Lõplikud automaadid ja -muundurid

Automaat

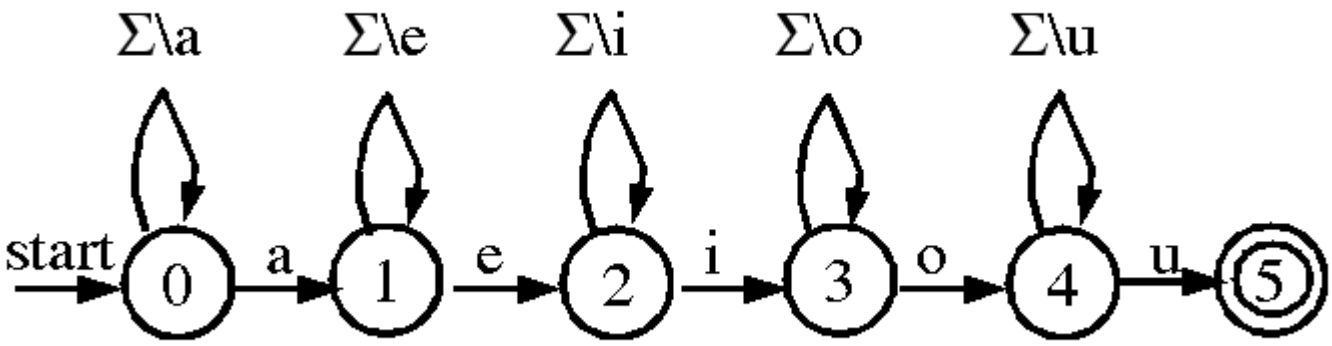
Jaak Vilo loengust: Automaadid, regulaaravaldised, grammatikad

Automaat *(i.k. automaton (sing.), automata (plural))*

Sageli tasub mõelda formaalselt näiteks programmi viibimisest mingis "olekus" (state).

Näiteks joogi-automaat on enne raha panemist nn. "algolekus". Kui raha on sisestatud, on olekus kus ootab konkreetse joogi tellimust. Kui tellimus on saadud ja korrektne (on vastavat toodet) siis väljastatakse toode, tagastatakse raha ja siirdub tagasi algolekusse.

Automaat, mis tunneb ära sõnad, milles on *a e i o u* (just selles järjekorras), nt. *abstemious*



Automaadi poolt ära tuntav keel

Σ^* alamhulk, mille puhul iga selle keele sõna aktsepteeritakse antud automaadi poolt.

Keele esitamiseks produktsioonireeglid:

$S_0 \rightarrow a\ S_1 \mid \Sigma\text{-}a\ S_0$

$S_1 \rightarrow e\ S_2 \mid \Sigma\text{-}e\ S_1$

$S_2 \rightarrow i\ S_3 \mid \Sigma\text{-}i\ S_2$

$S_3 \rightarrow o\ S_4 \mid \Sigma\text{-}o\ S_3$

$S_4 \rightarrow u\ S_5 \mid \Sigma\text{-}u\ S_4$

$S_5 \rightarrow \Sigma\ S_5 \mid \varepsilon$

Selle keele genereerimine:

$S_0 \Rightarrow a\ S_1 \Rightarrow a\ b\ S_1 \Rightarrow a\ b\ e\ S_2 \Rightarrow a\ b\ e\ i\ S_3 \Rightarrow a\ b\ e\ i\ o\ S_4 \Rightarrow a\ b\ e\ i\ o\ u\ S_5 \Rightarrow a\ b\ e\ i\ o\ u\ \varepsilon$

Formaalne definitsioon

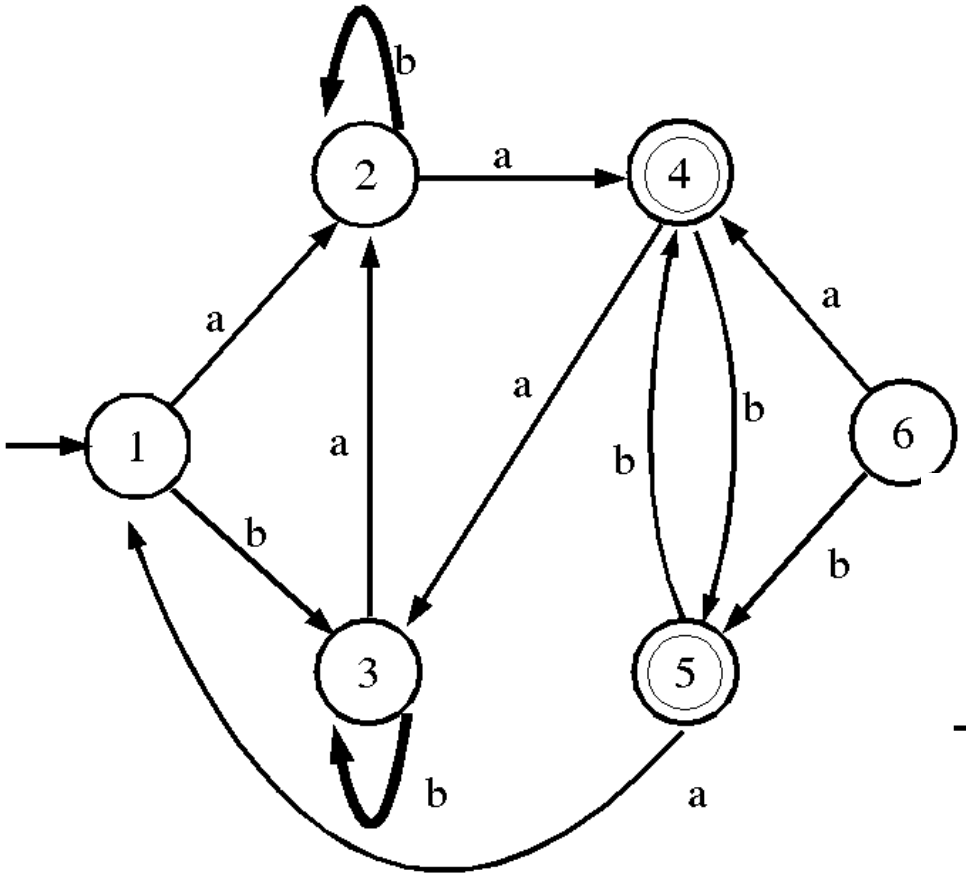
Lõplik automaat on viisik $M=(Q, \Sigma, \delta, q_0, F)$, kus

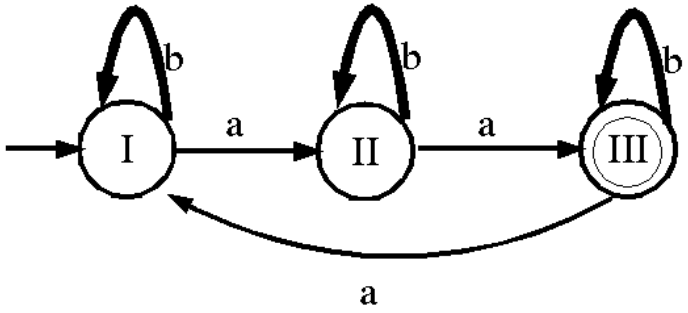
- Q on automaadi olekute lõplik hulk
- Σ on sisendtähestik
- automaadi üleminekuseos
  - determ. automaadil f-n $\delta : Q \times \Sigma \rightarrow Q$
  - mittedet. automaadil $\delta : Q \times (\Sigma \cup \varepsilon) \rightarrow P(Q)$
- $q_0 \in Q$ on automaadi algolek
- $F \subseteq Q$ on (aktsepteerivate) lõppolekute hulk

C-keelne programm

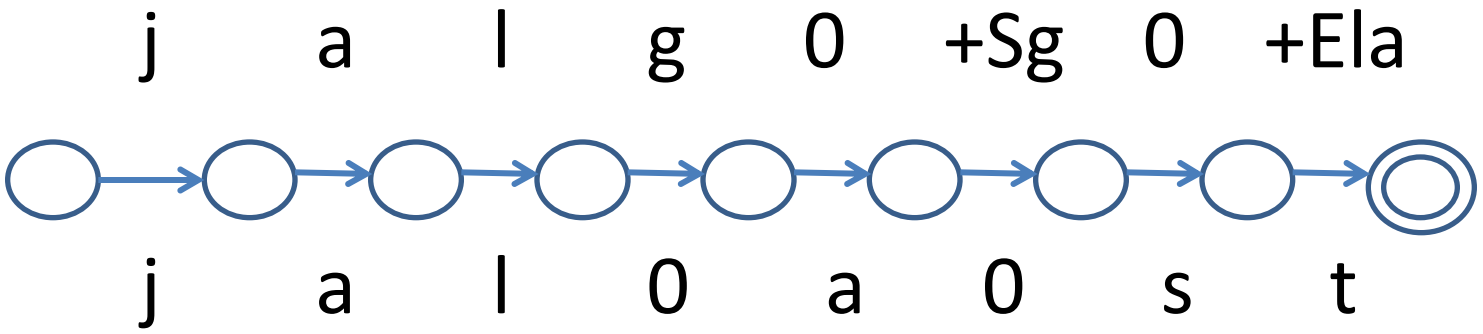
```
BOOLEAN aeiou( char *str )
{
        char *cp;       cp = str;       /* pointer to char in string */
        int state;      state = 0 ;     /* state machine's state     */
         while( *cp ){
                switch ( state ) {
                        case 0 : if( *cp == 'a' ) state = 1 ; break; /* state 0 */
                        case 1 : if( *cp == 'e' ) state = 2 ; break; /* state 1 */
                        case 2 : if( *cp == 'i' ) state = 3 ; break; /* state 2 */
                        case 3 : if( *cp == 'o' ) state = 4 ; break; /* state 3 */
                        case 4 : if( *cp == 'u' ) state = 5 ; break; /* state 4 */
                }
                cp++ ; /* next char */
        }
        if( state == 5 )  return TRUE ;
        else              return FALSE ;
}
```

Minimiseerimine
(b*ab*ab*)+





Lõplik muundur



... on nagu lõplik automaat, aga lisaks sisendi äratarbimisele ja töö lõpetamisele annab ka midagi välja

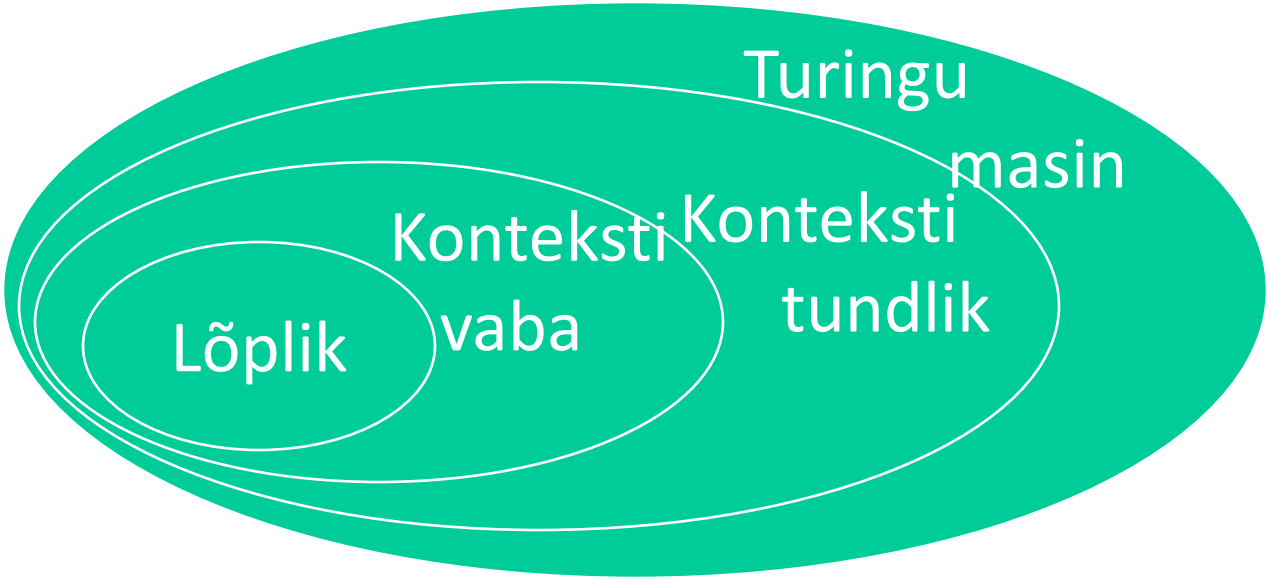
Insenerile

- Sõnade äratundmine ja loomine kiiresti ning mälusäästlikult
- Standardne andmeformaat
- Tööd teeb standardne, keelest sõltumatu programm

Lõplike automaatide koht keelegrammatikate hierarhias

"*English is not a finite state language.*"
(Chomsky "Syntactic structures" 1957)

Chomsky hierarhia:



Generatiivne fonoloogia

Chomsky, Halle (1968) kasutasid morf. sünteesiks kontekstitundlike ümberkirjutusreeglite järjestikust rakendamist, et teisendada abstraktne fonoloogiline esitus pindesituseks (sõnavormiks) läbi vahepealsete esituste.

Reeglite üldkuju: $x \rightarrow y / z \_ w$,

kus x, y, z, w on suvalise keerukusega tunnusstruktuurid.

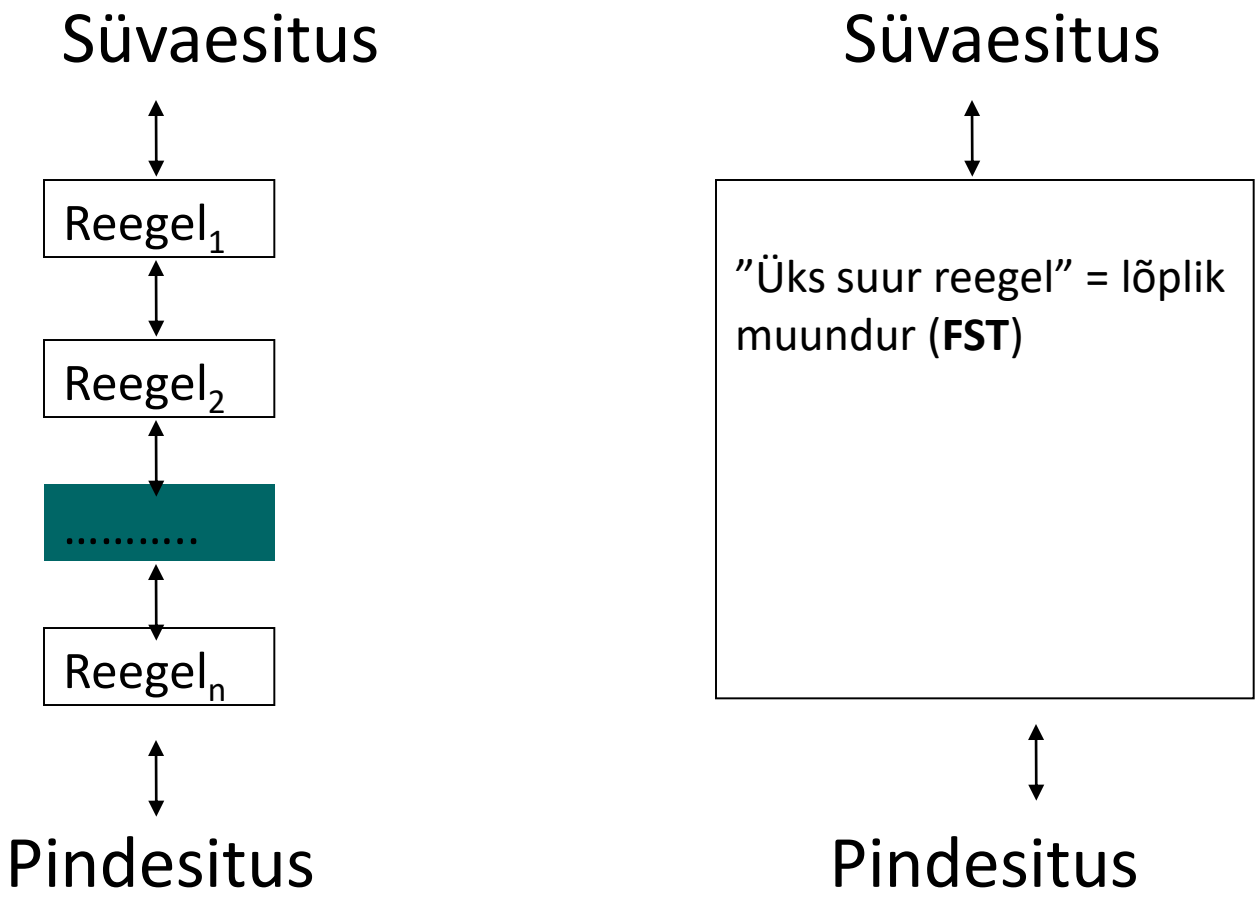
Aga:

D. Johnson, 1972: Fonoloogilised ümberkirjutusreeglid ei ole sisuliselt kontekstitundlikud, vaid neid saab kirjeldada lõplike muunduritena (*finite-state transducer*).

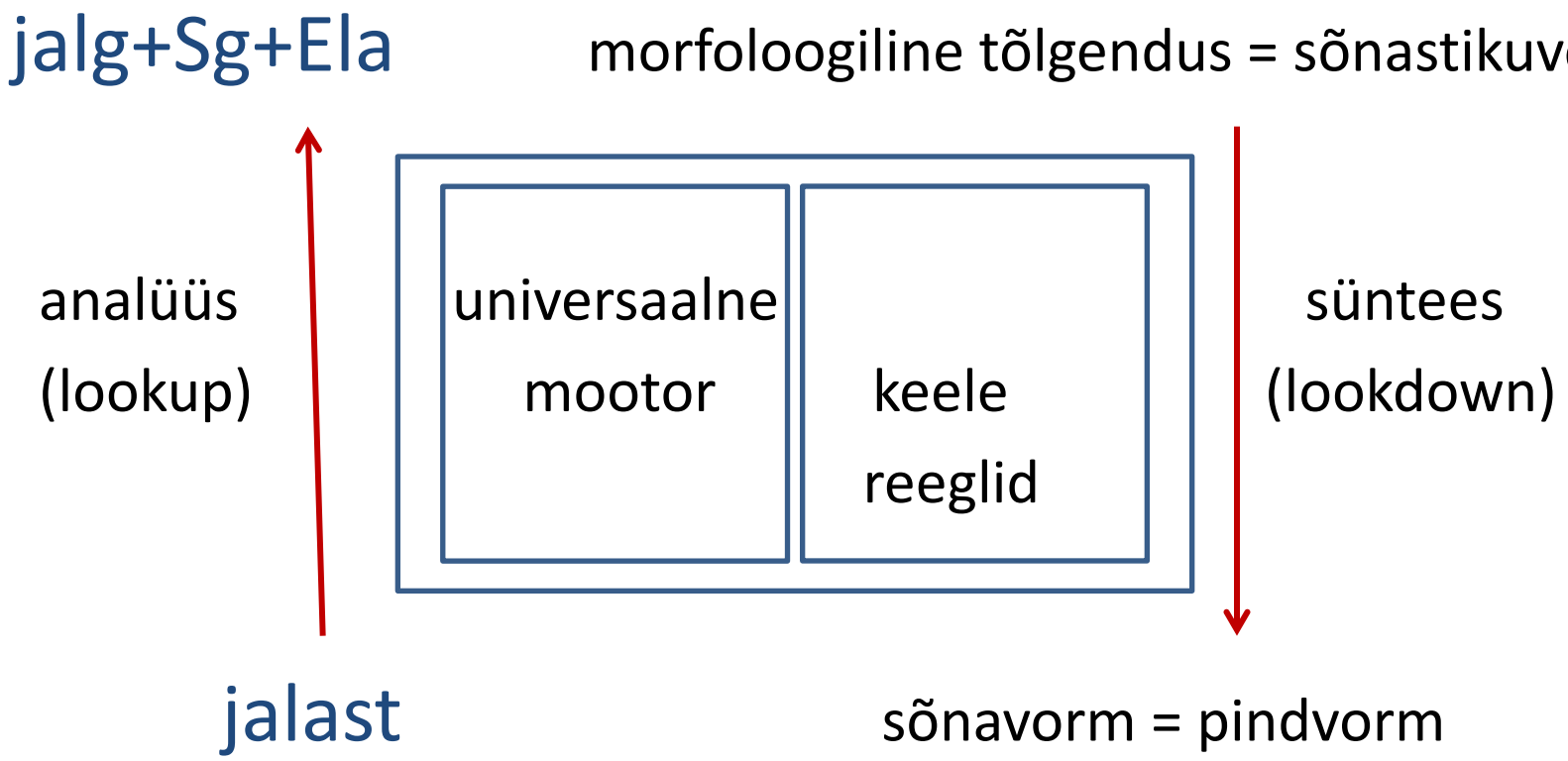
(Taasavastasid Kaplan, Kay 1980ndatel)

Schützenberger, 1961: Kui kaks lõplikku muundurit rakendada järjestikku, siis leidub üks lõplik muundur, mis on nende kahe kompositsioon.

Kompositsiooni üldistus n muundurile: saame läbi ilma vaheesitusteta – süvaesitus teisendatakse pindesituseks üheainsa monoliitse lõpliku muunduri abil!


Süvaesitus
Reegel1
Reegel2
.....
Reegeln
Pindesitus
Süvaesitus
”Üks suur reegel” = lõplik muundur (**FST**)
Pindesitus

Analüüs ja süntees FST kombel



Lõplikel muunduritel põhineva arvutimorfoloogia põhiväide:

Seos keele sõnavormide ja nende algvormide e. lemmade vahel on kirjeldatav regulaarse seosena.

Lõplik automaat ei tea, kuidas ta praegusesse olekusse sattus..

2 probleemi

- Morfotaktika
  - Sõnad koosnevad väiksematest üksustest, mida liidetakse kokku teatud järjekorras:

    kaarna-te-s on eesti keel, kaarna-s-te ei ole
- Morfofonoloogia
  - Nende väiksemate üksuste vorm võib kokkupanemisel muutuda:

    pida-ma, pea, pee-ti

Terminoloogiat

- Morfeem – väikseim tähenduslik tükk
- Morf – morfeemi esinemisvorm
- Allomorf - morfeemivariant (nt. de, te – mitmuse tunnus)
- Morfotaktika – morfeemide järjestuse ja kombineerimise tingimused (nt. tüvi + arv + käändelõpp, just selles järjekorras)
- Morfofonoloogia – kuidas morfeemid muutuvad, kui nad sõnasse kokku on pandud (nt. elama+tud =elatud, naerma+tud=naerdud, s.t. [l n r] + tud -> dud)

Eesti keel on aglutinatiivse (~ türgi, soome) ja flektiivse (~ saksa) kombinatsioon: morfeemide liitmine (äpi+le, äpi+ga jne), varieerimine (äpp, äpi, äppi).

Näide

Mitmus: allomorfid de, te, i, e, u

kala+de, aasta+te, aasta+i+l, õnnelik+e+l, kõrv+u+s

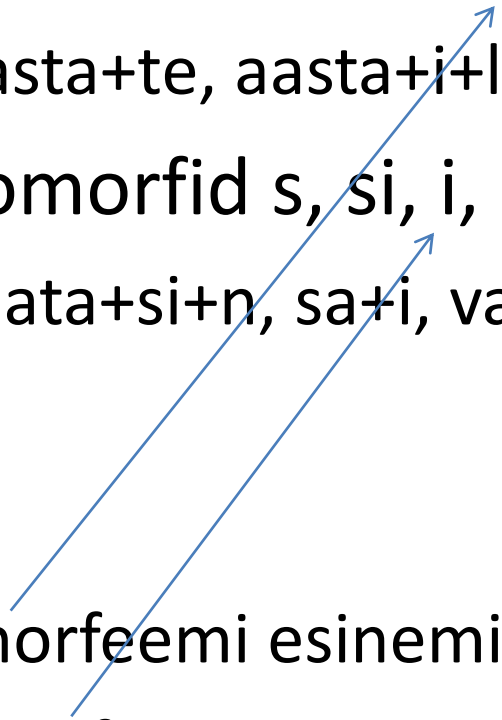
Minevik: allomorfid s, si, i, nu

vaata+s, vaata+si+n, sa+i, vaada+nu+ks

Morf i

mitmuse morfeemi esinemisvorm

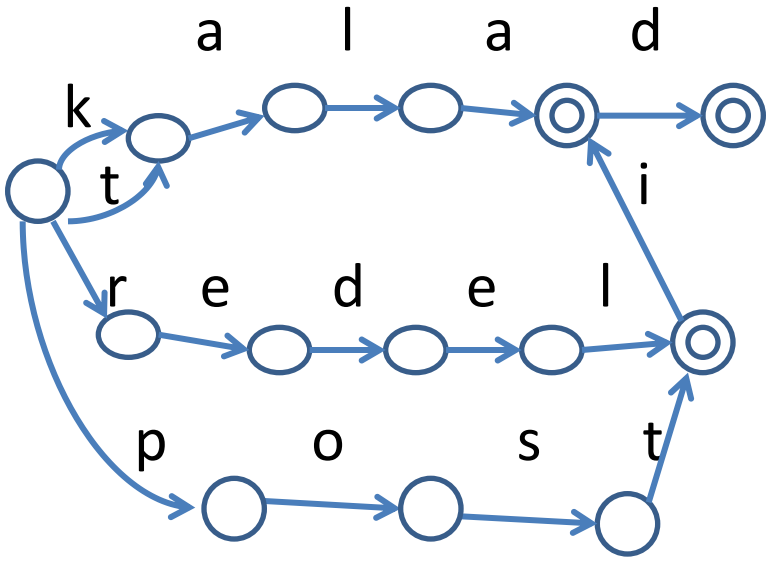
mineviku morfeemi esinemisvorm



Reeglipärane morfotaktika

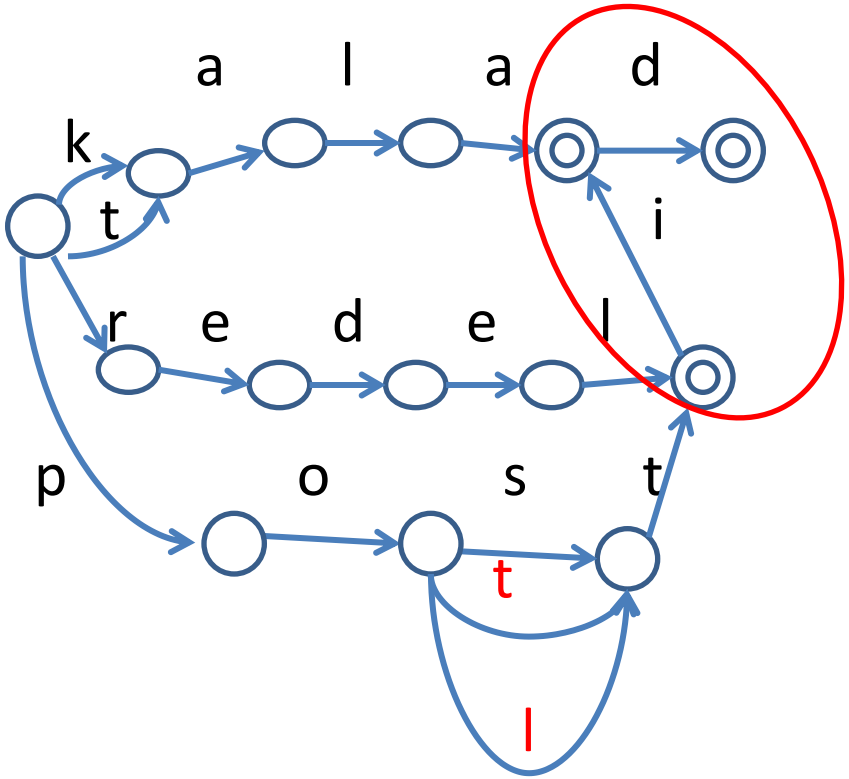
- Suur osa paljude keelte morfoloogiast on kirjeldatav kahe tehte abil:
  - ühend e. mittevälistav või;
  - konkatenatsioon e. jätkamine

Lihtne sõnastik (12 vormi)



kala, tala, redel, post: a. nim, a. om, mitm. nim.

Lihtne vigane sõnastik

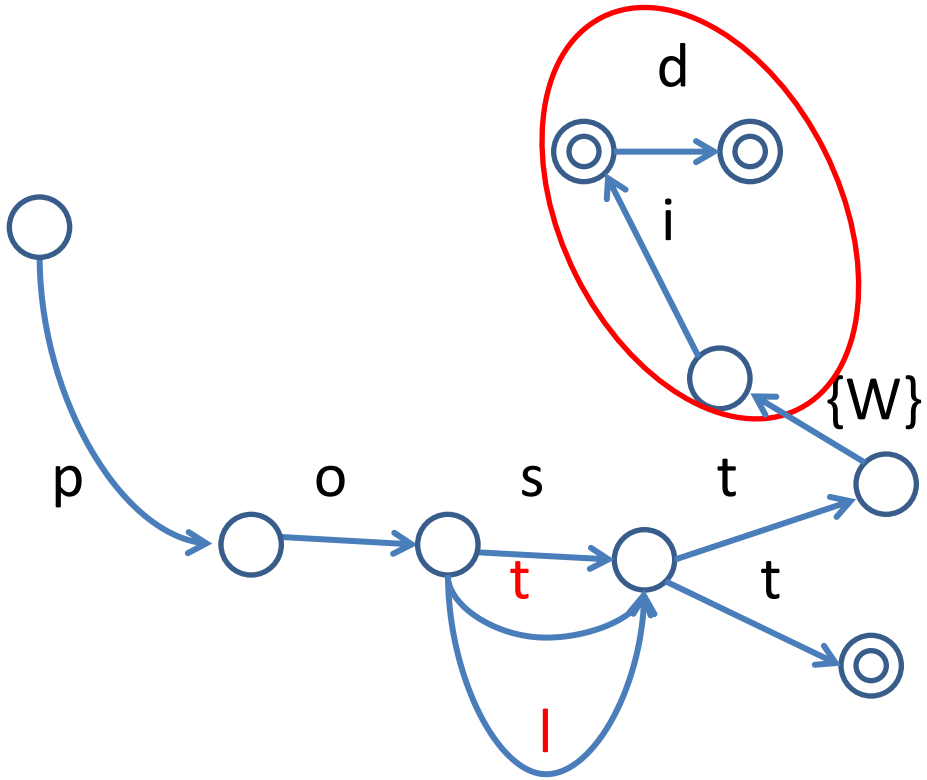


post, posti, postid, polt, polti, poltid, pott, potti, pottid

Kuidas mugavalt teha õigeid sõnastikke?

- Etapid
- Operatsioonid

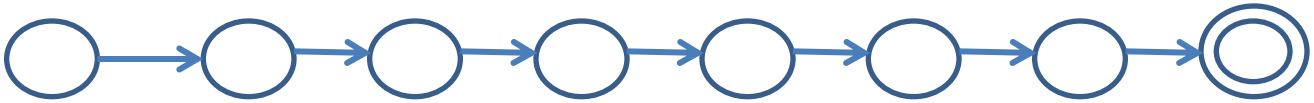
Lihtne pooleli sõnastik



{W} – nõrk aste post{W}id, pott{W}id, polt{W}id

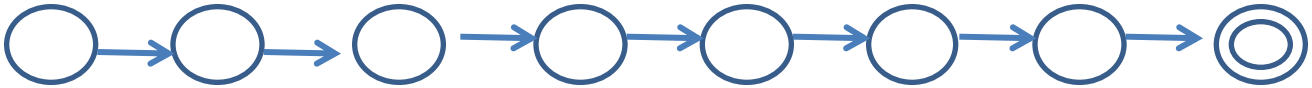
Järjest rakendamine e. kompositsioon

p o t t 0 +Sg +Gen



p o t t {W} i 0

p o t t {W} i 0



p o t 0 0 i 0

Kirjandus

- http://www.stanford.edu/~laurik/fsmbook/home.html
- http://web.stanford.edu/~laurik/.book2software/twolc.pdf