan di-	Tidak ada	di-	Null	Tidak Ada
12.	Imbuhan di-kan	Tidak ada	di-	-kan	Tidak Ada
13.	Imbuhan di-i	Tidak ada	di-	-i	Tidak Ada
14.	Imbuhan di-per	di-	per-	Null	Tidak Ada
15.	Imbuhan di-per-kan	di-	per-	-kan	Tidak Ada
16.	Imbuhan di-per-i	di-	per-	-i	Tidak Ada

17.	Imbuhan di-ke-kan	di-	ke-	-kan	Tidak Ada
18.	Imbuhan di-ber-kan	di-	ber-	-kan	Tidak Ada
19.	Imbuhan di-ter-kan	di-	ter-	-kan	Tidak Ada
20.	Imbuhan peN-	Tidak ada	peN-	Null	Tidak Ada
21.	Imbuhan peN-an	Tidak ada	peN-	-an	Tidak Ada
22.	Imbuhan peN-ber-an	peN-	ber-	-an	Tidak Ada
23.	Imbuhan per-	Tidak ada	per-	Null	Tidak Ada
24.	Imbuhan per-an	Tidak ada	per-	-an	Tidak Ada
25.	Imbuhan per-kan	Tidak ada	per-	-kan	Tidak Ada
26.	Imbuhan per-i	Tidak ada	per-	-i	Tidak Ada
27.	Imbuhan ke-	Tidak ada	ke-	Null	Tidak Ada
28.	Imbuhan ke-an	Tidak ada	ke-	-an	Tidak Ada
29.	Imbuhan ke-ber-an	ke-	ber-	-an	Tidak Ada
30.	Imbuhan ke-peN-an	ke-	peN-	-an	Tidak Ada
31.	Imbuhan ber-	Tidak ada	ber-	Null	Tidak Ada
32.	Imbuhan ber-an	Tidak ada	ber-	-an	Tidak Ada
33.	Imbuhan ber-kan	Tidak ada	ber-	-kan	Tidak Ada
34.	Imbuhan ber-peN-an	ber-	peN-	-an	Tidak Ada
35.	Imbuhan ber-ke-an	ber-	ke-	-an	Tidak Ada
36.	Imbuhan ter-	Tidak ada	ter-	Null	Tidak Ada
37.	Imbuhan ter-kan	Tidak ada	ter-	-kan	Tidak Ada
38.	Imbuhan -an	Tidak ada	Null	-an	Tidak Ada
39.	Imbuhan -kan	Tidak ada	Null	-kan	Tidak Ada
40.	Imbuhan -i	Tidak ada	Null	-i	Tidak Ada
41.	Kata Ulang Sejati Tanpa Imbuhan	Tidak ada	Reduplikasi	Reduplikasi	Tidak Ada
42.	Kata Ulang Sejati Dengan Imbuhan peN-	Reduplikasi	peN-	Null	Reduplikasi
43.	Kata Ulang Sejati Dengan Imbuhan peN-an	Reduplikasi	peN-	-an	Reduplikasi

44.	Kata Ulang Sejati Dengan Imbuhan per-an	Reduplikasi	per-	-an	Reduplikasi
45.	Kata Ulang Sejati Dengan Imbuhan ke-	Reduplikasi	ke-	Null	Reduplikasi
46.	Kata Ulang Sejati Dengan Imbuhan ke-an	Reduplikasi	ke-	-an	Reduplikasi
47.	Kata Ulang Sejati Dengan Imbuhan -an	Reduplikasi	Null	-an	Reduplikasi
48.	Kata Ulang Sebagian Dengan Imbuhan meN-	Reduplikasi	meN-	Reduplikasi	Null
49.	Kata Ulang Sebagian Dengan Imbuhan di-	Reduplikasi	di-	Reduplikasi	Null
50.	Kata Ulang Sebagian Dengan Imbuhan ber-	Reduplikasi	ber-	Reduplikasi	Null
51.	Kata Ulang Sebagian Dengan Imbuhan ter-	Reduplikasi	ter-	Reduplikasi	Null
52.	Kata Ulang Berimbuhan meN-	Reduplikasi	Kasus Khusus meN-	Reduplikasi	Null
53.	Kata Ulang Berimbuhan meN-	Reduplikasi	Kasus Khusus meN-	Reduplikasi	Null
54.	Kata Ulang Berimbuhan meN-kan	Reduplikasi	meN-	Reduplikasi	-kan
55.	Kata Ulang Berimbuhan di-kan	Reduplikasi	di-	Reduplikasi	-kan
55.	Kata Ulang Berimbuhan ke-an	Reduplikasi	ke-	Reduplikasi	-an
56.	Kata Ulang Berimbuhan ber-an	Reduplikasi	ber-	Reduplikasi	-an
57.	Kata Ulang Berimbuhan -an	Reduplikasi	Null	Reduplikasi	-an

Kelimapuluhtujuh macam variasi tersebut akan diterima berdasarkan rancangan morfologi yang dibangun. Kelimapuluh tujuh variasi tersebut memiliki spesifikasi preprefiks, prefiks, sufiks awal, sufiks akhir yang berbeda-beda.

Sebagai contoh kita lihat pada kata ulang berimbuhan –an di mana ia akan didefinisikan sebagai preprefiks reduplikasi, prefiks null, sufiks awal reduplikasi, dan sufiks akhir –an. Untuk variasi lainnya dapat dilihat lebih jauh pada Tabel 3-15 di atas.

Untuk permasalahan ditemukannya banyaknya pengecualian untuk saat ini diabaikan dulu. Jika nanti ditemukan penyebab yang pasti atas kemunculan pengecualian tersebut barulah akan diteliti lebih lanjut.

Setelah melalui aturan morfotaktik, peluluhan kata atau perubahan fonem yang terjadi akan dipaparkan lebih lanjut pada aturan-aturan morfofonemik di bawah ini.

3.6 Aturan-aturan Morfofonemik

Proses morfofonemik adalah proses yang menunjukkan proses perubahan suatu fonem menjadi fonem lain sesuai dengan fonem awal atau fonem yang mendahuluinya **[ALWI03]**. Pada rancangan morfologi untuk pengurai morfologi ini, perlu dipelajari terlebih dahulu bagaimana perilaku-perilaku morfofonemik pada suatu kata. Setelah ini akan diberikan aturan-aturan proses morfofonemik mengacu pada tata bahasa baku bahasa Indonesia **[ALWI03]**.

3.6.1 Morfofonemik Prefiks meN-

Tabel yang berisi aturan-aturan morfofonemik untuk prefiks meN- dapat dilihat pada tabel 3-16. Simbol "N" pada prefiks "meN" menunjukkan proses nasalisasi yang terjadi.

Tabel 3-16. Aturan Morfofonemik Prefiks meN-

Bentuk Dasar	Perubahan Fonem Prefiks	Perubahan Fonem Dasar	Contoh
Dimulai fonem /a/, /i/, /u/, /e/, /o/, /g/, /h/, /kh/	meng- /N/ *replacement with* /ng/	Tidak berubah	meN + ambil → mengambil meN + istirahatkan → mengistirahatkan meN + usulkan → mengusulkan meN + eja → mengeja

			meN + obati → mengobati meN + gunting → menggunting meN + hijau → menghijau meN + khawatirkan → mengkhawatirkan
Dimulai fonem /k/	me- /N/ *deletion*	/k/ *replaced by* /ng/	meN + kecil → mengecil
Dimulai fonem /L/, /m/, /n/, /r/, /y/, /w/	me- /N/ *deletion*	Tidak berubah	meN + lukis → melukis meN + makan→ memakan meN + naikkan → menaikkan meN + rancang → merancang meN + yakinkan → meyakinkan meN + wajibkan → mewajibkan
Dimulai fonem /d/, /c/, /j/, /sy/	men- /N/ *replacement with* n	Tidak berubah	meN + dirikan → mendirikan meN + cuci→ mencuci meN + jahit → menjahit meN + syukuri → mensyukuri
Dimulai fonem /t/	me- /N/ *deletion*	/t/ *replaced by* /n/	meN + tawarkan → menawarkan
Dimulai fonem /s/	me- /N/ *deletion*	/s/ *replaced by* /ny/	meN + simpan → menyimpan
Dimulai fonem /b/, /f/, /pr/	mem- /N/ *replaced by* /m/	Tidak berubah	meN + berikan → memberikan meN + fokuskan→ memfokuskan meN + prediksi→ memprediksi
Dimulai fonem /p/	me-	/p/ *replaced by* /m/	meN + putar→ memutar

	/N/ *deletion*		
Bersuku satu	menge- /N/ *replacement with* /nge/	Tidak berubah	meN + rem → mengerem

Bisa dilihat dari tabel di atas, untuk prefiks meng-, ada 9 macam aturan morfofonemik. Untuk prefiks meN- yang diikuti kata dasar berawalan fonem /a/, /i/, /u/, /e/, /o/, /g/, /h/, /kh/, terjadi perubahan fonem pada prefiks yaitu perubahan /meN/ menjadi /meng/ akan tetapi tidak terjadi perubahan fonem pada kata dasarnya.

Selanjutnya, untuk prefiks meN- yang diikuti kata dasar berawalan fonem /k/, perubahan terjadi pada fonem kata dasarnya, di mana /k/ tersebut berubah menjadi /ng/. Perubahan fonem pada kata dasar juga terjadi pada prefiks meN- yang diikuti kata dasar berawalan fonem /s/ dan /p/. Kata dasar berawalan fonem /s/ berubah menjadi /ny/, sedangkan pada kata dasar berawalan fonem /p/ berubah menjadi /m/.

Selain tiga contoh di atas masih ada satu kasus lagi yang mengalami perubahan fonem. Pada kata dasar berawalan fonem /t/, terjadi perubahan fonem yaitu fonem /t/ pada awal kata dasarnya berubah menjadi /n/. Pada aturan ini "meN" juga berubah menjadi "me".

Pada aturan lainnya perubahan dilakukan pada fonem prefiksnya saja. Fonem kata dasarnya tidak mengalami perubahan. Pada prefiks meN- yang diikuti kata dasar berawalan fonem /L/, /m/, /n/, /r/, /y/, /w/, /s/, /k/, /p/, "meN" berubah menjadi "me". Bisa dilihat ini memiliki kemiripan dengan prefiks meN- yang diikuti kata dasar berawalan fonem /t/ hanya saja kali ini tidak mengalami perubahan fonem pada kata dasarnya. Akan tetapi kemiripan-kemiripan seperti itu bisa kita manfaatkan untuk implementasi nantinya. Untuk 3 aturan lainnya yang mengalami perubahan pada fonem prefiksnya saja bisa dilihat lebih lanjut pada tabel di atas.

3.6.2 Morfofonemik Prefiks peN-

Tabel yang berisi aturan-aturan morfofonemik untuk prefiks peN- dapat dilihat pada tabel 3-17. Simbol "N" pada prefiks "peN" menunjukkan proses nasalisasi yang terjadi.

Tabel 3-17. Aturan Morfofonemik Prefiks peN-

Bentuk Dasar	Perubahan Fonem Prefiks	Perubahan Fonem Dasar	Contoh
Dimulai fonem /r/ atau suku pertama berakhir /er/ (kecuali kata "rajin → pengrajin")	pe- /N/ *deletion*	Tidak berubah	peN + rusak → perusak peN + kerja → pekerja
Dimulai fonem /a/, /i/, /u/, /e/, /o/, /g/, /h /	peng- /N/ *replacement with* /ng/	Tidak berubah	peN + atur → pengatur peN + ikat → pengikat peN + ukur → pengukur peN + endapan → pengendapan peN + olahan → pengolahan peN + garis → penggaris peN + hapus → penghapus
Dimulai fonem /k/	pe- /N/ *deletion*	/k/ *replaced by* /ng/	peN + kirim → pengirim
Dimulai fonem /L/, /m/, /n/, /w/	pe- /N/ *deletion*	Tidak berubah	peN + lukis → pelukis peN + main → pemain peN + nyanyi → penyanyi peN + wangi → pewangi

Dimulai fonem /c/, /j/	pen- /N/ *replacement with* /n/	Tidak berubah	peN + catatan → pencatatan peN + jual → penjual
Dimulai fonem /d/	pen- /N/ *replacement with* /n/	Tidak berubah	peN + dapat → pendapat
	pe- /N/ *deletion*	Tidak berubah	peN + dagang → pedagang
Dimulai fonem /t/	pe- /N/ *deletion*	Tidak berubah	peN + tani → petani
	pe- /N/ *deletion*	/t/ *replaced by* /n/	peN + tari → penari
Dimulai fonem /s/	pe- /N/ *deletion*	Tidak berubah	peN + suruh → pesuruh
	pe- /N/ *deletion*	/s/ *replaced by* /ny/	peN + sakit → penyakit
Dimulai fonem /b/, /f/	pem- /N/ *replaced by* /m/	Tidak berubah	peN + buru → pemburu peng + fitnah→ pemfitnah
Dimulai fonem /p/	pe- /N/ *deletion*	/p/ *replaced by* /m/	peN + pikir → pemikir peN + prosesan → pemrosesan
Bersuku satu	penge- /N/ *replacement with* /nge/	Tidak berubah	peN + bom → pengebom

Kata khusus (“ajar”)	pel- /N/ *replaced by* /L/	Tidak berubah	peN + ajar → pelajar

Seperti dapat kita lihat, prefiks peN- dapat mengalami 12 macam proses morfofonemik. Diantaranya ada yang mengalami perubahan pada fonem prefiks yaitu “peN” menjadi “pe”, “pen”, “penge”, dan “pel”.

Banyak juga perubahan fonem prefiks “peN” menjadi “pe” yang dipadu dengan perubahan fonem kata dasarnya. Diantaranya terjadi ketika /k/ berubah menjadi /ng/, /p/ berubah menjadi /m/, /t/ berubah menjadi /n/, dan /s/ berubah menjadi /ny/.

3.6.3 Morfofonemik Prefiks ber-

Tabel yang berisi aturan-aturan morfofonemik untuk prefiks ber- dapat dilihat pada tabel 3-18.

Tabel 3-18. Aturan Morfofonemik Prefiks ber-

Bentuk Dasar	Perubahan Fonem Prefiks	Perubahan Fonem Dasar	Contoh
Dimulai fonem /r/ atau suku pertama berakhir /er/	be- /r/ *deletion*	Tidak berubah	ber + runding → berunding ber + bekerja → bekerja
Kata khusus (“ajar”)	bel- /r/ *replaced by* /L/	Tidak berubah	ber + ajar → belajar
	ber- Tidak berubah	Tidak berubah	ber + lari → berlari ber + irama → berirama

Untuk prefiks ber- hanya ada tiga aturan morfofonemik. Ketiganya tidak mengakibatkan perubahan pada fonem kata dasarnya. Perubahan hanya mungkin

terjadi pada fonem prefiksnya. Untuk prefiks ber- yang diikuti kata dasar berawalan fonem /r/ atau suku pertama pada kata dasarnya berakhiran /er/, prefiks "ber" mengalami perubahan menjadi "be". Untuk prefiks ber- yang diikuti kata "ajar", prefiks "ber" berubah menjadi "bel". Untuk prefiks ber- yang diikuti kata dasar dengan fonem selain yang disebutkan, tidak ada perubahan yang terjadi.

3.6.4 Morfofonemik Prefiks ter-

Tabel yang berisi aturan-aturan morfofonemik untuk prefiks ter- dapat dilihat pada tabel 3-19.

Tabel 3-19. Aturan Morfofonemik Prefiks ter-

Bentuk Dasar	Perubahan Fonem Prefiks	Perubahan Fonem Dasar	Contoh
Dimulai fonem /r/	te- /r/ *deletion*	Tidak berubah	ter + rebut → terebut
	ter- Tidak berubah	Tidak berubah	ter + angkat → terangkat ter + percaya → terpercaya

Pada prefiks ter- seperti bisa kita lihat pada tabel di atas hanya ada dua macam aturan morfofonemik yang terjadi. Untuk prefiks ter- yang diikuti kata dasar berawalan fonem /r/, "ter" berubah menjadi "te". Untuk prefiks ter- yang diikuti kata dasar dengan fonem selain /r/, tidak mengalami perubahan.

3.6.5 Morfofonemik Prefiks di-

Tabel yang berisi aturan-aturan morfofonemik untuk prefiks di- dapat dilihat pada tabel 3-20.

Tabel 3-20. Aturan Morfofonemik Prefiks di-

Bentuk Dasar	Perubahan Fonem Prefiks	Perubahan Fonem Dasar	Contoh
	di- Tidak berubah	Tidak berubah	di + operasikan → dioperasikan di + kejar → dikejar

Prefiks di- tidak mengalami proses perubahan fonem baik di prefiksnya maupun di kata dasarnya.

3.6.6 Morfofonemik Prefiks -kan

Tabel yang berisi aturan-aturan morfofonemik untuk prefiks -kan dapat dilihat pada tabel 3-21.

Tabel 3-21. Aturan Morfofonemik Prefiks -kan

Bentuk Dasar	Perubahan Fonem Sufiks	Perubahan Fonem Dasar	Contoh
	-kan Tidak berubah	Tidak berubah	letak + kan → letakkan membuka + kan → membukakan

Sufiks -kan tidak mengalami proses perubahan fonem baik di sufiksnya maupun di kata dasarnya.

3.6.7 Morfofonemik Prefiks -an

Tabel yang berisi aturan-aturan morfofonemik untuk prefiks -an dapat dilihat pada Tabel 3-22.

Tabel 3-22. Aturan Morfofonemik Prefiks -an

Bentuk Dasar	Perubahan Fonem Sufiks	Perubahan Fonem Dasar	Contoh
	-an Tidak berubah	Tidak berubah	minum + an → minuman bersama + an → bersamaan

Sufiks -an tidak mengalami proses perubahan fonem baik di sufiksnya maupun di kata dasarnya.

3.6.8 Morfofonemik Prefiks -i

Tabel yang berisi aturan-aturan morfofonemik untuk prefiks -i dapat dilihat pada Tabel 3-23.

Tabel 3-23. Aturan Morfofonemik Prefiks -i

Bentuk Dasar	Perubahan Fonem Sufiks	Perubahan Fonem Dasar	Contoh
Selain yang diakhiri fonem /i/	-i Tidak berubah	Tidak berubah	akhir + i → akhiri

Sufiks -i tidak mengalami proses perubahan fonem baik di sufiksnya maupun di kata dasarnya. Hanya saja untuk sufiks -i tidak dapat bergabung dengan kata dasar yang diakhiri fonem /i/.

3.7 Rangkuman Aturan-aturan Morfofonemik Imbuhan

Berdasarkan perubahan fonem imbuhan dan kata dasar yang terjadi, aturan-aturan tersebut di atas bisa kita rangkum ke dalam Tabel 3-24 dan Tabel 3-25.

Tabel 3-24. Tabel Perubahan Fonem Imbuhan

Perubahan Fonem	Imbuhan	Bentuk Dasar	Contoh
/N/ *replacement with* /n/	meN-	Dimulai fonem /d/, /c/, /j/, /sy/	meN + dirikan → mendirikan meN + cuci→ mencuci meN + jahit → menjahit meN + syukuri → mensyukuri
	peN-	Dimulai fonem /c/, /j/, /d/	peN + catatan → pencatatan peN + jual → penjual peN + dapat → pendapat
/r/ *deletion*	ber-	Dimulai fonem /r/ atau suku pertama berakhir /er/	ber + runding → berunding ber + bekerja → bekerja
	ter-	Dimulai fonem /r/	ter + rebut → terebut
/N/ *deletion*	meN-	Dimulai fonem /L/, /m/, /n/, /r/, /y/, /w/,	meN + lukis → melukis

		/t/, /s/, /p/, /k/	meN + makan→ memakan meN + naikkan → menaikkan meN + rancang → merancang meN + yakinkan → meyakinkan meN + wajibkan → mewajibkan meN + tawarkan → menawarkan meN + putar→ memutar
	peN-	Dimulai fonem /L/, /m/, /n/, /w/, /r/, /d/, /t/, /s/, /p/, /k/	peN + lukis → pelukis peN + main → pemain peN + nyanyi → penyanyi peN + wangi → pewangi peN + rusak → perusak peN + dagang → pedagang peN + ternak → peternak peN + suruh → pesuruh peN + pikir → pemikir
	peN-	Suku pertama berakhir /er/ (kecuali kata "rajin → pengrajin")	peN + kerja → pekerja
/N/ *replaced by* /m/	meN-	Dimulai fonem /b/, /f/	meN + berikan → memberikan meN + fokuskan→ memfokuskan
	peN-	Dimulai fonem /b/, /f/	peN + buru → pemburu peN + fitnah→ pemfitnah
/N/ *replacement with* /nge/	meN-	Bersuku satu	meN + rem → mengerem

/N/ *replaced by* /L/	peN-	Kata khusus (“ajar”)	peN + ajar → pelajar
/r/ *replaced by* /L/	ber-	Kata khusus (“ajar”)	ber + ajar → belajar

Tabel 3-25. Tabel Perubahan Fonem Kata Dasar

Perubahan Fonem	Imbuhan	Bentuk Dasar	Contoh
/k/ *replaced by /ng/*	meN-	Dimulai fonem /k/	meN + kantuk → mengantuk
	peN-	Dimulai fonem /k /	peN + kirim → pengirim
/s/ *replaced by /ny/*	meN-	Dimulai fonem /s/	meN + siram → menyiram
	peN-	Dimulai fonem /s/	peN + sebaran → penyebaran
/p/ *replaced by /m/*	meN-	Dimulai fonem /p/	meN + pindahkan → memindahkan
	peN-	Dimulai fonem /p/	peN + pakai → pemakai
/t/ *replaced by* /n/	meN-	Dimulai fonem /t/	meN + tertawakan → menertawakan
	peN-	Dimulai fonem /t/	peN + tampilan → penampilan

Sesuai pada prinsip *two-level morphology* yang sudah dijelaskan pada BAB II, pemodelan *transducers* dari aturan-aturan tersebut akan lebih mudah untuk dibangun jika kita golongkan sesuai perubahan yang terjadi baik pada imbuhannya maupun pada kata dasarnya, sehingga akan lebih mudah juga untuk

mengimplementasikannya nanti. Atas dasar itulah kedua tabel rangkuman di atas dibuat.

Untuk kata ulang rancangannya sama saja dengan morfotaktik imbuhan. Hal tersebut terjadi diakibatkan proses morfofonemik yang terjadi pada kata ulang pada dasarnya terjadi pada imbuhan di dalam kata ulang itu sendiri.

Rancangan ini sedikit banyak juga mendapatkan pengaruh dari Laporan Tugas Akhir (TA) Hendra Hartono pada tahun 2002. Untuk level *tags* tentu saja berbeda cukup jauh, akan tetapi dari aturan-aturan morfofonemik Laporan TA Hendra dapat dijadikan pertimbangan, sehingga untuk bagian aturan-aturan morfofonemik rancangan ini juga merujuk kepada Laporan TA Hendra di samping sumber-sumber lainnya tentunya. Rancangan penelitian ini dan TA Hendra sama-sama mempergunakan prinsip *two-way morphology*.

Dari semua penjelasan di BAB 3 ini telah didapatkan rancangan yang dibutuhkan untuk melakukan implementasi pengurai morfologi. Sesuai pembahasan pada subbab 3.1, aturan-aturan morfofonemik maupun morfotaktik akan berperan besar dalam proses pengolahan morfologi suatu kata. Setelah semua rancangan selesai dirumuskan, rancangan tersebut bisa dilanjutkan ke tahapan implementasi di mana semua rancangan tadi akan direalisasikan ke dalam suatu program pengurai morfologi. Untuk implementasi tersebut akan dijelaskan pada BAB 4.

BAB 4 IMPLEMENTASI PENGURAI MORFOLOGI BAHASA INDONESIA

Dalam bab ini akan dijelaskan terlebih dahulu sekilas mengenai *tools* yang digunakan dalam mengembangkan pengurai morfologi ini, kemudian akan dilanjutkan paparan tentang tahapan implementasi yang telah dilalui mulai dari implementasi aturan morfotaktik hingga morfofonemik.

4.1 *Finite-State Authoring Software Tools*

Seperti halnya kebanyakan pengurai morfologi dalam bahasa lain, pengurai morfologi bahasa Indonesia ini, juga menggunakan *tools* untuk membangunnya. Adapun kali ini dipilih dua buah *tools* yang memudahkan dalam membangun *finite-state network* nantinya.

4.1.1 Xerox Finite-State Tool (XFST)

Xerox Finite-State Tool (XFST) adalah sebuah aplikasi yang dapat digunakan untuk melakukan komputasi ataupun manipulasi pada *finite-state networks*. Aplikasi ini dibangun dari riset yang dimulai oleh Ronald M. Kaplan dan Martin Kay pada awal tahun 1980-an bertempat di Xerox Palo Alto Research Center. Setelah pengembangan awal tersebut, riset terus dilakukan untuk menyempurnakan XFST ini terutama dalam hal manipulasi *regular expression* (regex).

XFST ini menerima masukan berupa aturan-aturan yang mendefinisikan sebuah *finite-state network*, lalu membangun model komputasi berdasarkan *network* tersebut. Aturan-aturannya dijabarkan dalam bentuk regex. Adapun bisa kita lihat contoh aturan sederhana yang dipakai oleh XFST:

```
define Digit [%0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9] ;
```

Contoh di atas mendefinisikan *simple network* dengan regex berupa angka 0-9. *Network* tersebut akan menerima angka dari 0-9. [%0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8

| 9] merupakan regex-nya yang akan dibentuk menjadi suatu *simple networks*, sedangkan "Digit" merupakan *symbol* yang didefinisikan sebagai *network* hasil regex tersebut.

Selain untuk membuat *simple network*, XFST juga bisa untuk membuat *transducer*. Berikut dapat dilihat contohnya:

```
read regex  %. -> %, , %, -> %. || Digit _ Digit ;
```

Contoh di atas mendefinisikan *transducer* yang melakukan *replacement rules*. *Replacement rules* dilakukan dengan memakai tanda "->". Jadi aturan di atas bisa dibaca sebagai replacement "." dengan "," yang diparalel dengan replacement "," dengan "." dengan syarat berada di antara "Digit".

Aturan-aturan yang telah didefinisikan dapat digabung dengan operasi paralel ataupun *compose*. Operasi paralel berarti penggabungan tersebut akan terjadi tanpa memperhatikan urutannya, sedangkan operasi *compose* akan memperhatikan urutannya. Berikut contoh penggabungan dengan operasi paralel:

```
define Paralel 1->2||Digit_Digit ,, 2->3||Digit_Digit;
```

Dengan oparasi paralel seperti contoh di atas, "1" akan digantikan oleh "2" setiap berada di antara "Digit" bersamaan dengan "2" akan digantikan oleh "3" setiap berada di antara "Digit". Operasi tersebut mengakibatkan "1" akan menjadi "2" dan "2" akan menjadi "3" setelah aturan ini. Paralelisasi menggunakan tanda "," atau ",,". Sedangkan untuk *compose*:

```
define Compose 1->2||Digit_Digit .o. 2->3||Digit_Digit;
```

Pada contoh *compose* seperti di atas, "1" akan digantikan oleh "2" setiap berada di antara "Digit" barulah kemudian "2" akan digantikan oleh "3" setiap berada di dalam "Digit". Operasi tersebut mengakibatkan baik "1" maupun "2" akan digantikan oleh "3" setelah aturan ini. *Compose* dilakukan dengan menggunakan tanda ".o.".

Kelebihan utama dari XFST yang membuat *tool* ini dipakai untuk mengembangkan pengurai morfologi adalah kemampuan XFST untuk membaca regex dalam bentuk *syntax* XFST maupun dalam bentuk *binary* kemudian meng-*compile*-nya menjadi satu atau lebih *finite-state networks* mulai dari *simple networks* sampai *transducers*. Berpijak dari kemampuannya itu, bisa dikembangkan berbagai macam komputasi dan manipulasi dengan *finite-state networks* untuk keperluan pembangunan pengurai morfologi ini. Hasil dari *finite-state networks* tersebut bisa disimpan baik pada *binary* maupun *text files*.

Satu lagi fungsi yang disediakan XFST dan sangat membantu pengembangan pengurai morfologi ini adalah *compile-replace*. *Compile-replace* ini ditujukan untuk menangani adanya *nonconcatenative morphology* pada bahasa Indonesia. Proses yang sebenarnya dilakukan pada *compile-replace* adalah menjalankan sebuah regex *compiler* lanjutan untuk output dari regex *compiler* sebelumnya. Hasilnya adalah sebuah *finite-state* yang sudah dimodifikasi **[BEES03]**. Agar hasil dari regex *compiler* sebelumnya dapat diproses oleh *compile-replace* pada regex *compiler* lanjutan, sebuah regex harus diberi *delimiter* yaitu simbol "^[" dan "^]". Dengan adanya *delimiter* tersebut, *compile-replace* akan membacanya dan melakukan kompilasi pada regex di dalamnya.

Pada pengurai morfologi bahasa Indonesia ini, *compile-replace* akan dimanfaatkan untuk menguraikan dan menganalisa bentuk kata ulang. Untuk kata ulang, kita perlu membatasi pengulangan agar kata yang dimaksud hanya muncul sebanyak 2 kali. Oleh karena itu dalam regex yang dibatasi oleh *delimiter* "^[" dan "^]" kita sisipkan "^2" yang akan membuat kata tersebut muncul dua kali. Hasil dari perintah *compile-replace* kemudian akan membuat kata tersebut berulang dua kali sehingga yang perlu kita lakukan selanjutnya adalah membuat aturan untuk memunculkan pemisah "-" di antara kata yang berulang tersebut. Di luar *compile-replace* dan aturan pemisah "-", kasus kata ulang mengalami proses morfofonemik dan morfotaktik juga seperti halnya kata-kata lainnya.

Compile-replace yang digunakan untuk pengurai morfologi bahasa Indonesia ini adalah *compile-replace lower* yaitu ketika proses *compile-replace* diimplementasikan pada bagian pita bawah (*lower tape*).

4.1.2 Finite-State Lexicon Compiler (LEXC)

Finite-state lexicon compiler (LEXC) adalah suatu *compiler* yang dapat mendeklarasikan atau mendefinisikan suatu *finite-state automata* dan *finite-state transducers* yang akan digunakan untuk membuat suatu *lexicon* dan *lexical transducers*. Lexc dibangun oleh Lauri Karttunen dan Todd Yampol pada tahun 1992-1993 bertempat di Xerox Palo Alto Research Center. Lexc *compiler* ditulis dalam bahasa C berdasarkan implementasi yang dilakukan Karttunen pada tahun 1990 menggunakan Common Lisp.

Lexc ini sendiri pada dasarnya melakukan manipulasi-manipulasi regex seperti yang dilakukan oleh XFST. Hanya saja jika kita ibaratkan seperti bahasa pemrograman pada umumnya, LEXC adalah bentuk *high-level language* dari XFST. Lexc lebih dikhususkan membuat *transducer* yang berhubungan dengan lexicon dan lebih memiliki alur yang eksplisit dalam prosesnya. XFST dapat membuat segala macam variasi *network* dengan deklarasi yang singkat sedangkan LEXC lebih spesifik dengan deklarasi yang lebih mendalam.

Proses implementasi pada LEXC berjalan dengan pergerakan dari suatu *continuation class* yang satu menuju ke *continuation class* yang lain. Berikut contoh sederhana sebuah *code* dalam LEXC dapat dilihat pada gambar 4-1.

```
LEXICON Root
Noun;
Verb;

LEXICON Noun
line NounSuffix;

LEXICON Verb
dine VerbSuffix;
line VerbSuffix;

LEXICON NounSuffix
s #;
#;

LEXICON VerbSuffix
s #;
d #;
#;
```

Gambar 4-1. Cuplikan Code Sederhana dalam Lexc

Code pertama kali akan bergerak dari *class* Root. Jika tidak ditemukan *class* Root maka akan bergerak dari *class* yang dideklarasikan paling atas. Kemudian dari *class* Root akan menuju ke *continuation class* Noun dan *continuation class* Verb. Dan begitu seterusnya sampai ke *terminator symbol* yang dilambangkan dengan "#".

Untuk manipulasi lebih lanjut, dapat kita gunakan *flag diacritics* pada regex. *Flag diacritics* adalah suatu tanda yang digunakan untuk pengaturan *feature structure and unification*. *Flag diacritics* ini memungkinkan dilakukan pembatasan-pembatasan aturan morfotaktik yang sah dan tidak sah. Contoh penggunaan *flag diacritics* dapat dilihat pada gambar 4-2.

```
LEXICON Root
NounVerb;

LEXICON NounVerb
<d i n e "@P.POS.Verb@"> Suffix;
<l i n e "@P.POS.Verb@"> Suffix;
<l i n e "@P.POS.Noun@"> Suffix;

LEXICON Suffix
<s> #;
<d "@R.POS.Verb@"> #;
#;
```

Gambar 4-2. Cuplikan Contoh Sederhana Penggunaan *Flag Diacritics*

Pada *code* di atas, ketika masuk ke *continuation class* Suffix, bisa diperiksa terlebih dahulu. Penambahan "d" hanya dapat dilakukan pada regex yang memiliki flag @P.POS.Verb@.

Dalam pengembangan pengurai morfologi ini, digunakan tiga macam *flag diacritics* yang membantu pekerjaan ini. Tiga macam *flags* tersebut di antaranya:

1. *Positive (Re)Setting* (@P.*feature*.*value*@). Merupakan *flag diacritic* yang digunakan untuk men-*set* suatu *feature* sesuai dengan *value*-nya, sehingga *feature* dengan *value* tersebut bernilai positif.

2. *Require Test* (@R.*feature*.*value*@). Merupakan *flag diacritic* yang digunakan untuk menjalankan tes yang hanya akan berhasil jika *feature* tersebut telah di-*set* sesuai *value*-nya. Dengan kata lain tes ini akan lulus jika ditemukan *feature* dengan *value* tersebut yang telah di-*set* positif dan akan gagal jika tidak ditemukan *feature* dengan *value* yang berkesesuaian ataupun jika *feature* dengan *value* tersebut di-*set* negatif.

3. *Disallow Test* (@D.*feature*.*value*@). Merupakan *flag diacritic* yang digunakan untuk menjalankan tes yang akan gagal jika *feature* dengan *value* tersebut sudah di-*set* positif. Dengan kata lain tes ini

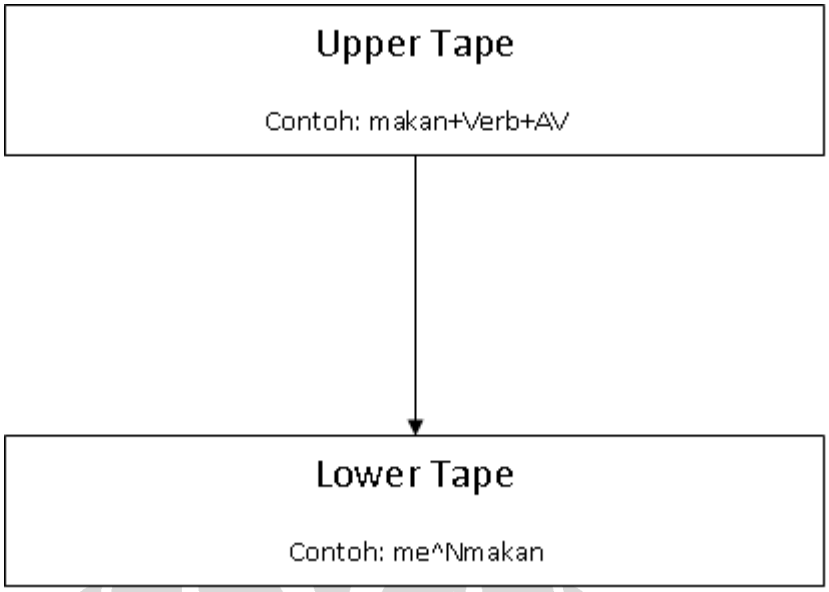
dijalankan untuk mencegah munculnya *feature* dengan *value* tertentu.

Dalam pengurai morfologi, LEXC lazim digunakan untuk morfotaktik dari sebuah bahasa. Hal ini dikarenakan kelebihannya dalam membuat *lexical transducers* yang memudahkan pendeklarasian aturan-aturan morfotaktik tersebut. Hasil dari LEXC ini adalah suatu *finite-state automaton* ataupun *transducer* yang kemudian akan digabungkan dengan aturan-aturan morfofonemik yang ditulis pada XFST.

4.2 Implementasi Aturan Morfotaktik

Aturan-aturan morfotaktik diimplementasikan dalam LEXC. Dalam LEXC tersebut dibuat sebuah *lexical transducer* yang memenuhi semua syarat yang terkandung dalam aturan morfotaktik yang sudah dirancang pada BAB 3. *Lexical transducer* tersebut akan berfungsi sebagai kumpulan aturan morfotaktik yang telah digabungkan untuk pengurai morfologi ini.

Sesuai yang sudah dirancang pada BAB 3, *lexical transducer* ini akan bergerak mulai dari prefiksnya. Pada tahapan ini digunakan keunggulan LEXC dalam memanipulasi regex untuk memasukkan prefiks tersebut dan juga untuk mengatur *tags* yang diperlukan. Dalam pengaturan regex ini perlu diperhatikan hal-hal yang perlu dimunculkan sebagai *input* dan sebagai *output*. Sesuai prinsip *two-level morphology* pada BAB 2, *input* akan berdiri pada pita atas (*Upper Tape*), sedangkan *output* akan berdiri pada pita bawah (*Lower Tape*) atau bisa juga sebaliknya tergantung cara kita memandangnya. Lebih jelasnya bisa kita lihat pada gambar 4-3.



Gambar 4-3. Contoh *Input-Output* Aturan Morfotaktik

Dari gambar di atas, bisa dilihat bahwa panah ke bawah tersebut merupakan proses morfotaktik yang diatur dalam LEXC. Pada tahapan memasukkan prefiks yang dibahas sebelumnya, ia baru memasukkan "me^N" pada pita bawah. Sedangkan pada pita atas belum terdapat apa-apa. Contoh pengimplementasian aturan tersebut dapat kita lihat pada gambar 4-4.

```
LEXICON Root
     Prefiks ;

LEXICON Prefiks
     ...
     PrefiksMeng ;
     ...
     ...
     ...

LEXICON PrefiksMeng
<[0 .x. m e "^N"] "@P.PREF.meng@">    Stems ;
...
...
...
```

Gambar 4-4. Cuplikan Pengimplementasian untuk Kasus meNmakan (1)

Dari Root bergerak menuju *continuation class* Prefiks, kemudian menuju *continuation class* PrefiksMeng. Pada *continuation class* PrefiksMeng, dilakukan *cross-product* pada regex-nya (menggunakan tanda ".x.") untuk memunculkan

“me^N” pada pita bawah. Kemudian kita *set flag diacritics* untuk prefiks tersebut *flag diacritics* yang di-set pada bagian ini nantinya akan menjadi input untuk pemeriksaan pada *classes* selanjutnya.

Kemudian dari tahapan pemasukan prefiks akan bergerak ke tahapan pemasukan kata dasar pembentuknya atau disebut juga tahapan *stems*. Pada tahapan ini, kata dasar pembentuknya akan dimasukkan. Jika dilihat pada ilustrasi di atas, pita atas akan terbentuk kata “makan”, sedangkan pada pita bawah terbentuk kata “me^Nmakan”. Sampai disini belum ditentukan batasan suatu prefiks bisa bergabung dengan kelas kata apa saja ataupun *tags* yang dihasilkan. Implementasi *code* dapat kita lihat pada gambar 4-5.

```
LEXICON Stems
<Vrb "@P.STEM.Vrb@">    Sufiks ;
...
...
...

LEXICON Sufiks
     SufiksNull ;
     ...
     ...
     ...
```

Gambar 4-5. Cuplikan Pengimplementasian untuk Kasus meNmakan (2)

Dari *class* Stems akan mengambil daftar *lexicon* kata dasarnya, kemudian di-*set flag diacritics* untuk tiap kelas kata dasarnya barulah selanjutnya bergerak menuju *continuation class* Sufiks yang akan mengarahkan kepada *continuation class* variasi akhiran.

Bisa bergabungnya suatu imbuhan dengan kelas kata apa saja dan apa saja *tags* yang dihasilkan baru diatur di tahapan terakhir. Pada tahapan semua variasi imbuhan ini akan ada yang memastikan bahwa suatu imbuhan hanya bisa bergabung dengan kelas kata tertentu saja. Pada tahapan ini semua *tags* juga diberikan. Lebih lanjut lagi, pada tahapan ini jugalah akhiran akan diberikan. Dapat dilihat pada ilustrasi di atas, pita atas dan bawah akan sempurna setelah tahapan ini. Implementasi dapat dilihat pada gambar 4-6.

```
LEXICON SufiksNull
...
...
...
! imbuhan meng-
<"+Verb":0 "+AV":0 "@D.PREPREF@" "@D.REDUP@" "@R.PREF.meng@" "@P.SUFF.null@">                    Redup2 ;
...
...
...

LEXICON Redup2
! untuk kata dasar dan imbuhan
<"@D.PREPREF.Redup@" "@D.PREF.Redup@" "@D.PREF.RedupKhusus@">          CausApplKan ;
...
...
...
```

Gambar 4-6. Cuplikan Pengimplementasian untuk Kasus meNmakan (3)

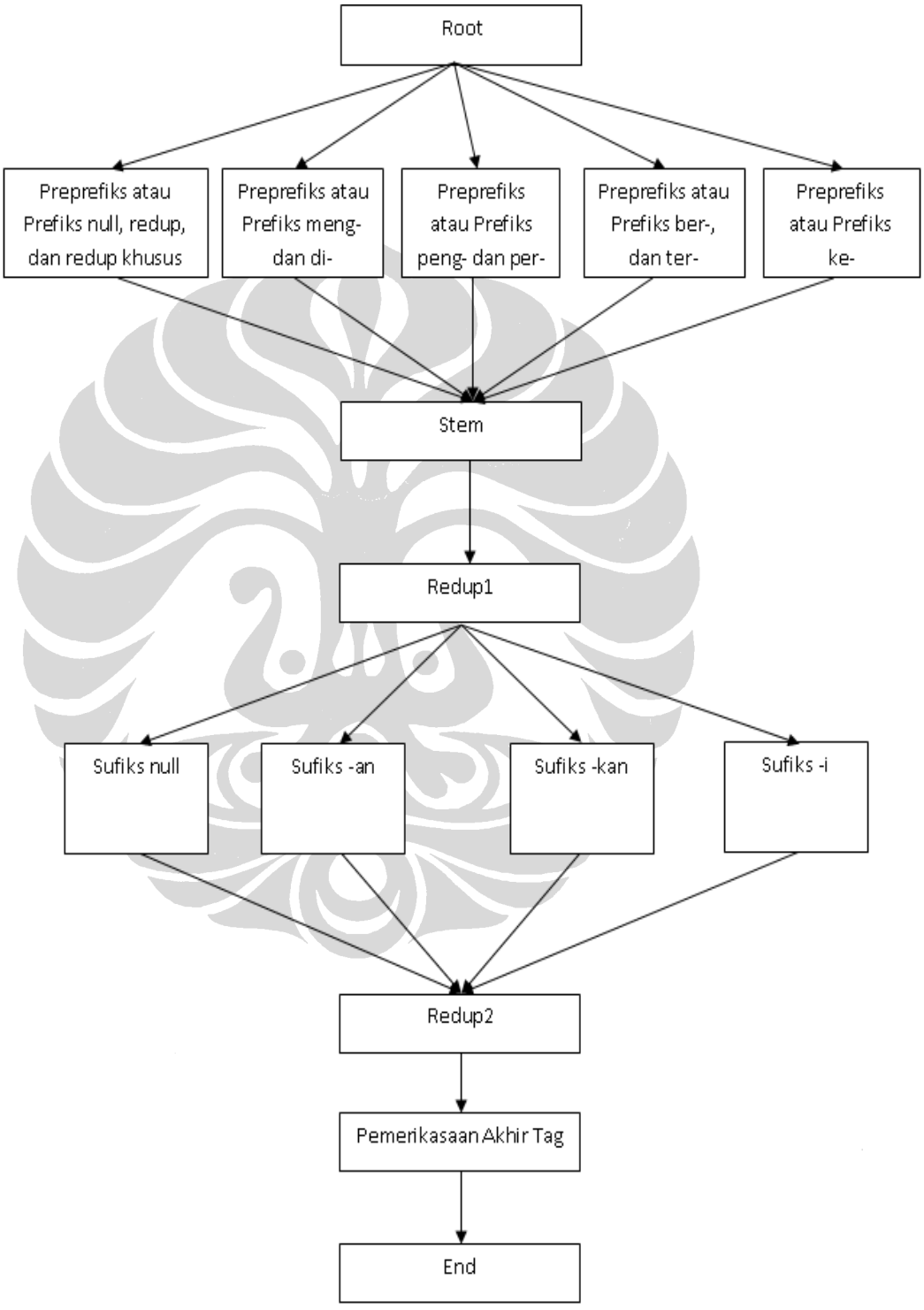
Class SufiksNull akan memberikan *flag diacritics* yang akan digunakan sebagai *input class* selanjutnya. *Class* SufiksRedup2 dibatasi dengan pembatasan *flag diacritics*-nya. Terminologi "Redup" yang digunakan sebagai nama untuk dua *classes* pada implementasi ini memiliki arti reduplikasi. Dalam *code* di atas, pembatasan dilakukan dengan tidak mengizinkan (*disallow flag*) regex yang memiliki *flag* PREPREF dan membutuhkan (*required flag*) regex yang memiliki *flag* PREF.meng. Pemberian *tags* juga dilakukan pada *class* ini. Pada bagian yang diberi tanda "…" sebenarnya berisi kemungkinan kasus-kasus lainnya yang dapat diterima oleh *transducers* yang dihasilkan.

Perilaku seperti pada penjabaran di atas berlaku untuk kasus umum. Ada dua macam kasus lainnya dimana tidak sepenuhnya berjalan seperti diatas.

Kasus pertama masih cukup sering ditemui, yaitu bila ada preprefiks yang mendahului suatu prefiks. Untuk kasus ini perbedaan dari skenario sebelumnya hanya terletak pada tahapan pertama. Pada tahapan ini, suatu prefiks bisa bergerak ke prefiks lainnya yang sah sebelum bergerak ke kata dasar pembentuknya. Padanan preprefiks, prefiks, kata dasar, dan sufiks yang sah bisa dilihat pada BAB 3. Setelah itu berjalan sesuai skenario biasa.

Kasus kedua adalah masalah *irregularity*. Ada beberapa kata yang memiliki struktur pembentuk yang tidak mengikuti kebanyakan kata lainnya. Untuk kata-kata seperti ini akan ditangani terpisah dengan perlakuan khusus.

Dengan mengacu kembali rangkuman aturan morfotaktik pada BAB 3, maka implementasi aturan morfotaktik dapat kita bagi menjadi beberapa bagian yang akan berjalan sequensial. Gambaran umum dari aturan morfotaktik dapat kita lihat pada gambar 4-7.



Gambar 4-7. Gambaran Umum Aturan Morfotaktik

Dari gambaran di atas dapat kita lihat alur yang berjalan tiap bagiannnya. Bagian Root akan dijelaskan pada subsubbab 4.2.1, bagian Prefiks dan Preprefiks akan dijelaskan pada subsubbab 4.2.2, bagian *Stems* dan Redup1 akan dijelaskan pada subsubbab 4.2.3, bagian Sufiks dan Redup2 akan dijelaskan pada subsubbab 4.2.4, bagian pemeriksaan akhir pada subsubbab 4.2.5. Berikut penjelasan selanjutnya mengenai bagian-bagian tersebut.

4.2.1 Bagian *Root*

Root merupakan bagian yang pertama kali dicari ketika program LEXC dijalankan. Jika ditemukan *Root*, maka program akan dijalankan mulai dari *class Root* tersebut. Implementasi *class Root* cukup sederhana dapat dilihat pada gambar 4-8.

```
! program selalu mulai dibaca dari root
! root memiliki continuation class prefiks
! prefiks akan menuju ke continuation classes yang berisi semua kemungkinan prefiks
LEXICON Root
     Prefiks ;

LEXICON Prefiks
     PrefiksNull ;
     PrefiksMeng ;
     PrefiksDi ;
     PrefiksPe ;
     PrefiksPeng ;
     PrefiksPer ;
     PrefiksBer ;
     PrefiksTer ;
     PrefiksKe ;
     PrefiksRedup ;
     PrefiksRedupKhusus ;
```

Gambar 4-8. Cuplikan Implementasi *Class Root*

Dari *Root* akan menuju ke *continuation class* prefiks yang untuk selanjutnya menuju ke beberapa *continuation class* prefiks yang akan dibahas di subsubbab setelah ini. Bagian regex dibiarkan kosong karena kita tidak melakukan manipulasi regex pada bagian ini. Hal yang terjadi pada bagian ini hanyalah mengarahkan alur program ke *continuation class* selanjutnya.

Kesebelas kemungkinan *continuation class* prefiks yang dituju menunjukkan semua kemungkinan prefiks dan preprefiks yang akan diterima.

4.2.2 Bagian Prefiks dan Preprefiks

Bagian ini dicapai setelah melewati *continuation class* Prefiks. Dalam kelas ini berisi semua kemungkinan prefiks dan preprefiks. Berikut implementasi beberapa prefiks dan preprefiks dapat dilihat pada gambar 4-9 dan 4-10.

```
LEXICON PrefiksMeng
<[0 .x. m e "^N"] "@P.PREF.meng@">     Stems ;
<[0 .x. m e "^N"] "@P.PREPREF.meng@">     PrefiksPer ;
<[0 .x. m e "^N"] "@P.PREPREF.meng@">     PrefiksBer ;
<[0 .x. m e "^N"] "@P.PREPREF.meng@">     PrefiksTer ;
<[0 .x. m e "^N"] "@P.PREPREF.meng@">     PrefiksKe ;
<[0 .x. m e "^N"] "@P.PREPREF.meng@">     PrefiksRedup ;

LEXICON PrefiksDi
<[0 .x. d i] "@P.PREF.di@">      Stems ;
<[0 .x. d i] "@P.PREPREF.di@">      PrefiksPer ;
<[0 .x. d i] "@P.PREPREF.di@">      PrefiksBer ;
<[0 .x. d i] "@P.PREPREF.di@">      PrefiksTer ;
<[0 .x. d i] "@P.PREPREF.di@">      PrefiksKe ;
<[0 .x. d i] "@P.PREPREF.di@">      PrefiksRedup ;

LEXICON PrefiksPe
<[0 .x. p e] "@P.PREF.pe@">      Stems ;

LEXICON PrefiksPeng
<[0 .x. p e "^N"] "@P.PREF.peng@">      Stems ;
<[0 .x. p e "^N"] "@D.PREPREF@" "@P.PREPREF.peng@">      PrefiksBer ;

LEXICON PrefiksPer
<[0 .x. p e r "+"] "@P.PREF.per@">     Stems ;
```

Gambar 4-9. Cuplikan Implementasi *Class* Prefiks dan Preprefiks (1)

```
LEXICON PrefiksBer
<[0 .x. b e r "+"] "@P.PREF.ber@">     Stems ;
<[0 .x. b e r "+"] "@D.PREPREF@" "@P.PREPREF.ber@">     PrefiksKe ;
<[0 .x. b e r "+"] "@D.PREPREF@" "@P.PREPREF.ber@">     PrefiksPeng ;
<[0 .x. b e r "+"] "@D.PREPREF@" "@P.PREPREF.ber@">     PrefiksRedup ;

LEXICON PrefiksTer
<[0 .x. t e r "+"] "@P.PREF.ter@">     Stems ;
<[0 .x. t e r "+"] "@D.PREPREF@" "@P.PREPREF.ter@">     PrefiksRedup ;

LEXICON PrefiksKe
<[0 .x. k e] "@P.PREF.ke@">       Stems ;
<[0 .x. k e] "@D.PREPREF@" "@P.PREPREF.ke@">       PrefiksBer ;
<[0 .x. k e] "@D.PREPREF@" "@P.PREPREF.ke@">       PrefiksPeng ;
<[0 .x. k e] "@D.PREPREF@" "@P.PREPREF.ke@">       PrefiksRedup ;

LEXICON PrefiksRedup
<[0 .x. "^[" "[" "{" ] "@P.PREF.Redup@">     Stems ;
<[0 .x. "^[" "[" "{" ] "@D.PREPREF@" "@P.PREPREF.Redup@">     PrefiksPeng ;
<[0 .x. "^[" "[" "{" ] "@D.PREPREF@" "@P.PREPREF.Redup@">     PrefiksPer ;
<[0 .x. "^[" "[" "{" ] "@D.PREPREF@" "@P.PREPREF.Redup@">     PrefiksKe ;
<[0 .x. "^[" "[" "{" ] "@D.PREPREF@" "@P.PREPREF.Redup@">     PrefiksNull ;

LEXICON PrefiksRedupKhusus
<[0 .x. "^[" "[" "{" ] "@P.PREF.RedupKhusus@">     Stems ;
```

Gambar 4-10. Cuplikan Implementasi *Class* Prefiks dan Preprefiks (2)

Cuplikan di atas memperlihatkan adanya interaksi antar *class* dimana *class* yang satu dapat menjadi *continuation class* yang lainnya. Hal tersebut dimaksudkan untuk membuat fungsi preprefiks.

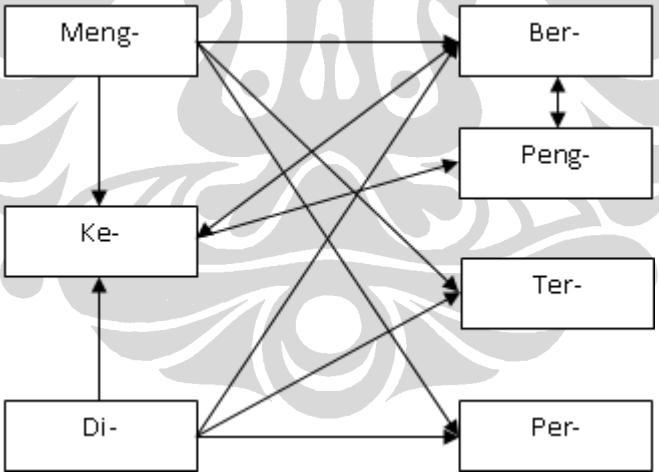
Bagian di antara "<" dan ">" menunjukkan regex yang ditambahkan pada *class* tersebut. Pada regex dapat dilakukan perubahan dari pita atas ke pita bawah maupun fungsi *flag diacritics* yang dibutuhkan. Pada bagian yang dibatasi "[" dan "]" terdapat penambahan prefiks yang diinginkan. Pada pita atas tidak akan ditambahkan apa-apa, sedangkan pada pita bawah dimasukkan regex yang menerima *strings* prefiks yang berkesesuaian. Untuk karakter "^[[{" merupakan pembatas kurung buka untuk kasus kata ulang yang akan dimanfaatkan selanjutnya pada proses *compile-replace*.

Flag diacritics yang diberikan pada regex ini nantinya akan menjadi input untuk pemeriksaan *flag diacritics* pada tahapan selanjutnya. *Flag diacritic* "@P.PREF.xxx@" akan men-*set* positif untuk *feature* PREF dengan *value* xxx, begitu juga "@P.PREPREF.xxx@" yang akan men-*set* positif untuk *feature*

PREPREF dengan *value* xxx. *Flags* ini akan menjadi *input* untuk *continuation class* selanjutnya untuk membatasi *input* yang diterima. Pembatasan tersebut dapat dilakukan baik dengan *required test* maupun dengan *disallow test*.

Pada bagian ini juga terdapat *disallow test* yang dijalankan untuk membatasi pada bagian tersebut ini berfungsi sebagai prefiks atau preprefiks saja. Dengan adanya "@D.PREPREF@" maka membatasi jalur tersebut tidak bisa dimasuki oleh *input* yang sebelumnya sudah mendapatkan *flag* positif untuk *feature* PREPREF. Pembatasan tersebut mengakibatkan jalur tersebut hanya dapat dimasuki oleh *input* yang memanfaatkan prefiks tersebut sebagai preprefiks bagi prefiks lainnya. Untuk kasus suatu prefiks menjadi preprefiks bagi prefiks lainnya, maka jalur selanjutnya adalah menuju *continuation class* prefiks lainnya barulah kemudian menuju *continuation class* Stems. Sedangkan untuk kasus lain dimana prefiks tersebut berfungsi sebagai prefiks, maka ia akan langsung menuju ke *continuation class* Stems.

Secara umum, interaksi prefiks dan preprefiks dapat dilihat pada gambar 4-11.



Gambar 4-11. Ilustrasi Interaksi Prefiks dan Preprefiks

Prefiks meng- dapat berfungsi sebagai preprefiks untuk prefiks ber-, peng-, ter-, per-, dan ke-. Akan tetapi tidak ada prefiks yang akan berfungsi sebagai preprefiks bagi prefiks meng-. Selanjutnya untuk prefiks lainnya dapat dilihat pada Gambar 4-11 di atas.

Khusus kelas PrefiksRedup, akan dianggap sebagai preprefiks untuk kata ulang sejati dan akan dianggap sebagai prefiks untuk kata ulang sebagian dan kata ulang berimbuhan. Hal ini disebabkan hasil analisis bahwa pada kata ulang sejati perulangan juga terjadi pada imbuhan yang diberikan, sehingga *delimiter* "^[" perlu diberikan sebelum prefiksnya. Sedangkan untuk kata ulang sebagian dan kata ulang berimbuhan perulangan tidak terjadi pada imbuhan, sehingga *delimiter* "^[" diberikan sesudah prefiksnya.

4.2.3 Bagian *Stems*

Setelah dari bagian prefiks dan preprefiks selanjutnya masuk ke bagian *Stems* dan Redup1. Implementasi beberapa bagiannya dapat dilihat pada gambar 4-12.

```
! Stems mengambil regex dummy lexicon dan menandakannya dengan flag diacritics yang berkesesuaian
! Sufiks menuju ke continuation classes yang berisi semua kemungkinan sufiks
LEXICON Stems
<Vrb "@P.STEM.Vrb@">   SufiksRedup1 ;
<Nom "@P.STEM.Nom@">   SufiksRedup1 ;
<Adj "@P.STEM.Adj@">   SufiksRedup1 ;
<Dll "@P.STEM.Dll@">   SufiksRedup1 ;

LEXICON SufiksRedup1
<[0 .x. "}" "%^REHYPH" "]" "^" 2 "^]"] "@P.REDUP.Ditutup@" "@R.PREF.Redup@">        Sufiks ;
                                                                                  Sufiks ;

LEXICON Sufiks
    SufiksNull ;
    SufiksAn ;
    SufiksKan ;
    SufiksI ;
    ! SufiksRedupKhusus ;
```

Gambar 4-12. Cuplikan Implementasi Bagian *Stems*

Class Stems akan mengambil *stem* dan memberi tanda ke setiap *stem* dari kelas kata yang berkesesuaian. Memberi tanda tersebut dapat dilihat dari dimasukkannya *flag diacritic* dengan *feature* STEM dan *value* sesuai kelas katanya pada regex di dalam *continuation class* Stems. *Flag diacritics* tersebut di-

set positif untuk menjadi *input* bagi *continuation class* selanjutnya. Dengan cara seperti itu, *continuation class* yang memerlukan pemeriksaan kelas kata dapat melakukan *required test* ataupun *disallow test*. Kemudian dari Stems akan menuju ke *continuation class* pemeriksaan SufiksRedup1. SufiksRedup1 ini akan melakukan *required test* untuk memeriksa manakah yang termasuk kata ulang sebagian dan kata ulang berimbuhan. Jika termasuk kata ulang sebagian atau kata ulang berimbuhan, maka akan ditambahkan *delimiter* kurung penutup ("}^REHYPH]^2^]") pada pita bawahnya untuk kemudian dimanfaatkan pada proses *compile-replace*. Untuk kata-kata yang termasuk kata ulang sebagian atau kata ulang berimbuhan, akan di-*set* sebuah *flag diacritic* lagi yang menunjukkan bahwa untuk kata ulang ini sudah dilakukan penutupan *delimiter* kurung terlebih dahulu.

Setelah pemeriksaan SufiksRedup1, program yang dijalankan akan menuju ke *continuation class* semua kemungkinan variasi sufiks. Terdapat empat kemungkinan variasi sufiks yaitu SufiksNull, SufiksAn, SufiksKan, dan SufiksI yang implemetasinya akan dijelaskan pada subsubbab setelah ini.

4.2.4 Bagian Sufiks

Selanjutnya tahapan sufiks. Pada tahapan ini pemeriksaan dilakukan dan dicegah semua kemungkinan yang tidak sah. Berikut sedikit cuplikannya pada gambar 4-13 sampai 4-15.

```
! dari sini semua prefiks ditambahkan
! juga pemberian tag dan pemeriksaan
LEXICON SufiksNull
! kata dasar
<"+BareVerb":0 "+UV":0 "@D.PREPREF@" "@D.REDUP@" "@R.STEM.Vrb@" "@R.PREF.null@" "@P.SUFF.null@">      Redup2 ;
<"+BareNoun":0 "@D.PREPREF@" "@D.REDUP@" "@R.STEM.Nom@" "@R.PREF.null@" "@P.SUFF.null@">             Redup2 ;
<"+BareAdjective":0 "@D.PREPREF@" "@D.REDUP@" "@R.STEM.Adj@" "@R.PREF.null@" "@P.SUFF.null@">        Redup2 ;
<"+BareEtc":0 "@D.PREPREF@" "@D.REDUP@" "@R.STEM.Dll@" "@R.PREF.null@" "@P.SUFF.null@">              Redup2 ;
! kata ulang sejati tanpa imbuhan
<"+BareVerb":0 "+Redup":0 "@D.PREPREF@" "@D.REDUP@" "@R.STEM.Vrb@" "@R.PREF.Redup@" "@P.SUFF.null@">          Redup2 ;
<"+BareNoun":0 "+Redup":0 "@D.PREPREF@" "@D.REDUP@" "@R.STEM.Nom@" "@R.PREF.Redup@" "@P.SUFF.null@">         Redup2 ;
! imbuhan meng-
<"+Verb":0 "+AV":0 "@D.PREPREF@" "@D.REDUP@" "@R.PREF.meng@" "@P.SUFF.null@">                          Redup2 ;
! imbuhan meng-per-
<"+Verb":0 "+AV":0 "@D.REDUP@" ["@R.STEM.Nom@"|"@R.STEM.Adj@"] "@R.PREPREF.meng@" "@R.PREF.per@" "@P.SUFF.null@">      Redup2 ;
! kata ulang sebagian imbuhan meng-
<"+Verb":0 "+AV":0 "+Redup":0 "@R.REDUP.Ditutup@" "@R.STEM.Vrb@" "@R.PREPREF.meng@" "@R.PREF.Redup@" "@P.SUFF.null@">      Redup2 ;
```

Gambar 4-13. Cuplikan Implementasi Bagian Sufiks (1)

```
LEXICON SufiksAn
! imbuhan -an
<["+Noun" .x. a n] "@D.PREPREF@" "@D.REDUP@" "@R.PREF.null@" "@P.SUFF.an@">                    Redup2 ;
! kata ulang sejati imbuhan -an
<["+Noun" .x. a n] "+Redup":0 "@D.PREPREF@" "@D.REDUP@" "@R.STEM.Vrb@" "@R.PREF.Redup@" "@P.SUFF.an@">          Redup2 ;
! kata ulang berimbuhan -an
<["+Noun" .x. a n] "+Redup":0 "@D.PREPREF@" "@R.REDUP.Ditutup@" "@R.STEM.Nom@" "@R.PREF.Redup@" "@P.SUFF.an@">          Redup2 ;
<["+Verb" .x. a n] "+Redup":0 "@D.PREPREF@" "@R.REDUP.Ditutup@" "@R.STEM.Vrb@" "@R.PREF.Redup@" "@P.SUFF.an@">          Redup2 ;
! imbuhan peng-an
<["+Noun" .x. a n] "@D.PREPREF@" "@D.REDUP@" "@R.PREF.peng@" "@P.SUFF.an@">                    Redup2 ;
! kata ulang sejati imbuhan peng-an
<["+Noun" .x. a n] "+Redup":0 "@D.REDUP@" ["@R.STEM.Vrb@"|"@R.STEM.Nom@"|"@R.STEM.Adj@"] "@R.PREPREF.Redup@" "@R.PREF.peng@" "@P.SUFF.an@">          Redup2
! imbuhan peng-ber-an
<["+Noun" .x. a n] "@D.REDUP@" "@R.PREPREF.peng@" "@R.PREF.ber@" "@P.SUFF.an@">                    Redup2 ;
! kata ulang sejati imbuhan per-an
<["+Noun" .x. a n] "+Redup":0 "@D.REDUP@" ["@R.STEM.Vrb@"|"@R.STEM.Nom@"] "@R.PREPREF.Redup@" "@R.PREF.per@" "@P.SUFF.an@">                    Redup2 ;
! imbuhan per-an
<["+Noun" .x. a n] "@D.PREPREF@" "@D.REDUP@" ["@R.STEM.Vrb@"|"@R.STEM.Nom@"|"@R.STEM.Dll@"] "@R.PREF.per@" "@P.SUFF.an@">          Redup2 ;
! imbuhan ber-an
<["+Verb" .x. a n] "@D.PREPREF@" "@D.REDUP@" "@R.STEM.Vrb@" "@R.PREF.ber@"  "@P.SUFF.an@">          Redup2 ;

LEXICON SufiksKan
! imbuhan -kan
<["+Verb" .x. k a n] "@D.PREPREF@" "@D.REDUP@" "@R.PREF.null@" "@P.SUFF.kan@">                    Redup2 ;
! imbuhan meng-kan
<["+Verb" .x. k a n] "+AV":0 "@D.PREPREF@" "@D.REDUP@" "@R.PREF.meng@" "@P.SUFF.kan@">                    Redup2 ;
! imbuhan meng-per-kan
<["+Verb" .x. k a n] "+AV":0 "@D.REDUP@" ["@R.STEM.Vrb@"|"@R.STEM.Nom@"|"@R.STEM.Dll@"] "@R.PREPREF.meng@" "@R.PREF.per@" "@P.SUFF.kan@">          Redup2 ;
! imbuhan meng-ber-kan
<["+Verb" .x. k a n] "+AV":0 "@D.REDUP@" ["@R.STEM.Vrb@"|"@R.STEM.Nom@"|"@R.STEM.Adj@"] "@R.PREPREF.meng@" "@R.PREF.ber@" "@P.SUFF.kan@">          Redup2 ;
! imbuhan meng-ter-kan
<["+Verb" .x. k a n] "+AV":0 "@D.REDUP@" ["@R.STEM.Vrb@"|"@R.STEM.Nom@"] "@R.PREPREF.meng@" "@R.PREF.ter@" "@P.SUFF.kan@">          Redup2 ;
! imbuhan meng-ke-kan
<["+Verb" .x. k a n] "+AV":0 "@D.REDUP@" "@R.STEM.Dll@" "@R.PREPREF.meng@" "@R.PREF.ke@" "@P.SUFF.kan@">                    Redup2 ;
! kata ulang berimbuhan meng-kan
<["+Verb" .x. k a n] "+AV":0 "+Redup":0 "@R.REDUP.Ditutup@" "@R.STEM.Vrb@" "@R.PREPREF.meng@" "@R.PREF.Redup@" "@P.SUFF.kan@">          Redup2 ;
! imbuhan di-kan
<["+Verb" .x. k a n] "+PASS":0 "@D.PREPREF@" "@D.REDUP@" "@R.PREF.di@" "@P.SUFF.kan@">                    Redup2 ;

LEXICON SufiksI
! imbuhan -i
<"+Verb":i "@D.PREPREF@" "@D.REDUP@" ["@R.STEM.Vrb@"|"@R.STEM.Nom@"|"@R.STEM.Adj@"] "@R.PREF.null@" "@P.SUFF.i@">          Redup2 ;
! imbuhan meng-i
<"+Verb":i "+AV":0 "@D.PREPREF@" "@D.REDUP@" ["@R.STEM.Vrb@"|"@R.STEM.Nom@"|"@R.STEM.Adj@"] "@R.PREF.meng@" "@P.SUFF.i@">          Redup2 ;
! imbuhan meng-per-i
<"+Verb":i "+AV":0 "@D.REDUP@" "@R.STEM.Nom@" "@R.PREPREF.meng@" "@R.PREF.per@" "@P.SUFF.i@">          Redup2 ;
! imbuhan di-i
<"+Verb":i "+PASS":0 "@D.PREPREF@" "@D.REDUP@" ["@R.STEM.Vrb@"|"@R.STEM.Nom@"|"@R.STEM.Adj@"] "@R.PREF.di@" "@P.SUFF.i@">          Redup2 ;
! imbuhan di-per-i
<"+Verb":i "+PASS":0 "@D.REDUP@" "@R.STEM.Nom@" "@R.PREPREF.di@" "@R.PREF.per@" "@P.SUFF.i@">          Redup2 ;
! imbuhan per-i
<"+Verb":i "@D.PREPREF@" "@D.REDUP@" "@R.STEM.Nom@" "@R.PREF.per@" "@P.SUFF.i@">                    Redup2 ;
```

Gambar 4-14. Cuplikan Implementasi Bagian Sufiks (2)

```
! tutup kurung untuk kata ulang sejati
LEXICON Redup2
! untuk kata dasar dan imbuhan
<"@D.PREPREF.Redup@" "@D.PREF.Redup@" "@D.PREF.RedupKhusus@">          CausApplKan ;
! untuk kata ulang sebagian dan kata ulang berimbuhan
<"@R.REDUP.Ditutup@">                          CausApplKan ;
! untuk kata ulang sejati dan kata ulang kasus khusus
<[0 .x. "}" "%^REHYPH" "]" "^" 2 "^]"] "@D.REDUP@" ["@R.PREPREF.Redup@"|"@R.PREF.Redup@"]>          CausApplKan ;
<[0 .x. "}" "%^REHYPH m e %^N" "]" "^" 2 "^]"] "@D.REDUP@" "@R.PREF.RedupKhusus@" "@R.SUFF.null@">    CausApplKan ;
```

Gambar 4-15. Cuplikan Implementasi Bagian Sufiks (3)

Bagian sufiks bisa dibilang merupakan bagian kunci dari pengurai morfologi ini. Karena pada bagian inilah dicegah segala kemungkinan yang tidak *valid*, diloloskan semua kemungkinan yang *valid*, dan diberikannya *tags* beserta akhiran yang sesuai untuk kemungkinan *valid* yang diloloskan tersebut.

Dengan melihat cuplikan-cuplikan *code* di atas, bagian yang dibatasi "<" dan ">" berisi regex yang akan menambahkan *tags* pada pita atas (*upper tape*) dan akhiran pada pita bawah (*lower tape*). Sebagai contoh, cuplikan regex [+Verb .x. k a n] akan memberikan hasil +Verb pada pita atas dan akhiran "k a n" pada pita bawahnya. Begitu pula untuk cuplikan regex lainnya.

Kalau diperhatikan lagi *codes* tersebut, maka akan jelas terlihat sederetan *required test* dan *disallow test* yang menyertai tiap baris regex. Kedua test ini berfungsi sebagai penyaring yang akan memeriksa kombinasi preprefiks, prefiks, stems, suffiks awal, dan sufiks akhir yang sah sesuai apa yang ditunjukkan pada subbab 3.4. *Disallow test* ("@D.yyy@") akan memcegah semua *input* yang telah di-*set* positif pada *feature* yyy dengan *value* apapun untuk melewati jalur tersebut, sedangkan *Required test* ("@R.yyy.xxx@") hanya akan meloloskan semua *input* yang sudah di-*set* positif pada *feature* yyy dengan *value* xxx. Dengan begitu dapat diimplementasikan semua variasi kata dengan spesifikasi masing-masing seperti yang ditunjukkan pada tabel 3-15 rancangan morfotaktik.

Selanjutnya tiap bagian sufiks tersebut akan memberikan tanda sesuai sufiks yang diberikan. Tanda tersebut diberikan berupa *flag diacritics* "@P.SUFF.xxx@" yang akan men-*set* positif *feature* SUFF dengan *value* yang

berkesesuaian. Setelah diberikan *flag diacritic* tersebut, maka program akan berlanjut ke *continuation class* Redup2.

Class Redup2 ini adalah *class* yang didefinisikan untuk memberikan kurung penutup ("}^REHYPH]^2^]") bagi kata ulang sejati. Untuk kata-kata yang tidak tergolong dalam kata ulang sejati akan langsung diteruskan ke *continuation class* CausApplKan. Akan tetapi, untuk kata ulang sejati akan diberikan penutup kurung pada pita bawahnya yang nantinya akan dipergunakan dalam proses *compile-replace*. *Required test* dan *disallow test* yang diberikan merupakan spesifikasi yang menunjukkan apakah ia termasuk kata ulang sejati atau tidak. Setelah pemberian tutup kurung, maka ia akan dilanjutkan ke *continuation class* CausApplKan.

4.2.5 Bagian Pemeriksaan Akhir

Bagian pemeriksaan akhir ini hanya dicapai dari *class* yang berhubungan dengan akhiran –kan, -i, dan awaaln peng-. Berikut cuplikan implementasinya pada gambar 4-16.

```
 ! tag-tag khusus yang dapat dikeluarkan ataupun tidak diberikan di tiga kelas ini
 ! tag-tag tersebut dikeluarkan bila lexicon yang digunakan memiliki tanda untuk itu
LEXICON CausApplKan
<"@D.kan@">     CausApplI ;
<"+Caus_kan":0 "@R.kan.Caus@">  CausApplI ;
<"+Appl_kan":0 "@R.kan.Appl@">   CausApplI ;

LEXICON CausApplI
<"@D.i@">       ActorInstrument ;
<"+Caus_i":0 "@R.i.Caus@">  ActorInstrument ;
<"+Appl_i":0 "@R.i.Appl@">   ActorInstrument ;

LEXICON ActorInstrument
<"@D.peN@">          # ;
<"+Actor":0 "@R.peN.Actor@">     # ;
<"+Instrument":0 "@R.peN.Instr@">       # ;
```

Gambar 4-16. Cuplikan Implementasi Bagian Pemeriksaan Akhir

Tiga *class* di atas hanya akan memeriksa *stems* yang memiliki tanda khusus yang memungkinkan *stems* tersebut mendapatkan *tags* yang akan diberikan pada *class* ini. Jika tidak memiliki tanda tersebut maka *tags* tidak diberikan.

Pertama akan dilihat *class* CausApplKan. *Class* ini akan memberikan *tag* "+Caus_kan" jika kata yang dimaksud sudah di-*set* positif untuk *feature* kan dengan *value* Caus, dan akan memberikan *tag* "+Appl_kan" jika kata yang dimaksud sudah di-*set* positif untuk *feature* kan dengan *value* Appl. Hal ini dapat dilakukan berkat adanya *required test* pada regexnya. *Required test* tersebut dapat ditunjukkan pada bagian "@R.kan.xxxx@". Untuk *disallow test* "@D.kan@" akan meloloskan kata-kata yang tidak di-*set* postitif untuk *feature* kan.

Setelah melewati *class* CausApplkan, maka akan masuk ke *continuation class* CuasApplI. Pada dasarnya apa yang dilakukan pada *class* CausApplI hampir sama dengan yang dilakukan pada CausApplKan. Perbedaannya hanya terletak pada *feature* yang diperiksa. Jika pada CausApplKan dilakukan pemeriksaan terhadap *feature* kan, maka pada CausApplI pemeriksaan dilakukan terhadap *feature* i.

Pemeriksaan terakhir terletak pada *class* ActorInstrument. Pemeriksaan yang dilakukan pada *class* ini juga hampir sama dengan dua *class* sebelumnya. Perbedaannya hanyalah terletak pada *feature* dan *value* dari *flag diacritic* yang diperiksa. Pada bagian ActorInstrument, pemeriksaan dilakukan apakah sudah di-*set flag* dengan *feature* peN atau tidak, dan jika sudah apakah *value*-nya Actor atau Instr.

Ketiga *flag diacritics* yang diperiksa pada bagian ini diberikan pada *level lexicon*. Untuk mendapatkan hasil dengan *tags* seperti yang diberikan pada bagian ini haruslah memiliki *lexicon* yang sudah ditandai sebelumnya.

4.2.6 Morfotaktik Untuk Kasus Kata Ulang

Morfotaktik untuk kata ulang sebenarnya sudah secara implisit dipaparkan pada subbab-subbab sebelumnya. Akan tetapi untuk penjelasannya akan diberikan pada subbab ini.

Screenshot code pada subbab 4.2.1 menunjukkan bahwa dari *class* Prefiks dapat menuju kepada *continuation class* PrefiksRedup dan PrefiksRedupKhusus. Inti dari *class* PrefiksRedup dan PrefiksRedupKhusu adalah pemberian karakter kurung untuk kegunaan *compile-replace*. Hal yang membedakan dua *class* tersebut yaitu pada *setting flag diacritics*. *Class* PrefiksRedup akan men-*set*

positive flag diacritics PREF dengan *value* redup yang ditujukan untuk semua jenis kata ulang kecuali kata ulang berimbuhan "meng-", sedangkan *class* PrefiksRedupKhusus akan men-*set positive flag diacritics* PREF dengan *value* redupKhusus yang ditujukan khusus untuk kata ulang berimbuhan "meng-". *Flag diacritic* PREF akan di-*set* untuk kata-kata yang tergolong kata ulang sebagian dan kata ulang berimbuhan, sedangkan PREPREF akan di-*set* pada kata ulang sejati. Hal ini dilakukan karena untuk kata ulang sejati maka imbuhan yang terdapat didalamnya haruslah diulang, sedangkan untuk kata ulang sebagian dan berimbuhan tidak. *Class* PrefiksRedup dan PrefiksRedupKhusus dapat dilihat pada *screenshot code* subbab 4.2.2.

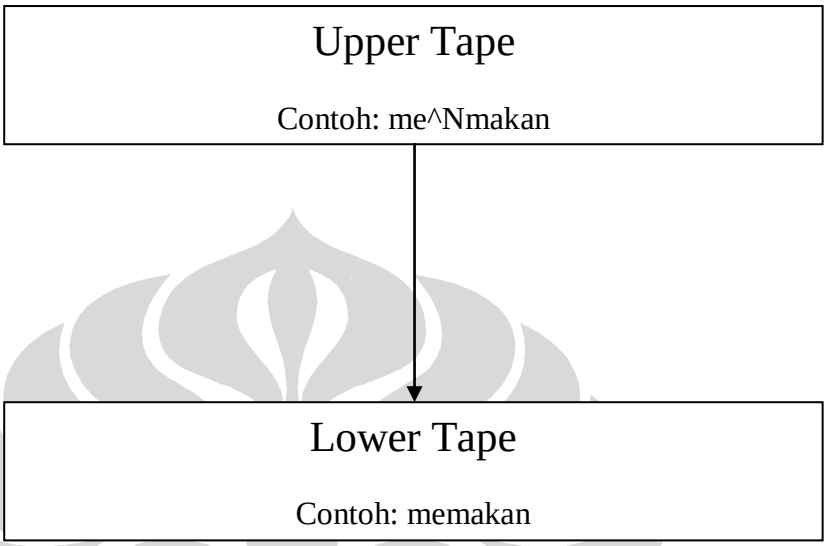
Setelah bersamaan dengan yang lainnya melewati *class* Stems, untuk kata ulang sebagian dan berimbuhan akan mengambil penutup karakter kurung untuk *compile-replace* pada bagian SufiksRedup1. Pada bagian itu juga akan ditambahkan *flag diacritics* yang akan digunakan sebagai *input* untuk *continuation classes* selanjutnya. Dari SufiksRedup1 akan menuju variasi akhiran kemudian menuju Redup2. Pada Redup2 inilah semua pemeriksaan *flag diacritics* dilakukan. Selain pemeriksaan, *class* Redup2 juga digunakan untuk menambahkan karakter penutup kurung ("}^REHYPH]^2^]"). Karakter penutup kurung pada *class* Redup2 ditujukan untuk kata ulang sejati, sedangkan pada kata ulang sebagian dan kata ulang berimbuhan penutupan kurung sudah dilakukan lebih dulu pada SufiksRedup1. Lagi-lagi hal ini disebabkan pengaruh ikut diulangnya suatu imbuhan atau tidak pada kata ulang tersebut. Implementasinya dapat dilihat pada cuplikan *code* pada subbab 4.2.3 dan 4.2.4.

Selanjutnya seperti hlanya kasus lainnya akan melanjutkan ke dalam bagian pemeriksaan akhir.

4.3 Implementasi Aturan Morfofonemik

Aturan-aturan morfofonemik dimplementasikan dalam XFST dimana dimungkinkan untuk bisa mendefinisikan satu per satu aturan yang telah dirancang pada BAB 3. Dari tiap aturan-aturan yang sudah dirancang pada BAB 3 akan dibentuk *transducer networks* yang berisi perubahan fonem yang terjadi baik pada imbuhan maupun kata dasar pembentuknya.

Pembentukan *input* dan *output* setelah melalui proses ini kurang lebih seperti pada aturan morfotaktik dimana *input* dan *output* akan ditunjukkan oleh pita atas (*Upper Tape*) dan pita bawah (*Lower Tape*). Berikut contohnya pada gambar 4-17.



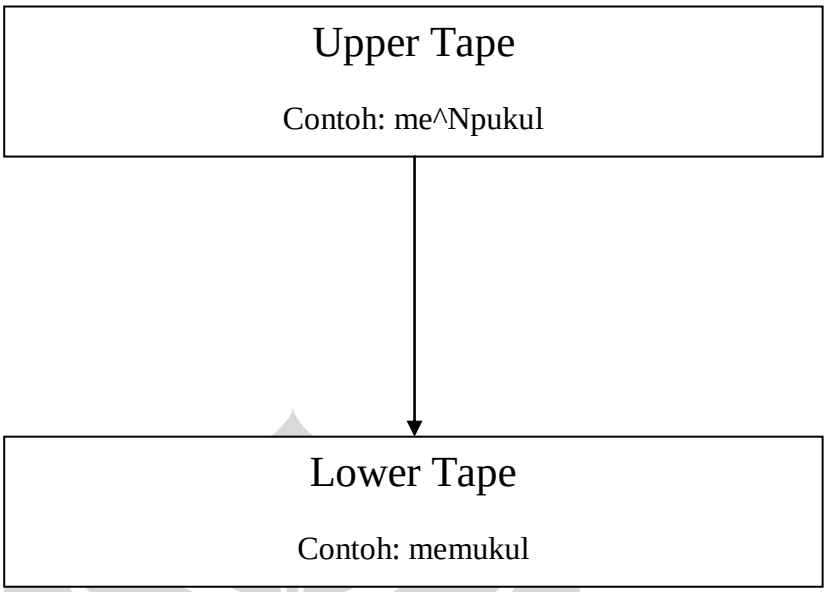
Gambar 4-17. Contoh *Input-Output* Aturan Morfofonemik (1)

Contoh implementasi aturan yang memungkinkan terjadinya hal di atas bisa dilihat sebagai berikut:

```
define R5 %^N->0||[m e|[p e]]_"lbraces"* [l|m|n|r|y|w|t|s|p|k];
! push defined R5;
```

Pada *code* di atas, symbol R5 akan berisi *transducer* yang mengizinkan "^N" *deletion* dengan syarat didahului "me" dan diikuti salah satu antara "l,m,n,r,y,w,t,s,p,k".

Contoh di atas memanfaatkan aturan "ng *deletion*". Akan tetapi ada kasus lain dimana diperlukan lebih dari satu aturan secara bersamaan. Contoh dapat dilihat pada Gambar 4-18.



Gambar 4-18. Contoh *Input-Output* Aturan Morfofonemik (2)

Pada kasus tersebut, dirasa perlu menggabungkan secata bersamaan aturan "ng *deletion*" dengan "p *replacement with* m". Oleh karena itu, ada beberapa aturan yang perlu digabungkan dengan paralelisasi. Implementasi aturan di atas dapat dilihat sebagai berikut pada gambar 4-19.

```
define RG4 %^N->0||[m e|[p e]]_"lbraces"* [l|m|n|r|y|w|t|s|p|k] ,,
          t->n||[m |p] e %^N "lbraces"*_ ,,
          p->m||[m|p] e %^N "lbraces"*_ ,,
          s->n y||[m|p] e %^N "lbraces"*_ ,,
          k->n g||[m|p] e %^N "lbraces"*_;
! push defined RG4;
```

Gambar 4-19. Cuplikan Implementasi Contoh pada Gambar 4-18

Code di atas mengimplementasikan *symbol* RG4 yang akan berisi *transducer* dimana *transducer* tersebut akan mengizinkan *deletion* "^N" dengan syarat didahului "me" atau "pe" untuk kemudian diparalel dengan "p" *replacement with* "m".

Sedangkan untuk aturan yang tidak perlu digabungkan secara parallel bisa digabungkan dengan meng-*compose* saja. Contoh implementasi dapat kita lihat sebagai berikut pada gambar 4-20.

```
! D1-D3 untuk kasus dimana perlu dideklarasikan aturan tambahan untuk prefiks bertingkat
define D1 %^N->m||[m e]_[p e r %+|b e r %+];
! push defined D1;
define D2 %^N->0||[m e]_[t e r %+] ,, t e r %+->n e r||m e %^N_;
! push defined D2;
define D3 %^N->n g||[m e]_[k e] ,, k e->e||m e %^N_;
! push defined D3;
```

Gambar 4-20. Cuplikan Implemenatsi untuk Aturan yang Harus Di-*compose*

Dapat kita lihat bahwa symbol DG menyimpan *network* hasil dari *compose* D1, D2, dan D3.

Contoh di atas merupakan gambaran secara khusus tiga contoh kasus yang terjadi pada aturan morfofonemik. Untuk gambaran secara umum kasus-kasus perubahan yang terjadi bisa dilihat sebagai berikut:

4.3.1 Implementasi Morfofonemik Prefiks meN- dan peN-

Aturan morfofonemik untuk imbuhan dengan prefiks meng- dan peng- banyak didefinisikan dalam satu aturan sebab memiliki beberapa kemiripan. Berikut beberapa implementasinya:

1. "^N" *replacement with* "n"

```
define R1 %^N->n||[m e]_"lbraces"* [d|c|j|[s y]];
! push defined R1;
define R2 %^N->n||[p e]_"lbraces"* [d|c|j];
! push defined R2;
```

Gambar 4-21. Cuplikan Implementasi "^N" *replacement with* "n"

Gambar 4-21 adalah cuplikan aturan yang mengubah "ng" atau dalam implementasi kali ini menggunakan tanda "^N" menjadi "n". Untuk meng- akan berubah menjadi men- jika bertemu "d", "c", "j", dan "sy". Sedangkan untuk peng- akan berubah menjadi pen- saat bertemu "d", "c", dan "j".

2. "^N" *replacement with* "m"

```
define R7 %^N->m||[[m e]|[p e]]_"lbraces"* [b|f];
! push defined R7;
```

Gambar 4-22. Cuplikan Implementasi "^N" *replacement with* "m"

Gambar 4-22 adalah cuplikan aturan yang mengubah “ng” atau dalam implementasi kali ini menggunakan tanda “^N” dengan “m”. Untuk meng- dan peng- akan berubah menjadi mem- dan pem- jika bertemu dengan “b” atau “f”.

3. “^N” *deletion* khusus peng-

```
define R9 r %^N->0||[p e]_"lbraces"* [r|["Cons"+ e r]];
! push defined R9;
```

Gambar 4-23. Cuplikan Implementasi “^N” *deletion* khusus peng-

Gambar 4-23 adalah cuplikan aturan yang menghilangkan “ng” atau dalam kasus ini menggunakan tanda “^N” khusus untuk peng- yang bertemu “r” atau suku kata pertama berakhiran “er”. Peng- akan menjadi pe- jika bertemu salah satu dari itu.

4. “e” *addition* khusus meng-

```
define R10 %^N->n g e||[m e]_"lbraces"* "Cons"* "Vow" "Cons"* .#.;
! push defined R10;
```

Gambar 4-24. Cuplikan Implementasi “e” *addition* khusus meng-

Gambar 4-24 adalah cuplikan aturan yang menambahkan “e”. Aturan tersebut mengakibatkan untuk setiap meng- yang bertemu kata dasar bersuku kata satu akan berubah menjadi “menge-”.

5. “^N” atau “ng” *deletion* yang diparalel dengan empat aturan perubahan fonem

```
define RG4 %^N->0||[m e|[p e]]_"lbraces"* [l|m|n|r|y|w|t|s|p|k] ,,
          t->n||[m |p] e %^N "lbraces"*_ ,,
          p->m||[m|p] e %^N "lbraces"*_ ,,
          s->n y||[m|p] e %^N "lbraces"*_ ,,
          k->n g||[m|p] e %^N "lbraces"*_;
! push defined RG4;
```

Gambar 4-25. Cuplikan Implementasi "^N" *deletion* yang diparalel dengan empat aturan perubahan fonem

Gambar 4-25 adalah cuplikan aturan yang memparalelkan aturan "ng" atau "^N" deletion dengan empat aturan perubahan fonem. Keempat aturan perubahan fonem tersebut antara lain: "t" *replacement with* "n", "p" *replacement with* "m", "s" *replacement with* "ny", dan "k" *replacement with* "ng". Meng- akan berubah menjadi me- untuk kasus-kasus ini.

4.3.2 Implementasi Morfofonemik Prefiks ber-, ter-, dan per-

Pada ber-, ter-, dan per- hanya terjadi satu macam perubahan yaitu "r" *deletion*. Berikut implementasinya pada gambar 4-26.

```
define R3 r %+->0||[b e]_"lbraces"* [r|["Cons"+ e r]];
! push defined R3;
define R4 r %+->0||[t e]_"lbraces"* r;
! push defined R4;
define R6 r %+->0||[p e]_"lbraces"* r;
! push defined R4b;
```

Gambar 4-26. Cuplikan Implementasi "r" *deletion*

Implementasi di atas akan mengakibatkan peluruhan "r" untuk ber- yang bertemu dengan "r" atau kata dasar dengan suku awal berakhiran "er' dan juga ter- dan per- yang bertemu "r". Untuk ketiga kasus tersebut ber-, ter-, dan per- berturut-turut akan menjadi be-, te-, dan pe-.

Sepeti yang telah dijelaskan sebelumnya, semua aturan itu akan digabungkan sebagaimana contoh pada Gambar 4-27.

```
define Morphrules DG .o. R1 .o. R2 .o. R3 .o. R4 .o. R6 .o. R7 .o. R9 .o. R10 .o. RG4 .o. R16 .o. R17;
! push defined Morphrules;
```

Gambar 4-27. Cuplikan Implementasi Penggabungan Semua Aturan Morfofonemik

Pada implementasi di atas, semua aturan yang telah dideskripsikan akhirnya dikomposisi menjadi satu aturan yang disebut Morphrules.

4.3.3 Implementasi Morfofonemik Kata Ulang

Untuk kata ulang pada dasarnya hanya perlu didefinisikan lbrace sebagaimana pada Gambar 4-28.

```
define lbraces ["^["|"["|"{"|"}"|"]"|"^"|2|"^]"];
! push defined lbraces;
```

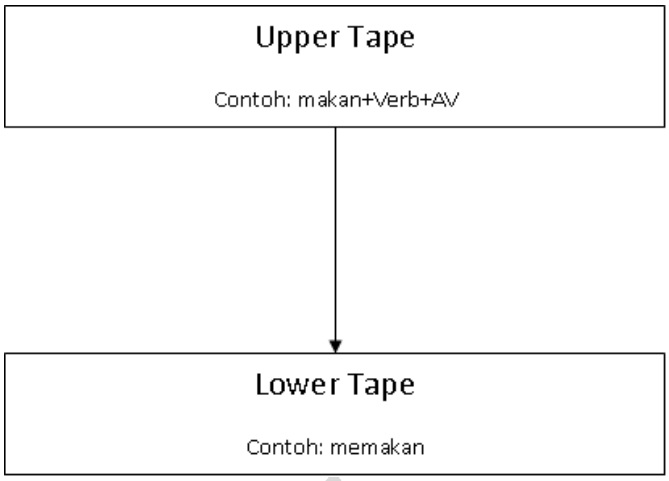
Gambar 4-28. Cuplikan Tambahan untuk Kata Ulang

Lbraces perlu didefisikan agar karakter-karakter penanda compile-replace tersebut dilewatkan. Kemudian setelah dilewatkan, implementasi untuk kata ulang dapat memakai apa yang ada di dalam aturan morfofonemik imbuhan dengan menambahkan “lbraces*” didalamnya.

Hal tersebut dapat dilakukan sebab pada dasarnya perilaku morfofonemik pada kata ulang terjadi pada kata imbuhan di dalam kata ulangnya. Perilaku tersebut mengakibatkan morfofonemik untuk kata ulang sama saja dengan morfofonemik untuk kata imbuhan.

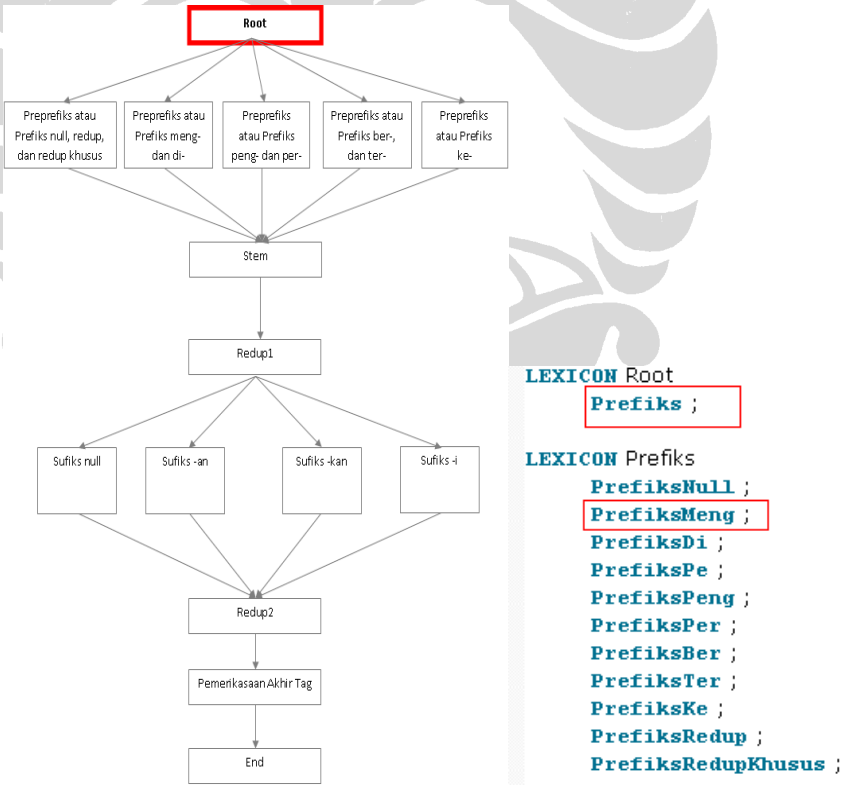
4.4 Contoh Kasus untuk Implementasi

Untuk lebih menjelaskan implementasi di atas, alur perjalanan aturan morfofonemik dan morfotaktik dijelaskan dengan sebuah contoh. Contoh tersebut memberikan ilustrasi proses yang dilalui suatu kasus spesifik. Contoh yang dipilih untuk ditampilkan adalah contoh pada gambar 4.29.



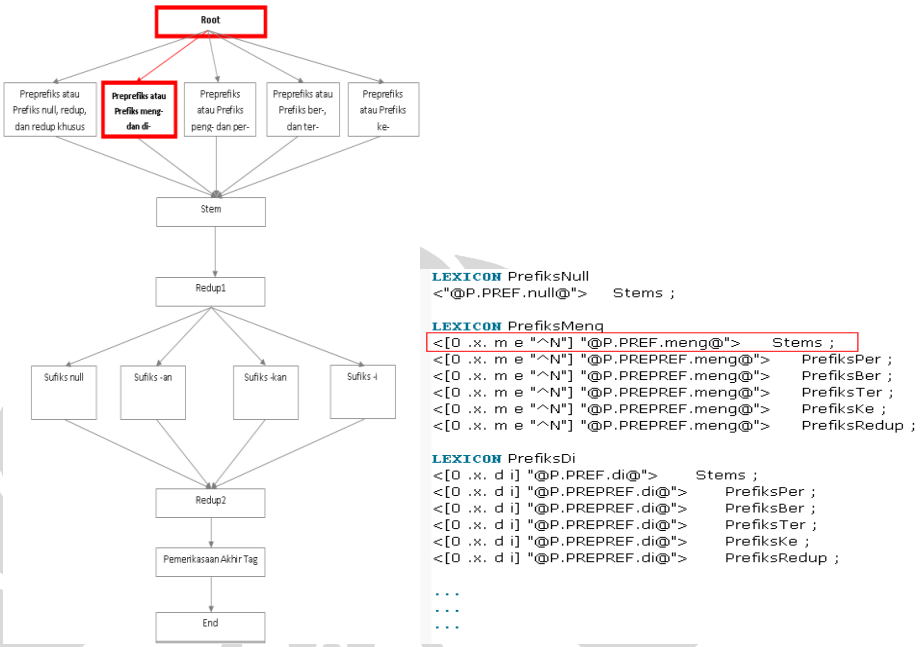
Gambar 4-29. Skema untuk Contoh Kasus Memakan

Contoh pada gambar 4.29 adalah contoh ketika kata "memakan" diuraikan sehingga menghasilkan analisis morfologi "makan+Verb+AV". Kasus tersebut akan bergerak mulai dari bagian *Root* pada aturan morfotaktik. Proses yang terjadi pada aturan morfotaktik bagian *Root* dapat dilihat pada gambar 4.30.



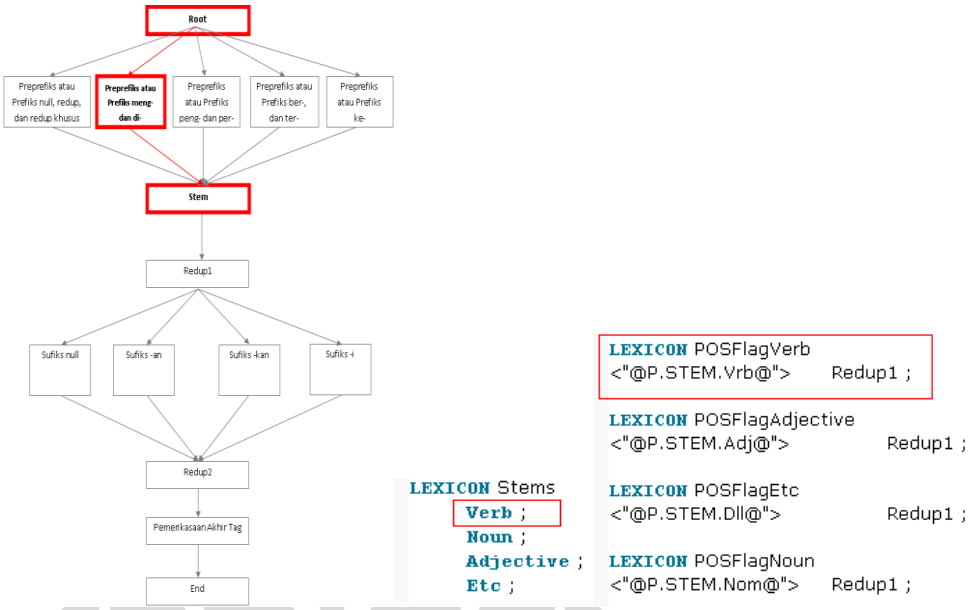
Gambar 4-30. Contoh Kasus Memakan Bagian *Root*

Pada gambar 4.30 dapat dilihat dari *class Root* akan menuju *continuation class* Prefiks. Kemudian sesuai prefiks yang diinginkan, dari *class* Prefiks akan bergerak ke *continuation class* PrefiksMeng. Tidak ada manipulasi regex yang terjadi pada tahapan ini. Setelah tahapan ini, contoh kasus penguraian kata "memakan" akan bergerak ke aturan morfotaktik bagian Preprefiks dan Prefiks.



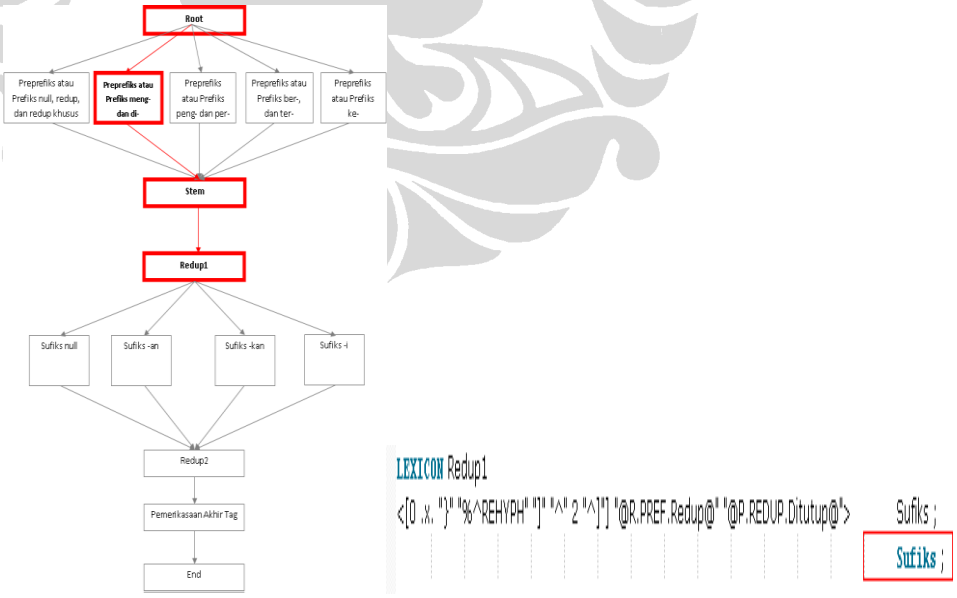
Gambar 4-31. Contoh Kasus Memakan Bagian Preprefiks dan Prefiks

Pada gambar 4.31 dapat dilihat pada class PrefiksMeng dilakukan pemberian prefiks "me^N" pada pita bawah (*lower tape*), sedangkan tidak ada penambahan pada pita atas (*upper tape*). Pada *class* PrefiksMeng juga diberikan *flag diacritic positive setting*. *Feature* "PREF" di-*set* positif untuk *value* "meng". Setelah tahapan ini akan bergerak ke *continuation class Stems* yang terdapat pada aturan morfotaktik bagian *Stems.*



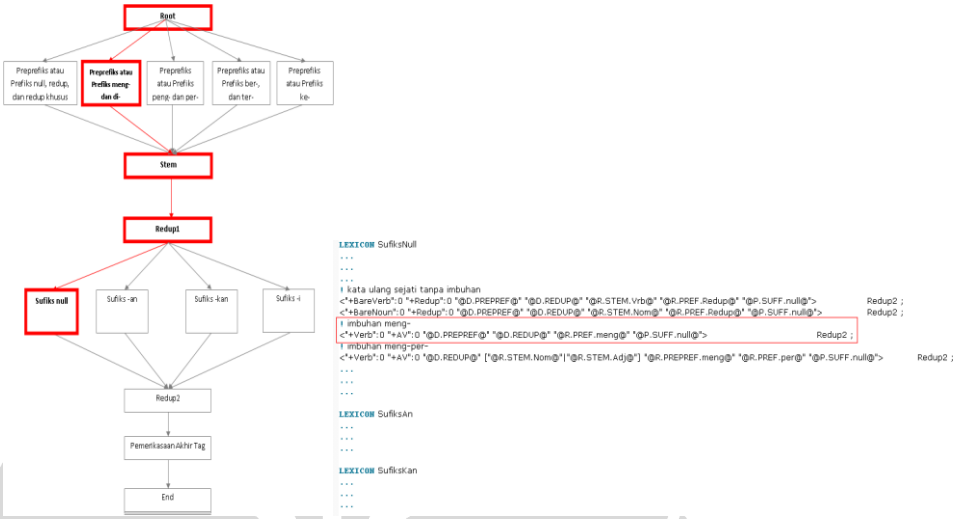
Gambar 4-32. Contoh Kasus Memakan Bagian *Stems*

Pada gambar 4.32 dapat dilihat dari *class Stems* akan menuju *continuation class Verb*. Pada *continuation class Verb* didapatkan kata "makan" yang diberikan kepada pita atas maupun pita bawah. Dari *class Verb* akan menuju *continuation class POSFlagVerb*. Pada class ini diberikan *flag diacritic positive setting*. *Feature* "PREF" di-*set* positif untuk *value* "meng". Setelah proses tersebut, kasus ini akan menuju continuation *class* Redup1.



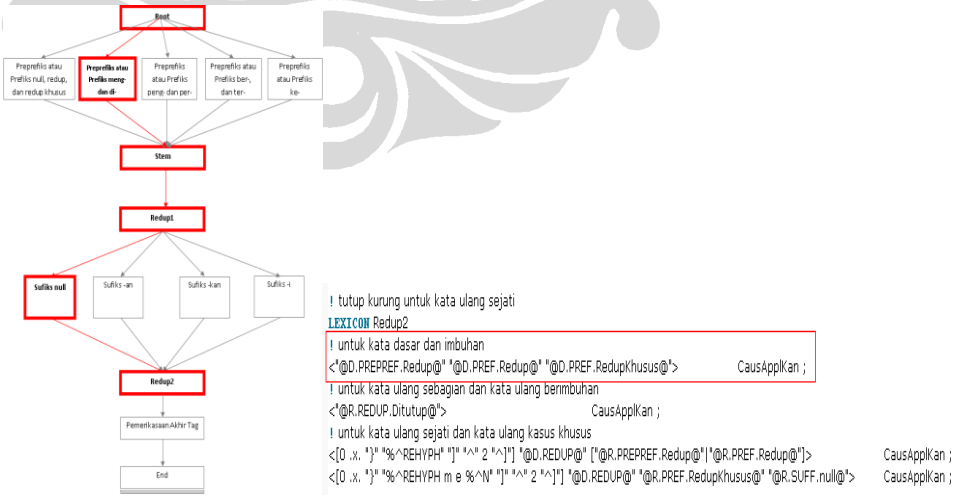
Gambar 4-33. Contoh Kasus Memakan Bagian Redup1

Pada gambar 4.33 dapat dilihat contoh kasus ini mengambil baris kedua pada *class* Redup1 yang akan bergerak ke *continuation class* Sufiks. Pengambilan baris kedua ini disebabkan contoh kasus penguraian "memakan" tidak memenuhi *required test* untuk *feature* PREF dengan *value* Redup.



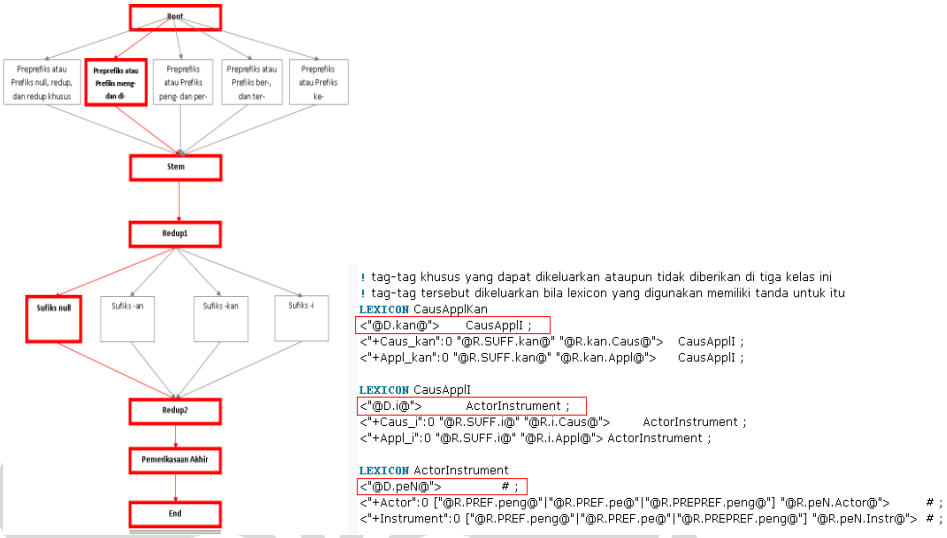
Gambar 4-34. Contoh Kasus Memakan Bagian Sufiks

Pada gambar 4.34 dapat dilihat contoh kasus ini mengambil baris yang sesuai untuk imbuhan "meN". Baris ini ditunjukkan dengan *flag diacritics* yang bisa dilewati oleh contoh kasus penguraian kata "memakan". Pada baris ini ditambahkan *tags* "+Verb" dan "+AV" pada pita atas, sedangkan pada pita bawah tidak ada yang ditambahkan. Selanjutnya contoh kasus akan bergerak ke *continuation class* Redup2.



Gambar 4-35. Contoh Kasus Memakan Bagian Redup2

Pada gambar 4.35 dapat dilihat contoh kasus ini mengambil baris pertama yang sesuai untuk kata dasar dan kata berimbuhan. Selanjutnya contoh kasus penguraian “memakan” akan bergerak ke *continuation class* CausApplKan yang terletak pada aturan morfotaktik bagian pemeriksaan akhir.



Gambar 4-36. Contoh Kasus Memakan Bagian Pemeriksaan Akhir

Pada gambar 4.36 dapat dilihat contoh kasus ini mengambil bagian pertama untuk ketiga *continuation class* yang terdapat pada bagian pemeriksaan akhir. Sampai sejauh ini sudah didapatkan hasil akhir dari bagian morfotaktik, sehingga contoh kasus akan bergerak ke bagian morfofonemik.

```
! Paralelisasi rule-rule sebelumnya yang perlu saja
define RG4 %^N->0||[m e|p e]_"lbraces"* [l|m|n|r|y|w|t|s|p|k] ,,
%^N->n g||[p e]_"lbraces"* r ,, t->n||[m |p] e %^N "lbraces"*_ ,,
p->m||[m|p] e %^N "lbraces"*_ ,,
s->n y||[m|p] e %^N "lbraces"*_ ,,
k->n g||[m|p] e %^N "lbraces"*_;
! push defined RG4;
```

Gambar 4-37. Contoh Kasus Memakan Bagian Aturan Morfofonemik

Pada gambar 4.37 dapat dilihat contoh kasus ini akan mendapatkan aturan morfofonemik pada baris pertama yaitu “^N” *deletion*. Setelah bagian ini, akan tercapai kata “memakan” pada pita bawah dan kata “makan+Verb+AV” pada pita atas sesuai yang diinginkan.

4.5 Menggabungkan Aturan Morfotaktik dan Morfofonemik

Untuk menggabungkan aturan morfotaktik dengan morfofonemik sebenarnya bisa dilakukan di kedua sisi, baik di XFST maupun di LEXC. Akan

tetapi dikarenakan suatu fungsi yang saya gunakan di implementasi aturan morfotaktik, untuk saat ini lebih baik digabungkan di sisi XFST.

Urutan cara menggabungkan dari XFST:

1. Lakukan kompilasi aturan morfotaktik LEXC: **read lexc < "nama file LEXC.ekstensinya"**. Dengan dijlankannya perintah tersebut, akan didapatkan *network* hasil kompilasi aturan morfotaktik di atas.
2. Hilangkan semua flag diacritics yang terdapat didalam aturan morfotaktik: **eliminate flag "nama flag diacritics"**. Dengan dijlankannya perintah tersebut, akan didapatkan *network* hasil kompilasi aturan morfotaktik bersih tanpa *flag diacritics*.
3. Definisikan *network* hasil kompilasi sebagai suatu *symbol*: **define "nama symbol poin2"**. Dengan dijlankannya perintah tersebut, *network* pada poin 2 akan didefinisikan ke dalam sebuah simbol.
4. Lakukan kompilasi aturan morfofonemik XFST: **source "nama file XFST.ekstensinya"**. Dengan dijlankannya perintah tersebut, akan didapatkan *network* hasil kompilasi aturan morfofonemik di atas.
5. *Compose* kedua *network* poin 2 dan 3: **read regex "nama symbol poin2" .o. "nama symbol poin3" ;**. Dengan dijlankannya perintah tersebut, akan didapatkan *network* hasil komposisi aturan morfotaktik dan morfofonemik.
6. Lakukan *compile-replace command* untuk kata ulang: **compile-replace-lower**. Dengan dijlankannya perintah tersebut, akan didapatkan *network* yang mengizinkan kata ulang sesuai yang telah didefinisikan.
7. Definisikan *network* hasil kompilasi sebagai suatu *symbol*: **define "nama symbol poin6"**. Dengan dijlankannya perintah tersebut, *network* pada poin 6 akan didefinisikan ke dalam sebuah simbol.
8. Lakukan kompilasi pengaturan tanda "-" untuk kata ulang pada XFST: **source "nama file XFST.ekstensinya"**. Dengan dijlankannya perintah tersebut, akan didapatkan *network* hasil kompilasi pengaturan tanda "-".

9. *Compose* kedua *network* poin 6 dan 7: **read regex "nama symbol poin6" .o. "nama symbol poin7" ;**. Dengan dijlankannya perintah tersebut, akan didapatkan *network* hasil komposisi aturan morfotaktik –morfofonemik beserta kata ulang yang diizinkan dengan pengaturan tanda "-".
10. Definisikan *network* hasil kompilasi sebagai suatu *symbol*: **define "nama symbol poin9"**. Dengan dijlankannya perintah tersebut, *network* pada poin 9 akan didefinisikan ke dalam sebuah simbol.
11. *Compose* kedua *network* poin 9 dan 2: **read regex "nama symbol poin9" .o. "nama symbol poin2" ;** . Dengan dijlankannya perintah tersebut, akan didapatkan *network* akhir.

Maka setelah itu penggabungan selesai dilakukan.

BAB 5 HASIL DAN ANALISIS PENGUJIAN

Pada bab ini, penulis akan memaparkan hasil ujicoba terhadap pengurai morfologi dan analisis terhadap hasil pengujian tersebut.

5.1 *Test Cases* Pengujian

Pengurai morfologi ini dibangun dengan mengacu kepada aturan-aturan tata bahasa Indonesia yang terkandung dalam buku Tata bahasa Indonesia **[ALWI03]**. Pengurai morfologi ini juga bertugas menangani kasus-kasus yang termasuk dalam cakupannya saja. Oleh karena dua hal tersebut, pengurai morfologi ini akan diujikan berdasarkan *test cases* aturan-aturan yang diimplementasikan oleh pengurai morfologi ditambah beberapa *test cases* untuk perilaku-perilaku yang seharusnya tidak diterima oleh pengurai morfologi ini.

Pemilihan *test cases* dilihat berdasarkan aturan-aturan yang diimplementasikan. Tiap *case* akan mewakili perilaku-perilaku morfologi yang tercakup dalam Pengurai morfologi ini. *Test cases*-nya dapat dilihat pada Lampiran A.

Pada tabel A-1 ada bagian yang diberi tanda bintang (*) pada kolom hasil uraian yang diharapkan. Maksud dari tanda bintang tersebut adalah hasil tersebut hanya bisa dicapai jika pada *lexicon* sudah diberikan tanda khusus agar fitur-fitur analisis morfologi tersebut dapat dihasilkan. Jika tidak ada tanda khusus pada *lexicon*, maka diharapkan akan dihasilkan keluaran yang tanpa tanda bintang.

Tabel A-1 menunjukkan *test cases* yang mewakili semua kemungkinan variasi kata yang diterima oleh aturan morfotaktik. Setiap aturan diwakili untuk semua kemungkinan variasi penggabungannya dengan kata dasar dari kelas kata tertentu.

Test cases pada tabel A-2 mewakili kemungkinan-kemungkinan morfotaktik yang tidak sah dan seharusnya tidak diterima oleh pengurai morfologi. Semua

kemungkinan yang tidak sah ini didapatkan dari kemungkinan-kemungkinan penggabungan prefiks dengan preprefiks yang tidak sah, kemudian juga kemungkinan penggabungan prefiks dengan sufiks yang tidak sah, dan juga kemungkinan variasi reduplikasi yang tidak sah.

Test cases untuk variasi kasus morfotaktik yang tidak sah ini tentunya belum dapat mencakup semua variasi yang tidak sah. Hal ini dikarenakan banyak sekali kemungkinan yang tidak sah. Jumlah prefiks dan preprefiks yang mungkin ada sembilan buah, sedangkan kemungkinan sufiks ada empat buah, dan juga kemungkinan penutupan tanda kurung untuk reduplikasi juga ada dua buah. Belum lagi ditambahkan dengan empat kemungkinan kelas kata dasar. Secara lengkap, untuk kemungkinan semua variasi ada: 9 x 9 x 4 x 2 x 4 = 2592 kemungkinan, sedangkan kemungkinan yang sah hanya 116 kemungkinan. Dari semua kemungkinan itu dipilih beberapa yang cukup mewakili untuk semua variasi yang tidak sah tersebut. Jika kita periksa dari *test cases* yang sah, sudah cukup ditunjukkan bahwa untuk variasi kemungkinan kelas kata dasar sudah dicapai. Oleh karena itu, untuk variasi yang tidak sah diambil dari kemungkinan penggabungan preprefiks dan prefiks, kemungkinan penggabungan prefiks dengan sufiks, dan juga kemungkinan variasi reduplikasi yang tidak sah.

Test cases untuk kasus morfofonemik pada tabel A-3 diambil dari semua kemungkinan variasi morfofonemik seperti yang telah dirancang dan diimplementasikan sebelumnya.

Test cases pada tabel A-4 didapatkan dari kemungkinan tidak bekerjanya suatu aturan morfofonemik. Untuk memeriksanya diperlukan beberapa aturan yang memastikan suatu aturan memaksa agar aturan tersebut harus dilaksanakan jika syaratnya tercapai, dan juga memastikan aturan tersebut hanya dilakukan jika syaratnya tercapai.

Pada tabel A-1 sampai tabel A-4 terkandung pengujian untuk aturan morfotaktik maupun morfofonemik. Tiap *test cases* dipilih berdasarkan kasus-kasus yang dicakup dalam rancangan dan implementasi pengurai morfologi ini. Kasus yang diwakili seperti dapat dilihat pada kolom 4 dan 5 merupakan kasus

yang dirujuk dari rancangan pada Bab 3 dan juga telah diimplementasikan seperti dilaporkan pada Bab 4.

Semua *test cases* di atas akan dijalankan dan dilihat apakah mengeluarkan hasil seperti yang diharapkan atau tidak. Setelah dijalankan dan mengeluarkan hasil, pada hasil yang dikeluarkan tersebut akan dilakukan proses sebaliknya dan akan dilihat apakah dapat menghasilkan kembali *test cases* tersebut. Dalam XFST, proses menjalankan *test cases* untuk melihat analisis morfologinya dilakukan dengan *command*: *apply up* <nama test case>, sedangkan proses sebaliknya dilakukan dengan *command*: *apply down* <hasil analisis morfologi>. Hasil dari pengujian dapat dilihat pada subbab 5.2.

5.2. Hasil Pengujian

Dari *test cases* pada subbab 5.1, dijalankan pengujian menggunakan XFST untuk melihat sampai di mana pengurai morfologi ini dapat menghasilkan analisis morfologi suatu kata. Pada bagian pengujian ini dilihat kesesuaian antara hasil yang diharapkan dengan hasil yang diperoleh (diujikan dengan *command apply up*) dan juga bisa atau tidaknya pengurai morfologi ini menghasilkan kata pada *test cases* jika dilakukan proses sebaliknya dari hasil analisisnya (diujikan dengan *command apply down*). Pada tabel B-1 sampai tabel B-2 dapat dilihat rangkuman hasil pengujiannya.

Pada tabel B-1 dapat dilihat untuk tags khusus seperti “+Caus_kan”, “+Appl_kan”, “+Caus_i”, “+Appl_i”, “+Actor”, dan “+Instrument” menghasilkan dua analisis dengan ataupun tanpa tags tersebut. Kejadian ini disebabkan penggunaan *multiple entries* untuk kata yang diberikan *tags*. *Multiple entries* yang dimaksud disini ialah ketika suatu kata dimasukkan lebih dari satu kali sehingga kata tersebut terdapat dalam beberapa *entries*. Dalam penelitian ini, *multiple entries* tersebut sengaja dibuat untuk *tags* yang berbeda. Pemberian *multiple entries* ini dirasakan penulis sebagai solusi yang paling mungkin untuk saat ini agar pemberian *tags* khusus tersebut dapat menghasilkan analisis sebagaimana mestinya.

Seperti yang dijabarkan pada tabel B-2, masalah pada kasus morfofonemik yang tidak sah adalah ketika didapatkan suatu kata yang memiliki lebih dari satu kelas kata stem di *lexicon*. Masalah ini muncul pada kasus nomer 1 dan 29. Kata "satu" dan "dua" memiliki dua kelas kata pada *lexicon* dan itu mengakibatkan kasus tidah sah menjadi diterima. Untuk melihat apakah kasus itu benar-benar bisa diterima atau tidak bisa digunakan kata "atau" yang berasal dari kelas kata yang sama.

Sedangkan untuk kasus nomer 25, kata "permakan" dapat dihasilkan karena dua hal. Pertama karena kata "makan" memiliki lebih dari satu kelas kata seperti halnya kata "satu" dan "dua". Kedua karena terdapat kata "permak" pada *lexicon* yang akan menghasilkan analisis morfologi tersebut sesuai aturan morfotaktik.

Untuk hasil pada tabel B-3 dapat dikatakan tidak ada keanehan secara khusus. Untuk hal-hal seperti menghasilkan lebih dari satu analisis akan dijelaskan bersamaan dengan tabel-tabel lainnya setelah hasil pengujian tabel B-4.

Pada tabel B-4 kata senan bisa dihasilkan, akan tetapi seperti halnya yang diceritakan di dalam tabel itu tidak menjadi masalah karena tidak dihasilkan dari kata "senang". Kejadian ini dapat terjadi karena ada kata "sen" didalam *lexicon* yang sesuai aturan morfotaktik memang dapat dihasilkan.

Dapat dilihat dari tabel-tabel di atas dari tabel B-1 hingga tabel B-4 ada kata-kata yang menghasilkan lebih dari satu hasil analisis morfologi. Sebagai contoh kita lihat no. 14 pada tabel B-1, kata "mengabari" diuraikan menjadi "abar+Verb+AV" dan juga "kabar+Verb+AV". Hal tersebut dapat terjadi dikarenakan ada kata abar di dalam kamus *lexicon* sehingga kata mengabari mungkin dihasilkan dari kata abar ataupun kata kabar. Faktor lainnya yang memungkinkan terjadinya hal tersebut adalah pengambilan keputusan suatu kata dasar dapat bergabung dengan suatu imbuhan dilihat dari kelas kata dasarnya. Sedangkan dalam bahasa Indonesia terdapat banyak inkonsistensi antara kata dasar yang dapat digabungkan dengan suatu imbuhan atau tidak. Sebagai contoh dapat dilihat *counter example* pada rancangan morfotaktik Bab 3. Oleh karena pengurai morfologi ini tidak menangani hal itu maka hasil seperti di atas dapat

diterima. Hal ini juga yang menyebabkan banyak dihasilkannya kata-kata yang terkesan kurang tepat pada kolom hasil proses sebaliknya tabel B1 sampai B4.

Inkonsistensi dalam bahasa Indonesia dan masalah dua *entry* dalam *lexicon* yang menghasilkan analisis morfologi, sebenarnya bisa diatasi dengan *lexicon* khusus. *Lexicon* khusus yang dimaksud di sini ialah *lexicon* yang memiliki analisis morfologi yang lebih mendalam. *Lexicon* tersebut dibatasi untuk tiap katanya kemungkinan bergabungnya kata tersebut dengan suatu imbuhan atau kata ulang tertentu. *Lexicon* khusus ini juga dapat digunakan untuk mengatasi kemungkinan kata dasar yang dapat bergabung dengan sufiks –i. Dengan memberi tanda pada *lexicon* suatu kata dasar berakhiran –i, maka dapat diputuskan ia tidak dapat bergabung dengan sufiks –i.

Untuk penjelasan selanjutnya dan analisis dari tabel pada subbab 5.2 di atas, dapat dilihat pada subbab 5.3 di bawah ini.

5.3. Analisis Hasil Pengujian

Dari hasil-hasil pengujian pada subbab 5.2, dapat dirangkum hasilnya menjadi tabel-tabel rangkuman untuk tiap-tiap kelompok *test cases*. Tabel-tabel rangkuman tersebut dapat dilihat pada tabel 5-1 sampai tabel 5-4.

Tabel 5-1. Tabel Rangkuman Hasil Pengujian untuk Kasus Morfotaktik yang Sah

Hasil Analisis	Proses Penguraian (apply up)	Proses Sebaliknya (apply down)	Total Kedua Proses
Tepat satu hasil benar	70	10	**80**
Lebih dari satu hasil, tapi ada satu yang benar	46	106	**152**
Hasil salah	0	0	**0**
Total	**116**	**116**	**232**

Pada kelompok *test cases* yang mewakili kasus morfotaktik yang sah, banyak hasil proses penguraian yang menghasilkan lebih dari satu *tags*. Akan tetapi dari beberapa *tags* yang dihasilkan ada *tags* yang benar. Hal ini terjadi

karena seperti yang telah dijelaskan pada subbab 5.2 karena beberapa faktor, di antaranya:

1. Adanya suatu kata di dalam *lexicon* selain kata yang dimaksud yang memang secara aturan morfotaktik dapat dihasilkan juga dan sah. Hal ini terkait inkonsistensi dalam bahasa Indonesia yang dapat diatasi jika diberikan *lexicon* yang memiliki analisis morfologi lebih mendalam.

2. Pada pengurai morfologi ini, penentuan dapat bergabungnya suatu kata hanya didasarkan pada kelas katanya. Sedangkan dalam bahasa Indonesia ada beberapa kata yang tidak bisa bergabung dengan suatu imbuhan meskipun kata lainnya dari kelas kata yang sama bisa bergabung dengan imbuhan tersebut. Masalah ini terkait dengan inkonsistensi bahasa Indonesia, maka cara mengatasinya sama seperti poin 1 yaitu dengan memberikan *lexicon* yang memiliki analisis morfologi lebih mendalam.

3. *Multiple entries* pada *lexicon* yang diimplementasikan untuk *tags* khusus. Hal ini dapat diatasi jika ditemukan solusi yang lebih sempurna untuk *tags* kasus tersebut.

Untuk proses sebaliknya, hasil dengan lebih dari satu keluaran akan tetapi ada yang benar lebih banyak lagi terjadinya. Kejadian seperti ini diakibatkan selain karena tiga faktor di atas juga karena satu faktor lainnya. Faktor tersebut yaitu karena banyaknya kesamaan *tags* yang dihasilkan untuk suatu kata. Faktor tersebut dapat diperbaiki dengan rancangan *tags* morfologi yang lebih mendalam, akan tetapi untuk *tags* yang ada sekarang, hasil yang dikeluarkan sudah seperti yang diharapkan. Selanjutnya dapat dilihat tabel rangkuman pengujian untuk kasus morfotaktik yang tidak sah.

Tabel 5-2. Tabel Rangkuman Hasil Pengujian untuk Kasus Morfotaktik yang Tidak Sah

Hasil Analisis	Proses Penguraian	Proses Sebaliknya	Total Kedua

	(apply up)	(apply down)	Proses
Tepat satu hasil benar	33	33	**66**
Lebih dari satu hasil, tapi ada satu yang benar	0	0	**0**
Hasil salah	3	3	**6**
Total	**36**	**36**	**72**

Seperti yang ditunjukkan pada tabel 5-2, ada tiga test cases yang mengeluarkan hasil yang salah. Hal ini disebabkan dua faktor yang juga terdapat pada kelompok test cases untuk kasus morfotaktik yang benar, yaitu:

1. Adanya suatu kata lainnya di dalam *lexicon* selain kata yang dimaksud yang memang secara aturan morfotaktik dapat dihasilkan juga dan sah. Hal ini terkait inkonsistensi dalam bahasa Indonesia yang dapat diatasi jika diberikan *lexicon* yang memiliki analisis morfologi lebih mendalam.

2. Pada pengurai morfologi ini, penentuan dapat bergabungnya suatu kata hanya didasarkan pada kelas katanya. Sedangkan dalam bahasa Indonesia ada beberapa kata yang tidak bisa bergabung dengan suatu imbuhan meskipun kata lainnya dari kelas kata yang sama bisa bergabung dengan imbuhan tersebut. Masalah ini terkait dengan inkonsistensi bahasa Indonesia, maka cara mengatasinya sama seperti poin 1 yaitu dengan memberikan *lexicon* yang memiliki analisis morfologi lebih mendalam.

Selanjutnya akan diperlihatkan tabel rangkuman hasil pengujian untuk kasus morfofonemik yang sah.

Tabel 5-3. Tabel Rangkuman Hasil Pengujian untuk Kasus Morfofonemik yang Sah

Hasil Analisis	Proses Penguraian	Proses Sebaliknya	Total Kedua Proses

	(apply up)	(apply down)	
Tepat satu hasil benar	30	0	**30**
Lebih dari satu hasil, tapi ada satu yang benar	6	36	**42**
Hasil salah	0	0	**0**
Total	**36**	**36**	**72**

Pada kolompok *test cases* untuk kasus morfofonemik yang sah dihasilkan juga banyak keluaran lebih dari satu hasil akan tetapi ada yang benar. Ini diakibatkan oleh faktor yang sama dengan kelompok *test cases* untuk kasus morfotaktik yang sah. Begitu juga untuk proses sebaliknya. Kejadian tersebut juga diakibatkan faktor yang sama dengan *test cases* pada kasus mordoraktik yang sah. Selanjutnya akan dipaparkan tabel rangkuman hasil pengujian untuk kasus morfofonemik yang tidak sah.

Tabel 5-4. Tabel Rangkuman Hasil Pengujian untuk Kasus Morfofonemik yang Tidak Sah

Hasil Analisis	Proses Penguraian (apply up)	Proses Sebaliknya (apply down)	Total Kedua Proses
Tepat satu hasil benar	21	21	**42**
Lebih dari satu hasil, tapi ada satu yang benar	0	0	**0**
Hasil salah	1	1	**2**
Total	**22**	**22**	**44**

Satu-satunya hasil salah pada kelompok *test cases* ini seperti halnya disebutkan pada subbab 5.2. Kejadian itu disebabkan kata lainnya yang terdapat di dalam *lexicon* dan sesuai aturan memang dapat dihasilkan. Cara yang dapat ditempuh agar kata yang dihasilkan juga merupakan kata yang sah menurut

kaidah bahasa Indonesia ialah dengan *lexicon* yang memiliki analisis morfologi lebih mendalam.

Kemudian dari empat tabel diatas dapat dirangkum menjadi tabel 5-5.

Tabel 5-5. Tabel Rangkuman Hasil Pengujian Keseluruhan

Hasil Analisis	Proses Penguraian (apply up)	Proses Sebaliknya (apply down)	Total Kedua Proses
Tepat satu hasil benar	154	64	**218**
Lebih dari satu hasil, tapi ada satu yang benar	52	142	**194**
Hasil salah	4	4	**8**
Total	**210**	**210**	**420**

Didapatkan hasil ujicoba keseluruhan dari pengurai morfologi ini seperti tabel 5-5 di atas.