Kategorialny Parser Składniowo-Semantyczny „ENIAM”

dokumentacja techniczna

Wojciech Jaworski

21 stycznia 2017

1 Ogólny opis

Kategorialny Parser Składniowo-Semantyczny „ENIAM” jest narzędziem generującym *formy logiczne/reprezentacje semantyczne* dla zdań w języku polskim. Parser pracuje na niepreparowanych danych, realizuje kolejne etapy przetwarzania tekstu: tokenizację, lematyzację, rozpoznawanie związków składniowych, anotację sensami słów oraz rolami tematycznymi, częściową dezambiguację oraz tworzenie reprezentacji semantycznej.

Poniższy dokument opisuje w jaki sposób zaimplementowane są poszczególne komponenty parsera.

2 Preprocessing

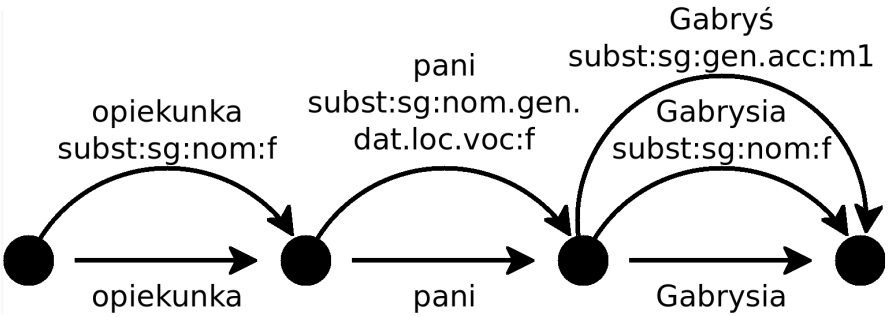
Kolejne etapy przetwarzania tekstu realizowane są w ramach rozmytego potoku przetwarzania. Parser nie dezambiguuje na bieżąco niejednoznaczności powstającej po każdym kroku przetwarzania tekstu. Zamiast tego tworzy zwartą reprezentację niejednoznacznego wyniku, którą przekazuję do następnego etapu, dezambiguację wykonuje pod koniec potoku przetwarzania.

Podejście to jest uzasadnione spostrzeżeniem, że dezambiguacja działa poprawnie jedynie w pewnym procencie przypadków, a złożenie kilku procedur, które nie działają całkowicie poprawnie radykalnie zmniejsza szansę uzyskania poprawnego wyniku. Np. tagery dla języka polskiego mają skuteczność ok 93%, w zdaniu występuje średnio 15 słów, więc prawdopodobieństwo poprawnego otagowania typowego zdania wynosi $0,93^{15} = 0,3367$.

Początkowe etapy przetwarzania tekstu (poprzedzające określenie struktury zależnościowej) będziemy określać mianem preprocesingu. Etapy te

realizowane są przez program `pre` stanowiący wolnostojący serwer sieciowy, z którym komunikuje się parser `eniam`.

Podczas preprocesingu tekst reprezentowany jest jako graf, którego krawędzie etykietowane są tokenami. Do takiego grafu dodawane są i usuwane nowe krawędzie i wierzchołki. Przykładowy graf:



Podczas preprocesingu następuje integracja zasobów. Przetwarzane zdanie anotowane jest m. in. informacjami pochodzącymi następujących zasobów: SGJP, Słowosieć, Walenty. Kolejne etapy preprocesingu to tokenizacja, lematyzacja, wykrywanie nazw własnych, określanie sensów słów, określanie walencji.

2.1 Tokenizacja

Przed właściwą tokenizacją następuje podział tekstu na znaki, klasyfikacja znaków na litery wielkie, litery małe, cyfry, symbole i pozostałe znaki. Literom wielkim przyporządkowywane są ich małe odpowiedniki. Analizowane są następujące bloki Unicode: Basic Latin, Latin-1 Supplement, Latin Extended-A, Latin Extended-B, Latin Extended Additional, General Punctuation, Currency Symbols.

Następnie ma miejsce łączenie znaków w tokeny. Tokeny należą do następujących typów:

- sekwencje cyfr wraz z typami np: liczby naturalne, sekwencje 3-cyfrowe, numery miesięcy
- liczby rzymskie wraz z tłumaczeniem na arabskie
- sekwencje liter z podziałem na sekwencje małych liter, sekwencje wielkich, sekwencje małych poprzedzone wielką i inne.
- symbole,

- proste i złożone znaki interpunkcyjne,
- emotikony.

Różnica pomiędzy symbolem a znakiem interpunkcyjnym polega na tym, że symbole są elementami złożonych tokenów, a znaki interpunkcyjne uczestniczą w analizie składniowej.

W ramach sekwencji liter następuje interpretacja użycia wielkich liter. Jeśli sekwencja zaczyna się od wielkiej litery może należeć do jednej z trzech kategorii: może być nazwą własną, być początkiem zdania lub nazwą własną na początku zdania. Jeśli sekwencja składa się z samych wielkich liter może być napisana tak z przyczyn stylistycznych lub być akronimem — zapis nie wskazuje wtedy czy jest to nazwa własna lub początek zdania. Rozpoznawane są też sufiksy: -em, -m, -eś, -ś, -eśmy, -śmy, -eście, -ście, -by i traktowane jako odrębne tokeny.

Podczas tokenizacji symboli następuje ujednolicenie symboli używanych do zapisania znaków interpunkcyjnych takich jak spacje, cudzysłowy, apostrofy, myślniki czy dywizy. Przykładowo znak ’’ interpretowany jest jako cudzysłów otwierający ’„’ lub cudzysłów zamykający ’”’, dwa następujące po sobie przecinki ’„’ interpretowane są jako cudzysłów otwierający, a dwa następujące po sobie apostrofy interpretowane są jako cudzysłów zamykający.

Haplologia kropki obsługiwana jest poprzez przypisanie temu symbolowi trzech możliwych interpretacji: tokenu końca zdania, tokenu symbolu (np. elementu skótu), tokenu końca zdania poprzedzonego tokenem symbolu. Podobnie rozwiązywana jest haplologia wielokropka. Przykładowo kropka fragment w poniższym tekstu

$$\ldots w\ XV\ w.\ \ Warszawa \ldots,$$

może w połączeniu z poprzedzającą literą ’w’ stanowić skrót formy ’wieku’, a niezależnie od tego może stanowić koniec zdania. Jeśli kropka ta oznacza koniec zdania, wielka liter w formie ’Warszawa’ może oznaczać początek zdania i nazwę własną, lub jedynie początek zdania.

Przecinek interpretowany jest jako początek i koniec zdania składowego, symbol (np. element notacji dziesiętnej), znak interpunkcyjny oznaczający koordynację.

Kolejnym krokiem jest rozpoznawanie złożonych tokenów obejmujące liczby

$$\begin{gathered}\text{pref3dig . 3dig . 3dig . 3dig} \rightarrow \text{natnum} \\ \text{day . month . year} \rightarrow \text{date} \\ \text{- natnum , dig} \rightarrow \text{realnum}\end{gathered}$$

odmienione akronimy i nazwiska obce

* - ów → *:subst:pl:acc:m1|subst:pl:gen:m1.m2.m3.n2
* - cie → *T:subst:sg:loc.voc:m3
* - cie → *TA:subst:sg:dat.loc:f
* - owscy → *-owski:adj:pl:nom.voc:m1.p1:pos
* ' ego → *:subst:sg:gen.acc:m1

leksemy wielotokenowe

ping - ponga → ping-pong:subst:sg:gen.acc:m2
rock ' n ' rollem → rock'n'roll:subst:sg:inst:m2

i skróty rozwijane do sekwencji wielu tokenów

br . → bieżący:adj:sg:$C:m3:pos rok:subst:sg:$C:m3

Wyszukiwane są tu najdłuższe dopasowania, a znalezione dopasowania zastępują rozpoznane sekwencje tokenów.

2.2 Lematyzacja

Lematyzacja wykonywana jest na podstawie SGJP-20160724.W pierwszym kroku na podstawie końcówki słowa odnajdywane są możliwe lematy i interpretacje. W drugim kroku następuje wybór znanych lematów spośród odnalezionych. Jeśli żaden znany lemat nie jest wystąpił wśród odnalezionych zwracane są wszystkie odnalezione lematy i interpretacje. Tokeny będące sekwencjami małych liter poprzedzonych wielką literą są lematyzowane w swojej pierwotnej postaci, a w razie niepowodzenia w wersji ze zmniejszoną pierwszą literą.

Kotem → kotem → kot subst:sg:inst:m2 → Kot subst:sg:inst:m2

Na tym etapie wykonywane jest też rozwijanie skrótów oraz rozpoznawanie wyrażeń wielosłownych.

2.3 Nazwy własne

Oznaczenie nazwami własnymi wykonywane jest za pomocą list wytworzonych na podstawie SGJP-20151020 oraz Polimorf-20151020, uzupełnionych o inicjały, nazwy dni tygodnia i miesięcy. Nazwom przyporządkowywane są ich typy np.: toponim, nazwisko oraz typy obiektów wskazywanych przez nazwy np.: obszar, osoba. Aktualnie tylko rzeczowniki mogą być nazwami własnymi. Rzeczowniki napisane wielką literą, ale nie znalezione w słowniku nazw własnych traktowane są jako nazwy własne nieznanego typu.

2.4 Sensy słów

Sensy słów przypisywane są do lematów na podstawie Słowosieci 2.1.0. Sensy reprezentowane są jako lematy zaopatrzone w numery wariantów i uzupełnione są o listy hiperonimów. Hiperonimy to synsety reprezentowane przez kanonicznie wybrane sensy. Nazwom własnym przypisywane są sensy wynikające z ich typów. Na przykład leksemowi *zamek* zostaną przypisane m. in. następujące sensy wraz z hiperonimami:

zamek 1 budowla: rezultat 1, wytwór 1, konstrukcja 1, budowla 1, budynek 1, dom 1, rezydencja 1, zamek 1

zamek 2 urządzenie do zamykania: obiekt 2, rzecz 4, przedmiot 1, zamknięcie 12, zamek 2

zamek 6 suwak: obiekt 2, rzecz 4, przedmiot 1, zamknięcie 12, zapięcie 2, zamek błyskawiczny 1

2.5 Walencja leksemów

Walencja leksemów zawartych w zdaniu ustalana jest na podstawie słownika walencyjnego Walenty w wersji dnia 2016.04.12. *Walenty* zawiera schematy opisujące cechy składniowe podrzędników wybranych czasowników, rzeczowników, przymiotników i przysłówków, ramy opisujące cechy semantyczne (role tematyczne i preferencje selekcyjne) podrzędników, oraz powiązania pomiędzy schematami i ramami. Na potrzeby parsowania łączymy ramy ze schematami.

Przed parsowaniem następuje dezambiguacja (selekcja) hiperonimów i preferencji selekcyjnych: pozostają tylko te hiperonimy, dla których występuje w segmencie odpowiednia preferencja selekcyjna i tylko te preferencje selekcyjne, dla których występuje w segmencie odpowiedni hiperonim. Predefiniowane w *Walenty* preferencje selekcyjne zostały dodane do relacji hiperonimii w *Słowocieci*.

Schematy zawarte w Walentym mają charakter ogólny opisują podrzędniki leksemów w sposób niezależny od ich formy gramatycznej. Aby zastosować je w praktyce trzeba je przetworzyć tak, by dostosować je do formy leksemu występującej w zdaniu. Proces ten ma w dużej mierze charakter dekompresji.

Etapy przetwarzania schematu walencyjnego zaprezentujemy na poniższym przykładzie:

subj,Initiator,{np(str),ncp(str,int)}+
Recipent,{refl}+Theme,{prepnp(o,loc);comprepnp(na temat)}

Mamy tu schemat walencyjny czasownika zawierającego podmiot realizowany jako fraza rzeczownikowa w przypadku strukturalnym lub fraza zdaniowa pytajna z korelatem (również w przypadku strukturalnym), partykułę *się* oraz argument realizowany jako fraza przyimkowa z przyimkiem *o* lub *na temat*. Podmiot ma tutaj rolę semantyczną Initiator, partykułę *się* — Recipent, fraza przyimkowa — Theme.

Pierwszym etapem przetwarzania jest wstawienie do schematów realizacji fraz, podtypów atrybutów i atrybutów równoważnych.

subj{np(str),ncp(str,int[co,czemu,czy,czyj,...])}+
{refl}+{prepnp(o,loc);comprepnp(na temat)}

Po nim następuje uzupełnienie pozycji o informację o opcjonalności.

subj,Agnt{pro,np(str),ncp(str,int[co,czemu,czy,czyj,...])}+
Ptnt{refl}+Arg{null,prepnp(o,loc);comprepnp(na temat)}

Usunięcie realizacji zawierających leksemy nie występujące w zdaniu.

subj,Agnt{pro,np(str),ncp(str,int[czemu])}+
Ptnt{refl}+Arg{null}

Przetwarzanie leksykalizacji (opisane w dalszym ciągu raportu). Ukonkretnienie schematu na podstawie klasy fleksemu i jego użycia (wskazanie konkretnych realizacji dla atrybutów str,part,żeby2,... oraz zmiana realizacji subj).

subj,Agnt{pro,np(nomagr),ncp(nomagr,int[czemu])}+
Ptnt{refl}+Arg{null}

Uzupełnienie schematu o modyfikatory.

subj,Agnt{pro,np(nomagr),ncp(nomagr,int[czemu])}+
Ptnt{refl}+Arg{null}+{null,advp}+{null,prepp}

Dodanie informacji o cechach semantycznych.

Przetwarzanie leksykalizacji zilustrujemy następującym przykładem:
balansować: {lex(prepnp(na,loc),sg,XOR('krawędź','skraj'),atr1({np(gen)}))}
Konwersja nazw fraz na nazwy fleksemów.
balansować: {lex([prep(loc),'na';subst(sg,loc),XOR('krawędź','skraj')],
atr1({np(gen)}))}
Zamiana list fleksemów na argumenty.
balansować: {lex(prep(loc),'na',ratr({lex(subst(sg,loc),XOR('krawędź','skraj'),
atr1({np(gen)}))}))}

Utworzenie schematów z leksykalizacji.
balansować: {lex(1,prep(loc),'na')}
lex(1,na): ratr({lex(2,subst(sg,loc),XOR('krawędź','skraj'))})
lex(2,krawędź): atr1({np(gen)})
lex(2,skraj): atr1({np(gen)}) Rozwinięcie modyfikacji.
lex(1,na): {lex(2,subst(sg,loc),XOR('krawędź','skraj'))}
lex(2,krawędź): {null;np(gen)}
lex(2,skraj): {null;np(gen)} Rozwinięcie list lematów.
lex(1,na): {lex(2,subst(sg,loc),'krawędź')}
lex(1,na): {lex(2,subst(sg,loc),'skraj')}

Dla leksemów nie występujących w Walentym generuję schematy typowe dla ich części mowy.

3 Parsowanie

Zaanowowany podczas preprocesingu tekst przetwarzany jest następnie przez parser działający w oparciu o gramatykę kategorialną. Mają tu miejsce następujące działania: generowane wpisu w leksykonie, określanie struktury zależnościowej, nadawanie walencji semantycznej, dezambiguacja, tworzenie reprezentacji semantycznej.

3.1 Gramatyka kategorialna (Type Logical Categorial Grammar)

Parsowanie gramatykach kategorialnych jest dowodzeniem twierdzeń w niekomutatywnej intuicjonistycznej logice liniowej. Można dzięki temu implementować fragmenty systemu dowodowego, uzyskując szybkie parsery i mając gwarancję poprawności.

Reguły gramatyczne to uniwersalne, niezależne od przetwarzanego języka reguły dowodzenia w logice. Powoduje to, że gramatyka jest w pełni zleksykalizowana, co ułatwia to integrację z zasobami słownikowymi np. Walentym

Spójniki LCG wyrażają podstawowe zjawiska występujące w języku:

- • konkatenacja (tworzenie wektorów cech)
- & niejednoznaczność
- /, \, | wymaganie argumentu
- $\oplus$ polimorficzne argumenty, sumowanie typów
- 1 pusty argument

- ? wielokrotny argument
- $\&$ parametryzowana niejednoznaczność
- $\star$ tworzenie listy

Umożliwiają bezpośrednie wyrażenie informacji dostarczanej przez lematyzację i zawartej w Walentym. Na przykład cechy fleksyjne leksemu *zielony* określone podczas lematyzacji jako

adj:sg:nom.voc:m1.m2.m3:pos|adj:sg:acc:m3:pos

może my wyrazić w gramatyce kategorialnej następująco:

$$(\text{adj} \bullet \text{sg} \bullet (\text{nom} \,\&\, \text{voc}) \bullet (\text{m1} \,\&\, \text{m2} \,\&\, \text{m3}) \bullet \text{pos}) \,\&\, (\text{adj} \bullet \text{sg} \bullet \text{acc} \bullet \text{m3} \bullet \text{pos})$$

Podstawową reguła w gramatyce jest aplikacja argumentu do funktora

$$\frac{\Gamma \vdash \psi \,/\, \varphi \qquad \Delta \vdash \varphi}{\Gamma, \Delta \vdash \psi}\,[/\,E] \qquad\qquad \frac{\Delta \vdash \varphi \qquad \Gamma \vdash \psi \setminus \varphi}{\Delta, \Gamma \vdash \psi}\,[\setminus E]$$

Oto przykładowy wywód gramatyczny. Będziemy w nim korzystać z leksykonu

$$\text{Jan}_1 \vdash \text{np},\ \text{widzi}_2 \vdash (\text{ip} \setminus \text{np}) \,/\, \text{np},\ \text{stół}_3 \vdash \text{np},\ .\!_4 \vdash \text{s} \setminus \text{ip}$$

i otrzymamy następujące drzewo wywodu:

$$\dfrac{\dfrac{\text{Jan}_1 \vdash \text{np} \qquad \dfrac{\text{widzi}_2 \vdash (\text{ip} \setminus \text{np}) \,/\, \text{np} \qquad \text{stół}_3 \vdash \text{np}}{\text{widzi}_2, \text{stół}_3 \vdash \text{ip} \setminus \text{np}}}{\text{Jan}_1, \text{widzi}_2, \text{stół}_3 \vdash \text{ip}} \qquad .\!_4 \vdash \text{s} \setminus \text{ip}}{\text{Jan}_1, \text{widzi}_2, \text{stół}_3, .\!_4 \vdash \text{s}}$$

Izomorfizm Curriego-Howarda wiąże reguły wnioskowania z termami liniowego rachunku lambda zadając sposób konstruowania formuł języka reprezentacji znaczenia podczas parsowania.

My jednak nie wykorzystujemy gramatyki kategorialnej bezpośrednio to tworzenia reprezentacji semantycznej a jedynie do budowy drzewa zależnościowego. Decyzję tę uzasadnimy w dalszym toku raportu.

Mamy tutaj do czynienia z dwoma językami: językiem reprezentacji w którym ma być wyrażona treść uzyskana w wyniku parsowania może być to np.: logika pierwszego rzędu lub — tak jak tutaj — drzewa zależnościowe oraz językiem konstruowania formuł, którym jest liniowy rachunek lambda.

Izomorfizm Curriego-Howarda wskazuje w jaki sposób należy zaopatrzyć reguły zaopatrzone w λ-termy opisujące wpływ aplikacji reguły na konstruowaną formułę:

$$\frac{\Gamma \vdash \psi \,/\, \varphi : M \qquad \Delta \vdash \varphi : N}{\Gamma, \Delta \vdash \psi : MN}\,[/\,E] \qquad \frac{\Delta \vdash \varphi : N \qquad \Gamma \vdash \psi \setminus \varphi : M}{\Delta, \Gamma \vdash \psi : MN}\,[\setminus E]$$

Leksykon uzupełniamy o formuły języka reprezentacji:

$$\mathrm{Jan}_1 \vdash \mathrm{np} : j, \quad \mathrm{widzi}_2 \vdash (\mathrm{ip} \setminus \mathrm{np}) \,/\, \mathrm{np} : \lambda x \lambda y.w(y,x),$$

$$\mathrm{stół}_3 \vdash \mathrm{np} : s, \quad {}_{.4} \vdash \mathrm{s} \setminus \mathrm{ip} : \lambda x.x$$

I w wyniku parsowania uzyskujemy drzewo wywodu:

$$\dfrac{\mathrm{Jan}_1 \vdash \mathrm{np} : j \qquad \dfrac{\mathrm{widzi}_2 \vdash (\mathrm{ip} \setminus \mathrm{np}) \,/\, \mathrm{np} : \lambda x \lambda y.w(y,x) \qquad \mathrm{stół}_3 \vdash \mathrm{np} : s}{\mathrm{widzi}_2, \mathrm{stół}_3 \vdash \mathrm{ip} \setminus \mathrm{np} : \lambda y.w(y,s)}}{\mathrm{Jan}_1, \mathrm{widzi}_2, \mathrm{stół}_3 \vdash \mathrm{ip} : w(j,s)}$$

Formuła $w(j,s)$ opisuje strukturę zależnościową parsowanego zdania. Opis pozostałych reguł gramatyki można znaleźć w publikacjach poświęconych logice liniowej.

3.2 Leksykon

Wszystkie informacje o języku polskim potrzebne w procesie parsowania zawarte są w leksykonie. Leksykon generowany jest dynamicznie dla poszczególnych zdań na podstawie informacji morfosyntaktycznej formy oraz schematu walencyjnego leksemu. Typy fraz występujące w gramatyce są zbudowane na podstawie typów fraz z Walentego oraz kategorii gramatycznych z SGJP

np • *number* • *case* • *gender* • *person*
nump • *number* • *case* • *gender* • *person*
adjp • *number* • *case* • *gender*

np są frazami nominalnymi z wyłączeniem liczebnikowych. Przykładowo dla słowa *Jan* wygenerowany zostanie wpis

np • sg • nom • m1 • ter

A token *zielony* oznaczony jako

zielony adj:sg:nom.voc:m1.m2.m3:pos|adj:sg:acc:m3:pos

zostanie przetłumaczony na następujące wpisy:

adjp • sg • (nom & voc) • (m1 & m2 & m3)

adj • sg • acc • m3

prepnp • *prep* • *case*
prepadjp • *prep* • *case*
comprepnp • *prep*

Frazy przyimkowe zawierają leksem przyimka. prepnp i comprepnp obejmują frazy przyimkowo-liczebnikowe.

cp • *ctype* • *comp*
ncp • *number* • *case* • *gender* • *person* • *ctype* • *comp*
prepncp • *prep* • *ctype* • *comp*
infp • *aspect*
advp
fixed • *lex*

Typy fraz nie występujące w Walentym

ip • *number* • *gender* • *person*
padvp
prepp • *case*
qub
inclusion
adja
aglt • *number* • *person*
aux-past • *number* • *gender* • *person*
aux-fut • *number* • *gender* • *person*
aux-imp
lex

pro oraz null są zamieniane w opcjonalny argument 1. *lex* to typ frazy zawierający tylko jeden leksem, nazywający się tak jak ten leksem, np. nie, się.

Argumenty wyglądają analogicznie do podstawowych wpisów. $\top$ oznacza, że wartość danego pola jest dowolna. Opcjonalność argumentów wyraża $\oplus$. $/, \backslash$ i $|$ wyrażają położenie argumentu względem nadrzędnika.

Oto przykładowe wpisy w leksykonie:

- Wpis dla przyimka w:loc

  prepnp • w • loc / np • $\top$ • loc • $\top$ • $\top$

- Wpis dla czasownika w formie osobowej mającego argument {null,prepnp(o,loc);comprepnp(na temat)}:

  ip • *number* • *gender* • *person*

  $|$ 1 $\oplus$ prepnp • w • loc $\oplus$ comprepnp • na temat

- Uzgodnienie rzeczownika z przymiotnikiem. Rzeczownik subst:sg:nom.acc:m3,
  którego podrzędnikiem jest przymiotnik (wersja uproszczona)

$$\underset{case\in\{\text{nom,acc}\}}{\&} \text{np} \bullet \text{sg} \bullet case \bullet \text{m3} \bullet \text{ter} \mid \text{adjp} \bullet \text{sg} \bullet case \bullet \text{m3}$$

  Formuła

$$\underset{x\in\{a_1,a_2,\ldots,a_n\}}{\&} \varphi(x)$$

  jest równoważna

$$\varphi(a_1)\ \&\ \varphi(a_2)\ \&\ \cdots\ \&\ \varphi(a_n)$$

  Pełny wpis

$$\underset{case\in\{\text{nom,acc}\}}{\&} \text{np} \bullet \text{sg} \bullet case \bullet \text{m3} \bullet \text{ter}\{$$

$$/\ 1 \oplus \text{adjp} \bullet \text{sg} \bullet case \bullet \text{m3},$$

$$\backslash ?\text{adjp} \bullet \text{sg} \bullet case \bullet \text{m3}\}$$

- Uzgodnienie czasownika. Czasownik w czasie przeszłym z podmiotem np i aglutynatem

$$\underset{gender\in\{\text{m1,m2,m3}\}}{\&} \underset{person\in\{\text{pri,sec}\}}{\&} \text{ip} \bullet \text{sg} \bullet gender \bullet person\{$$

$$\mid 1 \oplus \text{np} \bullet \text{sg} \bullet \text{nom} \bullet gender \bullet person,$$

$$\mid \text{aglt} \bullet \text{sg} \bullet gender \bullet person\}$$

  Czasowniki posiłkowe i aglutynaty są podrzędnikami leksemów do których przynależy schemat. Czas przeszły czasownika typu winien

$$\underset{gender\in\{\text{m1,m2,m3}\}}{\&} \underset{person\in\{\text{pri,sec}\}}{\&} \text{ip} \bullet \text{sg} \bullet gender \bullet person\{$$

$$\mid \text{aux-past} \bullet \text{sg} \bullet gender \bullet person,$$

$$\mid \text{aglt} \bullet \text{sg} \bullet gender \bullet person\}$$

- Uzgodnienia liczebnika. Uzgodnienie dla formy „dwóch” z interpretacją num:pl:gen.loc:m1.m2.m3.f.n2:congr

$$\underset{case\in\{\text{gen,loc}\}}{\&} \underset{gender\in\{\text{m1,m2,m3,f,n2}\}}{\&} \text{nump} \bullet \text{pl} \bullet case \bullet gender \bullet \text{ter}$$

$$/\,\mathrm{np} \bullet \mathrm{pl} \bullet \mathit{case} \bullet \mathit{gender} \bullet \mathrm{ter}$$

oraz z interpretacją num:pl:nom.acc.voc:n1.p1.p2:rec

$$\&_{\mathit{case}\in\{\mathrm{nom,acc,voc}\}} \&_{\mathit{gender}\in\{\mathrm{n1,p1,p2}\}} \mathrm{nump} \bullet \mathrm{sg} \bullet \mathit{case} \bullet \mathrm{n2} \bullet \mathrm{ter}$$

$$/\,\mathrm{np} \bullet \mathrm{pl} \bullet \mathit{case} \bullet \mathit{gender} \bullet \mathrm{ter}$$

Fraza liczebnikowa jako podmiot czasownika w formie osobowej

$$\mathrm{ip} \bullet \mathit{number} \bullet \mathit{gender} \bullet \mathit{person}$$

$$\mid \mathrm{nump} \bullet \mathit{number} \bullet \mathrm{nom} \bullet \mathit{gender} \bullet \mathit{person}$$

- Poprzyimkowe formy zaimków osobowych. Kluczowa jest forma „nie": jeśli nie uwzględnimy poprzyimkowości, uzyskamy dużą liczbę błędnych rozbiorów dla zdań z negacją. Wymaganie poprzyimkowości realizujemy za pomocą podniesienia typu. Na przykład formie „nią" ppron3:sg:acc.inst:f:ter:_:praep zamiast typu

  $$\mathrm{np} \bullet \mathrm{sg} \bullet (\mathrm{acc} \,\&\, \mathrm{inst}) \bullet \mathrm{f} \bullet \mathrm{ter}$$

  nadajemy typ

  $$\&_{\mathit{prep}} \&_{\mathit{case}\in\{\mathrm{acc,inst}\}} \mathrm{prepnp} \bullet \mathit{prep} \bullet \mathit{case}$$

  $$\backslash(\mathrm{prepnp} \bullet \mathit{prep} \bullet \mathit{case} \,/\, \mathrm{np} \bullet \mathrm{sg} \bullet \mathit{case} \bullet \mathrm{f} \bullet \mathrm{ter})$$

- Zaimki względne i pytajne. Spełniają dwie funkcje: realizują argumenty czasownika i zastępują ip przez cp. Przykłady

  – „Kto pyta?" kto subst:sg:nom:m1

  $$\mathrm{cp} \bullet \mathrm{int} \bullet \mathrm{kto} \,/(\mathrm{ip} \bullet \top \bullet \top \bullet \top \mid \mathrm{np} \bullet \mathrm{sg} \bullet \mathrm{nom} \bullet \mathrm{m1} \bullet \mathrm{ter})$$

  – „W związku z czym pyta?" czym subst:sg:inst.loc:n2

  $$\&_{\mathit{prep}} \&_{\mathit{case}\in\{\mathrm{inst,loc}\}} [\mathrm{cp} \bullet \mathrm{int} \bullet \mathrm{co} \,/(\mathrm{ip} \bullet \top \bullet \top \bullet \top \mid \mathrm{comprepnp} \bullet \mathit{prep})]$$

  $$\backslash(\mathrm{comprepnp} \bullet \mathit{prep} \,/\, \mathrm{np} \bullet \mathrm{sg} \bullet \mathit{case} \bullet \mathrm{n2} \bullet \mathrm{ter})$$

– „O którą książkę pyta?” którą adj:sg:acc.inst:f:pos

$$\&_{prep} \&_{case\in\{\text{acc,inst}\}} [[\text{cp}\bullet\text{int}\bullet\text{który} \,/ (\text{ip}\bullet\top\bullet\top\bullet\top \mid \text{prepnp}\bullet prep\bullet case)]$$

$$\backslash(\text{prepnp} \bullet prep \bullet case \,/\, \text{np} \bullet \text{sg} \bullet case \bullet \text{f} \bullet \text{ter})]$$

$$/(\text{np} \bullet \text{sg} \bullet case \bullet \text{f} \bullet \text{ter} \backslash \text{adjp} \bullet \text{sg} \bullet case \bullet \text{f})$$

Wykorzystanie podniesienia typu jest możliwe dzięki temu, że zaimki te są na początku zdania. Nie można byłoby z niego skorzystać, gdyby stały pomiędzy argumentami czasownika Jest to szczęśliwy zbieg okoliczności, albo przesłanka za tym, że opis w LCG odzwierciedla składnię języka polskiego.

- Generowanie wpisu w leksykonie dla leksykalizacji

$$\text{lex} \bullet 1 \bullet \text{na} \bullet \text{prep} \bullet \text{loc} \,/\, \text{lex} \bullet 2 \bullet \text{krawędź} \bullet \text{subst} \bullet \top \bullet \text{loc} \bullet \top \bullet \top$$

$$\text{lex} \bullet 2 \bullet \text{krawędź} \bullet \text{subst} \bullet number \bullet case \bullet \text{f} \bullet \text{ter}$$

$$\mid \text{np} \bullet \top \bullet \text{gen} \bullet \top \bullet \top$$

Gramatyka uzupełniona jest też o konstrukcje mowy niezależnej oraz zleksykalizowany opis określeń czasu.

Należy tutaj zaznaczyć, że ENIAM nie korzysta z gramatyki języka polskiego w tradycyjnym rozumieniu tego słowa. Więzy gramatyczne pomiędzy słowami zadane są przez słownik walencyjny uzupełniony o informację o możliwych modyfikatorach dla danego typu leksemu. Z racji tego, że gramatyka kategorialna jest w pełni zleksykalizowana nie było potrzeby tworzyć ogólnych reguł mówiących np. o tym, że rzeczownik uzgadnia się z przymiotnikiem pod względem przypadku, liczy i rodzaju. Informacja ta zawarta jest częściowo w walencji mówiącej, że rzeczownik może być modyfikowany przez uzgadniający się z nim przymiotnik, a częściowo w procedurach tłumaczących ramy walencyjne na leksykon gramatyki kategorialnej.

3.3 Parser

Parser bazuje na algorytmie CYK. Parsowanie wykonywane jest za pomocą reguł ograniczonego systemu dowodowego logiki liniowej, który parser w bezpośredni sposób implementuje. Parser wypełnia tablicę biorąc po dwa tokeny. Z pierwszego z nich próbuje wywnioskować $\psi \,/\, \varphi : M$ a z drugiego $\varphi : N$. Jeśli mu się uda dodaje $\psi : MN$ do tablicy, wykonując przy tym

redukcję MN. Analogicznie z drugiego próbuje wywnioskować $\psi \setminus \varphi : M$ a z pierwszego $\varphi : N$. Wnioskowania są przeprowadzane za pomocą ograniczonego systemu dowodowego logiki liniowej, który parser w bezpośredni sposób implementuje.

Parser ma siłę wyrazu gramatyki bezkontekstowej. Z racji tego, że formalizm kategorialny pozwala w zwarty sposób reprezentować niejednoznaczności wynikające z polskiej fleksji, rozmiar generowanego leksykonu jest wykładniczo mniejszy od rozmiaru odpowiadającej mu gramatyki bezkontekstowej.

Parser generuje na wyjściu drzewo zależnościowe dla zadanego zdania. Generowanie struktury zależnościowej pomiędzy tokenami odbywa się w sposób leniwy.

Niejednoznaczność powstająca w trakcie parsowania jest wyrażona w formie skompresowanego lasu. Kompresja niejednoznaczności ma miejsce w trakcie parsowania. Stosując aplikację w przód po wywnioskowaniu $\psi / \varphi : M$, parser bierze wszystkie tokeny znajdujące się na drugim polu. Z każdego z nich próbuje wywnioskować φ otrzymując listę możliwych semantyk $N_1, \ldots, N_k$. Jeśli lista ma jeden element dodaje do tablicy $\psi / \varphi : MN_1$. Jeśli lista ma więcej niż jeden element: tworzy nową etykietę e i dodaje etykietowany wariant $\psi / \varphi : M\langle e_1 : N_1, \ldots, e_2 : N_2\rangle$ do tablicy. Aplikacja w tył działa analogicznie.

3.4 Walencja semantyczna

Ramy semantyczne zawarte w Walentym dostarczają informacje o rolach tematycznych poszczególnych argumentów oraz ich preferencjach selekcyjnych. Preferencje selekcyjne są sensami ze Słowosieci (lub ich uogólnieniami). Sensy te powinny być bardziej ogólne od sensu podrzędnika. Spełnialność preferencji selekcyjnych przez sens danego słowa można określić, sprawdzając czy zbiór wszystkich jego hiperonimów ma niepuste przecięcie ze zbiorem preferencji selekcyjnych danego argumentu. Preferencje selekcyjne w Walentym umożliwiają m.in. rozstrzygnięcie, że w zdaniu *Kot aranżuje na fortepian*, *Kot* jest nazwą własną a nie rzeczownikiem pospolitym. Pomagają też rozróżniać argumenty od modyfikatorów i dzięki temu wskazywać właściwe role tematyczne:

- *Załadował bagażnik jabłkami*$_{Theme}$.
- *Załadował bagażnik koparką*$_{Instrument}$.
- *Załadował bagażnik wieczorem*$_{Time}$.

Sensy słów wprowadzają jednak olbrzymią niejednoznaczność, która tylko w niewielkim stopniu redukowana jest przez preferencje selekcyjne. Wynika to m.in. z tego, że poszczególne sensy danego leksemu są do siebie na tyle podobne, że wpadają w te same preferencje selekcyjne. Np. w zdaniu *Człowiek aranżuje* czasownik ma pięć ram/schematów (skojarzonych z 3 sensami), w których podmiot ma preferencje LUDZIE, bądź PODMIOTY; a rzeczownik ma 5 znaczeń, z czego znaczenia 2, 4 i 5 mają jako hiperonim znaczenie 1.

Z uwagi na niejednoznaczność walencja semantyczna jest wprowadzana dopiero po określeniu struktury zależnościowej. Poszczególne znaczenia i alternatywne ramy walencyjne są nakładane na strukturę zależnościową w taki sposób, by jedynie lokalnie zwiększać niejednoznaczność: pojedynczy węzeł w strukturze zależnościowej jest powielany proporcjonalną ilość razy do liczby jego interpretacji, a powielenie to nie propaguje się na resztę struktury. Konsekwencją takiego podejścia jest wymaganie by preferencje selekcyjne dotyczyły zawsze bezpośrednich podrzędników danego węzła. Wymusza to odejście od klasycznych zasad rozbioru składniowego: niesemantyczne przyimki, liczebniki, rzeczowniki użyte w znaczeniu pojemnikowym, czasowniki posiłkowe stają się teraz podrzędnikami swoich zwyczajowych argumentów.

Dezambiguacja Dezambiguacja odbywa się etapami. Najpierw ma miejsce badanie spełnialności preferencji selekcyjnych tam, gdzie mogą one wpłynąć na strukturę zależnościową, czy w przypadku argumentów, które nie mogą być modyfikatorami. Następnie wybierane są najbardziej prawdopodobnych lematów na podstawie listy frekwencyjnej z NKJP1M. Potem następuje badanie spełnialności preferencji selekcyjnych w pozostałych przypadkach. A na koniec wybór sensów słów.

Pozostałe typy niejednoznaczności, takie jak np. niejednoznaczność dowiązania frazy przyimkowej, pozostają aktualnie niezdezambiguowane. Na potrzeby prezentacji losowane jest 10 struktur zależnościowych.

Semantyka wyrażana jest za pomocą grafów semantycznych równoważnych Minimal Recursion Semantics. Rozwijana do formuł logiki pierwszego rzędu rozszerzonych o predykat metajęzykowy i kwantyfikatory specyficzne dla języka naturalnego. Język reprezentacji znaczenia (teoria opisu świata) wykorzystywany przez parser został szczegółowo opisany w ramach projektu Clarin-pl. Ontologia (zestaw pojęć) zadana jest przez Słowosieć. Relacje między pojęciami są rozszerzają zbiór ról tematycznych zdefiniowanych w Walentym.