Formális nyelvek, nyelvtanok, gépek

Nyelvi adatok feldolgozása – 2019/20 tavasz
3. óra

Simon Eszter

MTA Nyelvtudományi Intézet

Tartalom

A Chomsky-féle nyelvhierarchia

A nyelv formális meghatározása

Definíció (nyelv)

Egy tetszőleges véges A halmazból alkotott $A*$ halmaz tetszőleges részhalmazát (vagyis az A fölötti füzérekből összegyűjtött tetszőleges halmazt) nyelvnek nevezzük, az A halmazt pedig e nyelv ábécéjének.

A nyelvtan formális meghatározása

Definíció (nyelvtan)

A $G =< V_T, V_N, S, R >$ négyest formális nyelvtannak nevezzük, ahol

V_T: a terminális elemek ábécéje

V_N: a nem-terminális elemek ábécéje

S: a nyelvtan kezdőszimbóluma

R: a nyelvtan szabályainak a halmaza

A Chomsky-féle nyelv(tan)hierarchia

Definíció (Chomsky-féle nyelv(tan)hierarchia)

Egy $G =< V_T, V_N, S, R >$ nyelvtan i típusú ($i \in \{0, 1, 2, 3\}$) a Chomsky-féle nyelvtanhierarchia szerint, amennyiben az R szabályhalmaz minden elemére teljesül az adott típusban előírt, a szabály felépítésére vonatkozó előírás.

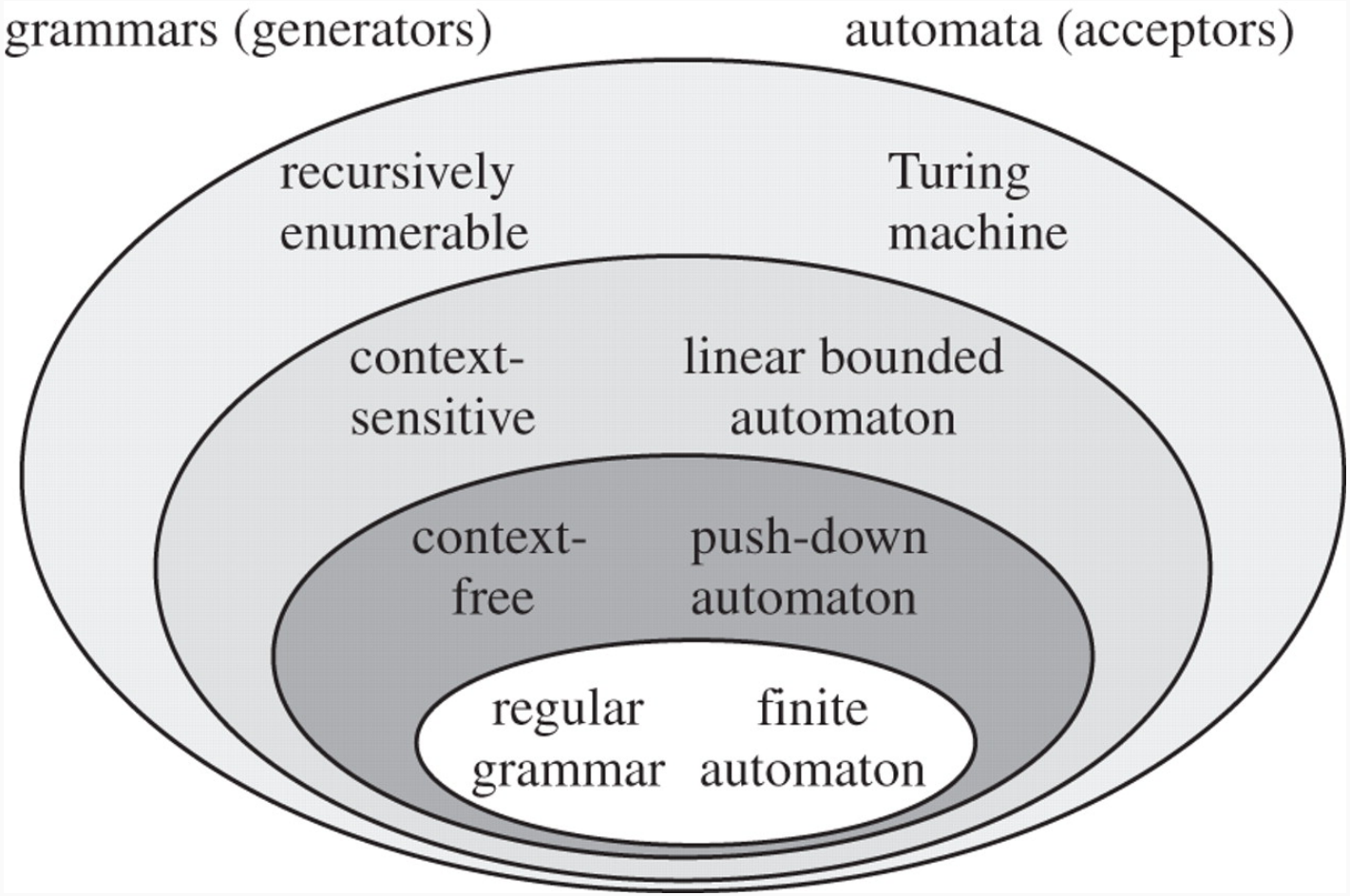
Jelölések: $\alpha, \beta, \gamma \in (V_T \cup V_N)*$

$A, B \in V_N$

$x \in V_T*$

0. típus (megszorítatlan újraíró rendszer)	$\alpha \to \beta$, ahol $\alpha \neq \varepsilon$
1. típus (környezetfüggő)	$\alpha A \beta \to \alpha \gamma \beta$, ahol $\gamma \neq \varepsilon$
2. típus (környezetfüggetlen)	$A \to \gamma$
3. típus (reguláris vagy jobblineáris)	$A \to xB$ vagy $A \to x$

Nyelvtanok és gépek

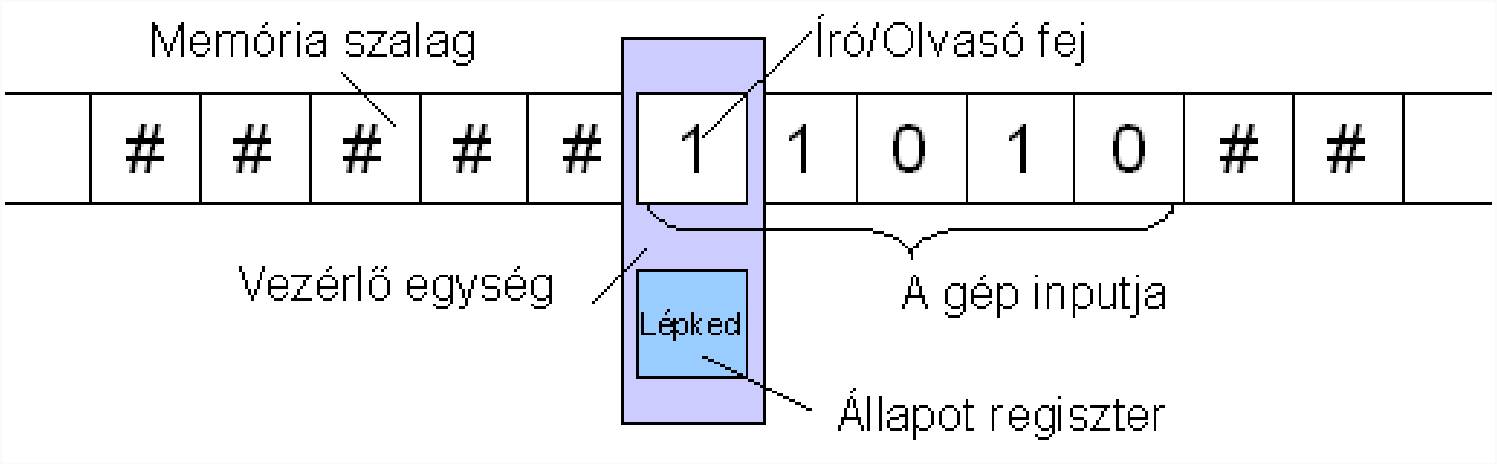


Megszorítatlan (0. típusú) nyelvek

$\alpha \to \beta$, ahol $\alpha \neq \varepsilon$

- megszorítatlan újraíró rendszer
- a leghasznosabb új eszköz a szimbólumcsere
- a 0. típusú nyelveket Turing-géppel lehet elfogadtatni $\to$ a legáltalánosabb nyelvtantípusnak a létező és elképzelhető legáltalánosabb absztrakt gép felel meg
- „a Turing-géppel minden kiszámolható, ami egyáltalán kiszámolható, és minden meghatározható, ami egyáltalán meghatározható emberi elménk számára”

Turing-gép


Memória szalag
Író/Olvasó fej
1 1 0 1 0 #
Vezérlő egység
Lépked
A gép inputja
Állapot regiszter

Turing-gép

- író-olvasó fej, egy kockákra osztott végtelennek tekintett szalag, amelynek minden kockáján egy szimbólum áll
- a fej jobbra és balra is mozoghat
- az író-olvasó fejnek különböző állapotai vannak, amelyek megszabják, hogy az éppen leolvasott szimbólumot átírja-e, vagy lépjen tovább
- a végtelennek tekintett szalag egy véges részén van csak információ, a többi # jelet tartalmazó üres hely: $...\#a_1a_2...a_n\#...$
- a számítás kezdetén a fej a bal szélső nem-üres szimbólumon áll, utána: $< q_i, a, q_j, X >$
- ha $X \in \{J, B\}$, akkor jobbra vagy balra lép egyet, különben marad és átír
- a Turing-gép akkor fogad el egy füzért, ha a számítás véges számú lépés után leáll

És így született meg a Turing-gép...

A környezetfüggő (1. típusú) nyelvek

$\alpha A \beta \to \alpha \gamma \beta$, ahol $\gamma \neq \varepsilon$

- a környezetfüggő szabály egyetlen nem-terminális szimbólum átalakításáról gondoskodik
- a bemeneti oldalon a szimbólum kétoldali környezetére is hivatkozhatunk
- a nem-terminális szimbólum megsemmisítése nem megengedett $\to$ hosszúságot nem csökkentő szabályok
- a környezetfüggő nyelvek absztrakt számítógépes jellemzésére a lineárisan korlátozott Turing-gépek szolgálnak $\to$ az író-olvasó fej nem léphet ki egy kezdő- és egy végszimbólumokkal jelölt területen kívülre

Környezetfüggő szabályok

Noam Chomsky and Morris Halle: The Sound Pattern of English (1968)

$$A \rightarrow B / [precontext\_\_postcontext]$$

tundrai nyenyec cirill–IPA konverzió

ю $\rightarrow$ *ju*/#__

ю $\rightarrow$ *ju*/__ [аеёиоуыэюя]

ю $\rightarrow^{j}$ *u*/ [бвгджзйклмнңпрстфхцчшщ] __

else:

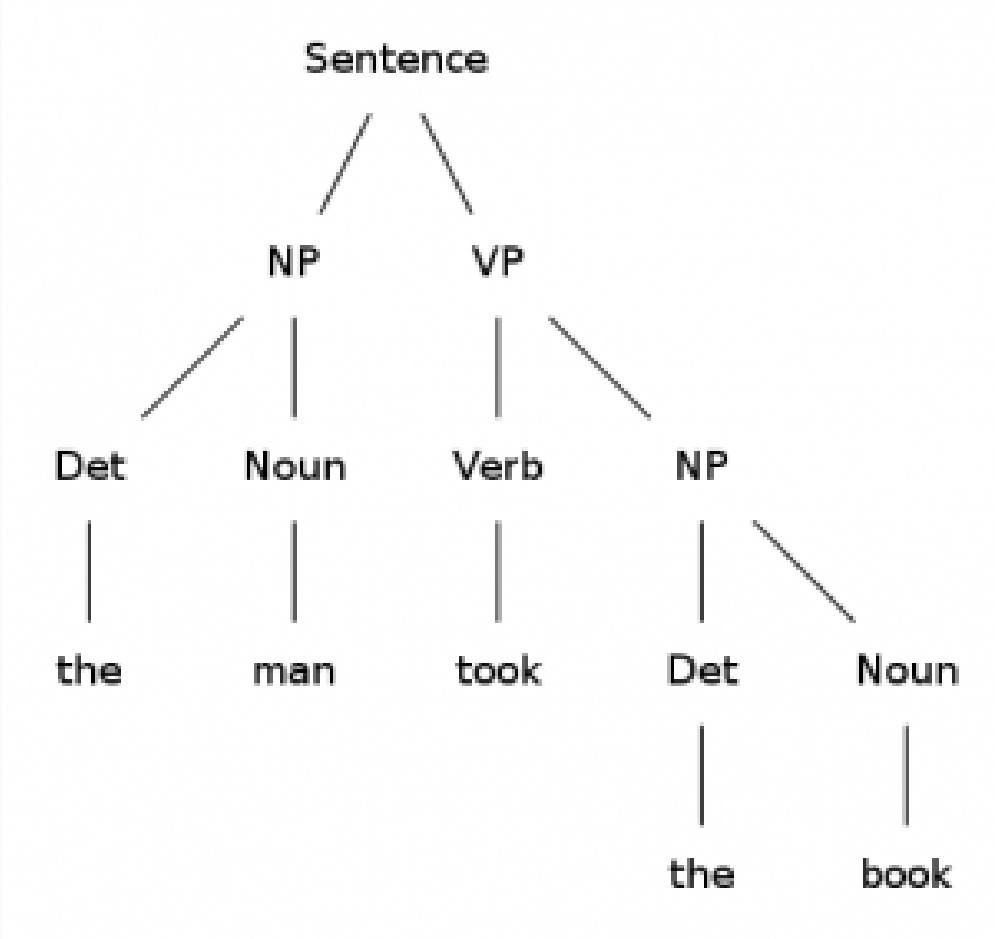
ю $\rightarrow$ *ju*/__

A környezetfüggetlen (2. típusú) nyelvek

$A \to \gamma$

- csak a szabály bemeneti oldala van korlátozva: egyetlen nem-terminális szimbólum állhat ott
- a kimeneti oldalon megengedett az üres sztring is
- Chomsky-féle normálalak: $A \to a$ vagy $A \to BC$
- elfogadó gép: veremautomata $\to$ mintha egy véges automata meg lenne toldva egy veremszerű memóriával
- verem: „last in, first out” (LIFO)

Környezetfüggetlen szabályok



Az első környezetfüggetlen elemzési fa (Chomsky, 1956: Three models for the description of language. *IRI Transactions on Information Theory, 2(3)*, 113–124.).

A reguláris (3. típusú) nyelvek

$A \to xB$ vagy $A \to x$

- a reguláris nyelveket leíró reguláris nyelvtanok reguláris kifejezésekkel ekvivalensek
- a reguláris nyelvtanok lehetnek jobblineárisak vagy ballineárisak
- egy jobblineáris szabály bemeneti oldalán egyetlen nem-terminális szimbólum állhat, és maximum egy nem-terminális állhat a kimeneti oldalán, és ez utóbbinak a legutolsónak kell lennie
- elfogadó gép: véges állapotú automata (finite state automaton, FSA)

Hol vannak a természetes nyelvek?

- valahol a környezetfüggetlen és a környezetfüggő között → enyhén környezetfüggő nyelvtanok
- környezetfüggetlen szabályokkal a természetes nyelvi jelenségek nagyon nagy része leírható, de nem minden → ellenpélda egy svájci német dialektusból

Shieber (1985): keresztező függőség az igék és tárgyuk között

Jan säit das...

...mer	em Hans	es huus	hälfed	aastriche
mi	Hans.DAT	a ház.ACC	segített	fest

'Jan azt mondta, hogy segítettünk Hansnak festeni a házat.'

két ugyanolyan füzér konkatenációjával előálló ismétléses füzérek környezetfüggő nyelve: $\{xx \mid x \in \{a, b\}*\}$

Automaták

Véges állapotú automata

Definíció (véges állapotú automata)

Az $M =< K, \Sigma, d, q_0, F >$ ötös egy véges állapotú automata, ahol

K: az automata állapotainak véges halmaza

Σ: az ábécé

q_0: a kezdőállapot

F: a végállapotok halmaza

d: az átmenetek halmaza

Determinisztikus bégetőautomata 1.

bee*!

$K = \{q_0, q_1, q_2, q_3, q_4\}$
$\Sigma = \{b, e, !\}$
$F = \{q_4\}$
$d = \{< q_0, b, q_1 >, < q_1, e, q_2 >, < q_2, e, q_3 >, < q_3, e, q_3 >,$
$< q_3, !, q_4 >\}$

Determinisztikus bégetőautomata 2.

$K = \{q_0, q_1, q_2, q_3, q_4, q_5\}$
$\Sigma = \{b, e, !\}$
$F = \{q_4\}$
$d = \{< q_0, b, q_1 >, < q_1, e, q_2 >, < q_2, e, q_3 >, < q_3, e, q_3 >,$
$< q_3, !, q_4 >, < q_0, e, q_5 >, < q_0, !, q_5 >, < q_1, b, q_5 >, < q_1, !, q_5 >,$
$< q_2, b, q_5 >, < q_2, !, q_5 >, < q_3, b, q_5 >, < q_4, b, q_5 >, < q_4, e, q_5 >,$
$< q_4, !, q_5 >, < q_5, b, q_5 >, < q_5, e, q_5 >, < q_5, !, q_5 >\}$

Determinisztikus vs. nem-determinisztikus

determinisztikus
nincs választási lehetőség, az algoritmus minden inputra egyértelműen tudja, hogy mit kell csinálni

nem-determinisztikus

1. a gép nem minden inputra tudja egyértelműen, hogy mit csináljon
2. egy ε-átmenet nem-determinisztikussá teszi az automatát, mert nem tudja, hogy merre menjen tovább, és ebben az input sem segít

Nem-determinisztikus bégetőautomaták

$K = \{q_0, q_1, q_2, q_3, q_4\}$
$\Sigma = \{b, e, !\}$
$F = \{q_4\}$
$d = \{< q_0, b, q_1 >, < q_1, e, q_2 >, < q_2, e, q_2 >, < q_2, e, q_3 >,$
$< q_3, !, q_4 >\}$

VAGY

$K = \{q_0, q_1, q_2, q_3, q_4\}$
$\Sigma = \{b, e, !, \varepsilon\}$
$F = \{q_4\}$
$d = \{< q_0, b, q_1 >, < q_1, e, q_2 >, < q_2, e, q_3 >, < q_3, \varepsilon, q_2 >,$
$< q_3, !, q_4 >\}$

Állapot-átmenet táblák I.

deterministic

	bemenet		
állapot	*b*	*e*	*!*
0	1	-	-
1	-	2	-
2	-	3	-
3	-	3	4
4:	-	-	-

Állapot-átmenet táblák II.

nem-determinisztikus 1.

	bemenet		
állapot	*b*	*e*	*!*
0	1	-	-
1	-	2	-
2	-	2,3	-
3	-	-	4
4:	-	-	-

nem-determinisztikus 2.

	bemenet			
állapot	*b*	*e*	*!*	ε
0	1	-	-	-
1	-	2	-	-
2	-	3	-	-
3	-	-	4	2
4:	-	-	-	-

Egy nyelvi példa

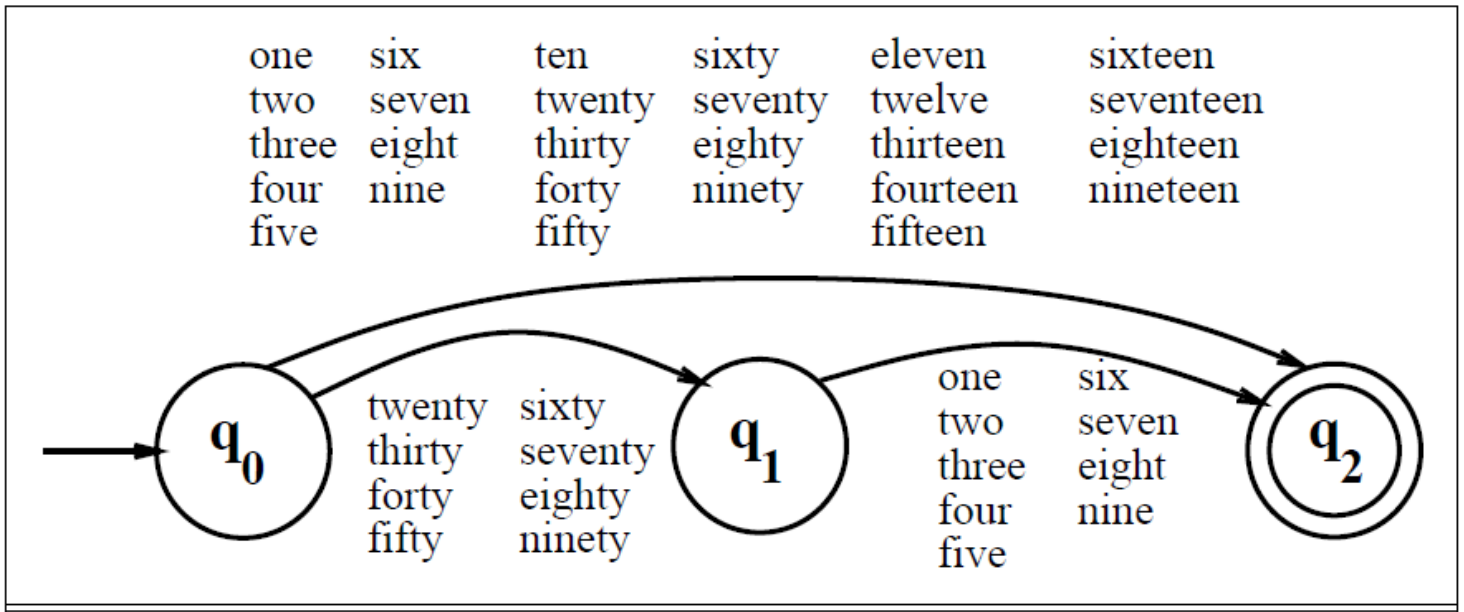


Figure 2.16 An FSA for the words for English numbers 1 – 99.

Nem-determinisztikusság

a nem-determinisztikus automata mehet rossz irányba → grammatikus sztringet utasít el → sztenderd megoldások:

- **backup:** megjelöljük a döntési pontot, így ha kiderül, hogy rosszfelé mentünk, akkor vissza tudunk oda térni, és mehetünk a másik ágon
- **look-ahead:** okosan előrenézünk az inputban, hogy el tudjuk dönteni, hogy merre érdemes menni
- **párhuzamosítás:** a döntési pontoknál minden alternatív utat párhuzamosan bejárunk

Recognition as Search

- az állapottér bejárása: a lehetséges megoldások terét szisztematikusan bejárjuk
- a hatékonyság kulcsa a sorrend
  1. verem (stack), mélységi bejárás (depth-first search), Last In First out (LIFO)
  2. cső (queue), szélességi bejárás (breadth-first search), First In First Out (FIFO)
- bonyolultabb problémák esetén: dinamikus programozás, A*

Megfelelőségek

- minden nem-determinisztikus automatának van egy determinisztikus megfelelője
- a reguláris kifejezések ekvivalensek az FSA-kkal: minden reguláris nyelvre építhető egy FSA, és minden FSA-hoz csinálható egy reguláris nyelv

Lewis, H. and Papadimitriu, C. (1981). *Elements of the Theory of Computation*. Prentice-Hall, Englewood Cliffs, NJ.
Hopcroft, J. E. and Ullman, J. D. (1979). *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, Reading, MA.

Morfológiai elemzés transzducerekkel

Morfológiai elemzés

a morfológiai elemzés során az input szót komponensekre bontjuk, és strukturált reprezentációt rendelünk hozzá

miért nem soroljuk fel az összes lehetséges szóalakot?

- produktív toldalékok → minden igéhez/főnévhez/stb. hozzátehetők
- a török igéknek 40.000 lehetséges alakja van – nem számolva a derivációkat
- a magyar igéknek XXX lehetséges alakja van – nem számolva a derivációkat
- plusz derivációk → elméletileg végtelen számú igealak

nem lehetséges az összes szóalak felsorolása → a szólakokat dinamikusan kell elemezni

Tövezés vs. lemmatizálás

tövezés (stemming)

foxes $\rightarrow$ fox

lemmatizálás (lemmatization)

sang, sung, sings $\rightarrow$ sing

Kétszintű morfológia

- felismerés → morfológiai elemzés
- Kimmo Koskenniemi (1983): megfeleltetés egy szó felszíni alakja és elemzése között

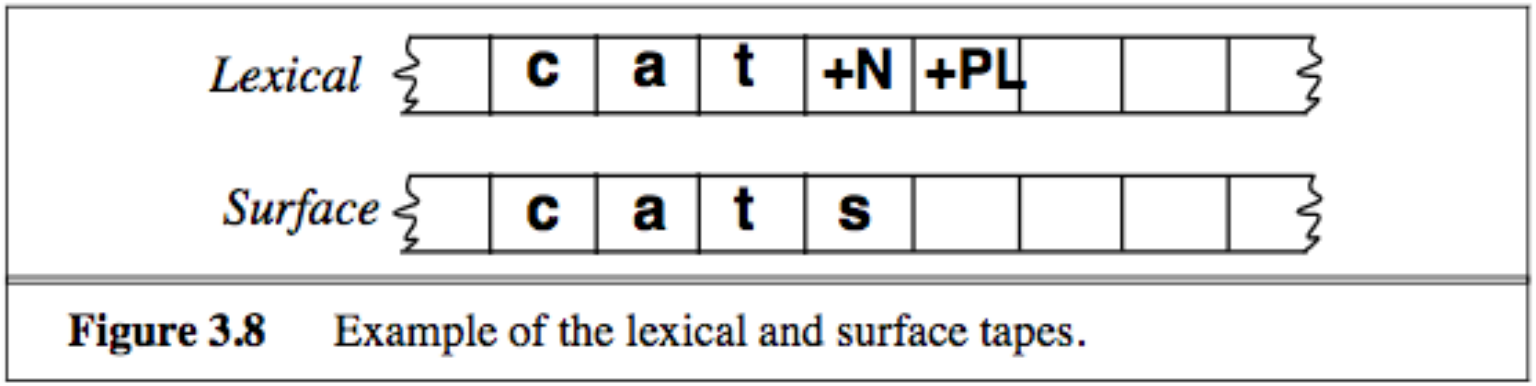


Figure 3.8 Example of the lexical and surface tapes.

Véges állapotú transzducer

Definíció (véges állapotú transzducer)

Az $M =< K, \Sigma, d, q_0, F >$ ötös egy véges állapotú transzducer, ahol
K: a transzducer állapotainak véges halmaza
Σ: az ábécé (komplex szimbólumok halmaza, ahol egy komplex szimbólum egy input-output pár $i{:}o$)
q_0: a kezdőállapot
F: a végállapotok halmaza
d: az átmenetek halmaza, pl. $< q_0, i : o, q_1 >$

Automatából transzducer 1.

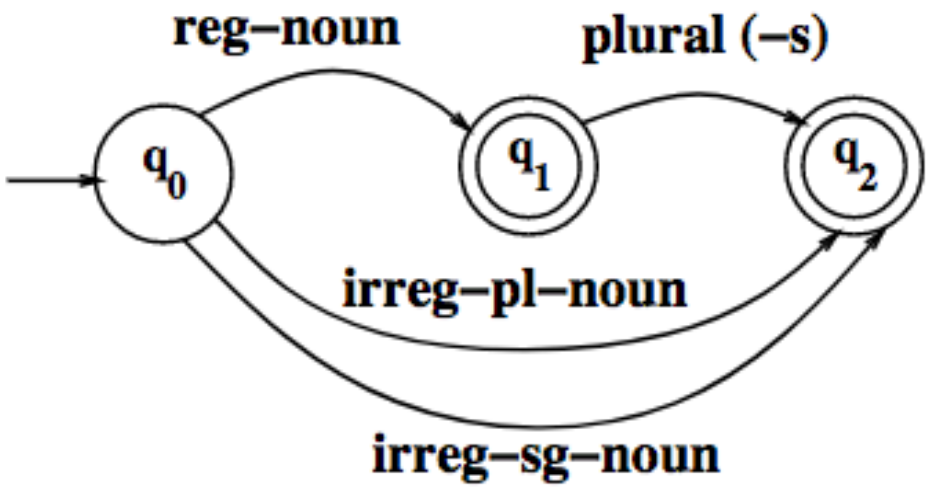


Figure 3.2 A finite-state automaton for English nominal inflection.

reg-noun	irreg-pl-noun	irreg-sg-noun	plural
fox	geese	goose	-s
cat	sheep	sheep	
dog	mice	mouse	
aardvark			

Automatából transzducer 2.

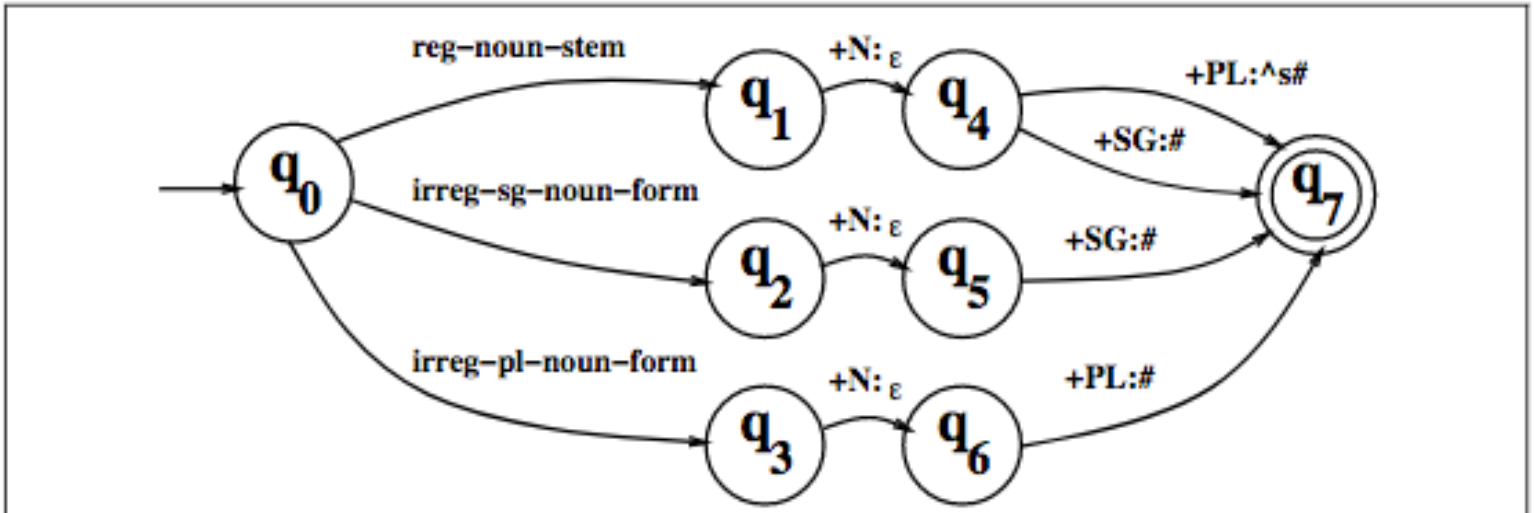


Figure 3.9 A transducer for English nominal number inflection T_{num}. Since both q_1 and q_2 are accepting states, regular nouns can have the plural suffix or not. The morpheme-boundary symbol ^ and word-boundary marker # will be discussed below.

reg-noun	irreg-pl-noun	irreg-sg-noun
fox	g o:e o:e s e	goose
cat	sheep	sheep
dog	m o:i u:ɛ s:c e	mouse
aardvark		

Allomorfok

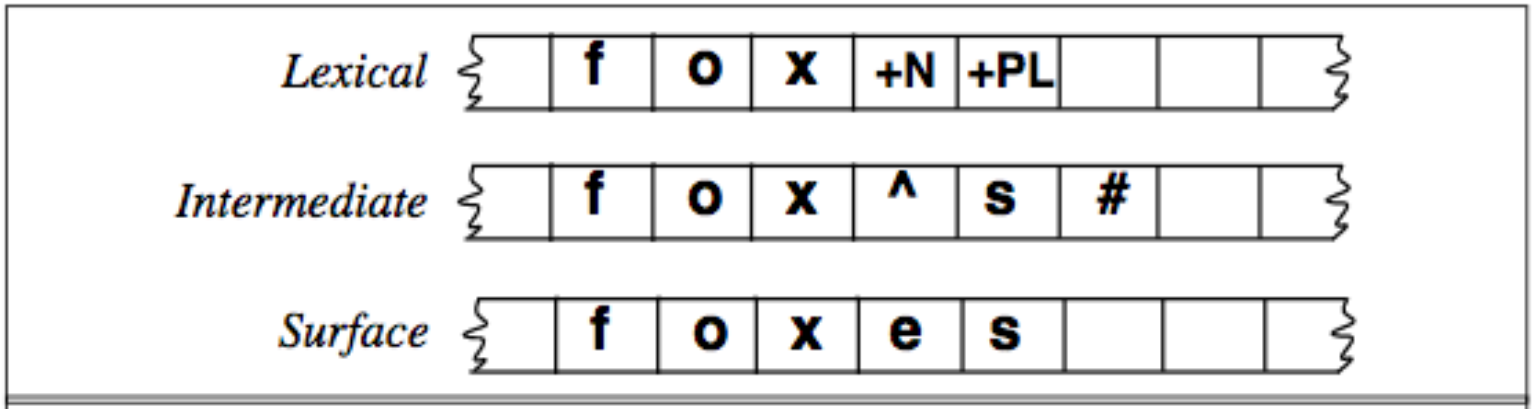


Figure 3.13 An example of the lexical, intermediate and surface tapes. Between each pair of tapes is a 2-level transducer; the lexical transducer of Figure 3.11 between the lexical and intermediate levels, and the E-insertion spelling rule between the intermediate and surface levels. The E-insertion spelling rule inserts an *e* on the surface tape when the intermediate tape has a morpheme boundary ^ followed by the morpheme *-s*.

Agglutinálunk (?)

morfémák:	igekötő	tő	képző	képző	eset	
morfok:	meg	emlék	ez	és	ül	
morfémák:	tő	szám	eset			
morfok:	barát	ok	ért			
morfémák:	igekötő	tő	LEH	képző	képző	eset
morfok:	fel	ismer	het	etlen	ség	ig
morfémák:	tő	Sing	1			
morfok:	könyv	em				
morfémák:	tő	Pl	2			
morfok:	tanul	unk				

Kiefer: A ragozás. In: STRMNy 3.

Kedveljük a rókákat.

emMorph-kimenetek:

1. terminálban futtatva magában:
   `kedvel[V]jük[P1.Def]`
   `róka[N]k[Pl]at[Acc]`
2. e-magyar:
   `kedvel[/V] + jük[Prs.Def.1Pl]`
   `róka[/N]=róká + k[Pl] + at[Acc]`

FST keretrendszerek

- Helsinki Finite-State Technology (HFST): https://hfst.github.io/
- Xerox Finite State Toolkit (XFST)
- Foma: https://fomafst.github.io/
- Stuttgart Finite State Toolkit (SFST): http://www.cis.uni-muenchen.de/~schmid/tools/SFST/
- OpenFST: http://www.openfst.org/twiki/bin/view/FST/WebHome

Házi feladat

Házi feladat

1. Csinálj egy olyan determinisztikus véges állapotú automatát, amely elfogadja az alábbi magyar szavakat és azok többesszámú alakját: *pók, póni, pék, póré, szék*. Prezentáld mindhárom tanult módon (állapotdiagram, formális leírás, állapot-átmenet tábla)! Fontos:

   - az automata csak és kizárólag a felsorolt sztringeket fogadja el, mást ne;
   - az automata a lehető legtömörebb, legegyszerűbb legyen, ne tartalmazzon redundáns állapotokat és átmeneteket.

2. Készítsd el a determinisztikus bégetőautomatának a negáltját, és prezentáld mindhárom tanult módon (állapotdiagram, formális leírás, állapot-átmenet tábla)!

Egy FSA negáltja azokat a sztringeket fogadja el, amiket az FSA elutasít, és azokat utasítja el, amiket az FSA elfogad (ugyanafölött az ábécé fölött).

Ajánlott irodalom

- Alberti Gábor: *Matematika a természetes nyelvek leírásában*. Segédkönyvek a nyelvészet tanulmányozásához 52. Tinta Könyvkiadó, Budapest, 2006.
- Daniel Jurafsky & James H. Martin: *Speech and Language Processing*. 2nd edition. Chapter 2, 16.