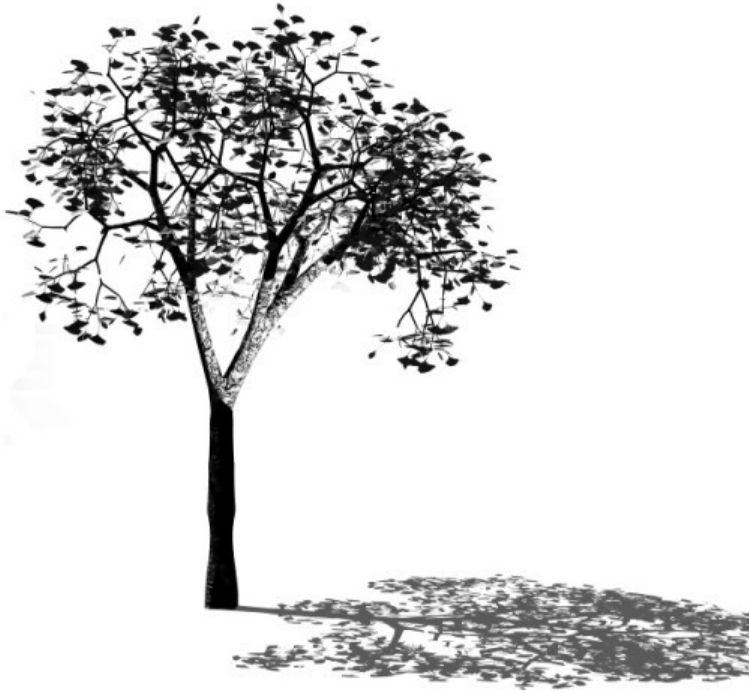
Parsery języków naturalnych

Mateusz Biliński

Sebastian Kuligowski

Left-corner parsing with LoPar



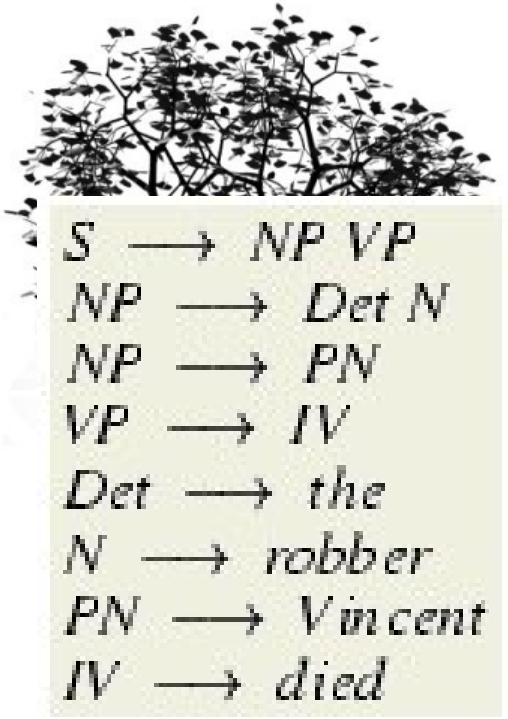
Skuteczność algorytmu top-down

Vincent died

I podejście
S -> NP VP
S -> Det N VP

Dla Det nie istnieje produkcja ze słowem Vincent

II podejście
S -> NP VP
S -> PN VP
S-> *Vincent* IV
S-> *Vincent died*



Skuteczność algorytmu bottom-up

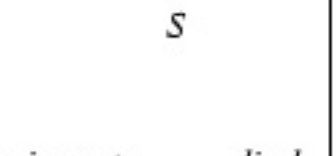
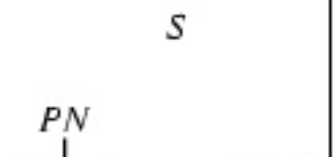
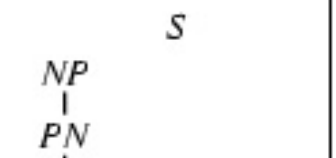
the plant died

the plant died
Det *plant died*
Det TV *died'*
Det TV IV
Det TV VP

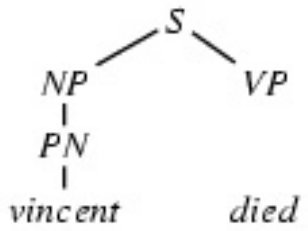
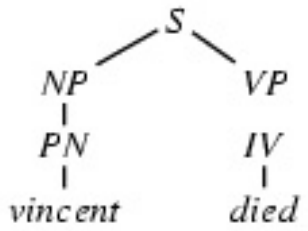
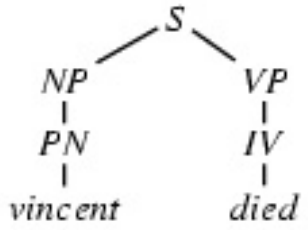
Nie występuje żadna redukcja
the plant died
Det *plant died*
Det N *died*
NP IV
NP VP
S

$$
\begin{aligned}
S &\longrightarrow NP\ VP \\
NP &\longrightarrow Det\ N \\
VP &\longrightarrow IV \\
VP &\longrightarrow TV\ NP \\
TV &\longrightarrow plant \\
IV &\longrightarrow died \\
Det &\longrightarrow the \\
N &\longrightarrow plant
\end{aligned}
$$

Hybryda: top-down + bottom-up (1)

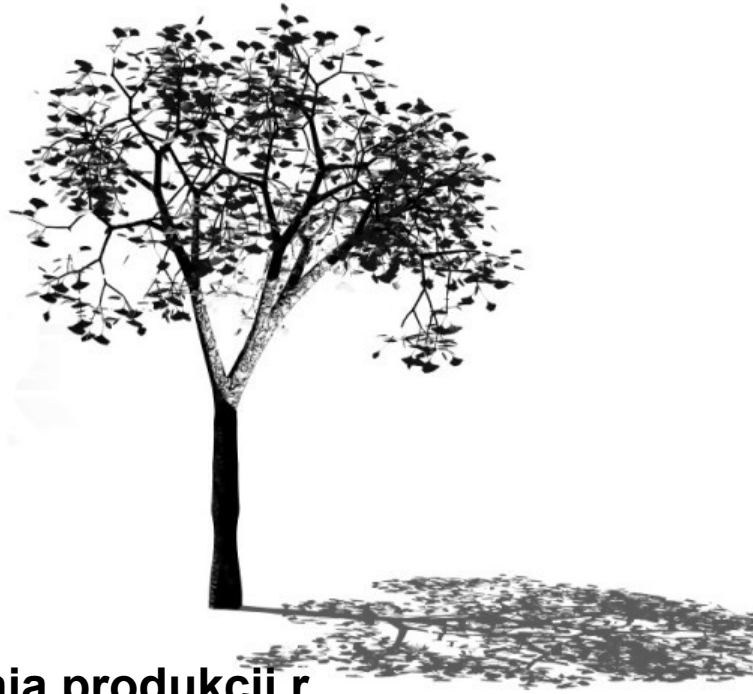
	**Krok 1**: Wprowadzenie **S**. (Top-down)
	**Krok 2**: Kategoria pierwszego słowa na wejściu to **PN**. (Bottom-up, reguła leksykalna)
	**Krok 3**: Wybranie reguły, która posiada **PN** po lewej stronie. (Bottom-up)

Hybryda: top-down + bottom-up (2)

	**Krok 4**: Wybranie reguły, która posiada **NP** po lewej stronie. Jest to **S-> NP VP** **Krok 5**: Następuje dopasowanie do istniejącej reguły.
	**Krok 6**: Rozpoznanie **VP** (Top-down) **Krok 7**: Kategoria słowa **died** to **IV**. (Bottom-up)
	**Krok 8**: Wybranie produkcji, w której po lewej stronie występuje IV. Jest nią **VP -> IV**. **Krok 9**: Przypasowanie i zakończenie parsowania.

PCFG i HPCFG

Probabilistyczna gramatyka bezkontekstowa jest taką gramatyką, która przypisuje prawdopodobieństwo *P(r)* dla każdej produkcji *r* danej gramatyki.



PCFG = CFG + prawdopodobieństwo P(r) wykonania produkcji r

W przypadku występowania jednakowych prawdopodobieństw dla kilku produkcji, stosuje się wyznaczanie tzw. głowy po prawej stronie produkcji. Przykładowo: **NP -> Det N'**

LoPar - pliki

Plik z gramatyką

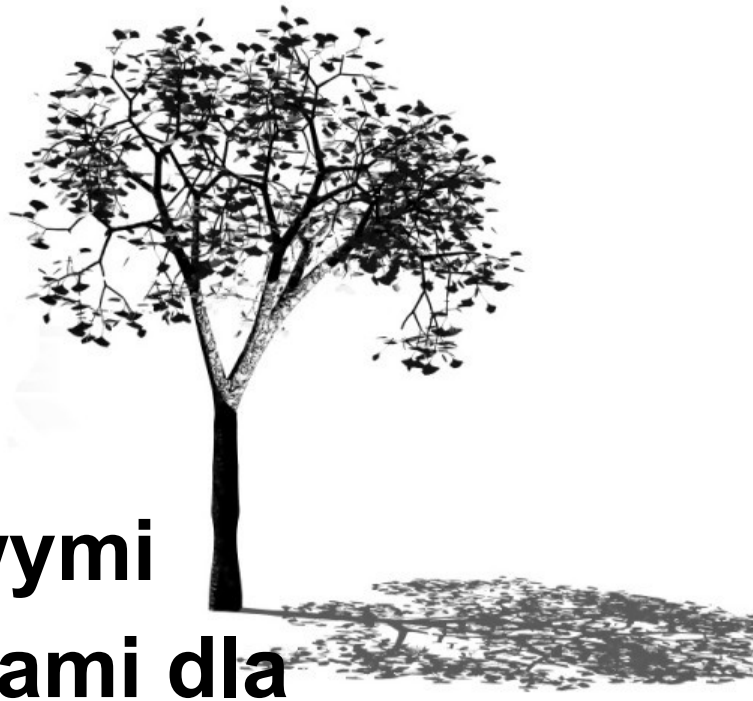
100 S NP VP'
200 VP VP' PP

Plik z leksykonem

Peter	NP 1 Peter
Mary	NP 1 Mary

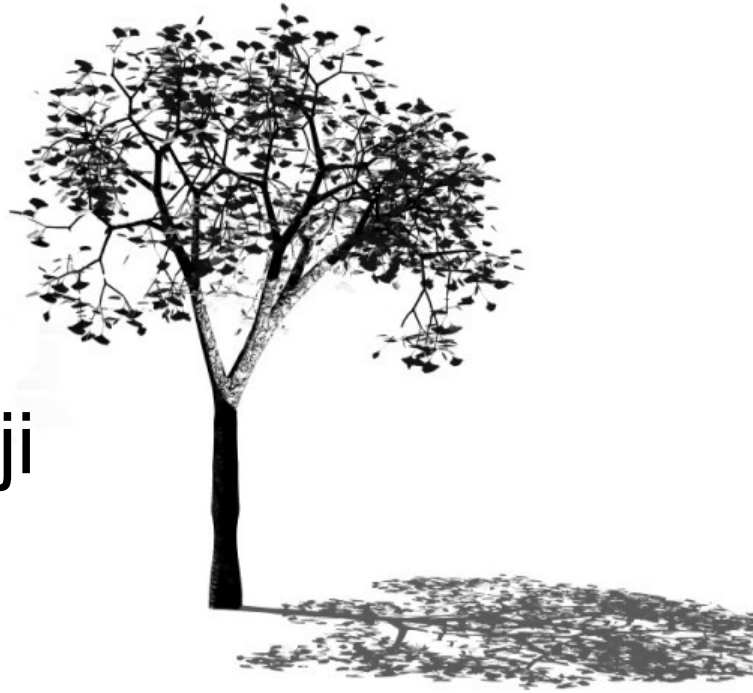
Plik z symbolami startowymi

Plik z otwartymi kategoriami dla nieznanych słów



LoPar – trenowanie gramatyki

Po stworzeniu odpowiedniej gramatyki należy poddać ją tzw. „treningowi”. Dokonuje się on w trzech krokach:



Trening bez leksykalizacji

Leksykalizacja

Trening z leksykalizacją

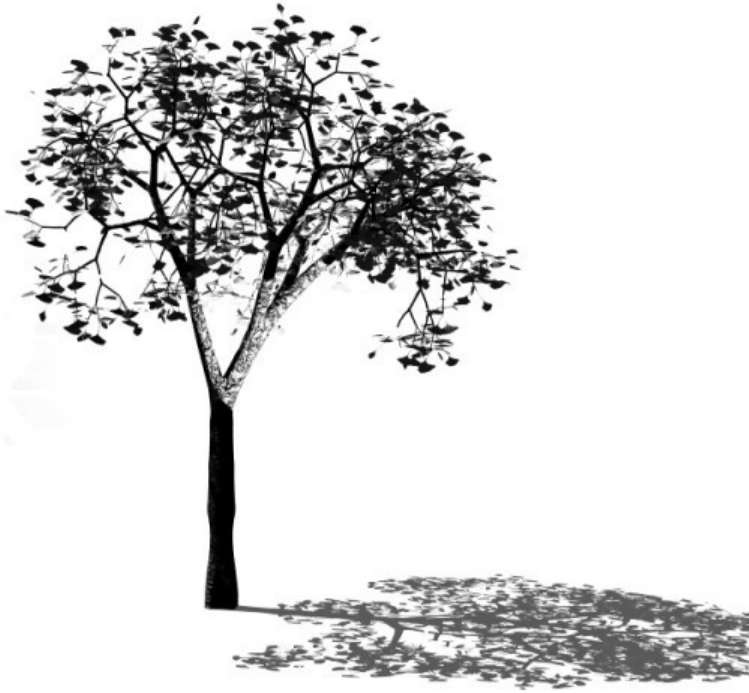
Przetestuj!

ssh 194.146.253.28

Login: lopar

Hasło: tk800lopar

cat README



Combinatory Categorial Grammar

http://openccg.sourceforge.net/

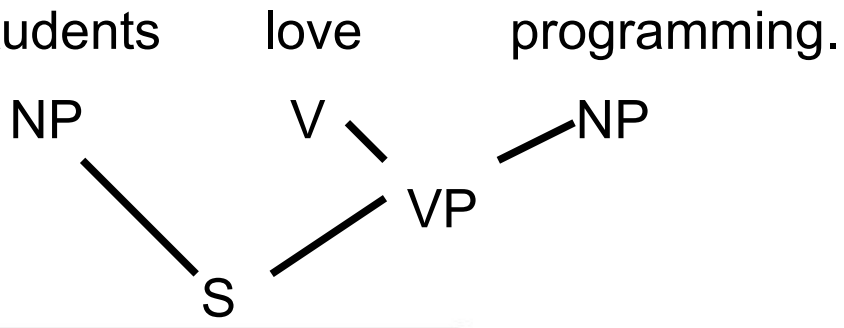
¡ OpenCCG

Students love programming.

NP (S\NP)/NP NP

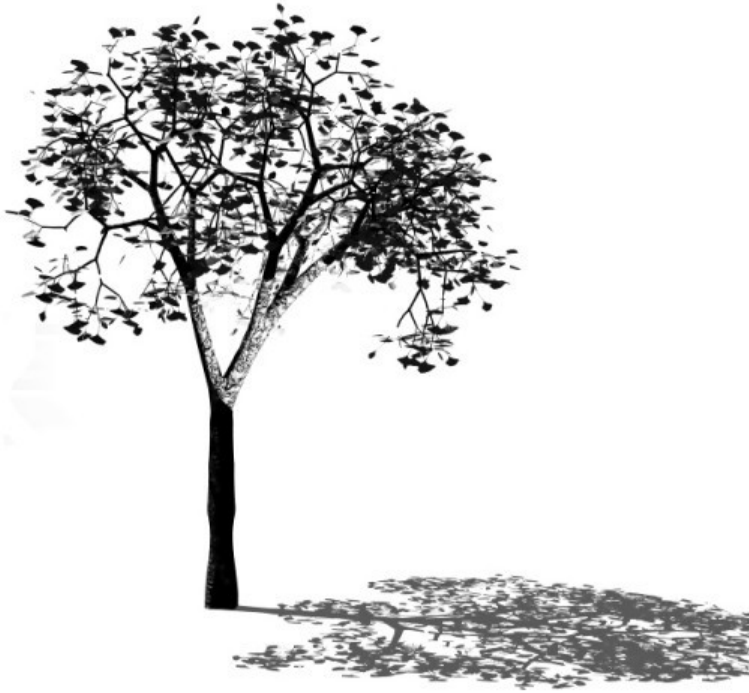
S\NP >

S <



Tworzenie gramatyk - kategorie

```
<family name="Noun" pos="N">
  <entry name="Primary">
    <atomcat type="n">
      <fs id="2">
        <feat attr="num">
          <featvar name="NUM"/>
        </feat>
      [...]
      </fs>
    </atomcat>
  </entry>
</family>
```

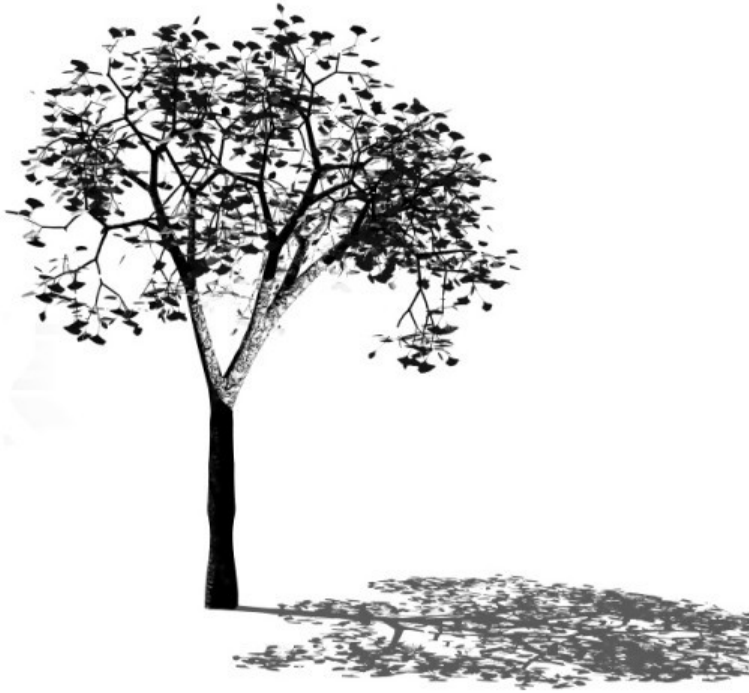


Tworzenie gramatyk

```
<family name="ProNP" pos="Pro" closed="true">
  <entry name="Primary">
    [...]
  </entry>
  <member stem="pro1"/>
  <member stem="pro2"/>
  <member stem="pro3f"/>
  <member stem="pro3m"/>
  <member stem="pro3n"/>
</family>
```

Czasownik przechodni

```
<family name="TransitiveVerbs" pos="V" closed="true">
  <entry name="Primary">
    <complexcat>
      <atomcat type="s">
        <fs id="1"> [...] </fs>
      </atomcat>
      <slash dir="\" mode="&lt;"/>
      <atomcat type="np">
        <fs id="2">
          <feat attr="case" val="nom"/>
          [...]
      </atomcat>
      <slash dir="/" mode="&gt;"/>
      <atomcat type="np">
        <fs id="3">
          <feat attr="case" val="acc"/>
          [...]
        </fs>
      </atomcat>
    </complexcat>
</family>
```



$$s_{\langle 1\rangle} \backslash np_{\langle 2\rangle nom} / np_{\langle 3\rangle acc}$$

Makra przypadków i osób

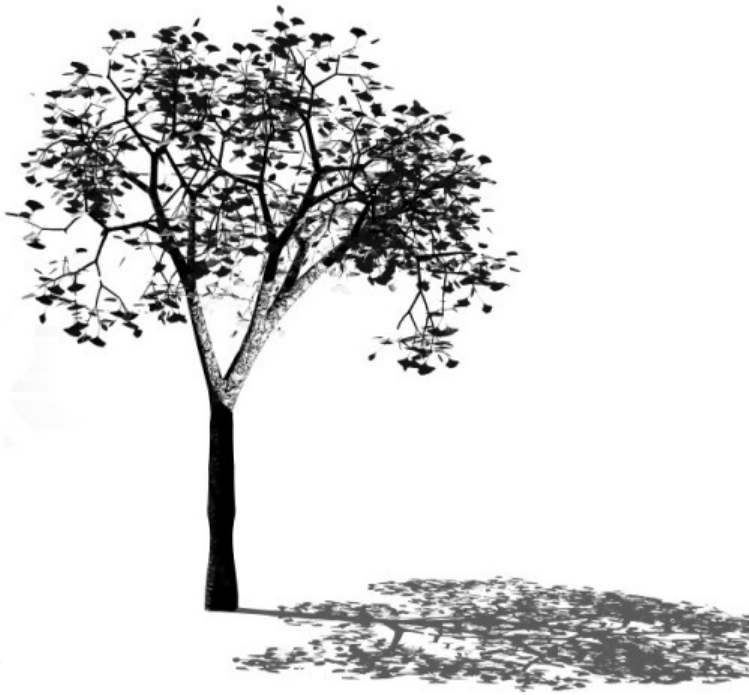
```
<entry pos="Pro" word="I" stem="pro1"
     macros="@1st @sg @nom .."/>
<entry pos="Pro" word="me" stem="pro1"
     macros="@1st @sg @acc .."/>
<entry pos="Pro" word="we" stem="pro1"
     macros="@1st @pl @nom .."/>
<entry pos="Pro" word="us" stem="pro1"
     macros="@1st @pl @acc .."/>
:
<macro name="@nom">
<fs id="2" attr="case" val="nom"/>
</macro>

<macro name="@acc">
<fs id="2" attr="case" val="acc"/>
</macro>
```

I |- np <2>*1st,sg,nom*

me |- np <2>*1st,sg,acc*

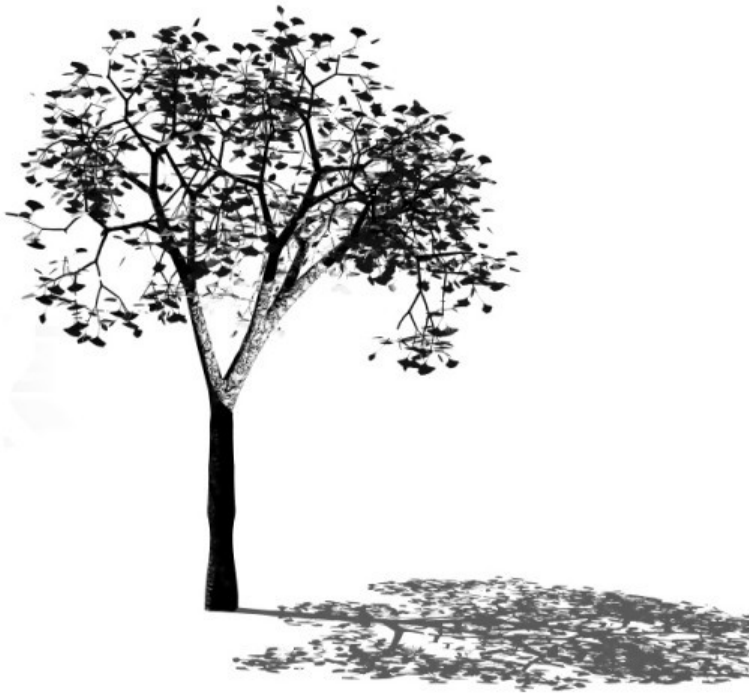
we |- np <2>*1st ,pl,nom*

us |- np <2>*1st ,pl,acc*



Makra przypadków i osób

```
<entry pos="V" word="buy" macros="@pres @non-3rd @sg"/>
<entry pos="V" word="buys" stem="buy"
     macros="@pres @3rd @sg"/>
<entry pos="V" word="buy" macros="@pres @pl"/>
<entry pos="V" word="bought" stem="buy" macros="@past"/>
:
<macro name="@1st">
  <fs id="2" attr="pers" val="1st"/> </macro>
<macro name="@2nd">
  <fs id="2" attr="pers" val="2nd"/> </macro>
<macro name="@3rd">
  <fs id="2" attr="pers" val="3rd"/> </macro>
<macro name="@non-3rd">
  <fs id="2" attr="pers" val="non-3rd"/>
</macro>
```



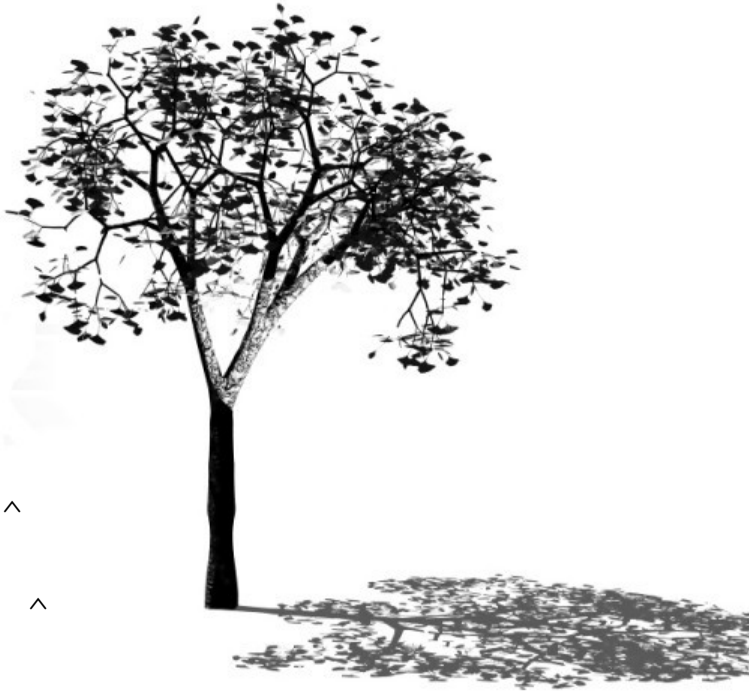
buy |- $s_{<1>} \backslash np_{<2>non3rd,sg,nom} / np_{<3>acc}$

Przykłady parsingu

```
tccg> she buys it
3 parses found.

Parse 1: s/pp :
  @b1:action(buy ^
             <tense>pres ^
             <Actor>(p1:animate-being ^ pro3f
                     <num>sg) ^
             <Beneficiary>x1:animate-being ^
             <Patient>(p2:thing ^ pro3n ^
                       <num>sg))
```

Przykłady parsingu

```
tccg> she buys the flower for the teacher
1 parse found.

Parse: s :
  @b1:action(buy ^
             <tense>pres ^
             <Actor>(p1:animate-being ^ pro3f ^
                     <num>sg) ^
             <Beneficiary>(t1:person ^ teacher ^
                           <det>the ^
                           <num>sg) ^
             <Patient>(f1:thing ^ flower ^
                       <det>the ^
                        <num>sg))
```



Przykłady parsingu

```
tccg> the teacher rented a DVD for her
1 parse found.

Parse: s :
  @r1:action(rent ^
             <tense>past ^
             <Actor>(t1:person ^ teacher ^
                     <det>the ^
                     <num>sg) ^
             <Beneficiary>(p1:animate-being ^ pro3f ^
                           <num>sg) ^
             <Patient>(d1:thing ^ DVD ^
                       <det>a ^
                       <num>sg))
```

Pliki gramatyk

grammar.xml – nazwa gramatyki i lista pozostałych plików

lexicon.xml – specyfikuje leksykalne rodziny (kategorie)

morph.xml – specyfikuje słowa gramatyki

rules.xml – określa, które kombinatoryczne reguły są dostępne dla gramatyki (aplikacja, kompozycja)