Uniwersytet Warszawski

Wydział Filozofii

Wojciech Stempniak

Nr albumu: 433855

Struktura zależnościowa koordynacji – analiza korpusów Universal Dependencies

Praca licencjacka

na kierunku KOGNITYWISTYKA

Praca wykonana pod kierunkiem

prof. dr. hab. Adama Przepiórkowskiego

Uniwersytet Warszawski

Warszawa, czerwiec 2024

Streszczenie

Istnieje wiele poglądów na temat struktury składniowej koordynacji, czyli konstrukcji współrzędnie złożonych. Poprzednie badania (Przepiórkowski i Woźniak, 2023) pokazują metodę pozwalającą na testowanie ich poprawności. Wykorzystują one zasadę Dependency Length Minimization (DLM, Temperley 2007), czyli tendencję do formułowania zdań tak, aby łączna długość relacji między słowami w zdaniu była jak najmniejsza.

Słowa kluczowe

koordynacja, struktura koordynacji, zależności składniowe, drzewa zależnościowe, Universal Dependencies (UD), Dependency Length Minimization (DLM), języki inicjalne, języki finalne, badanie korpusowe

Tytuł pracy w języku angielskim

Dependency structure of coordination – an analysis of Universal Dependencies corpora

Podziękowania

Pragnę złożyć serdeczne podziękowania mojemu promotorowi prof. dr. hab. Adamowi Przepiórkowskiemu za inspirację i pomoc w prowadzeniu badań oraz cierpliwość, wyrozumiałość i zaangażowanie podczas kierowania moją pracą.

Ukończenie niniejszej pracy nie byłoby możliwe bez pomocy mgr. Berkego Şenşekerci, któremu serdecznie dziękuję za pomoc w dostosowaniu stosowanych przeze mnie algorytmów do warunków języka tureckiego.

Dziękuję również Magdalenie Borysiak, Katarzynie Zrobek i Oskarowi Pruszyńskiemu za koleżeńską pomoc w prowadzeniu badań i zgłębianiu wiedzy na temat badanych przeze mnie zjawisk.

Spis treści

Rozdział 1

Przetwarzanie danych

1.1. Dane wejściowe

1.1.1. Korpusy zależnościowe

Analizie zostały poddane 72 korpusy zależnościowe 13 języków, spośród korpusów dostępnych na stronie internetowej Universal Dependencies (`https://universaldependencies.org/`).

Za główne kryterium doboru języka uznano istnienie korpusu zależnościowego opisanego w standardzie UD o objętości co najmniej 700 tysięcy tokenów.

Język arabski został wykluczony z analizy, ponieważ znaczna część jego korpusów zawierała wyłącznie relacje zależnościowe między tokenami. W takich korpusach treść zdań została zamieniona podkreślnikami, w związku z czym policzenie długości członów w słowach, sylabach i znakach nie była możliwa. Korpusy języka arabskiego zawierające słowa nie przekroczyły łącznej objętości 700 tys. tokenów.

Ze względu na fakt, że w korpusach UD dla języka japońskiego nie występują koordynacje (Kanayama i in., 2018, s. 79), język ten również został wykluczony z analizy.

Ponadto do analizy przyjęto korpusy dwóch języków finalnych z największymi korpusami, nie licząc wykluczonych japońskiego i arabskiego, tj. koreańskiego i tureckiego.

Dodatkowo uwzględniono korpusy języka polskiego jako języka ojczystego autora pracy.

Tabela 1.1 przedstawia informacje na temat korpusów użytych w badaniu.

Język	Korpus	Rozmiar	Sposób anotacji relacji zależnościowych
		Języki inicjalne	
angielski	GUM	184 478	ręcznie w formacie UD
	EWT	251 534	ręcznie w formacie UD
	Atis	61 879	ręcznie w formacie UD
	ParTUT	49 602	ręcznie w innym formacie, konwersja z poprawkami
	GENTLE	17 617	ręcznie w formacie UD
	PUD	21 058	ręcznie w formacie UD
	LinES	93 200	ręcznie w innym formacie, konwersja z poprawkami
	Pronouns	1 640	ręcznie w formacie UD
	ESLSpok	21 312	ręcznie w formacie UD
	GUMReddit	15 960	ręcznie w formacie UD
	Razem	**718 280**	
czeski	CAC	494 420	ręcznie w innym formacie, automatyczna konwersja
	PDT	1 527 257	ręcznie w innym formacie, automatyczna konwersja
	FicTree	166 747	ręcznie w innym formacie, automatyczna konwersja
	CLTT	36 011	ręcznie w innym formacie, automatyczna konwersja
	PUD	18 578	ręcznie w formacie UD
	Poetry	6 273	ręcznie w formacie UD
	Razem	**2 249 286**	
hiszpański	AnCora	555 670	ręcznie w innym formacie, automatyczna konwersja
	GSD	423 345	ręcznie w innym formacie, automatyczna konwersja
	PUD	22 822	ręcznie w innym formacie, automatyczna konwersja
	Razem	**1 001 837**	
islandzki	Modern	80 392	ręcznie w innym formacie, automatyczna konwersja
	IcePaHC	983 671	ręcznie w innym formacie, automatyczna konwersja
	PUD	18 831	automatycznie z poprawkami
	GC	99 611	ręcznie w innym formacie, automatyczna konwersja
	Razem	**1 182 505**	
polski	PDB	347 319	ręcznie w innym formacie, automatyczna konwersja
	LFG	130 967	ręcznie w innym formacie, automatyczna konwersja
	PUD	18 333	ręcznie w innym formacie, automatyczna konwersja
	Razem	**496 619**	
portugalski	PetroGold	232 333	ręcznie w formacie UD
	Porttinari	157 490	ręcznie w formacie UD
	Bosque	210 958	ręcznie w innym formacie, konwersja z poprawkami
	CINTIL	441 991	ręcznie w innym formacie, automatyczna konwersja
	GSD	296 169	ręcznie w innym formacie, automatyczna konwersja
	PUD	21 917	ręcznie w innym formacie, automatyczna konwersja
	Razem	**1 360 858**	
rosyjski	Taiga	197 001	ręcznie w formacie UD
	Poetry	64 112	ręcznie w formacie UD
	SynTagRus	1 517 881	ręcznie w innym formacie, automatyczna konwersja
	GSD	97 994	ręcznie w formacie UD
	PUD	19 355	ręcznie w formacie UD
	Razem	**1 896 343**	

Język	Korpus	Rozmiar	Sposób anotacji relacji zależnościowych
		Języki inicjalne	
rumuński	RRT	218 522	ręcznie w formacie UD
	SiMoNERo	146 020	ręcznie w innym formacie, automatyczna konwersja
	ArT	573	ręcznie w formacie UD
	Nonstandard	572 436	ręcznie w innym formacie, automatyczna konwersja
	Razem	**937 551**	
włoski	ISDT	278 460	ręcznie w innym formacie, automatyczna konwersja
	VIT	259 625	ręcznie w innym formacie, automatyczna konwersja
	Old	40 386	ręcznie w formacie UD
	ParTUT	51 614	ręcznie w innym formacie, konwersja z poprawkami
	ParlaMint	19 141	ręcznie w formacie UD
	TWITTIRO	28 384	ręcznie w innym formacie, automatyczna konwersja
	Valico	6 508	ręcznie w formacie UD
	PoSTWITA	119 334	automatycznie z poprawkami
	MarkIT	38 237	ręcznie w formacie UD
	PUD	22 182	ręcznie w innym formacie, automatyczna konwersja
	Razem	**863 871**	
		Języki mieszane	
łacina	ITTB	450 480	ręcznie w innym formacie, konwersja z poprawkami
	LLCT	242 391	ręcznie w innym formacie, konwersja z poprawkami
	UDante	55 286	ręcznie w formacie UD
	Perseus	28 868	ręcznie w innym formacie, automatyczna konwersja
	PROIEL	205 566	ręcznie w innym formacie, automatyczna konwersja
	Razem	**982 591**	
niemiecki	GSD	287721	ręcznie w innym formacie, automatyczna konwersja
	PUD	21 001	ręcznie w innym formacie, automatyczna konwersja
	LIT	40 340	ręcznie w formacie UD
	HDT	3 399 390	ręcznie w innym formacie, konwersja z poprawkami
	Razem	**3 748 452**	
		Języki finalne	
koreański	Kaist	350 090	ręcznie w innym formacie, automatyczna konwersja
	GSD	80 322	ręcznie w innym formacie, automatyczna konwersja
	PUD	16 584	ręcznie w innym formacie, automatyczna konwersja
	Razem	**446 996**	
turecki	Kenet	178 658	ręcznie w innym formacie, automatyczna konwersja
	Penn	183 555	ręcznie w innym formacie, automatyczna konwersja
	Tourism	91 152	ręcznie w innym formacie, automatyczna konwersja
	Atis	45 907	ręcznie w formacie UD
	GB	16 803	ręcznie w formacie UD
	FrameNet	19 223	ręcznie w innym formacie, automatyczna konwersja
	BOUN	121 835	ręcznie w formacie UD
	IMST	56 422	ręcznie w innym formacie, automatyczna konwersja
	PUD	16 535	ręcznie w formacie UD
	Razem	**730 090**	

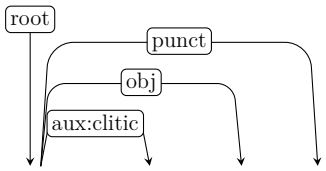
Tabela 1.1: Języki i korpusy analizowane w badaniu. Rozmiar korpusów podany jest w liczbie tokenów.

1.1.2. Format danych

Korpusy składają się ze zdań opisanych w formacie CONLL-U[1], zawierającym wszystkie informacje potrzebne to utworzenia drzewa zależnościowego. Poniższe schematy przedstawiają przykładowe zdanie (1), jego opis w formacie CONLL-U (2) oraz drzewo zależnościowe (3).

(1) Zacierałem ręce.

(2)
```
# sent_id = dev-1646
# text = Zacierałem ręce.
# converted_from_file = NKJP1M_1102000008_morph_6-p_morph_6.61-s-dis@1.xml
# genre = news
1 Zacierał zacierać VERB praet:sg:m1:imperf Aspect=Imp|Gender=Masc|Mood=Ind|Number=Sing|
SubGender=Masc1|Tense=Past|VerbForm=Fin|Voice=Act 0 root 0:root SpaceAfter=No
2 em  być  AUX  glt:sg:pri:imperf:wok Aspect=Imp|Number=Sing|Person=1| Variant=Long 1aux:clitic
1:aux:clitic _
3 ręce ręka NOUN subst:pl:acc:f Case=Acc|Gender=Fem|Number=Plur 1obj 1:obj SpaceAfter=No
4 .    .    PUNCT interp PunctType=Peri 1 punct 1:punct _
```

(3)


(Przepiórkowski i Patejuk, 2018)

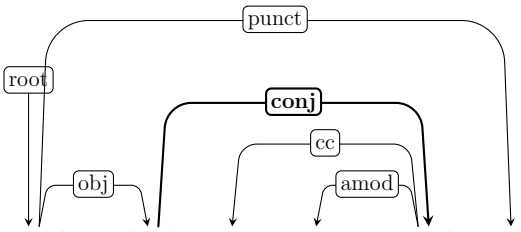
1.2. Wyciąganie koordynacji

W niniejszym punkcie omawiam proces wyciągania koordynacji, czyli zautomatyzowanego znajdowania i opisu konstrukcji współrzędnie złożonych w korpusach zależnościowych. Omawiana procedura zakłada, że zdania są anotowane w formacie UD, który przyjmuje stanfordzki model struktury koordynacji.

1.2.1. Relacja `conj`

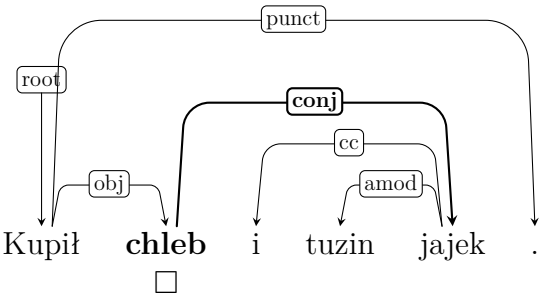
W standardzie Universal Dependencies zależność łącząca dwa człony koordynacji opisana jest zawsze etykietą `conj`. Taka etykieta sygnalizuje obecność konstrukcji współrzędnie złożonej, co pokazuje przykład (4). Ponieważ UD opisuje koordynacje według podejścia stanfordzkiego, to wiadomo, że każda krawędź drzewa podpisana etykietą `conj` łączy lewy człon koordynacji z jednym z pozostałych jej członów.

[1]Dokładny opis formatu znajduje się na stronie `https://universaldependencies.org/format.html`.

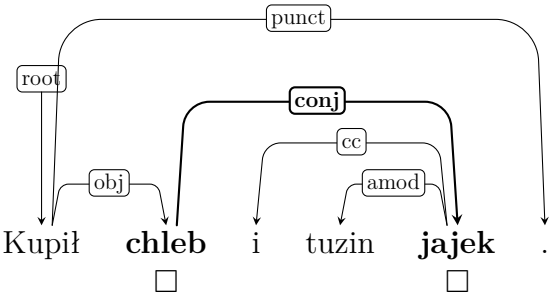


1.2.2. Wyznaczanie głów członów, nadrzędnika i spójnika koordynacji

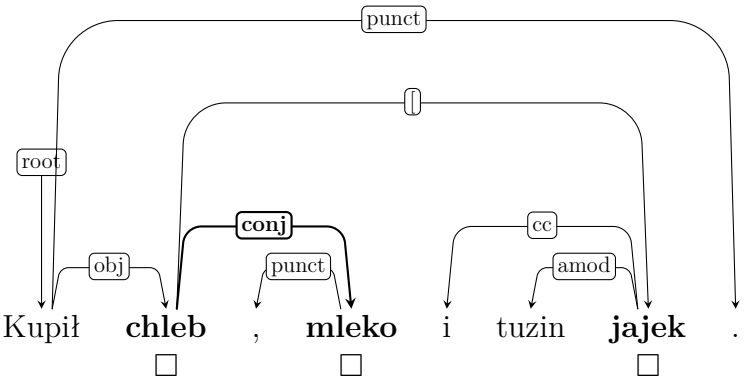
Głowa lewego członu Jeśli token jest nadrzędnikiem w relacji *conj*, to jest on głową lewego członu. W przykładzie (5) głową lewego członu jest token *chleb*.



Głowa prawego członu Jeśli z głowy lewego członu wychodzi tylko jedna relacja `conj`, to jej podrzędnik jest głową prawego członu. W (6) głową prawego członu jest token *jajek*.

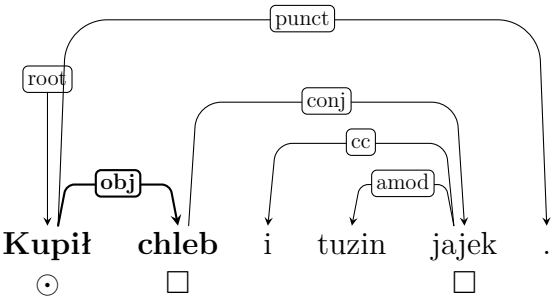


Jeśli natomiast głowa lewego członu ma kilka podrzędników z relacjami `conj`, jest to koordynacja wieloczłonowa. Podrzędniki tych relacji to głowy pozostałych członów. Głowa, która występuje w zdaniu jako ostatnia, jest głową prawego członu. W zdaniu (7) *chleb*, *mleko* i *jajek* są głowami członów jednej konstrukcji współrzędnie złożonej. *Chleb* jest głową lewego, zaś *jajek* prawego członu koordynacji.

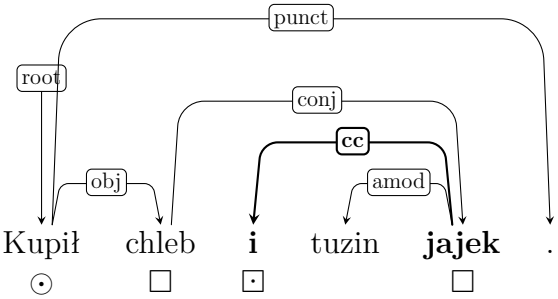


Ponieważ w analizie każda koordynacja traktowana jest jako binarna, środkowe człony (*mleko* w przykładzie (7)) są ignorowane.

Nadrzędnik Nadrzędnikiem koordynacji jest zawsze nadrzędnik głowy lewego członu. W przykładzie (8) jest to *Kupił*.



Spójnik Jeśli z głowy prawego członu wychodzi relacja `cc`, podrzędnik tej relacji jest spójnikiem koordynacji. W zdaniu (9) spójnikiem koordynacji *i*.



Podsumowując, procedura wyznaczania kluczowych elementów koordynacji wygląda następująco:

(10) Dla każdego tokenu L:
Jeśli L ma $n > 0$ podrzędników z relacją `conj` oraz:

- nadrzędnik L to G,
- podrzędniki L z relacją `conj` to $H_1, \ldots, H_n$,
- opcjonalne podrzędniki $H_1, \ldots, H_n$ z relacją `cc` to spójniki – oznaczane odpowiednio $C_1, \ldots, C_n$,

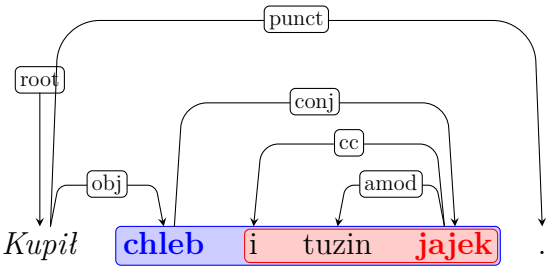
to należy rozpatrzeć taką koordynację, w której nadrzędnikiem jest G, głową lewego członu jest L, głową prawego członu H_n, a spójnikiem C_n.

H_i i C_i dla $i < n$ to odpowiednio pozostałe głowy i spójniki. Są one co do zasady pomijane w analizie[2].

1.2.3. Wyznaczanie granic członów

W celu automatycznego określenia granicy członu należy wziąć pod uwagę wszystkich potomków jego głowy:

(11)

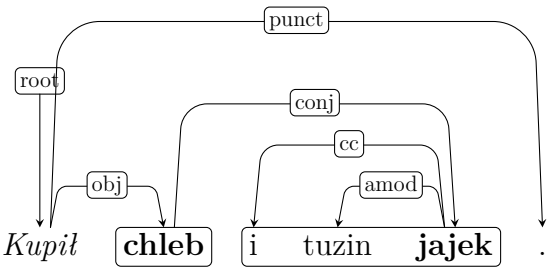


W zdaniu (11) potomkami głowy lewego członu *chleb* są *i*, *tuzin* i *jajek*, zaś potomkami głowy prawego członu *i* oraz *tuzin*.

Spośród podrzędników głowy lewego członu należy wykluczyć głowy pozostałych członów oraz wraz z ich podrzędnikami.

W ten sposób otrzymujemy wstępnie określone granice członów, co pokazuje przykład (12):

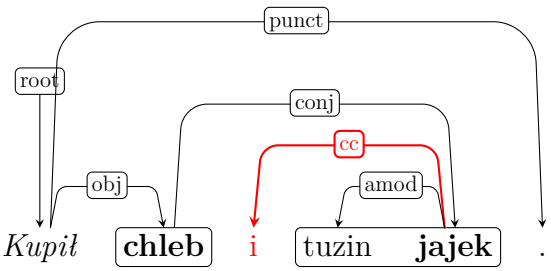
(12)



Następnie należy zastosować zestaw reguł w celu wykluczenia tokenów, które nie są elementami członów. W niniejszej pracy posługuję się następującymi heurystykami:

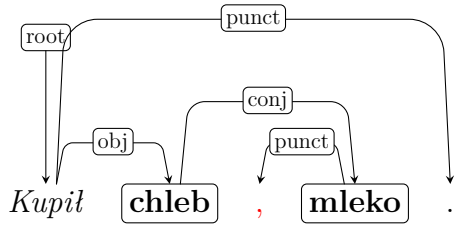
(H1) Człon nie może zaczynać się od spójnika (słowa, które jest połączone z głową członu relacją `cc`).

(13)



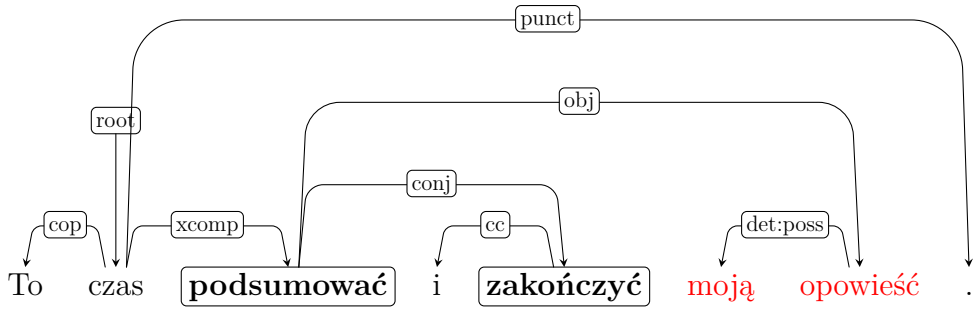
[2]Niektóre z heurystyk opisywanych w punktach 1.2.3 oraz 1.2.7 mogą brać pod uwagę istnienie lub treść środkowych członów i pozostałych spójników. Odnoszę się do tego faktu przy omawianiu tych heurystyk.

(H2) Człon nie może zaczynać się od znaku interpunkcyjnego (dokładniej rzecz ujmując, od przecinka, średnika, dwukropka, ani myślnika).

(14)


(H3) Potomkowie głowy lewego członu znajdujący się na prawo od prawego członu nie wchodzą w skład lewego członu.

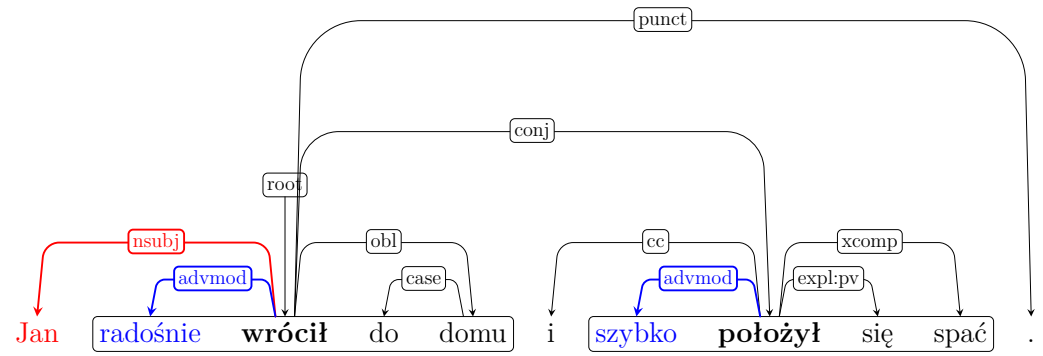
W rzeczywistości tokeny, o których mowa w (H3) są współdzielone przez oba człony, czyli należą do ich obu. Ponieważ interesuje mnie różnica długości członów, dla przejrzystości wykluczam ich wspólną część.

(15)


(H4) Podrzędnik głowy lewego członu po jego lewej stronie nie jest częścią lewego członu, jeśli z tą głową łączy go relacja o *unikalnej* etykiecie. Przez unikalną etykietę rozumiem taką, która nie występuje na żadnej relacji między głowami innych członów tej koordynacji i ich podrzędnikami.

Na potrzeby tej heurystyki etykiety `subj` i `subj:pass` oraz `nummod` i `nummod:gov` uznaję za identyczne.

W przykładzie (16) tokeny *szybko* i *radośnie* są połączone z głowami członów identycznymi etykietami `advmod`. Z tego powodu są potraktowane jako prywatne modyfikatory swoich nadrzędników. Token *Jan* jest podrzędnikiem relacji o unikalnej etykiecie `nsubj`, więc jest uznany za element obu członów.

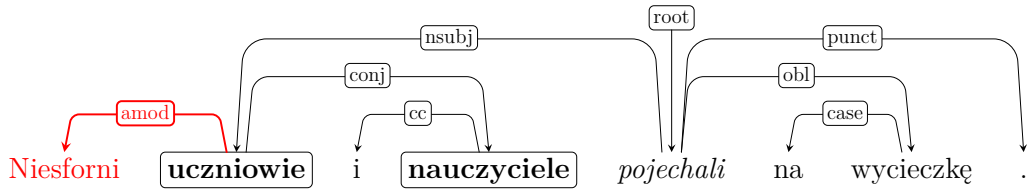
(16)


Chociaż reguły (H1) i (H2) niemalże zawsze działają prawidłowo, heurystyki (H3) i (H4) są bardziej zawodne. Ich celem jest rozróżnienie elementów „prywatnych” dla danego członu, takich jak okoliczniki *radośnie* i *szybko* w (16), od elementów „wspólnych” dla obu członów, takich jak podmiot *Jan* w (16) czy dopełnienie *moją opowieść* w (15).

Niemniej jednak rozróżnienie to nie zawsze działa prawidłowo. Przykładowo, zastosowanie (H4) do powtórzonego poniżej zdania (**??**) powoduje, że *Niesforni* zostaje uznane za element wspólny i prowadzi do interpretacji innej od tej, która wynika z semantyki.

(8) Niesforni uczniowie i nauczyciele pojechali na wycieczkę.

(34)



1.2.4. Określanie pozycji nadrzędnika

Na potrzeby przetwarzania korpusów UD (korzystających z podejścia stanfordzkiego) pozycja nadrzędnika koordynacji określana jest w następujący sposób:

ozn.	pozycja nadrzędnika	definicja (zakładająca opis w stylu UD)	przykład koordynacji
(L)	po lewej stronie	przed początkiem pierwszego członu	Drzewo *sadzą* [[Pat] i [Mat]].
(0)	brak nadrzędnika	głowa lewego członu jest korzeniem zdania	[[Posadzili] i [podlali]] drzewo.
(R)	po prawej stronie	po końcu ostatniego członu	[[Pat] i [Mat]] *posadzili* drzewo.
(M)	po środku	pozostałe przypadki	[[Pat] – *powtórzyłem* – oraz [Mat]].

Ze względu na małą liczbę koordynacji (M) oraz częste błędy w drzewach zdań zawierających koordynacje z nadrzędnikiem po środku w badaniu nie przeprowadzam osobnych analiz dla koordynacji (M).

1.2.5. Obliczanie długości członu

Drugą ważną z perspektywy analizy statystycznej cechą konstrukcji współrzędnie złożonej jest różnica długości jej członów.

Długość członu określana jest na cztery sposoby: jako liczba **słów**, **tokenów**, **sylab** i **znaków**.

Słowa Właściwym podejściem jest traktowanie słów jako podciągów tekstu rozdzielonych spacjami. Przyjęcie takiego rozwiązania skutkuje brakiem możliwości obliczenia długości członu, gdy jedynie część danego słowa należy do tego członu.

(35) Arma virumque canō.[3]

[3]Publiusz Wergiliusz Maro, *Eneida*, `https://www.thelatinlibrary.com/vergil/aen1.shtml`. W analizie morfologicznej użyto narzędzia Collatinus web (`https://outils.biblissima.fr/en/collatinus-web/`).

Arma vir-um que canō .
broń.ACC.PL mąż-ACC.SG i śpiewać.PRS.IND.ACT.1SG

„Śpiewam o broni i mężu.”

W zdaniu (35) znajduje się koordynacja binarna (36) ze spójnikiem *que*:

(36) [[Arma] [virum]que]

Uznanie *virumque* za jedno słowo oznaczałoby, że prawy człon koordynacji (36) składa się z niecałkowitej liczby słów. Nie wiadomo, jak wielu koordynacji może dotyczyć ten problem, jednak nawet jeśli wyżej opisane zjawisko jest rzadkie, powoduje ono dojście do niedopuszczalnych wniosków. W celu uniknięcia opisanego powyżej problemu, przez liczbę słów rozumiem liczbę wszystkich tokenów oprócz tych będących znakami interpunkcyjnymi.

Tokeny Tokenami są wyrazy i znaki interpunkcyjne, a także klityki, części kontrakcji i złożeń (Riedl i Biemann, 2018). W Universal Dependencies reguły tokenizacji różnią się nieznacznie między językami. Interpunkcja oraz części złożeń są odrębnymi tokenami, chyba że stanowią „integralną część lematu” danego słowa (De Marneffe i in., 2021).

W zdaniu (37) *śmy* jest klityką dołączoną do tokenu *Wywiesili*, zaś przymiotnik złożony *biało-czerwoną* składa się z tokenów *biało* i *czerwoną* oraz z łącznika (który również jest osobnym tokenem).

W zdaniu (38) *ins* jest kontrakcją dwóch tokenów: *in* oraz *das*. Słowo *Krankenhaus* (szpital), pomimo że jest złożeniem słów *Kranken* (chorzy) oraz *Haus* (dom), jest potraktowane jako jeden token.[4]

(37) Wywiesiliśmy biało-czerwoną flagę.

Wywiesili śmy biało - czerwoną flagę .

(38) Ich gehe **ins** Krankenhaus.

Ich gehe **in das** Krankenhaus .
ja iść.1SG.PRS do DET.SG.N.ACC szpital

„Idę do szpitala.”

[4]Istnieją argumenty za rozdzielaniem niemieckich czasowników złożonych na tokeny (Riedl i Biemann, 2018), jednak autorzy standardu UD podjęli decyzję, żeby tego nie robić (De Marneffe i in., 2021).

Sylaby Liczba sylab w obrębie danego członu jest sumą liczb sylab w poszczególnych słowach wchodzących w skład tego członu. Na potrzebę liczenia sylab przez „słowo" rozumiem część zdania oddzieloną od reszty spacją. Liczba sylab w tokenach została określona na podstawie opisanej niżej procedury.

- Algorytm sprawdza, czy słowo znajduje się na liście skrótów występujących w danym języku. Jeśli tak, rozpatruje rozwinięcie tego skrótu.
- Następnie algorytm sprawdza, czy słowo lub jego część jest liczbą. Jeśli tak, zamienia ją na formę słowną, wykorzystując do tego bibliotekę `numpy`[5].
- Ostatecznie algorytm dzieli słowo na sylaby, wykorzystując następujące biblioteki języka Python:

  `loguax` dla łaciny,

  `turkishNLP` dla języka tureckiego,

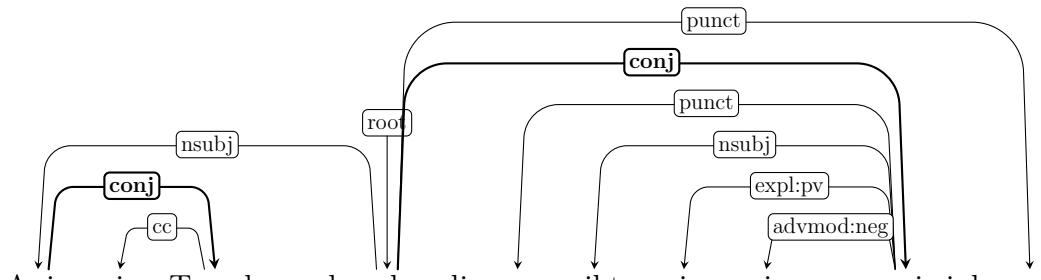
  `pyphen` dla pozostałych języków.

Znaki Długość członu wyrażona w znakach, tj. literach, spacjach i znakach interpunkcyjnych.

W przypadku języka koreańskiego każdy znak alfabetu (*jamo*) traktowany jest jak litera, zaś każdy blok znaków jako sylaba (Simpson i Kang, 2004). Przykładowo, słowo 꿀벌 (pszczoła) składa się z dwóch sylab (odpowiadających blokom 꿀 i 벌) oraz z sześciu znaków (ㄲ, ㅜ, ㄹ, ㅂ, ㅓ i ㄹ)[6].

1.2.6. Koordynacje zagnieżdżone

Czasami zdarzają się sytuacje, w których jedna koordynacja jest częścią innej. Takie koordynacje nazywają się zagnieżdżonymi.



(39) Ania i Tomek zdecydowali , nikt się nie sprzeciwiał .

Należy traktować je jako dwie osobne konstrukcje współrzędnie złożone:

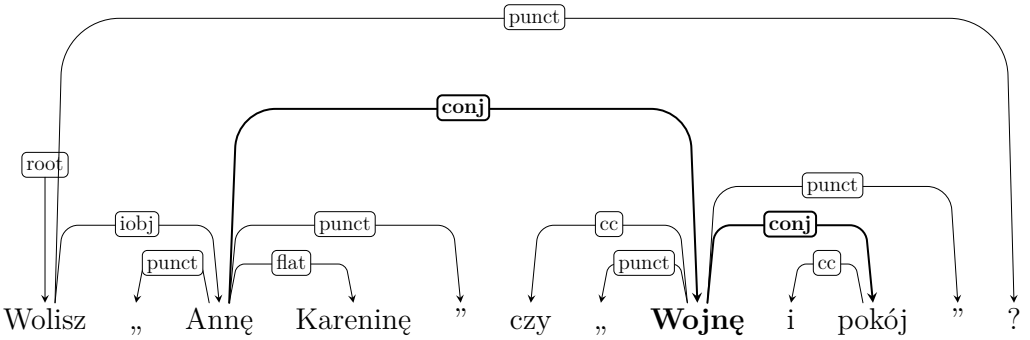
(40) [[Ania] i [Tomek]]

[5]Biblioteka ta nie obsługuje języka łacińskiego. Niemniej jednak prawie wszystkie (>99.5%) liczebniki w obrębie korpusów łaciny zapisane są słownie.

[6]Przykład pochodzi ze strony `https://www.korean.go.kr/eng_hangeul/principle/001.html`.

(41) [[Ania i Tomek zdecydowali], [nikt się nie sprzeciwiał]]

Jeden token może być głową lewego członu w jednej koordynacji i głową prawego w drugiej:

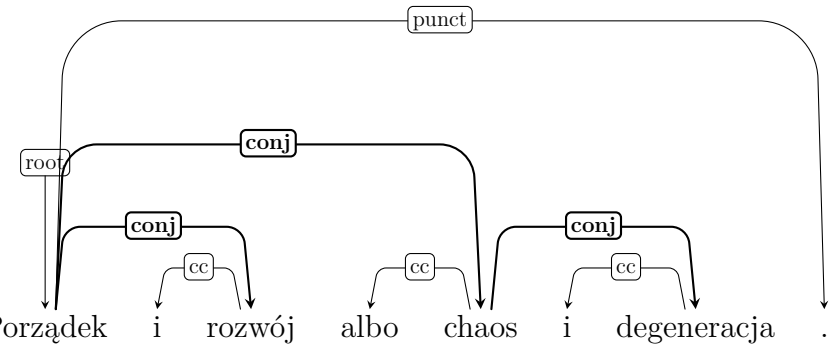
(42)


Nie stanowi to przeszkody dla wyciągania koordynacji według wyżej opisanych zasad. Algorytm (10) poprawnie wykrywa obie koordynacje:

(43) [[**Wojnę**] i [pokój]]

(44) [[„Annę Kareninę”] czy [„**Wojnę** i pokój”]]

Podejście używane przez UD dopuszcza sytuacje, w których jeden token jest głową lewego członu dwóch różnych koordynacji. W takich sytuacjach rozdzielenie koordynacji zagnieżdżonych jest trudniejsze. Pokazuje to przykład (45).

(45)


Według wcześniej opisanej procedury wyciągania koordynacji, zdanie (45) powinno zawierać następujące koordynacje:

(46) *[[Porządek] i [rozwój] albo [chaos i degeneracja]]

(47) [[chaos] i [degeneracja]]

Opis ten jest niepoprawny. W rzeczywistości w zdaniu (45) występują następujące koordynacje:

(48) [[Porządek i rozwój] albo [chaos i degeneracja]]

(49) [[Porządek] i [rozwój]]

(50) [[chaos] i [degeneracja]]

Ponieważ wyżej opisane reguły znajdowania głów członów koordynacji nie zadziałały poprawnie, należy zmodyfikować algorytm.

1.2.7. Procedura znajdowania koordynacji z uwzględnieniem koordynacji zagnieżdżonych

W celu poprawnej analizy zdań zawierających koordynacje zagnieżdżone w procedurze znajdowania koordynacji przyjmuję następującą regułę:

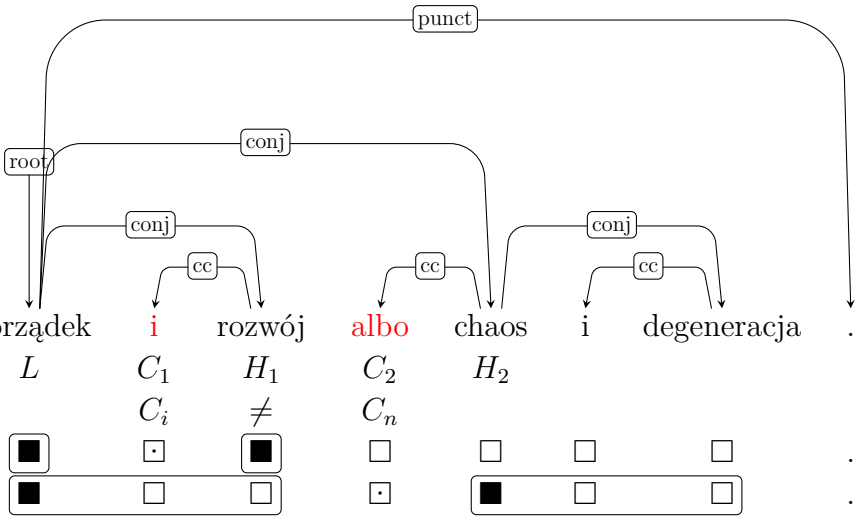
H5 Jeśli występuje sytuacja opisana w (10) oraz:

- istnieją przynajmniej dwa spójniki i
- jeden ze spójników C_i różni się od ostatniego spójnika C_n,

to należy dodatkowo rozpatrzeć koordynację, w której nadrzędnikiem jest G, głową lewego członu jest L, głową prawego członu H_i, a spójnikiem C_i.

Poniższe drzewo obrazuje zastosowanie (H5) do drzewa (45):

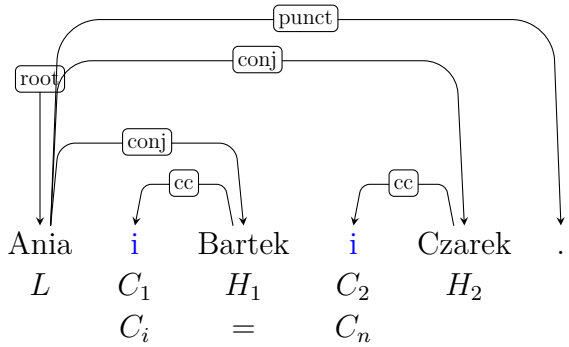
(51)



Ponieważ spójniki *i* oraz *albo* są różne, zgodnie z (H5) w zdaniu występują zagnieżdżone koordynacje (49) i (50).

W drzewie (52) występuje podobna sytuacja, jednak wszystkie spójniki podrzędników tokenu *Ania* są identyczne:

(52)



W związku z tym warunki (H5) nie są spełnione. Jest to zwykła koordynacja wieloczłonowa:

(53) [[Ania] i [Bartek] i [Czarek]]

1.3. Weryfikacja działania algorytmu

1.3.1. Ograniczenia

Wyżej opisana procedura wyciągania koordynacji nie jest niezawodna. Jest wiele powodów, dla których koordynacje mogą być źle opisane:

- nieprawidłowe dane w obrębie korpusu (np. ciągi losowych znaków, które nie są częścią języka naturalnego),
- nieprawidłowe drzewa zależnościowe (błędy w automatycznym opisie lub konwersji znakowania, błędy w ręcznym opisywaniu, błędy wynikające z niedoskonałości standardu UD),
- nieprawidłowe działanie heurystyk.

Z tych przyczyn algorytm został poddany ewaluacji. Opisana niżej procedura została opracowana na podstawie ewaluacji używanej w badaniu Przepiórkowski i Woźniak (2023) oraz replikującej je analizie Przepiórkowski i in. (2024).

1.3.2. Dobór języków

Ewaluacji zostały poddane zdania w następujących językach:

- język polski – dwóch natywnych recenzentów,
- język angielski – dwóch recenzentów,
- język turecki – jeden natywny recenzent.

1.3.3. Losowanie wyciągniętych koordynacji

W przypadku języka polskiego i angielskiego wylosowano 300 koordynacji z uwzględnieniem pozycji nadrzędnika – po 100 z nadrzędnikiem po lewej, po prawej i bez nadrzędnika. W przypadku języka tureckiego wylosowano 60 koordynacji bez rozróżnienia na pozycję nadrzędnika.

1.3.4. Ocena poprawności

Dwóch recenzentów niezależnie ocenia poprawność wyciągania koordynacji na podstawie dwóch kryteriów:

- Lewy i prawy człon koordynacji są **dokładnie** takie, jakie powinny być.
- Pozycja nadrzędnika jest poprawnie określona.

Następnie recenzenci spotykają się i rozstrzygają konflikty. Miarą oceny algorytmu jest stosunek liczby poprawnie opisanych koordynacji do liczby wszystkich koordynacji danego typu.

1.3.5. Wyniki

Tabela 1.2 przedstawia odsetek koordynacji ocenionych jako poprawnie wyciągnięte w językach poddanych ewaluacji.

Język	Wszystkie koordynacje	Nadrzędnik po lewej	Nadrzędnik po prawej	Brak nadrzędnika
polski	0.79	0.83	0.89	0.66
angielski	0.72	0.72	0.61	0.84
turecki	0.58[7]			

Tabela 1.2: Poprawność ewaluacji

Przed rozstrzygnięciem konfliktów współczynnik zgodności recenzentów κ wyniósł 56% dla języka polskiego i 54% dla języka angielskiego.

Zarówno współczynnik zgodności recenzentów, jak i odsetek poprawnych koordynacji algorytmu mają niższe wartości niż analogiczne miary uzyskane w pracy Przepiórkowski i Woźniak (2023). Należy jednak zwrócić uwagę, że w ich badaniu ewaluacja algorytmu polegała na sprawdzeniu wyłącznie poprawności pozycji nadrzędnika.

Przepiórkowski i in. (2024) ewaluując swój algorytm sprawdzali zarówno pozycję nadrzędnika, jak i granice członów. Stosując tę metodę uzyskali dokładność równą 50.1%. Oznacza to, że stosowana przeze mnie procedura wyciągania koordynacji uzyskała lepsze wyniki niż algorytmy używane w poprzednich badaniach.

Bibliografia

De Marneffe, M.-C., Manning, C. D., Nivre, J., i Zeman, D. (2021). Universal Dependencies. *Computational linguistics*, 47(2):255–308.

Kanayama, H., Han, N.-R., Asahara, M., Hwang, J. D., Miyao, Y., Choi, J. D., i Matsumoto, Y. (2018). Coordinate structures in Universal Dependencies for head-final languages. In *Proceedings of the Second Workshop on Universal Dependencies (UDW 2018)*, strony 75–84.

Przepiórkowski, A. i Patejuk, A. (2018). From lexical functional grammar to enhanced universal dependencies. In *Proceedings of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018)*, strony 2–4.

Przepiórkowski, A. i Woźniak, M. (2023). Conjunct lengths in English, dependency length minimization, and dependency structure of coordination. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, strony 15494–15512.

Przepiórkowski, A., Borysiak, M., i Głowacki, A. (2024). An argument for symmetric coordination from Dependency Length Minimization: A replication study. To appear in the proceedings of *LREC-COLING 2024*.

Riedl, M. i Biemann, C. (2018). Using semantics for granularities of tokenization. *Computational Linguistics*, 44(3):483–524.

Simpson, G. B. i Kang, H. (2004). Syllable processing in alphabetic Korean. *Reading and Writing*, 17:137–151.

Temperley, D. (2007). Minimization of dependency length in written English. *Cognition*, 105(2):300–333.