BAB 1

PENDAHULUAN

1.1 Latar Belakang

Pemrosesan bahasa alami atau dalam bahasa Inggris disebut dengan *natural language processing* (NLP) adalah cabang ilmu komputer dan linguistik yang mengkaji interaksi antara komputer dan manusia menggunakan bahasa alami. NLP sering dianggap sebagai cabang dari kecerdasan buatan dan bidang kajiannya bersinggungan dengan linguistik komputasional. Kajian NLP antara lain mencakup segmentasi tuturan *(speech segmentation)*, segmentasi teks *(text segmentation)*, penandaan kelas kata *(part-of-speech tagging)*, serta pengawataksaan makna *(word sense disambiguation)*. Salah satu alat yang digunakan oleh komputer dalam proses mengenali bahasa alami manusia adalah *morphological parser*.

Morphological parser berfungsi untuk membagi sebuah kata menjadi komponen-komponen penyusunnya. Proses ini dapat mengenali komponen kata seperti awalan, bentuk dasar, sisipan, dan akhiran serta dapat mengenali jika kata tersebut merupakan kata ulang maupun kata majemuk. Proses di mana *morphological parser* melakukan tugasnya dalam menguraikan kata menjadi komponen-komponen penyusunnya disebut dengan *morphological parsing*. Proses ini dapat membantu mengurangi ambiguitas selama proses mengetahui makna suatu kalimat. Sebagai contoh, kata "mengurus" bisa mempunyai makna menjadi kurus maupun mengerjakan sebuah urusan, bergantung pada apa bentuk dasar dari kata tersebut. Jika kita bisa membagi kata tersebut menjadi komponen penyusunnya, kita bisa lebih yakin mengenai makna dari kata tersebut dalam kalimat. *Morphological parsing* merupakan salah satu proses penting dalam NLP.

Morphological parser sudah banyak dibuat untuk beberapa bahasa yang ada di dunia, seperti bahasa Inggris, bahasa Turki, dan bahasa Bangla. Pisceldo et al. (2008) pernah membuat *morphological analyser* untuk bahasa Indonesia melalui pendekatan *two-level*, namun hanya dapat memproses kata hasil afiksasi dan reduplikasi. Dalam bahasa Indonesia, selain proses afiksasi dan

reduplikasi, dikenal ada satu lagi proses morfologi yang umum dilakukan, yaitu proses komposisi. Proses komposisi adalah proses penggabungan bentuk dasar dengan bentuk dasar lain untuk mewadahi suatu "konsep" yang belum tertampung dalam sebuah kata[1]. Dalam skripsi ini, akan dibuat sebuah perangkat lunak *morphological parser* yang dapat memproses kata dalam bahasa Indonesia yang merupakan hasil proses afiksasi, reduplikasi, dan komposisi.

1.2 Rumusan Masalah

Sehubungan dengan latar belakang yang telah diuraikan di atas, maka dibuat rumusan masalah sebagai berikut ini.

- Bagaimana aturan morfologi bahasa Indonesia?
- Bagaimana struktur data dari *lexicon* yang digunakan pada perangkat lunak?
- Bagaimana cara mengimplementasikan aturan morfologi bahasa Indonesia ke dalam perangkat lunak?
- Bagaimana performansi dari perangkat lunak yang dihasilkan?

1.3 Tujuan

Tujuan dari penelitian ini adalah sebagai berikut:

- Mengetahui aturan morfologi bahasa Indonesia
- Mengetahui struktur data dari *lexicon* yang digunakan pada perangkat lunak
- Mengimplementasikan aturan morfologi bahasa Indonesia ke dalam perangkat lunak
- Mengetahui performansi dari perangkat lunak yang dihasilkan

1.4 Batasan Masalah

Terdapat beberapa batasan masalah untuk penelitian ini:

- Kalimat yang dapat diproses adalah kalimat dalam bahasa Indonesia yang ditulis sesuai ejaan yang disempurnakan (EYD)

- Kata yang dapat diproses adalah kata yang merupakan bentuk dasar dan kata yang dibentuk dari proses morfologi berupa afiksasi, reduplikasi, dan komposisi
- Kata yang belum ada dalam Kamus Besar Bahasa Indonesia (KBBI) dan yang bukan merupakan hasil dari proses afiksasi, reduplikasi, dan komposisi dianggap sebagai bentuk asing
- Kata yang merupakan hasil proses penyisipan (infiksasi) dan belum ada dalam KBBI tidak dapat diproses karena infiksasi dianggap sudah tidak produktif dalam bahasa Indonesia pada saat ini

1.5 Metodologi Penelitian

Tahap-tahap yang akan dilakukan dalam penelitian ini adalah sebagai berikut:

1. Melakukan studi literatur tentang morfologi bahasa Indonesia dan perangkat lunak *morphological parser* yang sudah ada
2. Melakukan analisis pada *morphological parser* bahasa Indonesia dan *lexicon* yang digunakan serta merancang struktur data dari *lexicon*
3. Merancang dan mengimplementasikan *lexicon* dan *morphological parser* ke dalam perangkat lunak
4. Mengumpulkan contoh kalimat dalam bahasa Indonesia sebagai bahan pengujian
5. Melakukan pengujian terhadap perangkat lunak

1.6 Sistematika Pembahasan

Keseluruhan bab yang disusun dalam karya tulis ini terbagi ke dalam bab-bab sebagai berikut:

1. BAB 1 - PENDAHULUAN membahas mengenai latar belakang, rumusan masalah, tujuan, batasan masalah, metodologi penelitian, dan sistematika pembahasan.
2. BAB 2 - DASAR TEORI membahas mengenai morfem, proses morfologi bahasa Indonesia, *lexicon* bahasa Indonesia dan struktur data dari *lexicon*.
3. BAB 3 - ANALISIS membahas mengenai analisis *morphological parser* bahasa Indonesia, *lexicon* bahasa Indonesia, dan struktur data *lexicon* yang digunakan pada perangkat lunak *Morphological Parser*.

4. BAB 4 - PERANCANGAN membahas mengenai perancangan antarmuka dan struktur data pada perangkat lunak *Morphological Parser*.

5. BAB 5 - IMPLEMENTASI DAN PENGUJIAN membahas mengenai implementasi dan pengujian yang dilakukan pada perangkat lunak *Morphological Parser*.

6. BAB 6 - KESIMPULAN DAN SARAN membahas mengenai kesimpulan dan saran mengenai perangkat lunak *Morphological Parser*.

BAB 2

DASAR TEORI

Pada bab ini dijelaskan mengenai ....

2.1 Morfologi

Secara etimologi, kata *morfologi* berasal dari kata *morf* yang berarti 'bentuk' dan kata *logi* yang berarti 'ilmu'[1]. Secara harfiah, kata *morfologi* berarti 'ilmu mengenai bentuk'. Di dalam kajian linguistik, *morfologi* berarti 'ilmu mengenai bentuk-bentuk dan pembentukan kata'; sedangkan di dalam kajian biologi, *morfologi* berarti 'ilmu mengenai bentuk-bentuk sel-sel tumbuhan atau jasad-jasad hidup'. Kesamaan dari dua bidang kajian tersebut adalah keduanya mengkaji tentang bentuk.

Jika morfologi dalam kajian linguistik membicarakan tentang bentuk-bentuk dan pembentukan kata, maka segala bentuk dan jenis morfem yang merupakan satuan bentuk sebelum menjadi kata perlu dibicarakan juga. Pembicaraan mengenai pembentukan kata akan melibatkan pembicaraan mengenai komponen atau unsur pembentukan kata, yaitu morfem, baik morfem dasar maupun morfem afiks, dengan berbagai alat proses pembentukan kata, yaitu afiks dalam proses pembentukan kata melalui proses afiksasi, duplikasi atau pengulangan dalam proses pembentukan kata melalui proses reduplikasi, penggabungan dalam proses pembentukan kata melalui proses komposisi, dan sebagainya.

Ujung dari proses morfologi adalah terbentuknya *kata* dalam bentuk dan makna sesuai dengan keperluan dalam satu tindak pertuturan. Bila bentuk dan makna yang terbentuk dari satu proses morfologi sesuai dengan yang diperlukan dalam pertuturan, maka bentuknya dapat dikatakan berterima; tetapi jika tidak sesuai dengan yang diperlukan, maka bentuk itu dikatakan tidak berterima. Keberterimaan atau ketidakberterimaan bentuk itu dapat juga karena alasan sosial.

Objek kajian morfologi adalah satuan-satuan morfologi, proses-proses morfologi, dan alat-alat

dalam proses morfologi itu[1]. Satuan morfologi adalah:

1. Morfem (akar atau afiks).
2. Kata.

Lalu, proses morfologi melibatkan komponen:

1. Dasar (bentuk dasar).
2. Alat pembentuk (afiks, duplikasi, komposisi).

Morfem adalah satuan gramatikal terkecil yang bermakna. Morfem dapat berupa akar (dasar) dan dapat pula berupa afiks. Perbedaannya, morfem berupa akar dapat menjadi dasar dalam pembentukan kata, sedangkan morfem berupa afiks hanya "menjadi" penyebab terjadinya makna gramatikal. Kemudian, *kata* adalah satuan gramatikal yang terjadi sebagai hasil dari proses morfologis. Jika berdiri sendiri, setiap kata memiliki makna leksikal dan dalam kedudukannya dalam satuan ujaran memiliki makna gramatikal.

Dalam proses morfologi, dasar atau bentuk dasar merupakan bentuk yang mengalami proses morfologi. Dasar ini dapat berupa sebuah kata dasar maupun bentuk polimorfemis (bentuk berimbuhan, bentuk ulang, atau bentuk gabungan). Alat pembentuk kata dapat berupa afiks dalam proses afiksasi, pengulangan dalam proses reduplikasi, dan penggabungan dalam proses komposisi.

2.2 Morfem

Morfem adalah satuan gramatikal terkecil yang memiliki makna[1]. Dengan kata terkecil berarti "satuan" itu tidak dapat dianalisis menjadi lebih kecil lagi tanpa merusak maknanya. Sebagai contoh, bentuk *membeli* dapat dianalisis menjadi dua bentuk terkecil yaitu {me-} dan {beli}. Bentuk {me-} adalah sebuah morfem, yakni morfem afiks yang secara gramatikal memiliki sebuah makna; dan bentuk {beli} juga sebuah morfem, yakni morfem dasar yang secara leksikal memiliki makna. Kalau bentuk *beli* dianalisis menjadi lebih kecil lagi menjadi *be-* dan *li*, keduanya tidak memiliki makna apapun. Jadi, keduanya bukan morfem. Contoh lain, bentuk *berpakaian* dapat dianalisis ke dalam satuan-satuan terkecil menjadi {ber-}, {pakai}, dan {-an}. Ketiganya adalah morfem, di mana {ber-} adalah morfem prefiks, {pakai} adalah morfem dasar, dan {-an} adalah morfem sufiks. Ketiganya memiliki makna. Morfem {ber-} dan morfem {-an} memiliki makna gramatikal, sedangkan morfem {pakai} memiliki makna leksikal. Perlu dicatat dalam konvensi linguistik sebuah bentuk dinyatakan sebagai morfem ditulis dalam kurung kurawal ({...}).

2.2.1 Identifikasi Morfem

Satuan bahasa merupakan komposit antara bentuk dan makna[1]. Oleh karena itu, untuk menetapkan sebuah bentuk adalah morfem atau bukan didasarkan pada kriteria bentuk dan makna tersebut. Hal-hal berikut dapat menjadi pedoman untuk menentukan apakah sebuah bentuk adalah morfem atau bukan.

1. Dua bentuk yang sama atau lebih memiliki makna yang sama merupakan sebuah morfem. Umpamanya kata *bulan* pada ketiga kalimat berikut adalah sebuah morfem yang sama.
   - *Bulan* depan dia akan menikah.
   - Sudah tiga *bulan* dia belum bayar uang SPP.
   - *Bulan* November lamanya 30 hari.
2. Dua bentuk yang sama atau lebih bila memiliki makna yang berbeda merupakan dua morfem yang berbeda. Misalnya kata *bunga* pada kedua kalimat berikut adalah dua buah morfem yang berbeda.
   - Bank Indonesia memberi *bunga* 5 persen per tahun.
   - Dia datang membawa seikat *bunga*.
3. Dua buah bentuk yang berbeda, tetapi memiliki makna yang sama, merupakan dua morfem yang berbeda. Umpamanya, kata *ayah* dan kata *bapak* pada kedua kalimat berikut adalah dua morfem yang berbeda.
   - *Ayah* pergi ke Medan.
   - *Bapak* baru pulang dari Medan.
4. Bentuk-bentuk yang mirip (berbeda sedikit) tetapi maknanya sama adalah sebuah morfem yang sama, asal perbedaan bentuk itu dapat dijelaskan secara fonologis. Umpamanya, bentuk-bentuk *me-*, *mem-*, *men-*, *meny-*, *meng-*, dan *menge-* pada kata-kata berikut adalah sebuah morfem yang sama.
   - *me*lihat
   - *mem*bina
   - *men*dengar

- *meny*usul
- *meng*ambil
- *menge*cat

5. Bentuk yang hanya muncul dengan pasangan satu-satunya adalah sebuah morfem juga. Umpamanya bentuk *renta* pada konstruksi *tua renta*, dan bentuk *kuyup* pada konstruksi *basah kuyup* adalah juga morfem. Contoh lain, bentuk *bugar* pada *segar bugar*, dan bentuk *mersik* pada *kering mersik*.

6. Bentuk yang muncul berulang-ulang pada satuan yang lebih besar apabila memiliki makna yang sama adalah juga merupakan morfem yang sama. Misalnya bentuk *baca* pada kata-kata berikut adalah sebuah morfem yang sama.

   - mem*baca*
   - pem*baca*
   - pem*baca*an
   - *baca*an
   - ter*baca*
   - keter*baca*an

7. Bentuk yang muncul berulang-ulang pada satuan bahasa yang lebih besar, apabila mempunyai bentuk bahasa yang sama namun maknanya berbeda (polisemi) merupakan morfem yang sama. Umpamanya, kata *kepala* pada kalimat-kalimat berikut memiliki makna yang berbeda, tetapi tetap merupakan morfem yang sama.

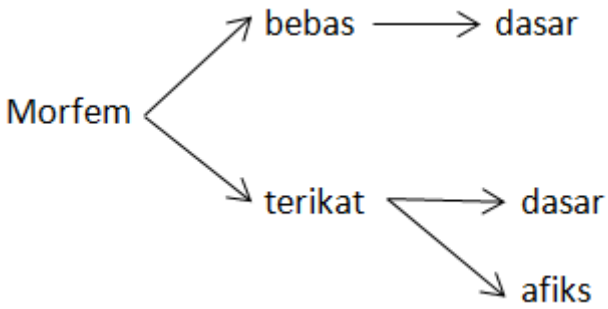
   - Ibunya menjadi *kepala* sekolah di sana.
   - Nomor teleponnya tertera pada *kepala* surat itu.
   - *Kepala* jarum itu terbuat dari plastik.
   - Setiap *kepala* mendapat bantuan sepuluh ribu rupiah.
   - Tubuhnya memang besar tetapi sayang *kepala*nya kosong.

2.2.2 Jenis Morfem

Dalam kajian morfologi biasanya dibedakan adanya beberapa morfem berdasarkan kriteria tertentu, seperti kriteria kebebasan, keutuhan, makna, dan sebagainya. Berikut adalah jenis-jenis morfem

tersebut.

1. Berdasarkan kebebasannya untuk dapat digunakan langsung dalam pertuturan, dibedakan adanya *morfem bebas* dan *morfem terikat*. Morfem bebas adalah morfem yang tanpa keterkaitannya dengan morfem lain dapat langsung digunakan dalam pertuturan. Misalnya, morfem {pulang}, {merah}, dan {pergi}. Morfem bebas ini tentunya berupa morfem dasar. Sedangkan morfem terikat adalah morfem yang harus terlebih dahulu bergabung dengan morfem lain untuk dapat digunakan dalam pertuturan. Dalam hal ini, semua afiks dalam bahasa Indonesia termasuk morfem terikat. Di samping itu, banyak juga morfem terikat yang berupa morfem dasar, seperti {henti}, {juang}, dan {geletak}. Untuk dapat digunakan, ketiga morfem ini harus terlebih dahulu diberi afiks atau digabung dengan morfem lain. Misalnya {juang} menjadi *berjuang*, *pejuang*, dan *daya juang*; *henti* harus digabung dulu dengan afiks tertentu seperti menjadi *berhenti*, *perhentian*, dan *menghentikan*; dan *geletak* harus diberi imbuhan dulu, misalnya menjadi *tergeletak*, dan *menggeletak*. Adanya morfem bebas dan terikat dapat digambarkan pada gambar 2.1 berikut.



Gambar 2.1: Morfem bebas dan terikat[1]

Berkenaan dengan bentuk dasar terikat, perlu dikemukakan catatan sebagai berikut:

Pertama, bentuk dasar terikat seperti *gaul*, *juang*, dan *henti* lazim juga disebut sebagai *prakategorial* karena bentuk-bentuk tersebut belum memiliki kategori sehingga tidak dapat digunakan dalam pertuturan.

Kedua, Verhaar (1978) juga memasukkan bentuk-bentuk seperti *beli*, *baca*, dan *tulis* ke dalam kelompok prakategorial, karena untuk digunakan di dalam kalimat harus terlebih dahulu diberi prefiks *me-*, prefiks *di-*, atau prefiks *ter-*. Dalam kalimat imperatif memang tanpa imbuhan bentuk-bentuk tersebut dapat digunakan. Namun, kalimat imperatif adalah hasil transformasi dari kalimat aktif transitif (yang memerlukan imbuhan).

Ketiga, bentuk-bentuk seperti *renta* (yang hanya muncul dalam *tua renta*), *kerontang* (yang

hanya muncul dalam *kering kerontang*), dan *kuyup* (yang hanya muncul dalam *basah kuyup*) adalah juga termasuk morfem terikat. Lalu, oleh karena hanya muncul dalam pasangan tertentu, maka disebut *morfem unik*.

Keempat, bentuk-bentuk yang disebut klitika merupakan morfem yang agak sukar ditentukan statusnya, apakah morfem bebas atau morfem terikat. Kemunculannya dalam pertuturan selalu terikat dengan bentuk lain, tetapi dapat dipisahkan. Umpamanya klitika *-ku* dalam konstruksi *bukuku* dapat dipisahkan sehingga menjadi *buku baruku*. Dilihat dari posisi tempatnya dibedakan adanya proklitika, yaitu klitika yang berposisi di muka kata yang diikuti seperti klitika *ku-* dalam bentuk *kubawa* dan *kauambil*. Sedangkan yang disebut enklitika adalah klitika yang berposisi di belakang kata yang dilekati, seperti klitika *-mu* dan *-nya* pada bentuk *nasibmu* dan *duduknya*.

Kelima, bentuk-bentuk yang termasuk preposisi dan konjungsi seperti *dan*, *oleh*, *di*, dan *karena* secara morfologis termasuk morfem bebas, tetapi secara sintaksis merupakan bentuk terikat (dalam satuan sintaksisnya).

Keenam, bentuk-bentuk yang oleh Kridalaksana (1989) disebut proleksem, seperti *a* (pada *asusila*), *dwi* (pada *dwibahasa*), dan *ko* (pada *kopilot*) juga termasuk morfem terikat.

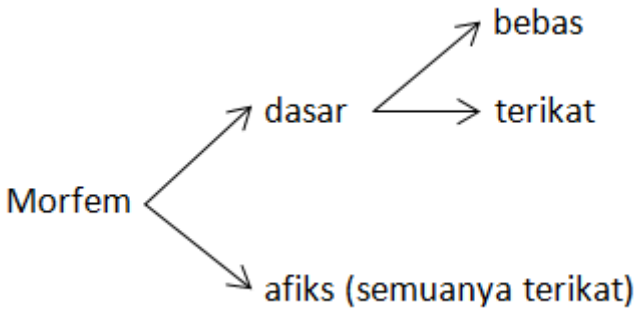
2. Berdasarkan keutuhan bentuknya dibedakan adanya *morfem utuh* dan *morfem terbagi*. Morfem utuh secara fisik merupakan satu-kesatuan yang utuh. Semua morfem dasar, baik bebas maupun terikat, serta prefiks, infiks, dan sufiks termasuk morfem utuh. Sedangkan yang dimaksud morfem terbagi adalah morfem yang fisiknya terbagi atau disisipi morfem lain. Karenanya semua konfiks (seperti *pe-an*, *ke-an*, dan *per-an*) adalah termasuk morfem terbagi. Namun, mengenai morfem terbagi ini ada dua catatan yang perlu diperhatikan.

   Pertama, semua konfiks adalah morfem terbagi; tetapi pada bentuk *ber-an* ada yang berupa konfiks dan ada yang bukan konfiks. Jika kata dalam bentuk *ber-an* tidak memiliki arti ketika hanya ditambahkan prefiks *ber-* atau sufiks *-an* saja, maka bentuk *ber-an* tersebut adalah berupa konfiks. Namun, jika kata tersebut memiliki arti ketika hanya ditambahkan prefiks *ber-* atau sufiks *-an* saja, maka bentuk *ber-an* tersebut adalah berupa *klofiks* (akronim dari kelompok afiks). Contoh, kata *bermunculan* adalah dasar *muncul* ditambahkan konfiks *ber-an* sementara kata *berpakaian* adalah prefiks *ber-* yang ditambahkan pada bentuk *pakaian*.

   Kedua, dalam bahasa Indonesia ada afiks yang disebut infiks, yaitu afiks yang ditempatkan di tengah (di dalam kata). Umpamanya infiks *-el-* pada dasar *tunjuk* menjadi kata *telunjuk*. Di sini infiks itu memecah morfem *tunjuk* menjadi dua bagian, yaitu *t-el-unjuk*. Dengan demikian

morfem *t-unjuk* menjadi morfem terbagi, bukan morfem utuh.

3. Berdasarkan kemungkinan menjadi dasar dalam pembentukan kata, dibedakan *morfem dasar* dan *morfem afiks*. Morfem dasar adalah morfem yang dapat menjadi dasar dalam suatu proses morfologi. Misalnya, morfem {beli}, {makan}, dan {merah}. Namun, perlu dicatat bentuk dasar yang termasuk dalam kategori preposisi dan konjungsi tidak pernah mengalami proses afiksasi. Sedangkan, yang tidak dapat menjadi dasar, melainkan hanya sebagai pembentuk disebut morfem afiks, seperti morfem {me-}, {-kan}, dan {pe-an}. Berdasarkan pembagian ini, maka dapat dibuat gambar 2.2 berikut.



Gambar 2.2: Morfem dasar dan afiks[1]

4. Berdasarkan ciri semantik dibedakan adanya *morfem bermakna leksikal* dan *morfem tak bermakna leksikal*. Sebuah morfem disebut bermakna leksikal karena di dalam dirinya, secara inheren, telah memiliki makna. Semua morfem dasar bebas, seperti {makan}, {pulang}, dan {pergi} termasuk morfem bermakna leksikal. Sebaliknya, morfem afiks seperti {ber-}, {ke-}, dan {ter-} termasuk morfem tak bermakna leksikal. Morfem bermakna leksikal dapat langsung menjadi unsur dalam pertuturan, sementara morfem tidak bermakna leksikal tidak dapat.

Dikotomi morfem bermakna leksikal dan tidak bermakna leksikal ini, untuk bahasa Indonesia timbul masalah. Morfem-morfem seperti {juang}, {henti}, dan {gaul} memiliki makna leksikal atau tidak. Kalau dikatakan memiliki makna leksikal, pada kenyataannya morfem-morfem itu belum dapat digunakan dalam pertuturan sebelum mengalami proses morfologi. Kalau dikatakan tidak bermakna leksikal, pada kenyataannya morfem-morfem tersebut bukan afiks.

2.2.3 Morfem Dasar, Bentuk Dasar, Akar, dan Leksem

Morfem dasar, bentuk dasar (lebih lazim dasar (*base*) saja), akar, dan leksem adalah empat istilah yang lazim digunakan dalam kajian morfologi. Namun, seringkali digunakan secara kurang cermat, malah seringkali berbeda. Oleh karena itu, ada baiknya istilah-istilah tersebut dibicarakan dulu sebelum pembicaraan mengenai proses-proses morfologi.

Istilah *morfem dasar* biasanya digunakan sebagai dikotomi dengan morfem afiks. Jadi, bentuk-bentuk seperti {beli}, {juang}, dan {kucing} adalah morfem dasar. Morfem dasar ini ada yang termasuk morfem bebas seperti {beli}, {kucing}, dan {pulang}; tetapi ada pula yang termasuk morfem terikat, seperti {juang}, {henti}, dan {tempur}. Sedangkan morfem afiks seperti {*ber-*}, {*di-*}, dan {*-an*} jelas semuanya termasuk morfem terikat seperti dijelaskan pada gambar 2.2 di atas.

Sebuah morfem dasar dapat menjadi bentuk dasar atau dasar (*base*) dalam suatu proses morfologi. Artinya, morfem dasar dapat diberi afiks tertentu dalam proses afiksasi, dapat diulang dalam proses reduplikasi, atau dapat digabung dengan morfem yang lain dalam suatu proses komposisi atau pemajemukan.

Istilah *bentuk dasar* atau *dasar (base)* biasanya digunakan untuk menyebut sebuah bentuk yang menjadi dasar dalam suatu proses morfologi. Bentuk dasar ini dapat berupa morfem tunggal, tetapi dapat juga berupa gabungan morfem. Umpamanya pada kata *berbicara* yang terdiri dari morfem {ber-} dan morfem {bicara}; maka morfem {bicara} adalah menjadi bentuk dasar dari kata *berbicara* itu, yang kebetulan juga berupa morfem dasar. Pada kata *dimengerti* bentuk dasarnya adalah *mengerti*, dan pada kata *keanekaragaman* bentuk dasarnya adalah *aneka ragam*. Pada bentuk reduplikasi *rumah-rumah* bentuk dasarnya adalah *rumah*, pada bentuk reduplikasi *berlari-lari* bentuk dasarnya berlari, dan pada bentuk reduplikasi *kemerah-merahan* bentuk dasarnya adalah *kemerahan*. Lalu, pada komposisi *sate ayam* bentuk dasarnya adalah *sate*, pada komposisi *ayam betina* bentuk dasarnya adalah *ayam*, dan pada komposisi *pasar induk* bentuk dasarnya adalah *pasar*. Jadi, bentuk dasar adalah bentuk yang langsung menjadi dasar dalam suatu proses morfologi. Wujudnya dapat berupa morfem tunggal, dapat juga berupa bentuk polimorfemis (terdiri dari dua morfem atau lebih).

Istilah *akar (root)* digunakan untuk menyebut bentuk yang tidak dapat dianalisis lebih jauh lagi. Artinya, akar adalah bentuk yang tersisa setelah semua afiksnya ditanggalkan. Misalkan pada kata *memberlakukan* setelah semua afiksnya ditanggalkan (yaitu prefiks *me-*, prefiks *ber-*, dan sufiks

-*kan*) dengan cara tertentu, maka yang tersisa adalah akar *laku*. Akar *laku* ini tidak dapat dianalisis lebih jauh lagi tanpa merusak makna akar tersebut. Contoh lain, kata *keberterimaan* kalau semua afiksnya ditanggalkan akan tersisa akarnya yaitu bentuk *terima*. Bentuk *terima* ini pun tidak dapat dianalisis lebih jauh lagi.

Istilah *leksem* ada digunakan dalam dua bidang kajian linguistik, yaitu bidang *morfologi* dan bidang *semantik*. Dalam kajian morfologi, leksem digunakan untuk mewadahi konsep "bentuk yang akan menjadi kata" melalui proses morfologi. Umpamanya bentuk PUKUL (dalam konvensi 'morfologi' leksem ditulis dengan huruf kapital semua) adalah sebuah leksem yang akan menurunkan kata-kata seperti *memukul*, *dipukul*, *terpukul*, *pukulan*, *pemukul*, dan *pemukulan*. Sedangkan dalam kajian semantik leksem adalah satuan bahasa yang memiliki sebuah makna. Jadi, bentuk-bentuk seperti *kucing*, *membaca*, *matahari*, *membanting tulang*, dan *sumpah serapah* adalah leksem.

Dari bentuk *leksem* ada bentuk-bentuk turunannya, yaitu *leksikon*, *leksikal*, *leksikologi*, dan *leksikografi*. Istilah leksikon dalam arti 'kumpulan leksem' dapat dipadankan dengan istilah *kosakata* atau *perbendaharaan kata*.

2.2.4 Morfem Afiks

Sudah dijelaskan pada subbab 2.2.2 bahwa morfem afiks adalah morfem yang tidak dapat menjadi dasar dalam pembentukan kata, tetapi hanya menjadi unsur pembentuk dalam proses afiksasi. Dalam bahasa Indonesia dibedakan adanya morfem afiks yang disebut:

1. *Prefiks*, yaitu afiks yang dibubuhkan di kiri bentuk dasar, yaitu prefiks *ber*-, prefiks *me*-, prefiks *per*-, prefiks *pe*-, prefiks *di*-, prefiks *ter*-, prefiks *se*-, dan prefiks *ke*-.

2. *Infiks*, yaitu afiks yang dibubuhkan di tengah kata, biasanya pada suku awal kata, yaitu infiks -*el*-, infiks -*em*-, dan infiks -*er*-.

3. *Sufiks*, adalah afiks yang dibubuhkan di kanan bentuk dasar, yaitu sufiks -*kan*, sufiks -*i*, dan sufiks -*an*.

4. *Konfiks*, yaitu afiks yang dibubuhkan di kiri dan di kanan bentuk dasar secara bersamaan karena konfiks ini merupakan satu kesatuan afiks. Konfiks yang ada dalam bahasa Indonesia adalah konfiks *ke-an*, konfiks *ber-an*, konfiks *pe-an*, konfiks *per-an*, dan konfiks *se-nya*.

5. *Klitika*[1], adalah imbuhan yang dalam ucapan tidak mempunyai tekanan sendiri dan tidak merupakan kata karena tidak dapat berdiri sendiri. Jadi, klitika merupakan bentuk yang

[1] *id.wikibooks.org/wiki/Bahasa_Indonesia/Klitika*

selalu terikat pada bentuk (kata) lain. Dilihat dari posisi tempatnya, dibedakan adanya *proklitika*, yaitu klitika yang berposisi di sebelah kiri kata yang diikuti seperti klitika *ku-* dan *kau-* dalam bentuk *kubawa* dan *kauambil*. Sedangkan yang disebut *enklitika* adalah klitika yang berposisi di belakang kata yang dilekati, seperti klitika *-ku*, *-mu*, *-nya*, dan *-lah* pada bentuk *bukuku*, *nasibmu*, *duduknya*, dan *pergilah*. Ada juga bentuk klitika yang ditulis terpisah dari kata yang diimbuhkan, yaitu klitika *pun* pada bentuk *kami pun*.

6. Dalam bahasa Indonesia ada bentuk kata yang *berklofiks*, yaitu kata yang dibubuhi afiks pada kiri dan kanannya; tetapi pembubuhannya itu tidak sekaligus, melainkan bertahap. Kata-kata berklofiks dalam bahasa Indonesia adalah yang berbentuk *me-kan*, *me-i*, *memper-*, *memper-kan*, *memper-i*, *ber-kan*, *di-kan*, *di-i*, *diper-*, *diper-kan*, *diper-i*, *ter-kan*, dan *ter-i*.

7. Dalam ragam nonbaku ada afiks nasal yang direalisasikan dengan nasal *m-*, *n-*, *ny-*, *ng-*, dan *nge-*. Kridalaksana (1989) menyebut afiks nasal ini dengan istilah *simulfiks*. Contoh: *nulis*, *nyisir*, *ngambil*, dan *ngecat*.

2.3 Proses Morfologi

Proses morfologi pada dasarnya adalah proses pembentukan kata dari sebuah bentuk dasar melalui pembubuhan afiks (dalam proses afiksasi), pengulangan (dalam proses reduplikasi), dan penggabungan (dalam proses komposisi)[1]. Prosedur ini berbeda dengan analisis morfologi yang mencerai-ceraikan kata (sebagai satuan sintaksis) menjadi bagian-bagian atau satuan-satuan yang lebih kecil. Sebagai contoh, jika dilakukan analisis morfologi terhadap kata *berpakaian*, mula-mula kata *berpakaian* dianalisis menjadi bentuk *ber-* dan *pakaian*; lalu bentuk *pakaian* dianalisis lagi menjadi bentuk *pakai* dan *-an*. Dalam proses morfologi, prosedurnya dibalik: mula-mula dasar *pakai* diberi sufiks *-an* menjadi *pakaian*. Kemudian kata *pakaian* itu diberi prefiks *ber-* menjadi *berpakaian*. Jadi, kalau analisis morfologi mencerai-ceraikan data kebahasaan yang ada, sedangkan proses morfologi mencoba menyusun dari komponen-komponen kecil menjadi sebuah bentuk yang lebih besar yang berupa kata kompleks atau kata yang polimorfemis.

Proses morfologi melibatkan komponen bentuk dasar dan alat pembentuk kata (afiksasi, reduplikasi, dan komposisi).

2.3.1 Bentuk Dasar

Pada subbab 2.2.3 telah disinggung bahwa *bentuk dasar* adalah bentuk yang kepadanya dilakukan proses morfologi. Bentuk dasar dapat berupa akar seperti *baca*, *pahat*, dan *juang* pada kata *membaca*, *memahat*, dan *berjuang*. Dapat pula berupa bentuk polimorfemis seperti bentuk *bermakna*, *berlari*, dan *jual beli* pada kata *kebermaknaan*, *berlari-lari*, dan *berjual beli*.

Dalam proses reduplikasi, bentuk dasar dapat berupa akar, seperti akar *rumah* pada kata *rumah-rumah*, akar *tinggi* seperti pada kata *tinggi-tinggi*, dan akar *marah* pada kata *marah-marah*. Dapat juga berupa kata berimbuhan seperti *menembak* pada kata *menembak-nembak*, kata berimbuhan *bangunan* pada kata *bangunan-bangunan*, dan kata berimbuhan *kemerahan* pada kata *kemerah-merahan*. Dapat juga berupa kata gabung seperti *rumah sakit* pada kata *rumah-rumah sakit*, dan *anak nakal* pada kata *anak-anak nakal*.

Dalam proses komposisi, bentuk dasar dapat berupa akar *sate* pada kata *sate ayam*, *sate padang*, dan *sate lontong*; dapat berupa dua buah akar seperti akar *kampung* dan akar *halaman* pada kata *kampung halaman*, atau akar *tua* dan akar *muda* pada kata *tua muda*.

Ada perbedaan bentuk antara *pelajar* dan *pengajar*. Menurut kajian tradisional dan struktural bentuk dasar dari kedua kata itu adalah sama, yaitu akar *ajar*. Dalam kajian proses di sini bentuk dasar kedua kata itu tidaklah sama. Bentuk dasar kata *pelajar* adalah *belajar* sedangkan bentuk dasar kata *pengajar* adalah *mengajar*. Ini dikarenakan makna gramatikal kata *pelajar* adalah 'orang yang belajar' sedangkan makna gramatikal kata *pengajar* adalah 'orang yang mengajar'. Contoh lain, bentuk dasar kata *penyatuan* adalah *menyatukan* karena makna *penyatuan* adalah 'hal/proses menyatukan'. Sedangkan bentuk dasar kata *persatuan* adalah *bersatu* atau *mempersatukan* karena makna gramatikalnya adalah 'hal bersatu' atau 'hal mempersatukan'. Namun, secara teoretis dapat juga dikatakan bentuk dasar kata *pelajar* dan *pengajar* adalah sama yaitu *ajar*; tetapi bentuk *pelajar* dibentuk dari dasar ajar melalui verba *belajar*, sedangkan *pengajar* dibentuk dari dasar *ajar* melalui verba *mengajar*. Demikian juga kata *penyatuan* dibentuk dari dasar *satu* melalui verba *menyatukan*, sedangkan kata *persatuan* dibentuk dari dasar *satu* melalui verba *bersatu* atau *mempersatukan*.

Dari uraian di atas, jelas bahwa konsep *bentuk dasar* tidak sama dengan pengertian *morfem dasar* atau *kata dasar*. Ini dikarenakan bentuk dasar dapat juga berupa bentuk-bentuk polimorfemis.

2.3.2 Pembentuk Kata

Komponen kedua dalam proses morfologi adalah alat pembentuk kata. Sejauh ini alat pembentuk kata dalam proses morfologi adalah (a) afiks dalam proses afiksasi, (b) pengulangan dalam proses reduplikasi, dan (c) penggabungan dalam proses komposisi.

Dalam proses afiksasi sebuah afiks diimbuhkan pada bentuk dasar sehingga hasilnya menjadi sebuah kata. Umpamanya pada dasar *baca* diimbuhkan afiks *me-* sehingga menghasilkan kata *membaca* yaitu sebuah verba transitif aktif; pada dasar *juang* diimbuhkan afiks *ber-* sehingga menghasilkan verba intransitif *berjuang*.

Berkenaan dengan jenis afiksnya, proses afiksasi dibedakan atas *prefiksasi*, yaitu proses pembubuhan prefiks, *konfiksasi* yakni proses pembubuhan konfiks, *sufiksasi* yaitu proses pembubuhan sufiks dan *infiksasi* yakni proses pembubuhan infiks. Perlu dicatat dalam bahasa Indonesia proses infiksasi sudah tidak produktif lagi. Dalam hal ini perlu juga diperhatikan adanya *klofiksasi*, yaitu kelompok afiks yang proses afiksasinya dilakukan bertahap. Misalnya pembentukan kata *menangisi*, mula-mula pada dasar *tangis* diimbuhkan sufiks *-i*; setelah itu baru dibubuhkan prefiks *me-*.

Proses prefiksasi dilakukan oleh prefiks *ber-*, *me-*, *pe-*, *per-*, *di-*, *ter-*, *ke-*, dan *se-*; infiksasi dilakukan oleh infiks *-el-*, *-em-*, dan *-er-*; sufiksasi dilakukan sufiks *-an*, *-kan*, dan *-i*; konfiksasi dilakukan oleh konfiks *pe-an*, *per-an*, *ke-an*, *se-nya*, dan *ber-an*; dan klofiksasi dilakukan oleh klofiks *me-kan*, *me-i*, *memper-*, *memper-kan*, *memper-i*, *ber-kan*, *di-kan*, *di-i*, *diper-*, *diper-kan*, *diper-i*, *ter-kan*, dan *ter-i*.

Alat pembentuk kedua adalah pengulangan bentuk dasar yang digunakan dalam proses reduplikasi. Hasil dari proses reduplikasi ini lazim disebut dengan istilah *kata ulang*. Secara umum dikenal adanya tiga macam pengulangan, yaitu pengulangan secara utuh, pengulangan dengan pengubahan bunyi vokal maupun konsonan, dan pengulangan sebagian.

Alat pembentuk ketiga adalah penggabungan sebuah bentuk pada bentuk dasar yang ada dalam proses komposisi. Penggabungan ini juga merupakan alat yang banyak digunakan dalam pembentukan kata karena banyaknya konsep yang belum ada wadahnya dalam bentuk sebuah kata. Misalnya, bahasa Indonesia hanya punya sebuah kata untuk berbagai macam warna merah. Oleh karena itulah dibentuk gabungan kata seperti *merah jambu*, *merah darah*, dan *merah bata*.

2.4 Morfofonemik

Morfofonemik (disebut juga morfofonologi) adalah kajian mengenai terjadinya perubahan bunyi atau perubahan fonem sebagai akibat dari adanya proses morfologi, baik proses afiksasi, proses reduplikasi, maupun proses komposisi[1]. Morfofonemik dalam pembentukan kata bahasa Indonesia terutama terjadi dalam proses afiksasi. Dalam proses reduplikasi dan komposisi hampir tidak ada. Dalam proses afiksasi pun terutama, hanya dalam prefiksasi *ber-*, prefiksasi *me-*, prefiksasi *pe-*, prefiksasi *per-*, konfiksasi *pe-an*, konfiksasi *per-an*, dan sufiksasi *-an*.

Berikut adalah beberapa jenis perubahan fonem dan bentuk-bentuk morfofonemik pada beberapa proses morfologi.

2.4.1 Jenis Perubahan

Dalam bahasa Indonesia ada beberapa jenis perubahan fonem berkenaan dengan proses morfologi ini. Di antaranya adalah proses:

1. *Pemunculan fonem*, yakni munculnya fonem (bunyi) dalam proses morfologi yang pada mulanya tidak ada. Misalnya, dalam proses pengimbuhan prefiks *me-* pada dasar *baca* akan memunculkan bunyi sengau [m] yang semula tidak ada.
   $me + baca \rightarrow membaca$

2. *Pelesapan fonem*, yakni hilangnya fonem dalam suatu proses morfologi. Misalnya, dalam proses pengimbuhan prefiks *ber-* pada dasar *renang*, maka bunyi [r] yang ada pada prefiks *ber-* dilesapkan. Juga, dalam proses pengimbuhan "akhiran" *-wan* pada dasar *sejarah*, maka fonem /h/ pada dasar *sejarah* itu dilesapkan. Contoh lain, pada proses pengimbuhan "akhiran" *-nda* pada dasar *anak*, maka fonem /k/ pada dasar *anak* dilesapkan atau dihilangkan.
   $ber + renang \rightarrow berenang$
   $sejarah + wan \rightarrow sejarawan$
   $anak + nda \rightarrow ananda$

   Dalam beberapa tahun terakhir ada juga gejala pelesapan salah satu fonem yang sama yang terdapat pada akhir kata dan awal kata yang mengalami proses komposisi. Misalnya.
   $pasar + raya \rightarrow pasaraya$
   $ko + operasi \rightarrow koperasi$

3. *Peluluhan fonem*, yakni luluhnya sebuah fonem serta disenyawakan dengan fonem lain dalam

suatu proses morfologi. Umpamanya, dalam pengimbuhan prefiks *me-* pada dasar *sikat*, maka fonem /s/ pada kata *sikat* itu diluluhkan dan disenyawakan dengan fonem nasal /ny/ yang ada pada prefiks *me-*. Hal yang sama juga terjadi pada proses pengimbuhan prefiks *pe-*.

$me + sikat \rightarrow menyikat$

$pe + sikat \rightarrow penyikat$

4. *Perubahan fonem*, yakni berubahnya sebuah fonem atau sebuah bunyi, sebagai akibat terjadinya proses morfologi. Umpamanya, dalam pengimbuhan prefiks *ber-* pada dasar *ajar* terjadi perubahan bunyi, di mana fonem /r/ berubah menjadi fonem /l/.

   $ber + ajar \rightarrow belajar$

2.4.2 Prefiksasi ber-

Morfofonemik dalam proses pengimbuhan prefiks *ber-* berupa: (a) pelesapan fonem /r/ pada prefiks *ber-*; (b) perubahan fonem /r/ pada prefiks *ber-* menjadi fonem /l/; dan (c) pengekalan fonem /r/ yang terdapat prefiks *ber-* itu.

1. Pelesapan fonem /r/ pada prefiks *ber-* itu terjadi apabila bentuk dasar yang diimbuhi mulai dengan fonem /r/, atau suku pertama bentuk dasarnya berbunyi [er]. Misalnya:

   $ber + renang \rightarrow berenang$

   $ber + ragam \rightarrow beragam$

   $ber + racun \rightarrow beracun$

   $ber + kerja \rightarrow bekerja$

   $ber + ternak \rightarrow beternak$

   $ber + cermin \rightarrow becermin$

2. Perubahan fonem /r/ pada prefiks *ber-* menjadi fonem /l/ terjadi bila bentuk dasarnya akar *ajar*; tidak ada contoh lain.

   $ber + ajar \rightarrow belajar$

3. Pengekalan fonem /r/ pada prefiks *ber-* tetap /r/ terjadi apabila bentuk dasarnya bukan yang ada pada poin 1 dan 2 di atas.

   $ber + obat \rightarrow berobat$

   $ber + korban \rightarrow berkorban$

   $ber + getah \rightarrow bergetah$

$ber + lari \rightarrow berlari$

$ber + tamu \rightarrow bertamu$

2.4.3 Prefiksasi me- (termasuk klofiks me-kan dan me-i)

Morfofonemik dalam proses pengimbuhan dengan prefiks *me-* dapat berupa: (a) pengekalan fonem; (b) penambahan fonem; dan (c) peluluhan fonem.

1. Pengekalan fonem di sini artinya tidak ada fonem yang berubah, tidak ada yang dilesapkan dan tidak ada yang ditambahkan. Hal ini terjadi apabila bentuk dasarnya diawali dengan konsonan /r, l, w, y, m, n, ng, dan ny/. Contoh:

   $me + rawat \rightarrow merawat$

   $me + lirik \rightarrow melirik$

   $me + wasiat \rightarrow wasiat$

   $me + yakin \rightarrow meyakinkan$

   $me + makan \rightarrow memakan$

   $me + nanti \rightarrow menanti$

   $me + nganga \rightarrow nganga$

   $me + nyanyi \rightarrow nyanyi$

2. Penambahan fonem, yakni penambahan fonem nasal /m, n, ng, dan nge/. Penambahan fonem nasal /m/ terjadi apabila bentuk dasarnya dimulai dengan konsonan /b/, /f/, dan /v/. Umpamanya:

   $me + baca \rightarrow membaca$

   $me + buru \rightarrow memburu$

   $me + fitnah \rightarrow memfitnah$

   $me + fokus \rightarrow memfokus(kan)$

   $me + vonis \rightarrow memvonis$

   Penambahan fonem nasal /n/ terjadi apabila bentuk dasarnya dimulai dengan konsonan /c/, /d/, dan /j/. Umpamanya:

   $me + cari \rightarrow mencari$

   $me + dengar \rightarrow mendengar$

   $me + jual \rightarrow menjual$

   Penambahan fonem nasal /ng/ terjadi apabila bentuk dasarnya dimulai dengan konsonan /g,

h, dan kh/ dan huruf vokal /a, i, u, e, dan o/. Contoh:

$me + goda \rightarrow menggoda$

$me + hina \rightarrow menghina$

$me + khayal \rightarrow mengkhayal$

$me + ambil \rightarrow mengambil$

$me + iris \rightarrow mengiris$

$me + ukur \rightarrow mengukur$

$me + elak \rightarrow mengelak$

$me + obral \rightarrow mengobral$

Penambahan fonem nasal /nge/ terjadi apabila bentuk dasarnya hanya terdiri dari satu suku kata. Misalnya:

$me + bom \rightarrow mengebom$

$me + cat \rightarrow mengecat$

$me + lap \rightarrow mengelap$

3. Peluluhan fonem terjadi apabila prefiks *me-* diimbuhkan pada bentuk dasar yang dimulai dengan konsonan bersuara /s, k, p, dan t/. Dalam hal ini konsonan /s/ diluluhkan dengan nasal /ny/, konsonan /k/ diluluhkan dengan nasal /ng/, konsonan /p/ diluluhkan dengan nasal /m/, dan konsonan /t/ diluluhkan dengan nasal /n/. Contoh:

$me + sikat \rightarrow menyikat$

$me + kirim \rightarrow mengirim$

$me + pilih \rightarrow memilih$

$me + tolong \rightarrow menolong$

2.4.4 Prefiksasi pe- dan konfiksasi pe-an

Morfofonemik dalam proses pengimbuhan dengan prefiks *pe-* dan konfiks *pe-an* sama dengan morfofonemik yang terjadi dalam proses pengimbuhan dengan prefiks *me-*, yaitu (a) pengekalan fonem; (b) penambahan fonem; dan (c) peluluhan fonem.

1. Pengekalan fonem, artinya tidak ada perubahan fonem, dapat terjadi apabila bentuk dasarnya diawali dengan konsonan /r, l, y, w, m, n, ng, dan ny/. Contoh:

$pe + rawat \rightarrow perawat$

$pe + latih \rightarrow pelatih$

$pe + yakin \rightarrow peyakin$

$pe + waris \rightarrow pewaris$

$pe - an + manfaat \rightarrow pemanfaatan$

$pe - an + nanti \rightarrow penantian$

$pe + nganga \rightarrow penganga$

$pe + nyanyi \rightarrow penyanyi$

2. Penambahan fonem, yakni penambahan fonem nasal /m, n, ng, dan nge/ antara prefiks dan bentuk dasar. Penambahan fonem nasal /m/ terjadi apabila bentuk dasarnya diawali oleh konsonan /b/. Contoh:

$pe + baca \rightarrow pembaca$

$pe + bina \rightarrow pembina$

$pe + buru \rightarrow pemburu$

Penambahan fonem nasal /n/ terjadi apabila bentuk dasarnya diawali oleh konsonan /c/, /d/, dan /j/. Contoh:

$pe + cari \rightarrow pencari$

$pe + dengar \rightarrow pendengar$

$pe + jual \rightarrow penjual$

Penambahan fonem nasal /ng/ terjadi apabila bentuk dasarnya diawali dengan konsonan /g, h, dan kh/ dan vokal /a, i, u, e, o/. Contoh:

$pe + gali \rightarrow penggali$

$pe + hambat \rightarrow penghambat$

$pe + khianat \rightarrow pengkhianat$

$pe + angkat \rightarrow pengangkat$

$pe + inap \rightarrow penginap$

$pe + usir \rightarrow pengusir$

$pe + elak \rightarrow pengelak$

$pe + obral \rightarrow pengobral$

Penambahan fonem nasal /nge/ terjadi apabila bentuk dasarnya berupa bentuk dasar satu suku. Contoh:

$pe + bom \rightarrow pengebom$

$pe + cat \rightarrow pengecat$

$pe + lap \rightarrow pengelap$

3. Peluluhan fonem, apabila prefiks *pe-* (atau *pe-an*) diimbuhkan pada bentuk dasar yang diawali dengan konsonan tak bersuara /s, k, p, dan t/. Dalam hal ini konsonan /s/ diluluhkan dengan nasal /ny/, konsonan /k/ diluluhkan dengan nasal /ng/, konsonan /p/ diluluhkan dengan nasal /m/, dan konsonan /t/ diluluhkan dengan nasal /n/. Contoh:
   $pe + saring \rightarrow penyaring$
   $pe + kumpul \rightarrow pengumpul$
   $pe + pilih \rightarrow pemilih$
   $pe + tulis \rightarrow penulis$

2.4.5 Prefiksasi per- dan konfiksasi per-an

Morfofonemik dalam pengimbuhan prefiks *per-* dan konfiks *per-an* dapat berupa: (a) pelesapan fonem /r/ pada prefiks *per-* itu; (b) perubahan fonem /r/ dari prefiks *per-* itu menjadi fonem /l/; dan (c) pengekalan fonem /r/ tetap /r/.

1. Pelesapan fonem /r/ terjadi apabila bentuk dasarnya dimulai dengan fonem /r/, atau suku kata pertamanya /er/. Contoh:
   $per + ringan \rightarrow peringan$
   $per + rendah \rightarrow perendah$
   $per + ternak \rightarrow peternak$
   $per + kerja \rightarrow pekerja$

2. Perubahan fonem /r/ menjadi /l/ terjadi apabila bentuk dasarnya berupa kata *ajar*.
   $per + ajar \rightarrow pelajar$

3. Pengekalan fonem /r/ terjadi apabila bentuk dasarnya bukan yang disebutkan pada poin 1 dan 2 di atas. Contoh:
   $per + kaya \rightarrow perkaya$
   $per + kecil \rightarrow perkecil$
   $per + lambat \rightarrow perlambat$
   $per + tegas \rightarrow pertegas$

2.4.6 Prefiksasi ter-

Morfofonemik dalam proses pengimbuhan dengan prefiks *ter-* dapat berupa: (a) pelesapan fonem /r/ dari prefiks *ter-* itu; (b) perubahan fonem /r/ dari prefiks *ter-* itu menjadi fonem /l/; dan (c) pengekalan fonem /r/ itu.

1. Pelesapan fonem dapat terjadi apabila prefiks *ter-* diimbuhkan pada bentuk dasar yang dimulai dengan konsonan /r/. Misalnya:
   $ter + rasa \rightarrow terasa$
   $ter + rangkum \rightarrow terangkum$
   $ter + rebut \rightarrow terebut$

2. Perubahan fonem /r/ pada prefiks *ter-* menjadi fonem /l/ terjadi apabila prefiks *ter-* itu diimbuhkan pada bentuk dasar *anjur*.
   $ter + anjur \rightarrow telanjur$

3. Pengekalan fonem /r/ pada prefiks *ter-* tetap menjadi /r/ apabila prefiks ter- itu diimbuhkan pada bentuk dasar yang bukan disebutkan pada poin 1 dan 2 di atas. Contoh:
   $ter + dengar \rightarrow terdengar$
   $ter + jauh \rightarrow terjauh$
   $ter + lempar \rightarrow terlempar$
   $ter + baik \rightarrow terbaik$

2.5 Peran Morphological Parser dalam Natural Language Processing[2]

Dalam bahasa Inggris, ketika kita ingin menulis sebuah kata benda plural, untuk sebagian besar kasus kita hanya tinggal menambahkan huruf 's' di belakang kata benda yang dimaksud. Misalnya, kita ingin membuat bentuk plural dari kata 'book' kita hanya perlu menambahkan huruf 's' di belakangnya sehingga menjadi kata 'books' yang merupakan bentuk plural dari 'book'. Namun, hal itu tidak berlaku jika kita ingin membuat bentuk plural dari kata 'baby', 'goose', atau 'fish'. Bentuk plural dari kata 'baby', 'goose', dan 'fish' adalah 'babies', 'geese', dan 'fish'.

Untuk mengenali bentuk 'books' dapat dipisahkan menjadi dua buah morfem 'book' dan 's' diperlukan proses yang disebut dengan **morphological parsing**. Dalam ranah ilmu temu kembali informasi (*information retrieval*), proses yang sama untuk memetakan bentuk 'books' menjadi 'book' disebut dengan *stemming*. Morphological parsing atau stemming tidak hanya dapat memisahkan

bentuk plural dari kata benda, tapi juga dapat untuk bentuk lain seperti bentuk kata kerja berakhiran 'ing' dalam contoh kata 'going', 'talking', dan 'eating'. Ketika dilakukan proses parsing terhadap bentuk tersebut akan didapatkan kata kerja 'go', 'talk', dan 'eat' yang ditambahkan morfem 'ing'.

Muncul pertanyaan, kenapa kita tidak simpan saja semua bentuk plural dari semua kata benda dan semua bentuk 'ing' dari semua kata kerja dalam bahasa Inggris yang ada dalam kamus? Alasan utamanya adalah karena bentuk 'ing' adalah sufiks yang sangat produktif, yang berarti bentuk tersebut dapat diterapkan pada semua kata kerja. Sama halnya dengan bentuk plural 's' yang bisa diterapkan di hampir semua kata benda. Oleh karena itu, ide untuk menyimpan semua bentuk plural dan bentuk 'ing' akan sangat tidak efisien.

Morphological parsing dibutuhkan tidak hanya untuk temu kembali informasi. Kita membutuhkan proses tersebut untuk supaya mesin dapat mengerti bahwa kata 'va' dan 'aller' dalam bahasa Perancis keduanya diterjemahkan menjadi kata kerja 'go' dalam bahasa Inggris. Kita juga membutuhkan proses tersebut untuk mengecek ejaan, karena dengan aturan morfologi kita dapat menentukan bahwa kata 'misclam' dan 'antiundoggingly' bukan merupakan kata yang valid dalam bahasa Inggris.

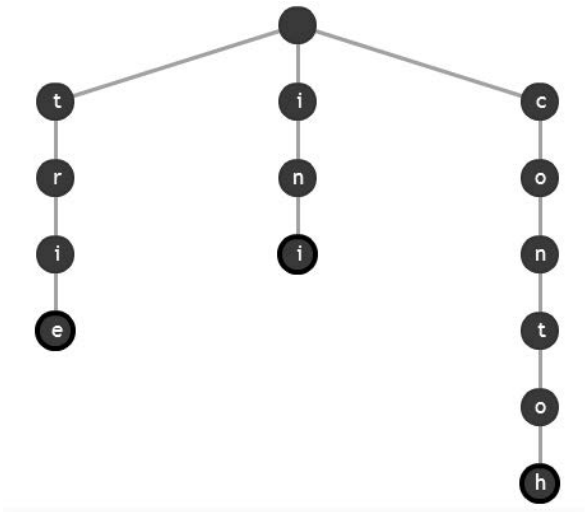
2.6 Finite-State Automata

Automata adalah ...

2.7 Struktur Data Trie

Trie adalah struktur data berupa pohon terurut untuk menyimpan suatu himpunan string di mana setiap node pada pohon tersebut mengandung awalan (prefix) yang sama[3]. Kata "trie" berasal dari kata *retrieval* yang berarti pengambilan. Struktur data trie ditemukan oleh seorang professor di MIT bernama Edward Fredkin. Trie sering digunakan pada masalah komputasi yang melibatkan penyimpanan dan pencarian string. Trie memiliki sejumlah keunggulan dibanding struktur data lain untuk memecahkan masalah serupa terutama dalam hal kecepatan dan memori yang digunakan.

Dalam trie, tidak ada node yang menyimpan kunci yang terkait dengan node tersebut, sebaliknya, posisinya di pohon menunjukkan kunci apa yang terkait dengannya. Setiap keturunan dari sebuah node memiliki prefix yang sama dengan string yang diwakilkan oleh node tersebut, dan akar menandakan sebuah string kosong.



Gambar 2.3: Trie dengan kata "trie","ini", dan "contoh"[3]

Gambar 2.3 di atas adalah contoh representasi struktur data trie yang menyimpan tiga buah string, yaitu "trie","ini", dan "contoh". Dari gambar tersebut kita bisa mendapat gambaran mengenai kompleksitas waktu yang diperlukan untuk mencari sebuah kata dalam trie. Jika panjang kata terpanjang dalam trie adalah L, maka untuk mencari sebuah kata dalam trie memerlukan waktu terburuk O(L)

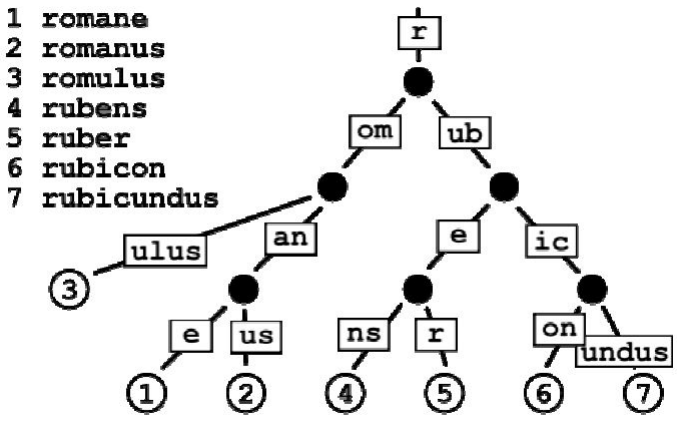
2.7.1 Bitwise Trie

Bitwise trie adalah salah satu variasi dari trie yang memiliki banyak kesamaan dengan trie berbasis karakter biasa, kecuali dalam representasi dengan bit individual yang biasanya digunakan untuk traversal secara efektif dan membentuk sebuah pohon biner. Secara umum, implementasinya menggunakan fungsi khusus CPU untuk dapat secara cepat mencari himpunan bit dengan panjang tertentu. Nilai ini lalu akan digunakan sebagai entri dari tabel dengan indeks 32 atau 64 yang menunujuk kepada elemen pertama dalam bitwise trie dengan sejumlah bilangan 0 di depan. Proses pencarian selanjutnya akan dilakukan dengan mengetes setiap bit dalam kunci dan memilih anak[0] atau anak[1] sesuai aturan hingga pencarian berakhir.

Walaupun proses ini mungkin terdengar lambat, tetapi sangat fleksibel karena kurangnya ketergantungan terhadap *register* dan oleh karena itu pada kenyataannya melakukan eksekusi dengan sangat baik pada CPU modern.

2.7.2 Patricia Trie

PATRICIA adalah variasi lain dari trie yang merupakan singkatan dari Practical Algorithm To Retrieve Information Coded In Alphanumeric. PATRICIA trie sendiri lebih dikenal dengan sebutan *pohon radix* atau *radix tree*. Pohon radix bisa diartikan secara sederhana sebagai trie yang kompleksitas ruangnya lebih efisien, di mana setiap node yang hanya memiliki satu anak digabung dengan anaknya sendiri. Hasilnya adalah setiap node paling dalam paling tidak memiliki 2 anak. Tidak seperti trie biasa, anak bisa diberi label deretan karakter maupun satu karakter. Ini membuat pohon radix jauh lebih efisien untuk jumlah string yang sedikit (terutama jika stringnya cukup panjang) dan untuk himpunan string yang memiliki prefix sama yang panjang.



Gambar 2.4: Pohon radix dengan 7 kata dengan prefix "r"[3]

Pohon radix memiliki fasilitas untuk melakukan operasi-operasi berikut, yang mana setiap operasinya memiliki kompleksitas waktu terburuk O(k), di mana k adalah panjang maksimum string dalam himpunan.

- Pencarian: Mencari keberadaan suatu string pada himpunan string. Operasi ini sama dengan pencarian pada trie biasa kecuali beberapa sisi mengandung lebih dari satu karakter.
- Penyisipan: Menambahkan sebuah string ke pohon. Kita mencari tempat yang tepat di pohon untuk menyisipkan elemen baru. Jika sudah ada sisi yang memiliki prefix sama dengan string masukan, kita akan memisahkan nya menjadi dua sisi dan memprosesnya. Proses pemisahan ini meyakinkan bahwa tidak ada node yang memiliki anak lebih banyak dari jumlah karakter string yang ada.
- Hapus: Menghapus sebuah string dari pohon. Pertama kita menghapus daun yang berkaitan.

Lalu, jika orangtuanya hanya memiliki satu anak lagi, kita menghapus orangtuanya dan menggabungkan sisi yang saling terhubung

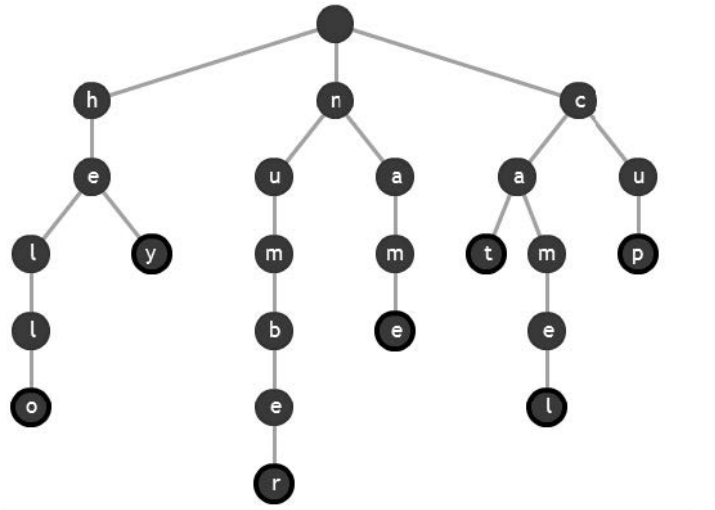
- Cari anak: Mencari string terbesar yang lebih kecil dari string masukan, sesuai dengan urutan alfabet.
- Cari orang tua: Mencari string terkecil yang lebih besar dari string masukan, sesuai dengan urutan alfabet.

Pengembangan yang umum dari pohon radix yaitu menggunakan node dua warna, hitam dan putih. Untuk mengecek apakah sebuah string masukan sudah ada di dalam pohon, pencarian dimulai dari puncak, dan terus menelusuri setiap sisi sampai tidak ada lagi jalan. Jika node akhir dari proses ini berwarna hitam, berarti pencarian gagal, jika node berwarna putih berarti pencarian telah berhasil. Hal ini membuat kita bisa menambahkan string dalam jumlah banyak yang memiliki prefix yang sama dengan elemen di pohon dengan menggunakan node putih, lalu menghapus sejumlah pengecualian untuk mengehemat memori dengan cara menambahkan elemen string baru dengan node hitam.

2.7.3 Implementasi Trie dalam Kamus

Salah satu implementasi dari struktur data trie yang paling populer adalah dalam kamus. Kamus terdiri dari kumpulan kata-kata yang sudah terurut menaik berdasarkan urutan alfabet. Dalam perkembangannya saat ini sudah banyak kamus yang hadir dalam bentuk perangkat lunak, yang bisa digunakan di komputer ataupun di telepon genggam. Kamus dalam bentuk perangkat lunak tentunya memiliki fitur-fitur yang memudahkan pengaksesannya, antara lain pencarian kata dan penambahan kata ke kamus.

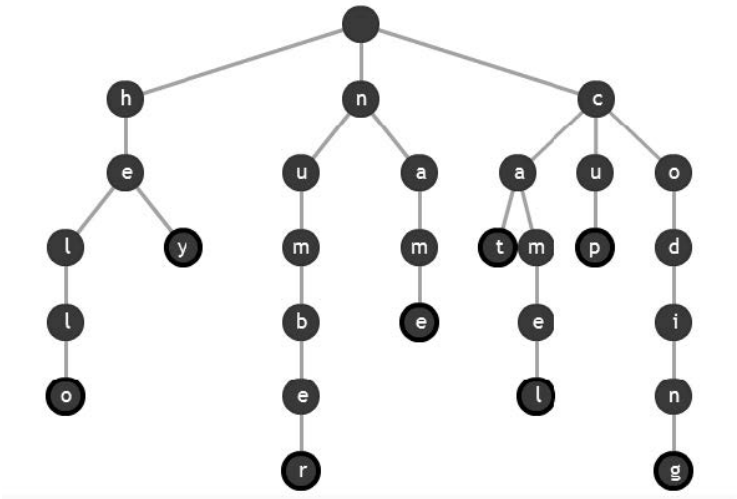
Berikut adalah ilustrasi kamus dengan struktur data trie.



Gambar 2.5: Trie dengan kata "hello","hey","number","name","cat","camel", dan "cup"[3]

Dapat dibayangkan, ketika kita mencari sebuah kata dalam kamus, kita akan mulai dengan karakter pertama dari kata tersebut. Jika tidak ada anak dari akar yang nilainya sama dengan karakter itu, maka langsung disimpulkan kata tidak ada di kamus. Jika ada, akan ditelusuri terus sampai ke dasar dari trie, jika kata ditemukan, maka akan dikembalikan info dari node itu, sedangkan jika tidak ketemu kita bisa mengembalikan kata yang memiliki prefix sama dengan kata yang dicari sebagai saran pencarian.

Sementara itu penambahan kata akan berakibat penambahan cabang baru bila kata itu belum ada sebelumnya.



Gambar 2.6: Penambahan kata "coding" pada trie di gambar 2.5[3]

2.7.4 Keunggulan Trie dibandingkan Struktur Data Lain

Trie sebagai turunan dari pohon memiliki keunggulan dibandingkan stuktur data yang sering digunakan untuk persoalan yang sama, yakni pohon pencarian biner dan tabel hash.

Berikut keunggulan utama trie dibandingkan pohon pencarian biner:

- Pencarian kunci dengan trie lebih cepat. Mencari sebuah kunci dengan panjang m menghabiskan waktu dengan kasus terburuk O(m). Pohon pencarian biner melakukan O(log(n)) perbandingan kunci, dimana n adalah jumlah elemen di dalam pohon, karena pencarian pada pohon biner bergantung pada kedalaman pohon, yang mana bernilai logaritmik terhadap jumlah kunci pencarian apabila pohonnya seimbang. Oleh karena itu pada kasus terburuk, sebuah pohon biner menghabiskan waktu O(m log n), yang mana pada kasus terburuk juga log n akan mendekati m. Operasi sederhana yang digunakan trie pada saat pencarian karakter, seperti penggunaan array index menggunakan karakter, juga membuat pencarian dengan trie menjadi lebih cepat.

- Trie menggunakan ruang lebih sedikit jika memuat string pendek dalam jumlah besar, karena kunci tidak disimpan secara eksplisit dan node dipakai bersama oleh kunci yang memiliki prefix yang serupa.

- Trie bisa memiliki fitur untuk menghitung kesamaan prefix terpanjang, yang membantu untuk mencari pengunaan kunci bersama terpanjang dari karakter-karakter yang unik.

Berikut keunggulan utama trie dibanding tabel hash:

- Trie bisa melakukan pencarian kunci yang paling mirip hampir sama cepatnya dengan pencarian kunci yang tepat, sementara tabel hash hanya bisa mencari kunci yang sama tepat karena tidak menyimpan hubungan antara kunci.

- Trie lebih cepat secara rata-rata untuk menyisipkan elemen baru dibandingkan dengan tabel hash. Hal ini terjadi karena tabel hash harus membangun ulang indeksnya ketika tabel sudah penuh, yang mana menghabiskan waktu sangat banyak. Oleh karena itu, trie memiliki kompleksitas waktu terburuk yang batasnya lebih konsisten, yang mana merupakan salah satu unsur penting pada jalannya sebuah program.

- Trie bisa diimplementasikan sedemikian sehingga tidak memerlukan memori tambahan. Tabel hash harus selalu memliki memori tambahan untuk menyimpan pengindeksan tabel hash.

- Pencarian kunci bisa jauh lebih cepat jika fungsi hashing dapat dihindarkan. Trie bisa menyimpan kunci bertipe integer maupun pointer tanpa perlu membuat fungsi hashing sebelumnya. Hal ini membuat trie lebih cepat daripada tabel hash pada hampir setiap kasus karena fungsi hash yang baik sekalipun cenderung overhead ketika melakukan hashing pada data yang hanya berukuran 4 sampai 8 byte.

- Trie bisa menghitung kesamaan prefix terpanjang, sedangkan tabel hash tidak.

BAB 3

ANALISIS

Pada bab ini dijelaskan mengenai ....

3.1 Analisis Perangkat Lunak Serupa

Pada tahun 2008, Pisceldo dkk pernah membuat sebuah artikel yang berjudul "A Two-Level Morphological Analyser for the Indonesian Language"[4]. Artikel ini berisi tentang usaha pembuatan perangkat lunak *morphological analyser* untuk bahasa Indonesia melalui pendekatan *two-level*. Perangkat lunak ini dapat memproses kata yang merupakan hasil afiksasi dan reduplikasi.

Proses morphological analysis mengambil informasi berupa kategori gramatikal dari sebuah kata berdasarkan aturan morfologi yang ada. Pada penelitian ini dipakai *two-level morphology approach*, prosesnya dipecah menjadi menentukan aturan morfotaktik dan morfofonemik. Aturan ini dimodelkan ke dalam network *finite-state transducers* dan diimplementasikan dengan **xfst** dan **lexc**. Sistem yang dihasilkan dari penelitian ini bisa menangani reduplikasi, proses morfologi yang tidak memerlukan penggabungan kata (non-concatenative).

3.1.1 Desain

Desain perangkat lunak dibagi dalam dua komponen:

- Morfotaktik: menentukan kelas morfem mana yang bisa mengikuti kelas morfem lain dalam sebuah kata
- Morfofonemik: memodelkan perubahan fonologi yang terjadi pada proses pembentukan kata

Adapun desain leksikon yang digunakan terdiri dari 4 kelas yaitu verbs, nouns, adjectives, dan 'etc' (pronouns, adverbs, numbers, dan particles)

Desain tag yang digunakan:

- Normal tags: $+VERB, +NOUN, +ADJ, +BAREVERB, +BARENOUN, +BAREADJ, +BAREETC, +AV, +PASS, +UV$, dan $+REDUP$
- Special tags: $+CAUS\_KAN, +APPL\_KAN, +CAUS\_I, +APPL\_I, +ACTOR, +INSTRUMENT$

Detailnya adalah sebagai berikut:

- Tag seperti $+VERB, +NOUN$, dan $+ADJ$ menandai kata yang sudah melalui proses morfologi
- Tag seperti $+BAREVERB, +BARENOUN, +BAREADJ$, dan $+BAREETC$ menandai kata dasar
- Tag seperti $+AV, +PASS$, dan $+UV$ menandai kategori gramatikal, $+AV$ untuk active voice, $+PASS$ untuk passive voice, dan $+UV$ untuk undergoer voice
- Tag $+REDUP$ untuk menandai reduplikasi
- Tag $+CAUS\_KAN, +APPL\_KAN, +CAUS\_I$, dan $+APPL\_I$ menandai kata tersebut merupakan causative atau applicative dengan melihat akhirannya
- Tag $+ACTOR$ dan $+INSTRUMENT$ menandai kata tersebut membawa makna actor atau instrument

Aturan morfotaktik untuk bahasa Indonesia ada 13 aturan, 10 aturan untuk pembubuhan afiks (meN-, peN-, di-, per-, ber-, ter-, ke-, -an, -kan, dan -i), dan 3 aturan untuk reduplikasi (utuh, sebagian, berimbuhan).

Contoh aturan morfotaktik pembubuhan afiks:

- membersihkan (meN+bersih+kan). Kata bersih adalah adjective, setelah digabung dengan meN-kan hasilnya adalah verb.
- pembelajaran (peN+ber+ajar+an). Kata ajar adalah verb, setelah digabung dengan peN-ber-an, hasilnya adalah noun.
- keberhasilan (ke+ber+hasil+an). Kata hasil adalah noun, setelah digabung dengan ke-ber-an, hasilnya tetap noun.
- terangi (terang+i). Kata terang adalah adjective, setelah digabung dengan -i, hasilnya adalah verb.

Contoh aturan morfotaktik reduplikasi:

- buku-buku. Kata buku-buku, adalah noun, dihasilkan dari kata buku, yang juga adalah sebuah noun.
- kekayaan-kekayaan (reduplikasi dari ke+kaya+an). Kata kaya adalah adjective. Setelah mengalami proses morfologi, hasilnya adalah noun.
- berlari-lari (ber+lari-lari). Kata lari adalah verb. Setelah mengalami proses morfologi, hasilnya adalah juga verb.
- tanam-menanam (tanam-meN+tanam). Kata tanam adalah verb. Setelah mengalami proses morfologi, hasilnya adalah juga verb.

Selama tahap penerapan aturan morfotaktik, ada beberapa langkah yang harus dilakukan, yaitu penambahan prefiks dan preprefiks, penambahan stems dan part-of-speech, penambahan sufiks, dan proses akhir berupa penambahan tags.

Aturan morfofonemik untuk bahasa Indonesia dibagi dalam dua kelompok, ada 4 aturan untuk memodelkan perubahan pada kata dasar dan ada 7 aturan untuk memodelkan perubahan pada afiks.

Aturan pada perubahan kata dasar:

- Mengganti /k/ menjadi /ng/ jika mendapat awalan meN- atau peN-. Contoh: meN+kantuk menjadi mengantuk.
- Mengganti /s/ menjadi /ny/ jika mendapat awalan meN- atau peN-. Contoh: peN+sebaran menjadi penyebaran.
- Mengganti /p/ menjadi /m/ jika mendapat awalan meN- atau peN-. Contoh: peN+pakai menjadi pemakai.
- Mengganti /t/ menjadi /n/ jika mendapat awalan meN- atau peN-. Contoh: meN+tertawakan menjadi menertawakan.

Aturan pada perubahan afiks:

- Penghapusan /N/ jika awalan meN- diikuti /l/, /m/, /n/, /r/, /y/, /w/, /t/, /s/, /p/, /k/ atau jika awalan peN diikuti /l/, /m/, /n/, /r/, /d/, /w/, /t/, /s/, /p/, /k/. Contoh: meN+lukis menjadi melukis.

- Penghapusan /r/ jika awalan ber-, ter-, atau per- diikuti oleh /r/ atau kata yang suku pertamanya berakhir dengan /er/. Contoh: ber+runding menjadi berunding.
- Penggantian /N/ dengan /n/ jika meN- diikuti oleh /d/, /c/, /j/, /sy/ atau jika peN- diikuti /d/, /c/, /j/. Contoh: peN+jual menjadi penjual.
- Penggantian /N/ dengan /m/ jika meN- atau peN- diikuti oleh /b/, /f/. Contoh: peN+buru menjadi pemburu.
- Penggantian /N/ dengan /nge/ jika meN- diikuti oleh kata dengan satu suku. Contoh: meN+rem menjadi mengerem.
- Penggantian /N/ dengan /l/ jika peN- diikuti dengan kata ajar. Contoh: peN+ajar menjadi pelajar.
- Penggantian /r/ dengan /l/ jika ber- diikuti dengan kata ajar. Contoh: ber+ajar menjadi belajar.

Desain dari aturan morfofonemik untuk reduplikasi secara umum sama dengan afiksasi karena proses morfofonemik pada reduplikasi terjadi saat proses afiksasi dilakukan.

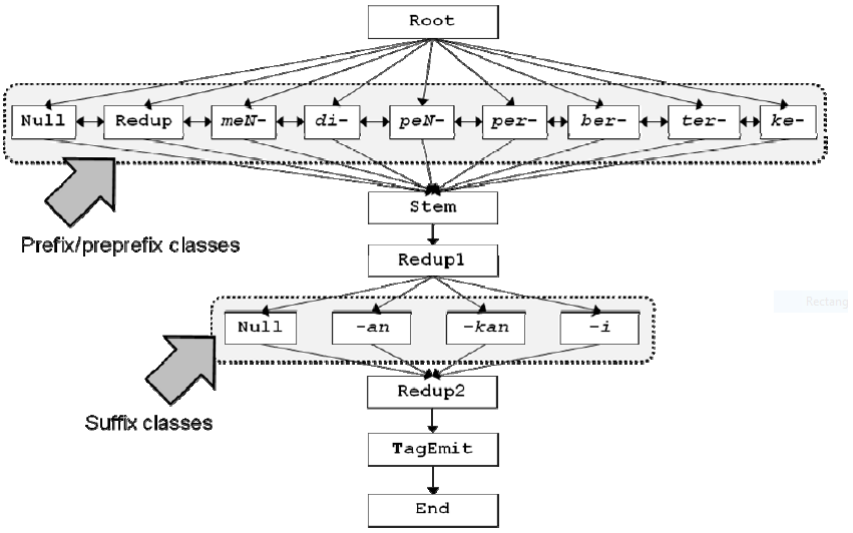
Pada kasus di mana perubahan terjadi pada afiks dan kata dasar sekaligus, perlu ada beberapa penyesuaian aturan untuk digunakan pada bentuk reduplikasi. Contoh pada aturan morfofonemik penggantian /k/ menjadi /ng/ dan penghapusan /N/ yang bekerja bersamaan, awalnya adalah penghapusan /k/ dan penggantian /N/ dengan /ng/. Namun, pendekatan ini tidak bekerja jika melibatkan reduplikasi. Kata mengotak-ngotakkan tidak akan dianalisis dengan tepat jika morphological analyser menggunakan aturan penghapusan /k/ dan penggantian /N/ dengan /ng/. Kata mengotak-ngotakkan dihasilkan dari kata dasar kotak yang dimodifikasi dengan reduplikasi berafiks meN-kan. Jika penghapusan /k/ dan penggantian /N/ dengan /ng/ digunakan, hasil dari proses tersebut adalah kata mengotak-otakkan yang mana tidak valid. Ini adalah alasan dalam reduplikasi aturannya diubah menjadi penggantian /k/ menjadi /ng/ dan penghapusan /N/ sehingga dihasilkan kata mengotak-ngotakkan.

3.1.2 Implementasi

Morphological analyser bahasa Indonesia ini diimplementasikan menggunakan **xfst** and **lexc**. Aturan morfotaktik diimplementasikan dengan **xfst** sementara aturan morfofonemik diimplementasikan dengan **lexc**.

Implementasi aturan morfotaktik dapat diilustrasikan dengan *finite-state automata* yang ditunjukkan pada gambar 3.1. Kata bahasa Indonesia yang valid, yaitu yang dihasilkan melalui proses morfologi yang benar, dapat diterima oleh automata, sementara yang tidak valid ditolak.

Berawal dari root, setiap state menentukan kemungkinan lanjutan state selagi mengambil simbol tertentu. Dalam **lexc**, state ini disebut *continuation classes*. Semua continuation classes yang bisa dicapai dari root menampilkan prefiks dan preprefiks. Perbedaan di antara keduanya diperlukan untuk mengenali variasi morfologi berupa adanya 2 prefiks, seperti prefiks memper-, diper-. Dari sana, continuation class selanjutnya adalah stem, di mana kata dasar diambil. Ini diteruskan dengan beberapa classes yang menampilkan kemungkinan sufiks, namun ada juga class Redup1 dan Redup2 yang muncul sebelum dan setelah sufiks. Fungsinya adalah untuk menangani reduplikasi. Terakhir, tagEmit class memproses semua tags yang belum ditangani oleh class sebelumnya.



Gambar 3.1: Ilustrasi proses morfotaktik[4]

Sudah dibahas di atas bahwa morfologi bahasa Indonesia mengandung proses reduplikasi. Sangat sulit menangani ini dengan regular grammar yang diimplementasikan dengan finite-state automata. Oleh karena itu, digunakan fitur compile-replace dalam **xfst**. Fitur ini memperbolehkan pengulangan bahasa yang kompleks dengan menggunakan tanda "^[" dan "^]" untuk menandai bagian reduplikasi. Tanda kurung siku kanan juga ditambah dengan ^2 untuk menandakan duplikasi; sehingga lengkapnya adalah "^[" dan "^2^]". Dengan definisi network ini, **xfst** melakukan compiles dan post-processes dari keterangan ini untuk menghasilkan network baru yang bisa mengenali reduplikasi. Contoh, "^[*buku*^2^]" akan dicompile menjadi bukubuku.

Idenya adalah untuk memasukkan "^[" dan "^2^]" pada posisi yang tepat. Karena banyaknya

tipe reduplikasi dalam bahasa Indonesia, aturan reduplikasi dapat ditemukan pada state Redup (pre)prefix dan state Redup1 dan Redup2. State prefix Redup mengambil tanda kurung siku "^[" dan mengeset flag yang tepat sebagai pengingat bahwa tanda kurung siku kanan dibutuhkan. State Redup1 bertanggung jawab untuk menutup reduplikasi sebagian dan reduplikasi berafiks, contoh di mana sufiks tidak diikutkan dalam reduplikasi, sementara state Redup2 bertanggung jawab untuk menutup reduplikasi utuh, contoh di mana sufiks menjadi bagian dari proses reduplikasi. Baik state Redup1 dan Redup2 mengecek nilai dari flag REDUP yang diset pada prefix state Redup.

Transducer yang utuh terdiri dari aturan morfotaktik dan morfofonemik sehingga keluaran dari implementasi aturan morfotaktik menjadi masukan bagi implementasi aturan morfofonemik. Dari contoh sebelumnya, keluaran dari implementasi aturan morfotaktik adalah me^Npukuli dan menjadi masukan bagi implementasi aturan morfofonemik.

Berbeda dengan implementasi aturan morfotaktik yang bisa direpresentasikan dengan diagram alur proses, implementasi dari aturan morfofonemik berisi aturan-aturan perubahan dan penggantian fonem seperti yang sudah dipaparkan sebelumnya. Pada gambar 3.2 berikut adalah contoh implementasi dari aturan 'RG4' yang mengkodekan penghapusan /N/ dan penggantian 4 jenis fonem pada kata dasar:

```
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 3.2: Implementasi aturan morfofonemik[4]

Aturan ini terdiri dari 5 aturan yang bekerja secara paralel. Aturan pertama adalah penghapusan /N/ pada prefiks jika /me/ atau /pe/ diikuti oleh kata dasar dengan fonem awal /l/, /m/, /n/, /r/, /y/, /w/, /t/, /s/, /p/, atau /k/.

Aturan selanjutnya terdiri dari 4 aturan perubahan fonem awal pada kata dasar yang diberikan prefiks /me^N/ atau /pe^N/, dan fonem awal pada kata dasarnya adalah /t/, /p/, /s/, atau /k/. Lebih detail, fonem /t/ akan diganti dengan /n/, fonem /p/ diganti dengan /m/, fonem /s/ diganti dengan /ny/, dan fonem /k/ diganti dengan /ng/.

Contoh masukan me^Npukuli pada proses sebelumnya memenuhi aturan penghapusan /N/ dan penggantian fonem awal /p/ dari kata dasar dengan fonem /m/. Kata me^Npukuli akan diproses menjadi kata memukuli. Kata tersebut merupakan kata yang valid dalam bahasa Indonesia dan proses berakhir di sini.

Pengujian dilakukan dengan menjalankan tes kasus dengan kata yang diambil dari Kamus Besar Bahasa Indonesia versi elektronik. Pengujian dilakukan pada implementasi dari aturan morfotaktik dan morfofonemik secara terpisah. Untuk menguji kemampuan dari analyser untuk menerima masukan yang valid dan tidak valid, tes kasus yang digunakan mengandung penulisan kombinasi morfem yang valid dan tidak valid. Hasil dari pengujian ditampilkan pada tabel 3.1, yang menunjukkan hasil pengujian morfotaktik, dan tabel 3.2, yang menunjukkan hasil pengujian morfofonemik. Kolom Analisis menampilkan hasil dari sistem yang memproses penguraian struktur morfologi dari kata dalam bahasa Indonesia. Contoh, diberikan masukan kata memukul, sistem diharapkan menghasilkan keluaran pukul+Verb+AV. Sementara, kolom Sintesis menangani situasi yang sebaliknya, di mana masukannya adalah tag morfologi dan sistem diharapkan menghasilkan keluaran berupa kata yang valid dalam bahasa Indonesia.

Hasil	**Analisis**	**Sintesis**	**Jumlah**
1. Hasil benar	103	43	146
2. Beberapa hasil, ada yang benar	46	106	152
3. Hasil salah	3	3	6
Jumlah	152	152	308

Tabel 3.1: Hasil pengujian morfotaktik[4]

Hasil	**Analisis**	**Sintesis**	**Jumlah**
1. Hasil benar	51	21	72
2. Beberapa hasil, ada yang benar	6	36	42
3. Hasil salah	1	1	2
Jumlah	58	58	116

Tabel 3.2: Hasil pengujian morfofonemik[4]

Ada tiga kategori hasil pengujian, kategori pertama adalah ketika sistem menghasilkan sebuah hasil analisis atau sintesis yang benar atau tidak menghasilkan apapun untuk tes kasus yang tidak valid. Kategori kedua adalah ketika sistem diberikan masukan valid dan menghasilkan beberapa jawaban, yang salah satunya adalah jawaban yang diharapkan. Terakhir, kategori ketiga adalah ketika sistem gagal menganalisis atau menyintesis tes kasus yang valid atau tidak memproduksi jawaban yang tepat ketika diberikan tes kasus yang tidak valid.

Dari tabel, dapat dilihat bahwa hasil untuk proses analisis lebih baik dari proses sintesis, di mana sistem cenderung memproduksi lebih dari satu kemungkinan jawaban. Contoh, sistem ini mampu menghasilkan kata memukul pada proses analisis, namun ketika proses sintesis untuk pukul+Verb+AV, sistem menghasilkan beberapa kemungkinan jawaban yaitu memukulkan, memperpukuli, mengepukulkan, dll. Ini menandakan diperlukan tag yang lebih baik untuk fitur

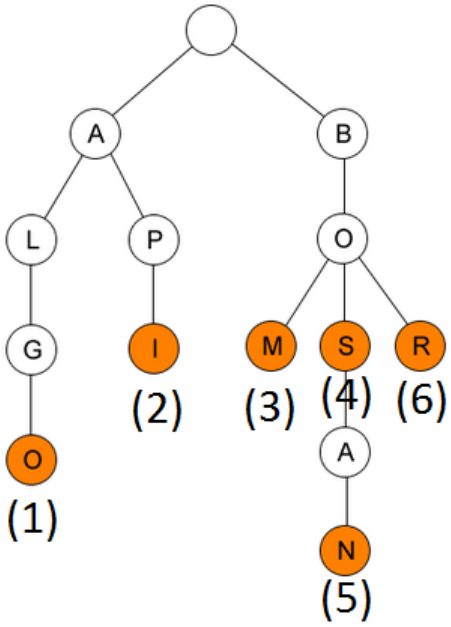
morfologi.

3.2 Leksikon

Leksikon, seperti ditulis pada subbab 2.2.3, dapat dipadankan dengan istilah *kosakata* atau *perbendaharaan kata*. Leksikon dibutuhkan pada proses *morphological parsing* untuk mengetahui apakah sebuah kata yang sedang diproses adalah sebuah kata dasar yang valid atau tidak dalam bahasa Indonesia. Leksikon menyimpan kumpulan kata dasar dan turunannya untuk nantinya diakses ketika proses *morphological parsing* dilakukan.

Leksikon dalam proses *morphological parsing* harus bisa diakses dengan cepat dan efektif. Hal ini dikarenakan leksikon akan diakses sangat sering dalam proses ini. Leksikon akan diakses sekitar 3-5 kali untuk setiap kata yang sedang diproses. Oleh karena itu, leksikon perlu disimpan pada struktur data yang memungkinkan waktu akses yang cepat supaya keseluruhan proses dapat dijalankan dalam waktu yang masuk akal.

Struktur data yang saat ini terkenal paling cepat untuk diakses adalah struktur data *trie*. Trie adalah struktur data berbentuk pohon yang menyimpan himpunan string yang jika ditelusuri setiap node mulai dari akar hingga daun akan membentuk suatu string yang merupakan kunci yang kita cari. Setiap string yang dihasilkan dari node awal yang sama akan mempunyai awalan (prefiks) yang sama, karena itulah trie disebut juga pohon prefiks.



Gambar 3.3: Struktur data trie

Struktur data trie yang digambarkan pada bagan 3.3 menyimpan enam string kunci dari dua buah awalan, yaitu string "A" dan "B". Jika kita telusuri dari node akar "A" sampai node daun "O", kita akan mendapat string "ALGO" yang ditandai dengan nomor (1). String lain yang disimpan pada contoh tersebut adalah string "API" pada nomor (2), string "BOM" pada nomor (3), string "BOS" pada nomor (4), string "BOSAN" pada nomor (5), dan string "BOR" pada nomor (6).

Perlu diperhatikan bahwa sebuah string kunci tidak harus disimpan dengan node terakhir ada pada posisi daun, seperti pada string "BOS" pada nomor 4. Node terakhir pada string tersebut merupakan node internal. Penyimpanan seperti ini bisa dilakukan dengan menandai setiap node yang merupakan akhir dari sebuah string yang membentuk kata.

Ada dua jenis kata yang disimpan dalam leksikon, yaitu kata dasar dan kata turunan. Contoh kata dasar adalah kata 'sapu', 'makan', dan 'kerja' sementara contoh kata turunan adalah kata 'menyapu', 'makan-makan', dan 'kerja bakti'. Kata-kata turunan ini adalah kata yang merupakan hasil dari proses morfologi berupa afiksasi, duplikasi, atau komposisi. Kata turunan disimpan sebagai bagian dari kata dasar dan dapat diakses ketika dibutuhkan.

Dalam Kamus Besar Bahasa Indonesia, ada beberapa kata yang merupakan hasil dari proses morfologi yang sudah diserap dan dianggap sebagai sebuah kata dasar. Contohnya adalah kata 'gerigi' yang merupakan hasil penyisipan infiks -er- pada kata 'gigi', kata 'abu-abu' yang merupakan hasil reduplikasi dari kata 'abu', dan kata 'rumah sakit' yang merupakan hasil komposisi dari kata 'rumah' dan kata 'sakit'. Dalam kasus tersebut, untuk kata yang merupakan hasil penyisipan infiks akan disimpan sebagai sebuah kata dasar dalam leksikon sementara untuk kata yang merupakan hasil reduplikasi dan komposisi akan disimpan sebagai kata turunan dari kata dasar yang bersangkutan.

Dalam Kamus Besar Bahasa Indonesia Dalam Jaringan (KBBI daring)[1], kata dasar dan kata turunan disimpan secara terpisah namun keduanya dapat dicari melalui kolom pencarian. Sementara pada Kamus Besar Bahasa Indonesia Luar Jaringan (KBBI luring)[2], hanya kata dasar saja yang bisa dicari melalui kolom pencarian namun semua kata turunannya juga disimpan sebagai bagian dari sebuah kata dasar. Pada penelitian kali ini akan digunakan struktur penyimpanan dan pencarian seperti pada KBBI luring.

Struktur penyimpanan seperti pada KBBI luring memungkinkan untuk mengenali perbedaan antara kata dasar dan kata yang telah melalui proses morfologi seperti afiksasi, reduplikasi, dan komposisi. Perangkat lunak yang dirancang pada penelitian ini harus dapat menentukan apakah sebuah kata merupakan kata dasar yang valid atau tidak dalam bahasa Indonesia. Pencarian kata

[1] https://kbbi.kemdikbud.go.id/
[2] http://ebsoft.web.id/kbbi-kamus-besar-bahasa-indonesia-offline-gratis/

dasar dalam lexicon harus dapat melakukan hal tersebut sehingga pencarian hanya bisa dilakukan terhadap kata dasar saja dan tidak dengan kata turunannya. Kata turunan perlu disimpan dalam lexicon untuk melakukan validasi terhadap hasil dari proses parsing yang dilakukan oleh perangkat lunak.

3.3 Proses *Morphological Parsing*

Pada subbab 2.3 telah dibahas mengenai proses morfologi, yang pada dasarnya adalah proses pembentukan kata melalui beberapa proses, yaitu pembubuhan afiks (afiksasi), pengulangan (reduplikasi), dan penggabungan (komposisi). Proses *morphological parsing* merupakan kebalikan dari proses morfologi. Masukan bagi proses *morphological parsing* adalah kata atau kalimat yang telah melalui proses morfologi dan keluarannya adalah komponen-komponen penyusunnya.

Proses *morphological parsing* untuk setiap kata dalam masukan dapat dituliskan sebagai berikut:

1. Periksa leksikon, jika kata tersebut ada dalam leksikon, masukkan sebagai salah satu kemungkinan keluaran
2. Periksa adanya kemungkinan afiks, baik itu prefiks, sufiks, maupun konfiks. Pisahkan afiks yang ditemukan dengan komponen kata yang lain dan lakukan proses parsing pada komponen kata tersebut
3. Periksa adanya simbol penghubung (-), yang menandakan hasil proses reduplikasi, lalu lakukan proses parsing terhadap bentuk dasar dari kata tersebut
4. Jika ada kata yang mengikuti, periksa kemungkinan kata yang sedang diproses dan kata yang mengikuti adalah dua kata hasil komposisi dengan melakukan pengecekan terhadap bentuk dasar dari kata tersebut

Lexicon yang dibuat dalam perangkat lunak ini juga menyimpan kata turunan yang valid dari setiap kata dasar yang ada. Setelah proses parsing selesai dilakukan, lexicon dapat melakukan validasi apakah kata turunan yang sudah diproses benar merupakan kata turunan yang valid dari kata dasar yang bersangkutan.

Sebagai contoh, jika dilakukan proses *morphological parsing* pada kata 'kemerah-merahan', maka prosesnya adalah sebagai berikut:

- Periksa leksikon, kata tersebut tidak ditemukan dalam leksikon

- Periksa kemungkinan afiks, ditemukan kemungkinan konfiks {ke-an} dan klofiks {ke-an}, lakukan proses parsing terhadap kata 'merah-merah'
- Ditemukan simbol penghubung (-) sehingga diketahui kata tersebut adalah hasil proses reduplikasi. Pisahkan kata dan lakukan proses parsing sehingga didapat hasilnya adalah reduplikasi dari kata dasar 'merah'
- Didapat dua kemungkinan hasil, yaitu reduplikasi kata 'merah' diikuti konfiksasi {ke-an} dan reduplikasi kata 'merah' diikuti klofiksasi {ke-an}
- Lakukan validasi pada lexicon, dan didapatkan kata turunan yang valid adalah reduplikasi kata 'merah' diikuti konfiksasi {ke-an}
- Hasil akhir proses parsing adalah bentuk dasar {merah} + reduplikasi + konfiks {ke-an}

Untuk kata dengan kemungkinan hasil parsing lebih dari satu, seperti kata 'beruang', prosesnya adalah sebagai berikut:

- Periksa leksikon, ditemukan bentuk dasar {beruang}, masukkan sebagai salah satu kemungkinan keluaran
- Periksa kemungkinan afiks pada kata 'beruang'
- Didapatkan prefiks {ber-} + bentuk dasar {uang}, masukkan sebagai salah satu kemungkinan keluaran
- Periksa kemungkinan adanya fonem yang dilesapkan pada bentuk dasar, yaitu fonem 'r', dan didapatkan prefiks {ber-} + bentuk dasar {ruang}, masukkan sebagai salah satu kemungkinan keluaran
- Lakukan validasi pada lexicon terhadap kata turunan dari bentuk dasar {uang} dan {ruang}
- Hasil akhir proses parsing adalah bentuk dasar {beruang}, prefiks {ber-} + bentuk dasar {uang}, dan prefiks {ber-} + bentuk dasar {ruang}

Bentuk-bentuk yang tidak secara khusus ada dalam bahasa Indonesia seperti angka, nama orang, dan kata dalam bahasa asing ditulis sebagai *bentuk asing* dalam keluaran dari proses parsing.

Beberapa contoh yang sudah dibahas di atas adalah contoh proses parsing yang dilakukan pada sebuah kata dalam bahasa Indonesia. Perangkat lunak *morphological parser* yang dirancang pada

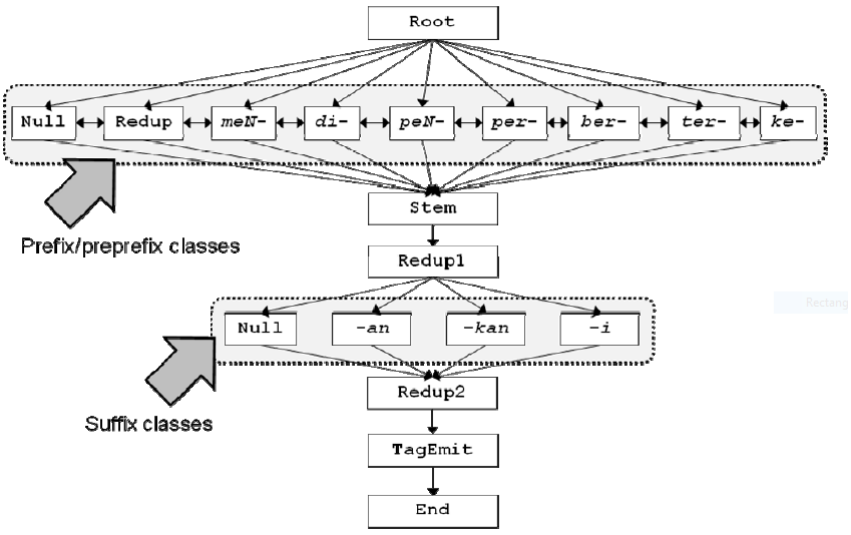
penelitian ini akan dapat memproses tidak hanya kata tapi juga kalimat dan paragraf yang ditulis dalam bahasa Indonesia. Proses parsing pada kalimat dan paragraf memerlukan beberapa langkah tambahan yaitu:

1. Hilangkan tanda baca yang tidak diperlukan dalam proses parsing. Tanda baca yang diperlukan dalam proses parsing hanya tanda baca penghubung kata (-) sebagai tanda hasil proses reduplikasi
2. Gantikan tanda baca yang dihilangkan dengan karakter kosong ("")
3. Pisahkan setiap kata berdasarkan karakter spasi yang memisahkan kata lalu lakukan proses parsing untuk setiap kata tersebut

3.4 Morfotaktik

Pada subbab 2.2.3 dan 2.2.4 telah dijelaskan mengenai morfem dasar dan morfem afiks. Morfem dasar adalah morfem yang dapat menjadi dasar dalam suatu proses morfologi. Sementara morfem afiks adalah morfem yang tidak dapat menjadi dasar dalam pembentukan kata, tetapi hanya menjadi unsur pembentuk dalam proses afiksasi. Kedua morfem tersebut dapat digabungkan dalam proses morfologi berupa proses afiksasi untuk membentuk sebuah kata. Proses penggabungan antara kedua morfem tersebut tidak boleh dilakukan secara sembarangan sehingga diperlukan aturan khusus yang mengatur penggabungan antara morfem dasar dan morfem afiks. Aturan ini disebut dengan morfotaktik.

Aturan morfotaktik tidak hanya mengatur tentang proses penggabungan antara morfem dasar dengan morfem afiks, namun juga mengatur tentang proses pengulangan morfem dasar dalam proses reduplikasi dan proses penggabungan antara morfem dasar dengan morfem dasar lain dalam proses komposisi. Aturan morfotaktik ini dapat diilustrasikan dengan sebuah *finite-state automata* yang dapat dilihat pada gambar 3.4. Kata yang dibentuk dari proses morfologi bahasa Indonesia yang benar akan dapat diterima oleh automata, sementara yang tidak akan ditolak.

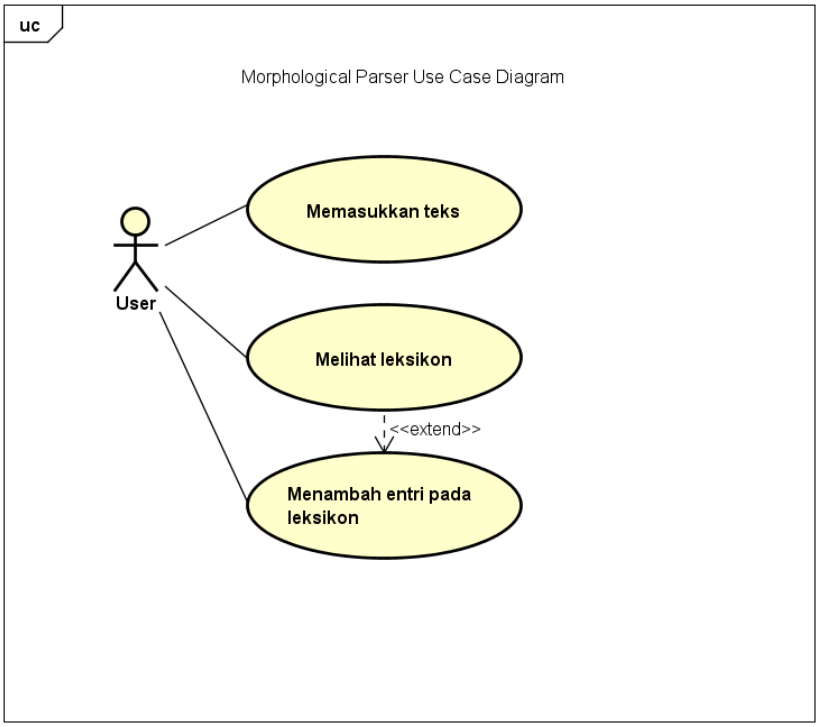


Gambar 3.4: Ilustrasi proses morfotaktik

Automata dimulai dari state *Akar*, lalu bisa berlanjut ke state *Null*, yang berarti tidak digabungkan dengan apapun, atau bisa berlanjut ke state *Redup*, yang berarti dilakukan proses reduplikasi, atau bisa berlanjut ke state *Kompo*, yang berarti dilakukan proses komposisi, atau bisa berlanjut ke salah satu state prefiks, yaitu state *me-, pe-, per-, di-, ber-, ter-, se-,* atau *ke-*. Dari state ini, automata dapat bergerak ke state *Pangkal* atau dapat bergerak ke salah satu state dalam kelas yang sama untuk kasus di mana ada dua prefiks yang diimbuhkan. Lalu dari ...

3.5 Diagram Use Case

Perangkat lunak *morphological parser* yang akan dibangun dapat memproses masukan berupa teks dalam bahasa Indonesia yang dapat dimasukkan ke dalam perangkat lunak melalui dua cara, melalui kolom masukan dan melalui file teks. Perangkat lunak juga memiliki leksikon yang isinya dapat dilihat oleh user. Selain itu, user juga dapat menambahkan entri pada leksikon melalui perangkat lunak ini. Fitur-fitur ini dapat digambarkan dalam diagram use case pada gambar 3.5.



Gambar 3.5: Diagram use case perangkat lunak morphological parser

Dari diagram tersebut dapat dituliskan use case scenario sebagai berikut:

MEMASUKKAN TEKS

Name: Memasukkan teks

Actors: User

Goals: User berhasil memasukkan teks yang akan diproses ke dalam sistem

Precondition: Teks sudah disiapkan

Steps:

Actor actions	System responses
1. User memilih pilihan untuk memasukkan teks melalui file	2. Sistem menampilkan kotak dialog untuk memilih file
3. User mengarahkan kotak dialog ke direktori tempat file teks masukan	
4. User menekan tombol "OK"	5. Sistem mengeluarkan keterangan "Teks berhasil dimasukkan"

Alternate flow:

Actor actions	System responses
1a. User memilih pilihan untuk memasukkan teks melalui kolom masukan 2a. User mempaste teks ke dalam kolom masukan 3a. User menekan tombol "OK"	4a. Sistem mengeluarkan keterangan "Teks berhasil dimasukkan"

MELIHAT LEKSIKON

Name: Melihat leksikon

Actors: User

Goals: User dapat melihat leksikon yang ada dalam sistem

Precondition: Sistem sudah memuat leksikon ke dalam program

Steps:

Actor actions	System responses
1. User memilih pilihan untuk melihat leksikon	2. Sistem menampilkan leksikon yang ada dalam sistem

MENAMBAH ENTRI PADA LEKSIKON

Name: Menambah entri pada leksikon

Actors: User

Goals: User berhasil menambah entri pada leksikon

Precondition: Sistem sudah memuat leksikon ke dalam program

Steps:

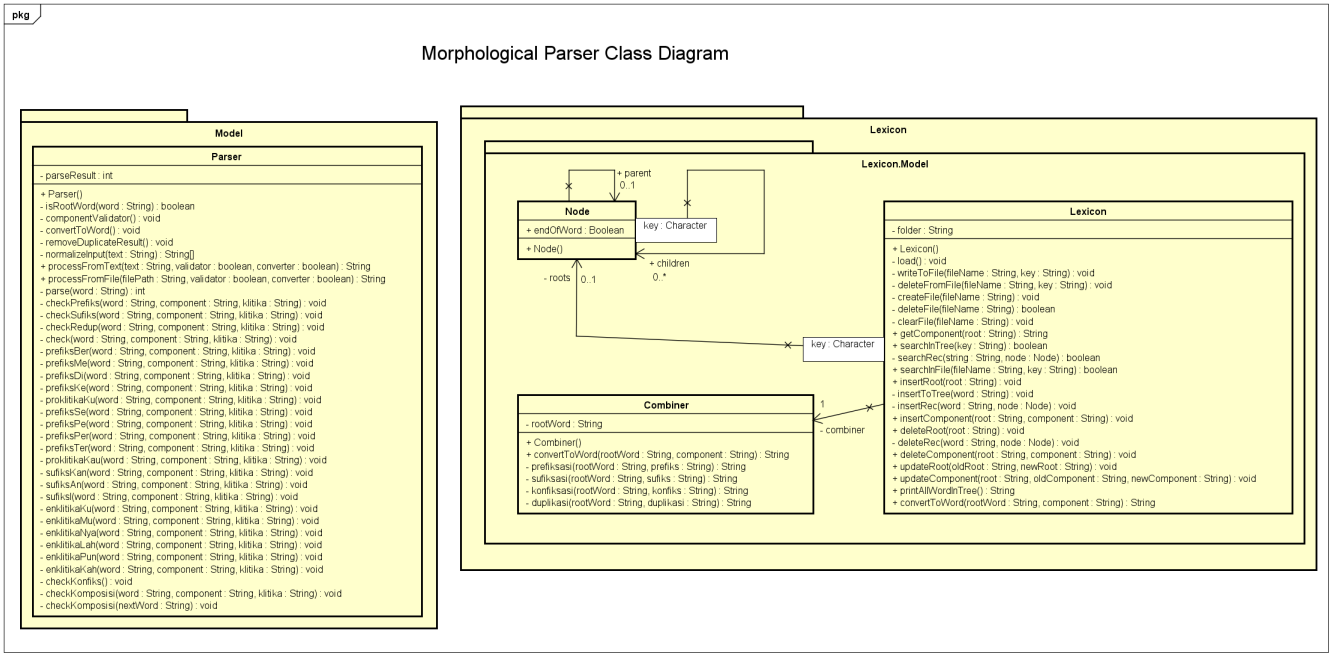
Actor actions	System responses
1. User memilih pilihan untuk menambah entri pada leksikon	2. Sistem menampilkan form untuk menambah entri pada leksikon
3. User mengisikan entri baru pada form 4. User menekan tombol "OK"	5. Sistem mengeluarkan keterangan "Entri berhasil dimasukkan"

Alternate flow:

Actor actions	System responses
	5a. Sistem mengeluarkan keterangan "Format pengisian entri salah, ulangi lagi"

3.6 Diagram Kelas

Berikut adalah diagram kelas yang dibuat untuk perangkat lunak ini.



Gambar 3.6: Diagram kelas perangkat lunak morphological parser

DAFTAR REFERENSI

[1] Chaer, A. (2008) *Morfologi Bahasa Indonesia (Pendekatan Proses)*. Rineka Cipta, Jakarta.

[2] Jurafsky, D. dan Martin, J. H. (2009) *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition*, 2nd edition. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

[3] Najogie, R. D. (2010) Pengenalan trie dan aplikasinya. *Makalah IF2091 Struktur Diskrit - Sem. I Tahun 2010/2011*, **1**, 91–95.

[4] Pisceldo, F., Mahendra, R., Manurung, R., dan Arka, I. W. (2008) A two-level morphological analyser for the indonesian language. *Proceedings of the Australasian Language Technology Association Workshop 2008 (ALTA 2008)*, Hobart, Australia, 8-10 December, pp. 142–150. Australasian Language Technology Association.