Gorana Levačić

Prirodni jezik za komunikaciju

- Gramatike strukture fraza
- Sintaktička analiza (parsiranje)

Prirodni jezik za komunikaciju

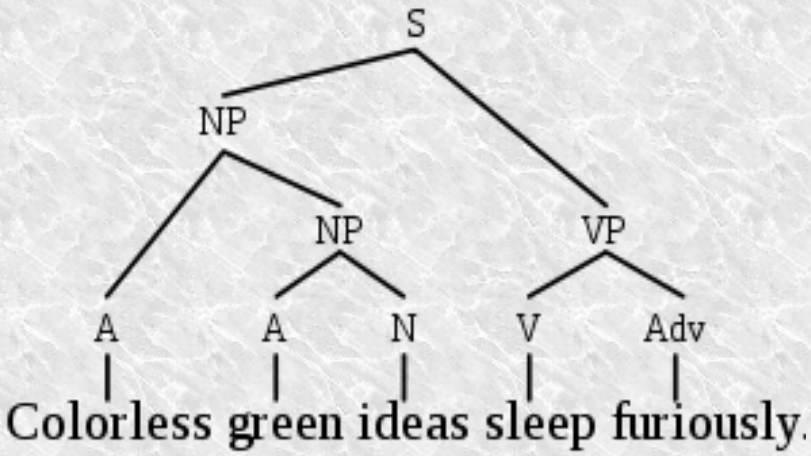
- **komunikacija** – namjerna razmjena informacija koja se ostvaruje produkcijom i percepcijom određenih znakova koji dolaze iz unaprijed dogovorenog skupa znakova za komunikaciju
- agentu komunikacija s drugim agentima omogućuje da sazna nove informacije o svom svijetu

Gramatike strukture fraza

- **Leksičke kategorije:**
  - imenice, pridjevi, glagoli...
  - čine leksikon – skup leksema, pojmova (leksem 'trčati' odgovara riječima 'trčao', 'trčala', 'trčimo'...)

- **Sintaktičke kategorije**:
  - skup riječi ili fraza koje imaju slična svojstva
  - glagolske fraze / verb phrase (VP): *“Kralj je brzo počeo bježati od divlje svinje.”*
  - imeničke fraze / noun phrase (NP): *“Svim normalnim ljudima Zagreba je dosadila dosadna i nepraktična hladnoća.”*

- **struktura fraza** – kombinacija elemenata sintatičkih kategorija, oni grade stablo
- ne mora biti smislena iako je sintaktički točna



- **formalna gramatika** – precizan opis formalnog jezika, tj. skupa nizova znakova
- **generativne gramatike** – skupovi pravila za generiranje nizova znakova jezika

- postoji više modela jezika utemeljenih na ideji struktura fraza
- opisat ćemo **probabilističku kontekstno – neovisnu gramatiku** (probabilistic context-free grammar, **PCFG**)
- **gramatika** – skup pravila koja definiraju jezik kao skup dozvoljenih nizova riječi
- "**probabilistički**" – gramatika pridružuje vjerojatnost svakom nizu
- "**kontekstno-neovisna**" – lijevu stranu pravila produkcije čini jedan neterminalni simbol, a na desnoj strani je niz terminalnih i/ili neterminalnih simbola
- **terminalni simboli** – elementarni simboli jezika koje definira formalna gramatika (riječi)
- **neterminalni simboli** / **sintaktičke varijable** budu po pravilima produkcije zamijenjene terminalnim znakovima

Primjer:

- alfabet: a,b
- početni simbol: S
- pravilo (1): S → aSb
- pravilo (2): S → ba
- pravilo (1) daje string *aSb*, ponovnom primjenom dobijemo *aaSbb*
- pravilo (2) mijenja *S* s *ba* pa dobijemo *aababb*

- primjer pravila produkcije PCFG:

  VP → Verb [0.70]

  → VP NP [0.30]

- VP (verb phrase), NP (noun phrase) su neterminalni simboli
- terminalni simboli su stvarne riječi (*banana, vitez, gomolj*)
- pravilo kaže da se VP s vjerojatnosti 0.7 sastoji samo od glagola, a s vjerojatnosti 0.3 se sastoji od VP koju slijedi NP

- definirat ćemo gramatiku za sitan dio engleskog jezika koji je prikladan za komunikaciju među agentima koji istražuju Wumpusov svijet

- prvo definiramo **leksikon** – listu dozvoljenih riječi
- riječi su grupirane u **leksičke kategorije**:
  - imenice, zamjenice, imena
  - glagoli – označavaju događaje
  - pridjevi – modificiraju imenice
  - prilozi – modificiraju glagole
  - funkcijske riječi – čestice, prijedlozi, veznici

- **otvorene klase** – nemoguće nabrojati sve riječi, kategorija završava s "..."
- **zatvorene klase** – moguće nabrojati sve riječi

Imenice	stench [0.05], breeze [0.10], Wumpus [0.15], pits [0.05]...
Glagoli	is [0.10], feel [0.10], smells [0.10], stinks [0.05] ...
Pridjevi	right [0.10], dead [0.05], smelly [0.02], breezy [0.02] ...
Prilozi	here [0.05], ahead [0.05], nearby [0.02] ...
Zamjenice	me [0.10], you [0.03], I [0.10] ...
Odnosne -||-	that [0.40], which [0.15], who [0.20], whom [0.02] ...
Imena	John [0.01], Mary [0.01], Boston [0.01] ...
Čestice	the [0.40], a [0.30], an [0.10], every [0.05] ...
Prijedlozi	to [0.20], in [0.10], on [0.05], near [0.10] ...
Veznici	and [0.50], or [0.10], but [0.20], yet [0.02] ...
Znamenke	0 [0.20], 1 [0.20], 2 [0.20], 3 [0.20], 4 [0.20]

- potom definiramo gramatiku – kombiniramo riječi u fraze
- **stablo parsiranja / sintaktičke analize**:
  - predstavlja sintaktičku analizu rečenice
  - daje konstruktivan dokaz da je dobiven niz riječi pravilna rečenica
- **problemi**:
  - stvaranje neispravnih fraza: *Me go eat Boston.*
  - odbacivanje ispravnih rečenica: *I think Wumpus is smelly.*
- **sintaktičke kategorije**:
  - rečenica (Reč)
  - imenička, glagolska fraza (NP, VP)
  - lista pridjeva (Adjs)
  - propozicijska fraza (PP)
  - odnosna (su)rečenica (OdnReč)

Pravila:

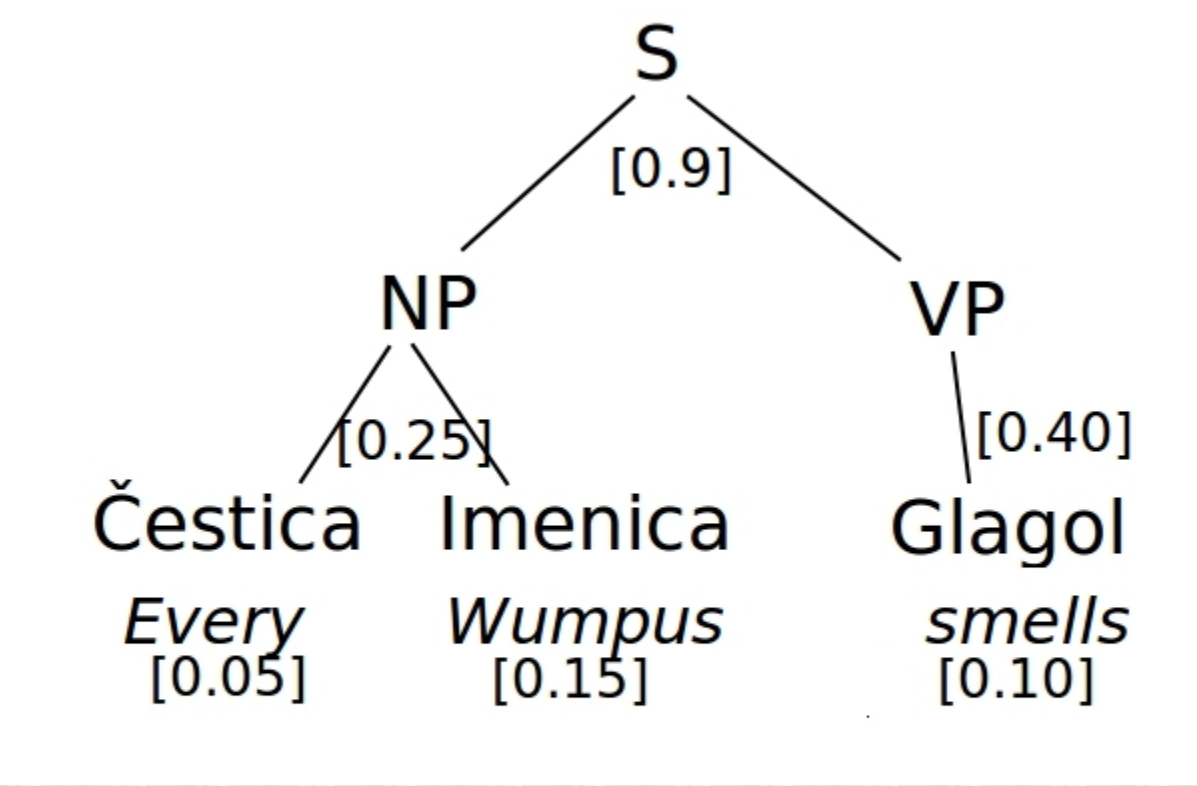
Reč → NP VP	[0.90] I + feel a breeze
→ Reč Conj Reč	[0.10] I feel a breeze + and + it stinks
NP → Zamjenica	[0.30] I
→ Ime	[0.10] John
→ Imenica	[0.10] pits
→ Čestica Imenica	[0.25] the + Wumpus
→ Čestica Adjs Imenica	[0.05] the + smelly dead + Wumpus
→ Znamenka Znamenka	[0.05] 3 4
→ NP PP	[0.10] the wumpus + in 1 3
→ NP OdnReč	[0.05] the wumpus + that is smelly

Pravila:

VP → Glagol	[0.40] stinks
→ VP NP	[0.35] feel + a breeze
→ VP Pridjev	[0.05] smells + dead
→ VP PP	[0.10] is + in 1 3
→ VP Prilog	[0.10] go + ahead
Adjs → Pridjev	[0.80] smelly
→ Pridjev Adjs	[0.20] smelly + dead
Adjs → Pridjev	[0.80] smelly
→ Pridjev Adjs	[0.20] smelly + dead
PP → Prijedlog NP	[1.00] to + the east
OdnReč → OdnZamj. VP	[1.00] that + is smelly

Stablo parsiranja za rečenicu *Every Wumpus smells*:

- svaki čvor ima za oznaku svoju vjerojatnost
- vjerojatnost stabla je 0.9 * 0.25 * 0.40 * 0.05 * 0.15 * 0.10 = 0.0000675
- ovo je jedino moguće stablo za ovu rečenicu, pa je gornji broj ujedno i vjerojatnost same rečenice



Parsiranje / sintaktička analiza

- proces analiziranja niza riječi s ciljem otkrivanja njegove strukture fraza, prema pravilima gramatike

Pristupi:

- počnemo sa simbolom S i od vrha prema dnu tražimo (gradimo) stablo koje ima riječi iz niza kao listove
- počnemo od niza riječi i idemo prema vrhu dok ne dobijemo stablo čiji korijen je S
- oba mogu biti neučinkovita jer mogu zaglaviti u slijepoj cesti

Primjer:

Have the students in section 2 of Computer Science 101 take the exam.

Have the students in section 2 of Computer Science 101 taken the exam?

- prvih 10 riječi je jednako, ali analize su različite (pitanje ili naredba?)
- algoritam parsiranja slijeva udesno bi morao pogađati je li prva riječ dio naredbe ili pitanja i tek kod 11.riječi bi znao je li dobro pogodio
- ako krivo pogodi, mora se vraćati na početak i iznova analizirati rečenicu ⇒ koristi se dinamičko programiranje
- "the students in section 2 of Computer Science 101" je NP i to zabilježimo u stukturi podataka zvanoj chart (grafikon)
- algoritmi koji se time bave se zovu chart parseri, ima ih mnogo, mi ćemo opisati **CYK algoritam**

- CYK algoritam za dani niz riječi nalazi najvjerojatniji izvod za cijeli niz i svaki podniz
- vraća tablicu P u kojoj je P(X,start,len) vjerojatnost najvjerojatnijeg niza X duljine *len* koji počinje na poziciji *start*
- ako nema X te veličine na toj poziciji, vjerojatnost je 0
- CYK algoritam zahtijeva gramatiku u **Chomsky normalnoj formi**
  - leksička pravila su oblika X → riječ
  - sintaktička pravila su oblika X → Y Z
  - bilo koja CFG se može prebaciti u normalnu formu

- za rečenicu duljine n i m neterminalnih simbola u gramatici:
  - prostorna složenost: *O(nm)* = *O(n)* jer je m konstantan
  - *vremenska složenost:* $O(n^3)$ ⇒ najbrži algoritam za općenite CFG

- uz A* algoritam pretraživanja i dobre heuristike CYK algoritam može imati i linearnu složenost ⇒ ne trebaju nam sva rješenja, nego samo najvjerojatnije (ili par najvjerojatnijih)

function CYK-PARSE($words$, $grammar$) **returns** P, a table of probabilities

$N \leftarrow$ LENGTH($words$)
$M \leftarrow$ the number of nonterminal symbols in $grammar$
$P \leftarrow$ an array of size $[M, N, N]$, initially all 0
/ Insert lexical rules for each word */*
for $i = 1$ **to** N **do**
  for each rule of form $(X \rightarrow words_i\ [p])$ **do**
    $P[X, i, 1] \leftarrow$ p
/ Combine first and second parts of right-hand sides of rules, from short to long */*
for $length = 2$ **to** N **do**
  for $start = 1$ **to** $N - length + 1$ **do**
    for $len1 = 1$ **to** $N - 1$ **do**
      $len2 \leftarrow length - len1$
      for each rule of the form $(X \rightarrow Y\ Z\ [p])$ **do**
        $P[X, start, length] \leftarrow \text{MAX}(P[X, start, length],$
          $P[Y, start, len1] \times P[Z, start + len1, len2] \times p)$
return P

Učenje:

- PCFG ima puno pravila, svako pravilo ima svoju vjerojatnost ⇒ bolje naučiti gramatiku iz podataka nego graditi znanje od nule
- učenje je najlakše iz skupa točno parsiranih rečenica (treebank)
  - The Penn Treebank je najpoznatiji – 3 milijuna riječi uz stabla parsiranja i dodatne podatke

- iz danog skupa stabala možemo stvoriti PCFG samo prebrojavajući:
  - imamo 100 000 (pod)stabala sa S u korijenu
  - imamo 60 000 čvorova oblika [S [NP . . .][VP . . .]]

  ⇒ stvaramo pravilo S → NP VP [0.60]

- moguće je učiti gramatiku samo iz neanaliziranih rečenica, ali je teže:
  - problemi: naučiti strukturu pravila i vjerojatnosti vezane uz pravila
  - imamo imena leksičkih i sintaktičkih kategorija
  - gramatika uključuje sva moguća pravila oblika X → Y Z ili X → riječ

- koristi se **inside-outside** algoritam
  - pretpostavimo da su vjerojatnosti pravila neki nasumični ili jednoliko raspoređeni brojevi, potom pokušavamo saznati stvarne vjerojatnosti pravila
  - spor algoritam – $O(n^3m^3)$, n broj riječi u rečenici, m broj kategorija
  - često daje nesuvisle gramatike i može zapeti u lokalnom maksimumu (krivo procjeniti vjerojatnosti)

- zato se često koristi kombinirani pristup
  - uvodimo prototipe – ubacimo jednostavna pravila, naučimo kompliciranija
  - koristimo treebanke kao temelj, ali učimo nova pravila iz rečenica koje nisu u treebanku

Usporedba kontekstno-neovisnih i Markovljevih modela:

- u PCFG je razlika između P (“eat a banana”) i P (“eat a bandanna”) utemeljena na (Noun → “banana”) tj. P (Noun → “bandanna”), a ne na vezi između “eat” i odgovarajućeg objekta
- Markovljev model višeg reda, ako mu damo dovoljno veliki skup rečenica, će znati da je prva rečenica puno vjerojatnija – to je model definiran Markovljevim lancima u kojem vjerojatnost n-te riječi ovisi o prethodnim riječima
- zato se često ti pristupi kombiniraju