Tagmondatok és megszakítatlan összetevők kinyerése függőségi elemzésből

Szécsényi Tibor

SzTE, Általános Nyelvészeti Tanszék
szecsenyi@hung.u-szeged.hu

Kivonat: A természetesnyelv-feldolgozás során sokszor szükség van egy mondat tagmondatokra bontására, vagy az egy tagmondaton belüli legfelsőbb szintű összetevők meghatározására. Az itt bemutatott `dg2psg` eszköz egy mondat meglevő függőségi elemzése alapján képes elvégezni ezeket a feladatokat, vagyis egy mondat vagy mondatszakasz összes maximális megszakítatlan összetevőjét felsorolni rekurzív módon. Az eszköz kizárólag a függőségi elemzés éleit használja, az élcímkéket vagy a szavak egyedi tulajdonságait nem veszi figyelembe, ezért tetszőleges függőségi elemzési rendszerrel használható. A top-down elemző az összetevők felszíni pozícióját adja vissza. Egy külső definiálású tesztfüggvény segítségével képes az összetevők közül a tagmondatok kiválasztására.

1 Bevezetés

A nyelvfeldolgozáshoz vagy a nyelvleíráshoz szükséges összefüggések megállapításakor sokszor van szükség arra, hogy a nyelvi adatokat, mondatokat ne teljes egészében tekintsük, hanem annak csak bizonyos részeit vegyük figyelembe. Az igék argumentumszerkezetének meghatározásához vagy az anaforikus kifejezések antecedenseinek megtalálásához ismerni kell a mondat belső szerkezetét is, legalább olyan szinten, hogy az egyes tagmondatokat elkülöníthessük egymástól, illetve azonosítani kell a tagmondatban található maximális összetevőket is. Szerencsére már rendelkezésünkre állnak olyan eszközök, amelyek a szövegek, mondatok szintaktikai szerkezetét is megadják használható pontossággal, ezek az eszközök azonban legtöbbször a mondat függőségi elemzését végzik el, amiből közvetlenül nem hozzáférhetőek a keresett mondatszegmentumok.

Ez a tanulmány egy olyan egyszerű eszközt mutat be, amely egy mondat meglevő függőségi elemzésből közvetlenül visszaadja a mondatban található tagmondatok listáját, valamint az egyes tagmondatokban levő maximális összetevőket is. Az eszköz valójában ennél többet is tud, először a függőségi szerkezetet átalakítja frázisstruktúra-szerkezetté, majd ezután (vagy eközben) kikeresi a tagmondatokat (vagy bármilyen speciális összetevőt). A frázisstruktúra-szerkezetté alakítás közben a függőségi elemzésből kizárólag a függőségi éleket veszi figyelembe, azok címkéit nem, ezért tetszőleges függőségi elemzésnél alkalmazható – eddig a Szeged Dependency Treebank (Vincze és mtsai., 2010) függőségi elemzéseivel, valamint a magyarlanc (Zsibrita és mtsai., 2013) és az e-magyar (Váradi és mtsai., 2018) automatikus elemzők kimeneteivel használtuk, de más függőségi elemzővel is alkalmazható, mint például a HuSpaCy (Orosz

és mtsai., 2022, 2023). A tagmondatok kiválasztásához azonban függőséginyelvtan-specifikus információk is szükségesek, ezek a pluszinformációk azonban teljesen szabadon megfogalmazhatók. Az eszköz emiatt különbözik a (Dömötör és Nemeskey, 2023) által bemutatottól, amely függőségi elemzés éleivel és címkéivel megfogalmazott sémák alapján azonosítja a tagmondatokat és a maximális összetevőket, ezért függ az adott függőségi elemzés egyedi tulajdonságaitól.

Az eszköz a maximális összetevők meghatározása során a felszíni pozíciót veszi figyelembe, és így kezeli a távoli függőséget és a megszakított összetevőket is (minden összetevőt ahhoz a tagmondathoz köt, amelyikben megjelenik), sőt az olyan esetleges kódolási hibák esetén is működik, mint a függőségi elemzés során keletkezett zárványok (egymástól körkörösen függő szavak).

A tanulmány 2. szakaszában a függőségi elemzések és a frázisstruktúra elemzések tulajdonságait és egymásba való átalakíthatóságukat mutatom be. A 3. szakaszban az eszköz működésének leírásához használt fogalmakat ismertetem, majd a 4. szakaszban magának az eszköznek a működését. Az eszköz Python library formájában elérhető a githubon: https://github.com/szecsenyi/dg2psg

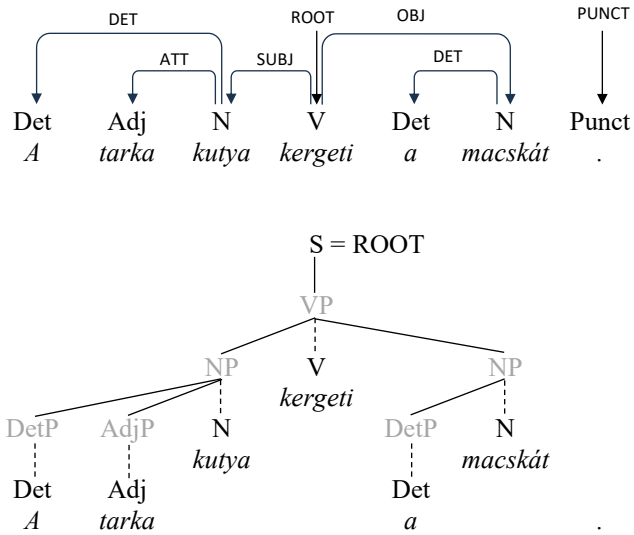
2 Dependencia és frázisstruktúra

Bár első pillanatra a függőségi elemzés és a frázisstruktúra elemzés nagyon különbözőnek tűnhet, valójában elég könnyen átalakítható az egyik elemzési mód a másikká. Ebben a fejezetben a két reprezentáció különbségeit és a reprezentációk közötti transzformációs nehézségeket fogom bemutatni, a fejezetben előforduló alapvető terminusokról részletesebben (Carnie, 2013) és (Kübler és mtsai., 2009) adnak részletesebb magyarázatot.

A függőségi elemzés során a mondatot alkotó szavak közötti függőségi viszonyok adják a függőségi fa éleit, az élek címkéjét pedig a függőségi viszony típusa határozza meg. A függőségi fa terminális és nem terminális csomópontjaihoz is szavakat rendelünk. Az összetevős szerkezeti elemzés során kizárólag az elemzési fa terminálisait címkézzük szavakkal, a nem terminális csomópontok pedig a mondat „összetevőinek" feleltethetőek meg valamilyen helyettesítési osztály címkéjével ellátva, vagyis olyan kifejezéseknek, amelyek bármilyen más, ugyanolyan címkéjű kifejezéssel helyettesíthetőek. Az összetevős szerkezeti fa nem terminálisai az egyik kitüntetett, fejnek tekintett szavának a szófajának, szintaktikai kategóriájának (N, V, Adj, Adv stb.) a kiterjesztett változatával vannak címkézve (NP, VP stb.). A függőségi elemzésben frázisnak tekinthetjük az egyes nem terminális csomóponthoz kapcsolt (rekurzívan értelmezett) dependenseket, amelynek a feje a nem terminális csomóponthoz tartozó szó.

A függőségi fából összetevős szerkezeti fát kaphatunk, ha a függőségi éleket közvetlen dominanciát kifejező éleknek tekintjük, továbbá a nem terminális csomópontokból egy újabb közvetlen dominancia élt húzunk, fej-leányként „leengedve" így a csomóponthoz tartozó szót (szófajcímkéjével együtt). A nem terminális csomópontokat ezután a fejük szófajának/szintaktikai kategóriájának a kiterjesztett (frazális) változatával címkézzük.

Az *A tarka kutya kergeti a macskát* mondat függőségi elemzését és az abból kapott összetevős elemzését láthatjuk a X ábrán:



1. Ábra. Az *A tarka kutya kergeti a macskát* mondat függőségi szerkezetének összetevős szerkezetté alakítása. Az eredeti függőségi élek címkézetlenül megmaradtak (nem szaggatott vonalak), de minden szó bevezet egy új összetevőségi élt (szaggatott vonalak). A nem terminálisok címkézése az alattuk levő terminális szófaja alapján alakul ki (szürke).

Ez az eljárás azonban nem eredményez feltétlenül jólformált összetevős szerkezeti fát. Ha a függőségi fa nem volt projektív, azaz tartalmazott egymást metsző éleket, akkor a kapott összetevős szerkezeti fa ágai is metszik egymást. A szabad szórendű nyelvekben, mint amilyen a magyar is, gyakran találkozhatunk ilyen nem projektív függőségi elemzéssel, pl. a *Péternek kiesett a foga* mondat esetében, ahol a *foga* fejhez tartozik a *Péternek* birtokos, ezért az őket összekötő él keresztezi a *kiesett* ige `root` címkéjű élét. Ez a rosszul formáltság megszüntethető azzal, ha a jobban beágyazott (a root elemtől távolabbi) él fej-végpontját magasabban levő csomóponthoz kötjük, ezáltal az összetevőt a felszíni pozíciójának megfelelő fejhez csatoljuk.

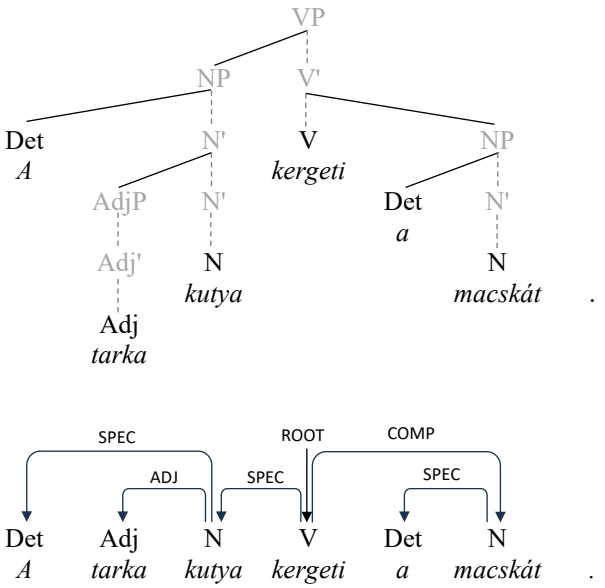
További problémaként jelentkezik az is, hogy a függőségi elemzésben az ugyanahhoz a fejhez tartozó közvetlen dependensek egyenrangúak, vagyis a kialakuló összetevős szerkezetben egymásnak (és a fejnek) testvérei lesznek, ezáltal nem érvényesül az összetevők belső hierarchikussága, ami például az X-vonás elméletben a fej bővítményinek a típusát szabályozza (specifikáló, adjunktum, komplementum).

A függőségi szerkezet X-vonásos frázisstruktúra szerkezetté alakítását írja le pl. (Xia és Palmer, 2001). A jelen tanulmányban bemutatott transzformációs eljárás ennél egyszerűbb, mivel a legtöbb esetben nincs szükség ilyen részletes frázisstruktúra-reprezentációra.

Az összetevős szerkezet függőségi szerkezetté alakítása során az előzőekben ismertetett eljárásnak pont a fordítottját kell elvégezni: mivel a függőségi elemzésben a nem terminálisok címkéi a mondat szavai, ezért az összetevős szerkezetben a terminális csomópontokban szereplő szavakat (X fejeket) kell „felemelni” abba a nem terminális csomópontba, amelynek ők a fejei (XP), a fej-leány ágakat pedig törölni, továbbá a hierar-

chikusabb összetevős szerkezetek esetén a nem fej leányokat a közbülső (X') csomópontok helyett az maximális projekcióba kell bekötni, majd a csomópontok korábbi szintaktikai címkéit törölni. A különböző összetevős szerkezeti reprezentációkban a nem fej összetevők viszonyát a fejhez különbözőféleképpen fejezik ki, az X-vonás elméletben például az XP hierarchiában elfoglalt helyük alapján megkülönböztetünk specifikálót, adjunktumot és komplementumot, a HPSG-ben (Pollard és Sag, 1994) pedig a leányok vannak címkézve (HEAD-DTR, COMP-DTRS, ADJUNCT-DTR, FILLER-DTR). Ezek alapján a transzformált függőségi élek is címkézhetők, akár még ennél specifikusabban is: a VP vagy TP összetevő specifikálója például SUBJ-ként.

Az összetevős szerkezet függőségi szerkezetté alakítását mutatja be a 2. ábra.



2. Ábra. Az *A tarka kutya kergeti a macskát* mondat X-vonás elméleti összetevős szerkezetének függőségi szerkezetté alakítása. A nem terminális csomópontok címkéi (szürke) törlésre kerülnek. A terminális csomópontok a fölöttük levő maximális projekciós csomópontokba másolódnak a szaggatott élek mentén. A nem maximális csomópontból induló élek a maximális csomóponthoz kerülnek (pl. N'→AdjP helyett NP→AdjP). Az élek irányítása a magasabban levő csomópontokból az alacsonyabban levők felé történik. Az élek címkézése az X-vonás elméletnek megfelelő SPEC, ADJ és COMP lesz. A legfelső csomópontba egy ROOT címkéjű él mutat.

Az ilyen irányú átalakításnál a nehézséget az okozza, hogy a függőségi elemzésnél a nem terminális csomópontok címkéi a mondat szavai, a többi dependens pedig ehhez van kapcsolva függőségi élekkel. Emiatt csak akkor problémamentes az átalakítás, ha az összetevős szerkezet endocentrikus, azaz minden egyes összetevőnek pontosan egy fej összetevője van. Ez a feltétel viszont a frázisstruktúra nyelvtanok esetében sokszor nem teljesül, például a funkcionális kategóriák sokszor zéró (meg nem jelenő) fejjel rendelkeznek, a mellérendelő szerkezetekben pedig egynél több fej található. További problémát jelentenek a funkcionális és a lexikai kategóriák egymáshoz való viszonya, például a TP-VP esetében annak eldöntése, hogy a segédige vagy az ige a fej, vagy a

DP-NP esetében a determináns és a főnév viszonya, illetve hogy transzformációs elemzéseknél a felszíni szerkezet adja vissza a szavak sorrendjét, de a valódi függőségi viszonyokat a kiinduló szerkezetből kellene származtatni.

Az összetevős szerkezetek függőségi szerkezetté alakításáról a magyarban (Simkó és mtsai., 2015) ad bővebb tájékoztatást, a függőségi és összetevős elemzés közti átalakításról lásd még (Zsibrita és mtsai., 2013).

3 A frázisstruktúrává alakítás során használt fogalmak

Az igei argumentumszerkezetek korpuszalapú vizsgálatát és jellemzését célzó korábbi kutatásunk során szükséges volt a korpuszban található mondatokban levő tagmondatok, és a tagmondatokban megjelenő legfelsőbb szintű összetevők azonosítása. Az erre a feladatra elkészített `dg2psg` eszköz a szavak, kifejezések felszíni pozíciója és a mondatok más elemzésekből származó függőségi szerkezetét használta fel. Az eszköz egy megadott mondatszakaszban – kiindulásként a teljes mondatban – keresi meg a maximális nagyságú megszakítatlan összetevőket, visszaadva azok kezdő és végpontját, valamint az összetevők fejének a pozícióját. A megtalált maximális megszakítatlan összetevők fej előtti és fej mögötti részén a keresést rekurzívan elvégezve az eszköz a mondatszakasz teljes, bár nagyon leegyszerűsített, felszíni összetevős szerkezetét megadja.

Az összetevők keresésénél kizárólag a mondatszakasz függőségi szerkezete lett figyelembe véve, a szavak egyedi tulajdonságai és a függőségi élek címkézése nem. A függőségi szerkezet kódolását a szokásosnak feltételezi az eszköz: minden szó egyedi címkével, azonosítóval rendelkezik, a függőségi jelölés pedig azt írja le, hogy az egyes szavak melyik másik szónak a módosítója, azaz a dependensekhez hozzá van rendelve a fej azonosítója. A következő példákban az `n:`*`szó`*`:m` egyszerűsített jelölést használom, ahol az n a szó azonosítója, m pedig a függőségi él által megadott fej azonosítója. A teljes mondat fejét 0 függőségi címkével jelölöm.

Megszakítatlan összetevőnek azt a megszakítatlan szósorozatot nevezzük, amelynél a szavak függőségi címkéi egy kivételével mind a szósorozaton belülre mutatnak. Az egyetlen, a szósorozatból kimutató függőségi címkéjű szót tekintjük a megszakítatlan összetevő fejének. Nem megszakítatlan összetevő az a szósorozat, amelynek egynél több feje van, vagy nincs egy sem.

Nem minden megszakítatlan összetevő valódi összetevője a mondatnak, az `1:`*`ugat`*`:0 2:`*`a`*`:4 3:`*`tarka`*`:4 4:`*`kutya`*`:1` mondatban a `3:`*`tarka`*`:4 4:`*`kutya`*`:1` megszakítatlan összetevő 3 címkéjű fejjel, de nem maximális, mivel nem része az `2:`*`a`*`:4` szó. **Maximális megszakítatlan összetevőnek** nevezzük azt a szósorozatot, amely az ugyanazzal a fejjel rendelkező megszakítatlan összetevők közül a leghosszabb.

Egy mondatszakaszban egy megszakítatlan maximális összetevőt nem bottom-up, a fejből kiindulva lehet meghatározni rekurzívan a fejhez csatolva a fej dependenseit, mert akkor nem tudjuk biztosítani a megszakítatlanságot. Ehelyett top-down módszerrel azonosítjuk a megszakítatlan maximális összetevőket egy mondatszakaszon belül. Először a teljes mondatszakaszt vizsgáljuk, hogy megszakítatlan összetevő-e. Ha igen akkor egyúttal a mondatszakasz maximális megszakítatlan összetevője is, ismert fejjel

és kezdő- és végponttal. Ha nem, azaz egynél kevesebb vagy több feje (a mondatszakaszon kívüli fejjel rendelkező függőségi él) van, akkor csökkentjük a mondatszakasz hosszát, elhagyva annak az utolsó elemét, és a megmaradt mondatszakaszon ismételten végrehajtjuk az ellenőrzést. A mondatszakasz hosszának a csökkentését egészen addig folytatjuk, amíg a maradék mondatszakasz nem lesz megszakítatlan összetevő (vagy egy szó hosszúságú mondatszakasz). Ezzel a módszerrel megtaláljuk az eredeti mondatszakasz első maximális megszakítatlan összetevőjét. A megtalált megszakítatlan összetevő utáni mondatszakaszon ismételten végrehajtva a maximális megszakítatlan összetevő keresését megkapjuk az eredeti mondatszakasz összes maximális megszakítatlan összetevőjét, azaz a közvetlen összetevőit, felsorolva azok fejét, illetve első és utolsó elemét.

	1:*tegnap*:2	2:*este*:7	3:*a*:4	4:*fiú*:7	5:*a*:6	6:*barátjával*:7	7:*sakkozott*:0	
1-6		|1	|2	|2	|3	|3		:1-2 *tegnap este*
3-6				|1	|2	|2		:3-4 *a fiú*
5-6						|1		:5-6 *a barátjával*

3. Ábra. A *Tegnap este a fiú a barátjával sakkozott* mondat ige előtti részében a megszakítatlan maximális összetevők azonosítása

A 3. ábrán a *Tegnap este a fiú a barátjával sakkozott* mondat ige előtti szakaszának a maximális megszakítatlan összetevők azonosítását mutatja be. Az első sor a mondat függőségi elemzését tartalmazza. Először a teljes 1-6 mondatszakaszban vizsgáljuk, hogy hány szó függőségi éle mutat ki a vizsgált szakaszból (3), majd a vége felől fokozatosan csökkentve a mondatszakasz hosszát ugyanezt megismételjük (1-5:3, 1-4:2, 1-3:2, 1-2:1) egészen addig, amig már csak egy él mutat ki a szakaszból. Az első ilyen mondatszakasz az 1-2 szakasz lesz, ez az első maximális megszakítatlan összetevő az 1-6 szakaszban: *tegnap este*. Ezután a következő körben az 1-6 szakasz maradék részében, vagyis a 3-6 szakaszban keressük meg az első megszakítatlan összetevőt: 3-4: *a fiú*, és így tovább.

A maximális megszakítatlan összetevő meghatározása során a vizsgált mondatszakasz végétől kezdődő visszanyesésekkel dolgozik, ezért az olyan zárvány függőségi szerkezeteket is képes feldolgozni, amelynek egyetlen eleme sem rendelkezik a mondatszakaszból kimutató függőségi éllel, pl. `1:w1:2 2:w2:3 3:w3:1`.

1-3	|1	|0	1-2:w1w2
3-3		|1	3-3:w3

4. Ábra. Megszakítatlan maximális összetevők zárvány függőségi szerkezetben

Ilyen függőségi szerkezet elméletileg nem létezhet, de elemzési hibaként alkalmanként előfordul, például kézi elemzés után. A definíció az 1-3 mondatszakaszt nem tekinti maximális megszakítatlan összetevőnek, de mondatszakasz jobbról történő lerövidítésével kapott 1-2 mondatszakaszt már igen, és a maradék 3-3 mondatszakasz szintén megszakítatlan maximális összetevő. Ez azért fontos, mert így az eszköz nagyon

robusztus, nem akasztják meg a működését az ilyen jellegű kódolási hibák sem. A kézzel annotált Szeged Dependency Treebank három zárványos elemzésű mondatot tartalmaz, mindhárom a szerzői jogi részkorpusz található: a 3019., a 3170. és a 3172. mondatokban. A mondatok megtalálhatóak a https://github.com/szecsenyi/dg2psg repozitóriumban.

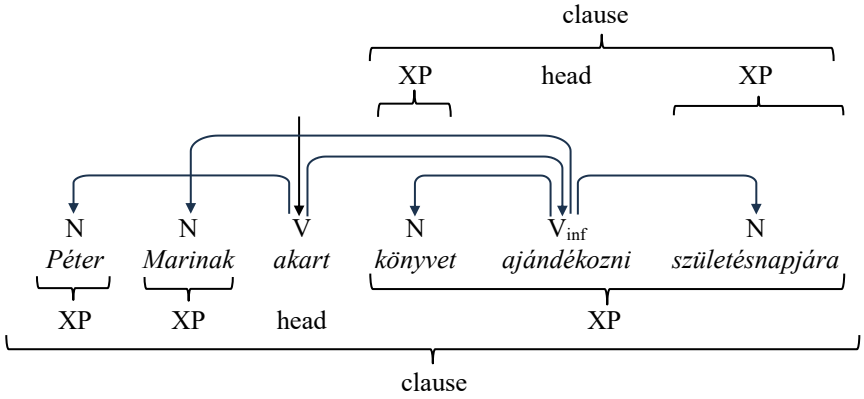
Az így megtalált maximális megszakítatlan összetevők mind legmagasabb szintűek, a megadott mondatszakasz közvetlen összetevői. Az alacsonyabb szinten levő, azaz a beágyazott maximális megszakítatlan összetevőket is azonosíthatjuk, ha az összetevők fej előtti, illetve a fej utáni szakaszán rekurzívan elvégezzük a maximális megszakítatlan összetevők keresését. Így hozzáférhetünk a teljes mondat valamennyi összetevőjének a listájához, amelyből rekonstruálhatjuk a mondat összetevős szerkezetét is.

Az igei argumentumszerkezet leírásához a tagmondatok listájára van szükség. Ehhez egyszerűen a kapott összetevőlistát kell szűrni: azokat a maximális megszakítatlan összetevőket kell kiválasztani, amelyek valamilyen teszt alapján tagmondatnak minősíthetőek. Vizsgálatunk során azokat az összetevőket tekintettük tagmondatnak, amelyeknek a feje igei POS-taggel rendelkeztek. Más típusú összetevőket is kilistázhatunk azonban, például ha a határozott ragozást kiváltó kifejezéseket kívánjuk vizsgálni, akkor azokat az összetevőket gyűjtjük ki, amelynek a feje tárgy esetű, az összetevő első szava pedig határozott névelő.

Az előzőekben megadott maximális megszakítatlan összetevő definíciójával az összetevők a felszíni pozíciójukban jelennek meg az elemzés során. A `1:`*Péternek*`:4 2:`*elveszett*`:0 3:`*a*`:4 4:`*kalapja*`:2` mondatban a függőségi elemzés *Péternek a kalapja* birtokos szerkezetben a *Péternek* birtokot a *kalapja* birtokhoz kapcsolja dependensként, az összetevős elemzés során azonban a *Péternek a kalapja* nem megszakítatlan összetevő, hanem a *Péternek* birtokot és az *a kalapja* birtokost az ige előtti és mögötti összetevőként lesz felsorolva, vagyis a birtok az *elveszett* fejű tagmondat önálló maximális megszakítatlan összetevője, függetlenül attól, hogy az nem az igei fej közvetlen dependense. Ezzel szemben az `1:`*elveszett*`:0 2:`*Péternek*`:4 3:`*a*`:4 4:`*kalapja*`:1` mondatban a birtok és a birtokos (*Péternek a kalapja*) már egyetlen összetevő lesz a tagmondatban.

Végül egy utolsó példa: a *Péter Marinak akart könyvet ajándékozni születésnapjára* mondat függőségi elemzése látható az 5. ábrán.

A függőségi szerkezet nem projektív, a főnévi igenévi fejet és a datívuszi dependensét összekötő függőségi él két másik élt is metsz. A mondatban két maximális megszakítatlan összetevőt azonosíthatunk tagmondatként (clause). Az ábra felső részén jelölve láthatjuk a főnévi igenévi fejű tagmondatot (*könyvet ajándékozni születésnapjára*), amiben két maximális megszakítatlan összetevő (XP) van a legfelső szinten: *könyvet* és *születésnapjára*. A főnévi igenév datívuszi bővítménye a felszínen nem a tagmondat összetevője. A másik tagmondat, a teljes mondat, az ábra alján van részletezve. A ragozott igei fejű tagmondatban a fej mellett három maximális megszakítatlan összetevőt találunk, a *Péter* alanyt, a *könyvet ajándékozni születésnapjára* tagmondatot, valamint a datívuszi *Marinak* főnévi csoportot. Ez utóbbi a függőségi elemzés alapján nem közvetlen dependense a ragozott igei fejnek, de a felszínen annak a bővítménye, fókusza.



5. Ábra. A *Péter Marinak akart könyvet ajándékozni születésnapjára* mondat függőségi elemzése, valamint a mondatban található tagmondatok és azok maximális megszakítatlan összetevői

4 A `dg2psg` eszköz bemutatása

A https://github.com/szecsenyi/dg2psg github repozitóriumban is megtalálható `dg2psg` python library az eddig ismertetett módon adja meg egy mondat maximális megszakítatlan összetevőinek vagy a tagmondatainak a listáját. Az eszköz a tagmondatmeghatározást végző `allSpecXP` függvény kivételével kizárólag a függőségi éleket veszi figyelembe.

A függőségi él reprezentációjaként egy egész számot használ, ami az adott dependens fejének mondatbeli pozíciójára utal, a mondatbeli szavakat 1-től számozva. A mondat feje esetén ez az érték 0. A n szavas mondat releváns függőségi elemzése tehát egy n elemű lista, a korábban elemzett `1:`*Péternek*`:4 2:`*elveszett*`:0 3:`*a*`:4 4:`*kalapja*`:2` függőségi elemzésnél ez a `[4,0,4,2]` lista. természetesen ettől eltérő függőségi kódolást is használhatunk, ebben az esetben a `getDep` függvénnyel alakíthatjuk át a meglevő függőségi elemzést az eszköz által használt alakra.

Az eszköz központi eleme a `closedXP` függvény, amely eldönti, hogy egy mondatszakasz megszakítatlan összetevő-e. Ennek megfelelően három argumentuma van, az első a mondatszakasz első, a második a mondatszakasz utolsó szavának a sorszáma, a harmadik pedig a függőségi élek listája. A –1 vagy –2 visszaadott érték jelzi, ha a mondatszakaszban nincs kívülre mutató függőségi él, vagy ha egynél több ilyen is van, egyébként pedig a megszakítatlan összetevő fejének a sorszámát adja vissza.

Ugyanilyen argumentumokkal használható a `firstMaxXP` és az `allMaxXP` függvény is, amelyek egy mondatszakasz első, illetve összes legfelsőbb szintű maximális megszakítatlan összetevőjét adják vissza, az előbbi az összetevő első elemének, fejének és utolsó elemének sorszámából álló három elemű tuple-t, a második pedig egy ilyenekből álló listát. Az `allMaxXP` függvény rekurzív változata az `allMaxXP_recursive`, amely nem csak a mondatszakasz legfelsőbb szintű összetevőinek a listáját adja vissza, hanem az alacsonyabb szintűeket is: a megtalált maximális megszakítatlan összetevők belsejében, azok fej előtti és fej utáni mondatszakaszában is megkeresi az

összetevőket (és azokban is stb.). A visszaadott érték az azonosított összetevőket jellemző `(start,head,end)` tuple-k listája, ahol az összetevők egymásba ágyazottsága nincs jelölve.

A tagmondatok azonosításához az `allMaxXP_recursive` kibővített változatát használhatjuk, amelyben lehetőség van az összetevők tesztelésére is: `allSpecXP`. Ehhez egy külső tesztfüggvényt használunk, amit az `allSpecXP` argumentumként kap meg, a mondat elemzését tartalmazó további információval együtt. A tesztfüggvényre az `allSpecXP`-ben `XPtest`-ként hivatkozhatunk, aminek négy argumentummal kell rendelkeznie, az összetevőt azonosító start, head és end értékeken kívül a mondatszerkezet leírását és az eddig is használt, függőségi éleket felsoroló listát. Argumentumszerkezeti vizsgálataink során az e-magyar (Váradi és mtsai., 2018) előfeldolgozóval elemzett emtsv kódolású mondatokat olyan listaként reprezentáltuk, amelyben a szavak mezőiből álló dictionary-k volt voltak a lista elemei. Ekkor a következő teszfüggvényt adtuk át argumentumként az `allSpecXP`-nek:

```
def CPtest(sentence, start, head, end, deps):
    return sentence[head-1]['upostag'] == 'VERB'
```

vagyis a mondat `head-1`-dik szavának (ez a tagmondat feje) upostag jegyének az értékét vizsgáltuk, hogy az megegyezik-e az igék `VERB` szófajával. A tagmondatok felsoroltatását pedig az

```
allSpecXP(sentence, 1, len(sentence), deps, CPtest)
```

függvényhívással kezdeményeztük (ahol a `sentence` teljes mondat emtsv-reprezentációja, a `deps` pedig az ebből kinyert függőségi lista).

A 4. ábrán is bemutatott zárvány szerkezeteket is tudja kezelni az eszköz, az ábrán bemutatott módon alakítva ki az összetevőket. A zárványokat a `closedXP` függvény érzékeli. Ha az allMaxXP vagy az allMaxXP_recursive függvények checkNohead argumentumát True értékre állítjuk, akkor a függvények zárvány észlelésekor '`Dependency graph is cyclic`' exception üzenetet adnak vissza, így az eszköz a zárványok keresésére is használható.

5 Kiértékelés, hibaelemzés

Az eszköz megbízhatóságának mérésére a Szeged Dependency Treebank (Vincze és mtsai., 2010) nyolcadik osztályos elbeszélő fogalmazásainak függőségi elemzéséből gyűjtöttem ki a tagmondatokat, és a tagmondatlistát a Szeged Treebank 2.0 (Csendes és mtsai., 2005) ugyanezen részkorpuszának összetevős elemzésének a tagmondatlistájához hasonlítottam.

A tagmondatokat mindkét esetben a tartalmazó mondat sorszámával, valamint a tagmondat első és utolsó szavával jellemeztem, így a két tagmondatlista könnyen összehasonlítható volt. A kezdő és záró szavak sorszáma pontosabbnak tűnhet, de a két korpuszban néhol eltérő a szegmentáció, illetve a dependenciakorpuszban találhatóak üres elemek is, ami az tagmondatok egymáshoz illesztését nehezítette volna.

A Szeged Treebank (SzTB) XML kódolású korpusz, amelyben a szöveg tokenjei köré a szintaktikai szerkezetet jelölő tag-ek kerültek. Ezek közül a `CP` tag jelölte a tagmondatokat. A tagmondatlista meghatározása során a mondatokban található `CP` tagen belüli első és utolsó `w` (szó) elemek tartalmát, azaz a szó formáját vettem figyelembe, a `c` (írásjel) elemeket nem. A korpusz 7575 mondata 19 372 tagmondatot tartalmaz.

A dependenciakorpuszban a tagmondatok azonosítása során használt `CPTest` fejének a szófaja `VAN` (zéró létige), `ELL` (elliptált ige) vagy `V` (ige), de a tagmondatok közül kiszűrtem a főnévi igeneves fejűeket (`Mood=n`), mivel az összetevős elemzés azokat nem jelölte külön tagmondatnak. A tagmondatok első és utolsó szavának meghatározásánál átléptem az első és utolsó `VAN` és `ELL` tokeneket, valamint a pont és vessző írásjeleket. A `dg2psg` így összesen 15447 tagmondatot azonosított.

A tagmondatsorok összehasonlítását a Notepad++ szerkesztő compare funkciójával végeztem, amely minimal edit distance módszerrel jelöli a különbségeket (azonosság, beszúrás, törlés, csere). A pontosság és fedés számítását Excelben végeztem: true positive találatnak tekintettem a tagmondatok azonosságát, false positive találatnak a cserét és a törlést, false negative találatnak a cserét és a beszúrást. A teljes korpuszban így 13176 tp, 2271 fp és 6196 fn találat volt, ami 0,85 pontosságot (P), 0,68 fedést (R) és 0,76 F1 értéket adott.

Hibaelemzésként a korpusz első száz mondatán vizsgáltam meg, hogy milyen tipikus hibák jelentkeztek. A száz mondatban a `dg2psg` összesen 163 tagmondatot azonosított, ebből 150 tp, 13 fp találat volt, ezen kívül 49 fn tagmondatot nem jelzett.

A 49 fn találatból 32-t a mellérendelések függőségi és az összetevős elemzők általi különböző elemzése okozott. Összetevős elemzésnél a mellérendelt tagmondatok egyenként is `CP` jelölést kaptak, valamint együtt is, így két ténylege tagmondat esetén három `CP` lett jelölve. A függőségi elemzésben az első mellérendelt tagmondat fejéhez kapcsolódik a mellérendelő kötőszó, amihez a következő tagmondat feje, így az eszköz a mellérendelést két tagmondat alárendelésének látja. A mellérendelések miatti fn hibákat szükség esetén a tagmondatlista utólagos vizsgálatával és javításával kiküszöbölhetjük. További false negative találatokat okoztak azok a mondatok (5 db), amelyek valójában nem is mondatok, hanem például címek, felkiáltások (pl. *Gól!!!*), vagy a függőségi elemzésben nem jelölt ellipszisek (6 db).

A 13 false positive közül 2-t a múlt idejű feltételes módú igék (pl. *futott volna*) *volna* részének önálló tagmondatként való bejelölése okozta. Ha ezt hibának tekintjük, könnyen kiküszöbölhető az egyszavas *volna* tagmondatok kiszűrésével. A többi fp találat nagy rész kapcsolatban áll a mellérendelés eltérő ábrázolásával.

Ha a koordinációs hibát nem tekintjük fn hibának a *volna* hibát pedig fp hibának, az első 100 mondatot figyelembe véve 150 tp, 11 fp és 17 fn hibát találunk, ezekkel számolva a dg2psg P=0,93 pontossággal, R = 0,90 fedéssel és 0,91 F1 értékkel rendelkezik.

A kiértékeléshez használt Jupyter notebook és a párosított tagmondatlista megtalálható a https://github.com/szecsenyi/dg2psg github repozitóriumban.

6 Összegzés

A tanulmányban bemutatott `dg2psg` eszköz egy mondat meglevő függőségi elemzése alapján képes a mondat összetevőinek a meghatározására. Az eszköz a mondatnak azt a maximális hosszúságú szakaszát tekinti összetevőnek, amelynek a függőségi elemzés alapján pontosan egy feje, azaz egyetlen a mondatszakaszon kívülre mutató függőségi éle van. Mivel a mondat minden eleméhez (szavához) tartozik egy ilyen függőségi él, az eszköz a mondat minden eleméhez hozzárendel egy ilyen maximális megszakítatlan összetevőt.

Az eszköz fölülről lefelé, a legnagyobb összetevőtől a legkisebb felé haladva azonosítja az összetevőket. Az eszköz egy külső definiálású tesztfüggvény segítségével használható a mondat összes valamilyen tulajdonságú összetevőinek, például az összes tagmondatának a kigyűjtésére, vagy egy megadott mondatszakasz, például egy tagmondat összes legfelsőbb szintű összetevőjének az azonosítására egyaránt.

Az eszköz használhatóságát növeli, hogy az összetevő-keresés során kizárólag a függőségi éleit veszi figyelembe, a függőségi élek címkézését vagy a mondat szavainak egyedi tulajdonságát nem, ezért bármilyen függőségi elemzési rendszernél használható. A robusztusságot növeli az a tény, hogy az esetleges függőségi elemzési hibákat, például a zárványokat is tudja kezelni.

Az eszköz elsősorban igei fejű tagmondatok, és a tagmondatokban található összetevők meghatározására alkalmazható. Emiatt kiválóan alkalmas előre rögzített igék argumentumszerkezetének tanulmányozásához, különböző igék argumentumszerkezeteinek összehasonlításához, mint például (Gyulai, 2019, 2021, 2023; Szécsényi, 2019; Szécsényi és Kovács, 2020; Szécsényi & Virág, 2022).

Bibliográfia

Carnie, A.: *Syntax: A generative introduction* (Third Edition). Wiley-Blackwell (2013)

Csendes, D., Csirik, J., Gyimóthy, T., Kocsor, A.: The Szeged Treebank. In V. Matoušek, P. Mautner, T. Pavelka (Szerk.), *Text, Speech and Dialogue.* pp. 123–131. Springer Berlin Heidelberg (2005) http://link.springer.com/10.1007/11551874_16

Dömötör A., Nemeskey D.: Tagmondatokra bontás és NP-chunking függőségi alapon. In Berend G., Gosztolya G., Vincze V. (Szerk.), *XIX. Magyar Számítógépes Nyelvészeti Konferencia.* pp. 457–469. Szegedi Tudományegyetem, Informatikai Intézet (2023)

Gyulai L.: Nem kompozicionális igekötős igék argumentumszerkezetének korpuszalapú vizsgálata. In Ludányi Z., Gráczi T. E. (Szerk.), *Doktoranduszok tanulmányai az alkalmazott nyelvészet köréből 2019. XIII. Alkalmazott Nyelvészeti Doktoranduszkonferencia.* pp. 44–58. MTA Nyelvtudományi Intézet (2019) https://doi.org/10.18135/Alknyelvdok.2019.13.4

Gyulai L.: Az igekötők legjellemzőbb argumentumszerkezetváltoztató hatásainak korpuszalapú vizsgálata. In Gráczi T. E., Ludányi Z. (Szerk.), *Alknyelvdok15. Doktoranduszok tanulmányai az alkalmazott nyelvészet köréből.* pp. 176–198. Nyelvtudományi Kutatóközpont (2021) http://www.nytud.hu/alknyelvdok21/proceedings/gyulai.pdf

Gyulai L.: Az igék csoportosítása az igekötők argumentumszerkezetben okozott változása alapján. *Alkalmazott nyelvtudomány*, *23*(2), 157–173 (2023) https://doi.org/10.18460/ANY.K.2023.2.009

Kübler, S., McDonald, R., Nivre, J.: *Dependency parsing*. Morgan and Claypool Publishers (2009)

Orosz, G., Szabó, G., Berkecz, P., Szántó, Z., Farkas, R.: Advancing Hungarian text processing with HuSpaCy: Efficient and accurate NLP pipelines. In K. Ekštein, F. Pártl, M. Konopík (Szerk.), *Text, Speech, and Dialogue*. pp. 58–69. Springer Nature Switzerland (2023)

Orosz G., Szántó Z., Berkecz P., Szabó G., Farkas R.: HuSpaCy: An industrial-strength Hungarian natural language processing toolkit. In *XVIII. Magyar Számítógépes Nyelvészeti Konferencia*. pp. 59–73. Szegedi Tudományegyetem, Informatikai Tanszékcsoport (2022)

Pollard, C., Sag, I. A.: *Head-driven phrase structure grammar*. CSLI, University of Chicago Press (1994)

Simkó K. I., Vincze V., Szántó Z., Farkas R.: Konstituensfák automatikus átalakítása függőségi fákká vagy kézi annotáció? In Tanács A., Varga V., Vincze V. (Szerk.), *XI. Magyar Számítógépes Nyelvészeti Konferencia*. pp. 49–60. Szegedi Tudományegyetem, Informatikai Intézet (2015) https://m2.mtmt.hu/api/publication/2807221

Szécsényi T.: Argumentumszerkezet-variánsok korpusz alapú meghatározása. In Berend G., Gosztolya G., Vincze V. (Szerk.), *XV. Magyar Számítógépes Nyelvészeti Konferencia*. pp. 315–329. Szegedi Tudományegyetem, Informatikai Intézet (2019)

Szécsényi T., Kovács V.: A topikalizálhatóságot befolyásoló tényezők statisztikai vizsgálata. In Dékány É., Halm T., Surányi B. (Szerk.), *Általános nyelvészeti tanulmányok XXXII.: Újabb eredmények a grammatikaelmélet, nyelvtörténet és uralisztika köréből*. pp. 237–247. Akadémiai Kiadó (2020) http://publicatio.bibl.u-szeged.hu/19744/

Szécsényi T., Virág N.: Az ige helyhatározói bővítményeinek megkülönböztetése és az argumentumszerkezeti variánsok korpusz alapú szétválasztása. In Berend G., Gosztolya G., Vincze V. (Szerk.), *XVIII. Magyar Számítógépes Nyelvészeti Konferencia*. pp. 535–647. Szegedi Tudományegyetem, Informatikai Intézet (2022)

Váradi, T., Simon, E., Sass, B., Mittelholcz, I., Novák, A., Indig, B., Farkas, R., Vincze, V.: E-magyar – A Digital Language Processing System. In N. Calzolari (Szerk.), *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. pp. 1307–1312. European Language Resources Association (ELRA) (2018)

Vincze, V., Szauter, D., Almási, A., Móra, G., Alexin, Z., Csirik, J.: Hungarian Dependency Treebank. In *Proceedings of the Seventh Conference on International Language Resources and Evaluation*. pp. 1855–1862. European Language Resources Association (2010) http://www.lrec-conf.org/proceedings/lrec2010/index.html

Xia, F., Palmer, M.: Converting dependency structures to phrase structures. In *Proceedings of the first international conference on Human language technology research—HLT '01*. pp. 1–5. Association for Computational Linguistics (2001) https://doi.org/10.3115/1072133.1072147

Zsibrita, J., Vincze, V., Farkas, R.: magyarlanc: A Toolkit for Morphological and Dependency Parsing of Hungarian. In *Proceedings of RANLP 2013*. pp. 763–771 (2013) http://publicatio.bibl.u-szeged.hu/3981/