Ismert névelemek felismerése és morfológiai annotálása szabad szövegben

Tikk Domonkos[1], Szidarovszky Ferenc P.[2], Kardkovács Zsolt Tivadar[1], Magyar Gábor[1]

[1] Budapesti Műszaki és Gazdaságtudományi Egyetem, Távközlési és Médiainformatikai Tanszék, H-1117 Budapest, Magyar Tudósok krt. 2.
{tikk, kardkovacs, magyar}@tmit.bme.hu
[2] Szidarovszky Kft. H-1392 Budapest, Pf. 283.
ferenc.szidarovszky@szidarovszky.com

Kivonat: „A szavak hálójában” projekt[61] keretében készülő internetes keresőszolgáltatásnak egyik célja az, hogy lehetőséget nyújtson természetes nyelvű magyar kérdésekkel internetes adatbázisok tartalmában az ún. mélyhálóban való keresésre. Az adatbázisokból ki lehet nyerni azokat az *egyedi azonosítókat*, amelyek együttese lehetővé teszi, hogy a felhasználói keresések információigénye és a mélyhálós tartalmak között kapcsolatot teremtsünk. Az egyedi azonosítókat *névelemnek* nevezzük. A természetes nyelvű kérdések feldolgozásának kiemelt fontosságú része a bennük szereplő ismert névelemek felismerése, valamint a kérdésben betöltött szerepük meghatározásához a felismert névelemek morfológiai jegyeinek meghatározása. Cikkünkben bemutatjuk a probléma megoldására javasolt és megvalósított algoritmusunkat, amely számítási igényt tekintve is hatékonyan oldja meg a felvázolt feladatokat.

1 Bevezetés

„A szavak hálójában” projekt[2] keretében készülő internetes keresőszolgáltatást végző alkalmazásnak egyik célja az, hogy lehetőséget nyújtson természetes nyelvű magyar kérdésekkel internetes adatbázisok tartalmában az ún. mélyhálóban való keresésre. Az adatbázisokból az adatgazdákkal történő együttműködés eredményeként ki lehet nyerni azokat az *egyedi azonosítókat* pl. könyvtári adatbázis esetén a szerzők, kiadók, műcímek stb. nevének listáját , amelyek együttese lehetővé teszi, hogy a felhasználói keresések információigénye, és a mélyhálós tartalmak között kapcsolatot teremtsünk, és ezzel a kérdés megválaszolását megkönnyítsük. A mélyháló jellegzetességei és keresésének jelentősége [1, 6], valamint a projekt keretében kidolgozott mélyhálós internettartalmak keresését végző rendszerünk [4, 5] felől érdeklődő Olvasók számára – terjedelmi okok miatt – a megadott irodalmi forrásokat ajánljuk.

Cikkünk felépítése a következő. Először a 2. szakaszban meghatározzuk az általunk feldolgozott névelemek körét, és ismertetjük, hogy milyen problémákat kell

[61] NKFP-0019/2002 projekt

megoldania a névelem felismerő algoritmusnak. A 3. szakaszban részletesen ismertetjük az általunk javasolt algoritmust, majd a 4. szakaszban működését példákon keresztül is bemutatjuk. Végül az 5. szakaszban röviden összegezzük a cikk lényeges eredményeit.

2 Névelemek és felismerésük problematikája

Az egyedi azonosítókat *szótári*, vagy *ismert névelemnek* nevezzük, amelyeket a *névelemtárban* tárolunk. A szótári jelzőt a *minták alapján felismert névelemektől* (pl. dátumok, postai és internetes címek, stb.) való megkülönböztetésre használjuk, hangsúlyozandó azt, hogy a névelemtárban szereplő névelem bejegyzéseket szótári (kanonikus) alaknak tekintjük. A szótári névelemek nagy részét a fenti meghatározás miatt a tulajdonnevek teszik ki, azonban alkalmazásunkban a fogalomba beleértjük az olyan rögzített alakú közneveket is, amelyeknek kiemelt szerepe van bizonyos minták alapján felismert névelemtípusok (mennyiségek, címek, stb.) és egyéb, az elemzett kérdés további feldolgozása szempontjából fontos fogalmak azonosítása során. Eszerint névelemnek tekintjük pl. az alábbi csoportokba tartozó közneveket: a pénznemek jelölései (forint, euró, stb.), nemzetiségnevek (magyar, szlovák, angol, stb.), közterülettípus (út, utca, tér, stb.), stb.

A névelemtárnak az adatbázisból történő feltöltése során szemantikai információkat rendelünk az egyes elemekhez, amelyeket az adat adatbázisbeli séma- és attribútum-információiból nyerünk ki. A névelemtárban lehetőség van a kanonikus alak lehetséges szinonimáinak[62] megadására is (pl. *Petőfi Sándor* bejegyzéshez *Petőfi* szinonima, vagy a *forint* bejegyzéshez a *HUF* szinonima).

A névelemtár elemei meghatározzák azt az információs teret, amelyben a felhasználó kérdésére választ tudunk adni. Ez azt jelenti, hogy csak azokat a kérdéseket tudjuk megválaszolni a mélyhálós tartalmak segítségével, amelyekben ezen tartalmakból kinyert névelemek szerepelnek. Összességében az alábbi megszorításokat tesszük a felhasználó kérdéseire vonatkozóan, a listában szerepelnek a tartalmi vonatkozású megkötések is:

- csak egyszerű, azaz nem összetett mondatokat fogadunk el;
- csak helyesen írt, és nyelvtanilag helyes mondatokat fogadunk el;
- csak kérdőszóval kezdődő, nem eldöntendő kérdést fogadunk el; a lehetséges kérdőszavakat is korlátozzuk;
- Szubjektív (*Hány éves a kapitány?*), ok-okozati viszonyra irányuló (*Miért tört ki a II. világháború?*), vagy egyéb nem tényszerű, illetve nem a fenti információs térben található mondatok helyes megválaszolását nem garantáljuk.

A természetes nyelvű kérdések feldolgozásának tehát kiemelt fontosságú része a bennük szereplő ismert névelemek felismerése, valamint a kérdésben betöltött szerepük meghatározásához a felismert névelemek morfológiai jegyeinek meghatározása .Ez a toldalékoló magyar nyelv esetén korántsem egyszerű feladat, mivel a névelemek nem feltétetlen rögzített alakjukban (beleértve a szinonimákat) fordulnak elő, hanem többnyire toldalékolt alakban. A toldalék megváltoztathatja a névelem szótövét, illetve ha a szótári alak már eleve toldalékolt, akkor ezt is módosíthatja[63]. Továb-

[62] Nem toldalékolt alakok, csak lehetséges különböző előfordulásai a kanonikus alaknak

[63] Ld. *Vissza a jövőbe* és *Hol adják a Vissza a jövőbét?*

bi gondot jelenthet az egymásba ágyazott névelemeknél a névelem határainak meghatározása[64] [3]. Ha ez utóbbi esetben több értelmezés lehetséges, akkor alternatívákat állítunk elő. A morfológiai jegyek meghatározásánál a nem alanyesetű kanonikus alakok és a nem magyar (azaz morfológiai elemző által fel nem ismert) névelemek *speciális* esetei kívánnak külön megfontolást.

Cikkünkben bemutatjuk a probléma megoldására javasolt és megvalósított algoritmusunkat, amely azon kívül, hogy a fenti feladatokat megoldja, mindezt a számítási igényt tekintve hatékonyan valósítja meg. Az ismertetett módszer a HunMorph [2] szabad forráskódú statisztikai alapú morfológiai elemzőt használja, ennek megfelelően a példáknál található morfológiai elemző eredmények is a HunMorph kódolása szerint vannak megadva.

Fontosnak tartjuk kiemelni, hogy a módszer *nem felügyelt tanuláson alapul*, mivel célja nem ismeretlen névelemek felismerése, hanem az ismertek pontos azonosítása.

3 Szótári névelemek felismerése

A szótári névelem (ezentúl itt csak *névelem*) felismerőnek két fő célja van:

– *keresés*: a mondatban szereplő névelemek megtalálása;

– *annotálás* (vagy *címkézés*): a névelemek morfológiai jegyeinek meghatározása.

A keresés és annotálás folyamata általában összekapcsolódik, így önmagukban nem hajthatók végre.

Mivel egy névelem több szóból is állhat, a kérdőmondat tetszőleges szegmense (szavak rögzített sorrendű sorozata) lehet névelem. Egy n szavas kérdőmondat szegmenseinek száma $n(n+1)/2$. Egy átlagos kérdőmondat 7-10 szóból áll, míg a névelemtár mérete 10^6 nagyságrendű is lehet. Így sokkal hatékonyabb a mondatszegmensekből kiindulva keresni, mint a névelemtárból kiindulva. Egy kifejezés keresése a névelemtárban gyorsítható a névelemtár elemeinek hash-elésével. A mondatszegmensek összevetése a névelemtárral a szegmensek hossza szerint csökkenő sorrendben történik.

A névelem felismerés egy másik problémája az, hogy egy névelem tartalmazhat egy másikat (pl. a *The New York Times* egy napilap). Míg a Blitz NL feldolgozó [3] a felismert névelemek közül csak egyet választ ki konfidencia értékek alapján, mi fel kívánjuk ismerni az összes névelemet, különböző mondat alternatívákat létrehozva. Ebből kifolyólag az összevetés a keresés eredményétől függetlenül tovább folytatódik a rövidebb szegmensekkel.

A szegmensek összevetése az alábbi sorrendben történik:

1. A teljes mondattal kezdjük: [1,...,n], és vesszük az első szóval kezdődő egyre rövidebb szegmenseket: [1,...,j], ahol $j=n$-1,...,1.
2. Vesszük a második szóval kezdődő egyre rövidebb szegmenseket: [2,...,j], ahol $j=n$,...,2.
3. Általánosan, az összes szegmenst megvizsgáljuk a kezdőszó mondatbeli pozíciója szerint növekvő, majd azon belül a szegmens hossza szerint csökkenő sorrendben: [i,...,j], ahol i=3,...,n, $j=n$,...,i.

[64] *New York Times sport rovata* tartalmazza a New York, York, Times, és New York Times-t.

1. megjegyzés: Nyilván nem mindegyik mondatszegmens lehet valóban névelem. Ha figyelembe vesszük, hogy a mondat első szavának a megszorítások miatt feltétlenül kérdőszónak kell lennie, akkor kezdhetünk a 2. lépéssel ([2,...,*n*] szegmenstől), a vizsgálandó részletek számát $n(n-1)/2$-re csökkentve.

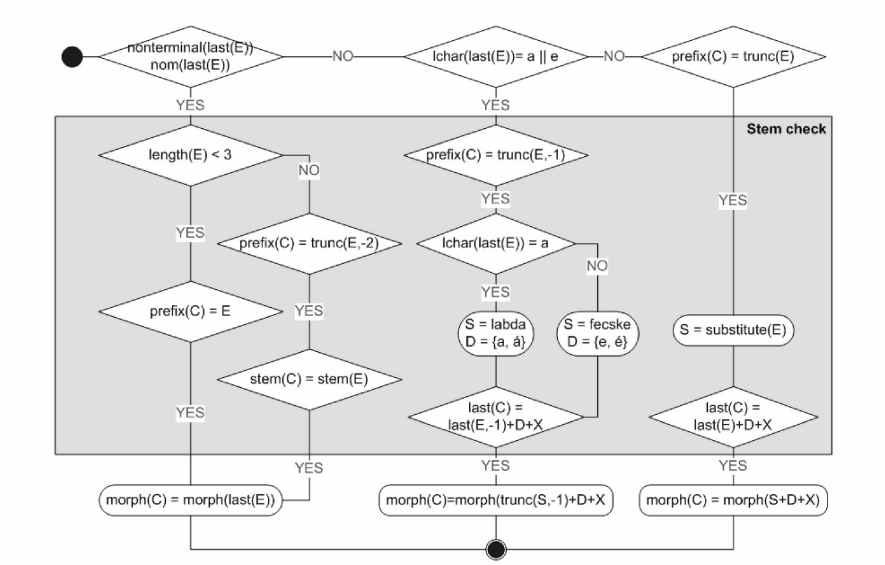
A továbbiakban a névelem felismerést egy konkrét mondatszegmens (ezentúl *jelölt*) kapcsolatban ismertetjük. A magyar nyelvben a szavak töve változhat toldalékolásnál. Az esetek nagy részében a szótőnek csak az utolsó két betűje változhat (*tűz* □ *tüzet*; *álom* □ *álmot*). Hasonlóan, egy toldalék megváltozhat egy következő toldaléktól (ez csak akkor fordul elő, ha a névelem magában is toldalékolt, és azt a mondatban tovább toldalékoljuk, ld. 3. lábjegyzet). Ebben az esetben csak az utolsó betű változhat. Mindezeket a névelem felismerés keresés fázisában figyelembe kell vennünk.

A névelemek jelentős része nem magyar nyelvű, így a morfológiai elemző nem képes azokat elemezni. Ennek ellenére a névelem felismerő ezen névelemeket is el kell lássa morfológiai jegyekkel. Erre a feladatra ún. *helyettesítő szavakat* használunk. A helyettesítő szónak a névelemek toldalékainak meghatározásánál van szerepe. Feltételezzük, hogy minden névelemhez rendelkezünk egy helyettesítő szóval, mely morfológiailag elemezhető és pontosan ugyanúgy ragozódik (kiejtés szerint azonos hangrendű, főnév), mint a névelem utolsó szava. Ez gyakran a névelem utolsó szava (ha az egy alanyesetű magyar főnév), vagy algoritmikusan előállítható mikor a névelem bekerül a névelemtárba. A helyettesítő szónak mindig főnévnek kell lennie, mivel az ismert névelemek előfordulásai egyedi entitásokat jelölnek, tehát a mondatban főnévi szerepben állnak és eszerint ragozódnak. Kivételt képeznek a 2. szakaszban ismertetett egyéb névelemtípusokat egyes esetei, de ezek a morfológiai elemző által ismert magyar szavak, ahol tehát a morfológiai jegyek megállapítására nincs szükség helyettesítő szóra.

Az alábbi jelöléseket használjuk:

– last(x) jelöli az x kifejezés utolsó szavát.
– length(x) jelöli az x szó betűinek számát.
– trunc(x,i) jelöli az x szót az utolsó i betűje nélkül.
– lchar(x) jelöli az x szó utolsó betűjét.

Továbbá jelölje *C* a jelöltet, *S* a helyettesítő szót és *E* a névelemet. Az algoritmus folyamatábráját az 1. ábra szemlélteti:



1. ábra Az algoritmus folyamatábrája

1. Ha last(*E*) toldalékolható, alanyesetű, magyar szó (azaz a morfológiai elemző felismeri)
   1.1 keresés
      1.1.a ha length(last(*E*)) ≥ 3, ellenőrizzük, hogy *C* trunc(*E*,2)-vel kezdődik-e.
      1.1.b ha length(last(*E*)) < 3, ellenőrizzük, hogy *C E*-vel kezdődik-e.
   1.2 szótő ellenőrzés: Ha 1.1.a igaz, azaz *C* trunc(*E*,2)-vel kezdődik, akkor meg kell határozni, hogy last(*C*) és last(*E*) szótöve megegyezik-e. Erre azért van szükség, mert a betűelhagyás miatt a csonkolt szó több értelmes szónak is a prefixe lehet. Ez a lépés kihagyható, ha 1.1.b igaz.
   1.3 annotáció: Ha 1.2-ben a szótövek megegyeznek, akkor *C* az *E* névelem, melynek morfológiai jegyei a last(*C*) jegyei. Ha *E* és *C* egyaránt rendelkezik nem záró morfémával, azt kihagyjuk az annotációból (lásd 4. példa).
2. Ha last(*E*) nem felel meg az 1. feltételeinek, azaz a morfológiai elemző nem ismeri fel, vagy nem toldalékolható, vagy nem alanyesetű.
   2.1 keresés
      2.1.a Ha lchar(last(*E*)) = *a* vagy = *e*, ellenőrizzük, hogy *C* trunc(*E*,1)-vel kezdődik-e.
      2.1.b Ha lchar(last(*E*)) ≠ *a* és ≠ *e*, ellenőrizzük, hogy *C E*-vel kezdődik-e.
   2.2 helyettesítő szó megállapítása
      2.2.a Ha 2.1.a igaz és lchar(last(*E*)) = *a*, akkor S = *labda*, ha lchar(last(*E*)) = *e*, akkor S = *fecske*.
      2.2.b Ha 2.1.b igaz, akkor vesszük a névelemtárban *E*-hez megadott *S*-t.
   2.3 annotáció

2.3.a *C* utolsó szavának alakja a következő: [trunc(last(*E*),1){*a*,*e*}*marad*], ahol *marad* a (*C*) végén lévő maradék betűkből áll (ha vannak). A következő szövegeket elemeztetjük a morfológiai elemzővel: [trunc(last(*S*),1){*á*}marad], ill. [trunc(last(*S*),1){*é*}marad] ha lchar(*E*) = *a*, ill. lchar(*E*) = *e*, azaz a szóvégi magánhangzót hosszúra cseréljük . Csak az egyik szöveg lesz helyes szó, és ismeri fel a morfológiai elemző. A *C* morfológiai jegyei a helyes szó jegyei lesznek.

2.3.b *C* utolsó szavának alakja a következő: [last(*E*) *marad*]. A következő szöveget elemeztetjük a morfológiai elemzővel: [*S marad*]. A *C* morfológiai jegyei az [*S marad*] szó jegyei lesznek.

1. megjegyzés: Látható, hogy az első esetben a keresés bonyolultabb, mert a toldalékolható szavak esetén a helyes szótő azonosítása nehezebb. A második esetben viszont az annotálás a bonyolultabb, mert a toldalékok meghatározása csak egy megfelelő helyettesítő szóval lehetséges.

2. megjegyzés: A névelemek keresett alakja a névelemtár feltöltésekor számítható és tárolható, így jelentős időt nyerünk a keresésnél.

3. megjegyzés: A 2.3-nál ha length(*marad*) = 0, akkor kihagyható a morfológiai elemző használata, mert ez azt jelenti, hogy a névelemen nincsenek toldalékok és az egy alanyesetű főnévnek tekinthető.

4. megjegyzés: A 2.2.b-ben használt, a névelemhez rendelt helyettesítő szó meghatározásánál egy fél-heurisztikus algoritmust használunk. A helyettesítő szavakat már a névelemtár feltöltésekor offline, a névelem utolsó mássalhangzója és az utolsó szavának magánhangzói alapján határozzuk meg. Míg ez (kiejtett) magánhangzóra végződő szavak esetén triviális, mássalhangzóra végződő szavak esetén több körültekintést igényel. Ez az eljárás pl. az idegen szavak kiejtés követő toldalékolása miatt nem 100%-osan tökéletes, de az esetek túlnyomó többségében (több mint 98%-ban) jó helyettesítő szavakat eredményez.

4 Példák

A továbbiakban néhány példán keresztül bemutatjuk az algoritmus működését.

1. példa: Lásd 2. ábra

Milyen költők vannak Arany Jánostól József Attiláig?

E = *József Attila*, last(*E*)-t felismeri a morfológiai elemző mint

Attila[noun_prs]+[NOM]

így ez az 1-es eset. A keresés *József Atti* kifejezéssel végezzük,, ami alapján a *C* = *József Attiláig* szegmenst találjuk (mivel ezekben a példákban a *C* választása triviális, a következőkben külön nem térünk ki rá). A last(*C*) morfológiai elemzése

Attila[noun_prs]+[TERM]

Így az *E* névelemet felismertük *C*-ben és a morfológiai jegyei [TERM].

2. példa: Lásd 2. ábra

Ki rendezte az Anyádat ist?

E = *Anyádat is*, ez a 2 (b) eset, mert az *is* kötőszó, mely nem toldalékolható. Legyen S a *kés*, így a morfológiai elemzővel a *kést* szöveget elemeztetjük. Az eredmény

kés[noun]+[ACC]

így a felismert névelem: *Anyádat is*$_{\text{névelem}}$+[ACC].

3. példa: Lásd 2. ábra

Mennyit kell fizetnem az Interjú a vámpírralért?

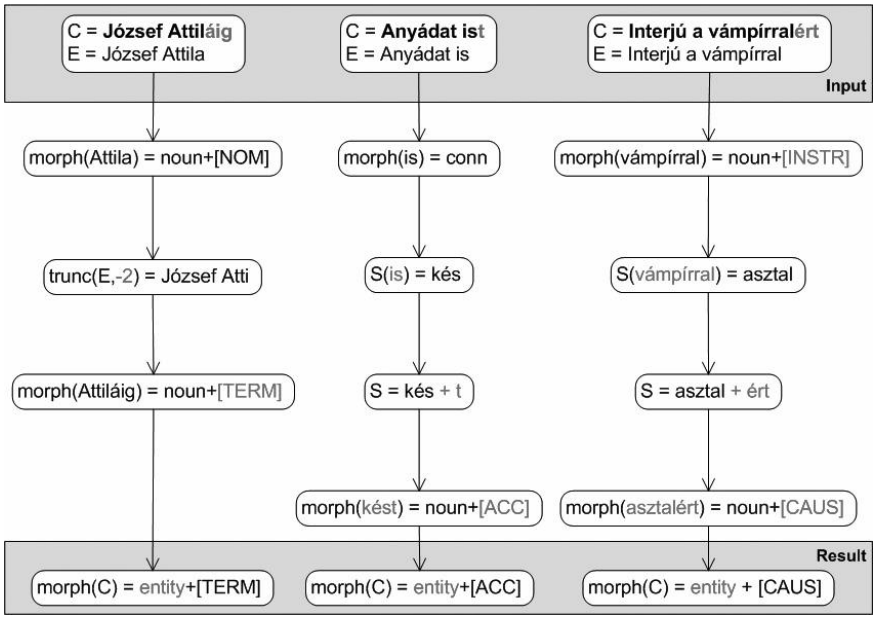
E = *Interjú a vámpírral*, ez is a 2 eset, mert last(*E*) már toldalékolt:

vámpír[noun]+[INSTR]

Legyen S az *asztal*, így a morfológiai elemzővel az *asztalért* szöveget elemeztetjük. Az eredmény

asztal[noun]+[CAUS/FIN]

így a felismert névelem: *Interjú a vámpírral*$_{\text{névelem}}$+[CAUS/FIN].



2. ábra Illusztráció az 1-3. példákhoz

4. példa: Lásd 3. ábra

Ki rendezte Az én kis mosodámat?

E = Az én kis mosodám. A névelem utolsó szava birtokos toldalékú, amit a névelem egészére mint entitásra vonatkozóan tárgyrag követ. Ebből következően a névelemet csak a tárgyraggal kell felcímkézni. Az utolsó szó morfológiai elemzése a névelem az algoritmus mindkét fő ágát aktiválja, hiszen

mosoda[noun]+[POSS_SG_1]+[ACC]
mosoda[noun]+[POSS_SG_1]+[NOM]

Az első sor a 2-es esetet aktiválja. Legyen S a *karám*, így a morfológiai elemzővel a *karámat* szöveget elemeztetjük. Mivel ezt a szót a morfológiai elemző nem ismeri fel, ez az ág nem talál névelemet.

A második sor az 1-es esetet aktiválja. A last(*E*) = *mosodám* és last(*C*) = *mosodámat* szótöve egyezik, és *C E*-vel kezdődik. Végül a morfológiai jegyeket a last(*C*) és last(*E*) morfológiai jegyeinek különbözetéből kapjuk: *Az én kis mosodám*$_{\text{névelem}}$+[ACC].

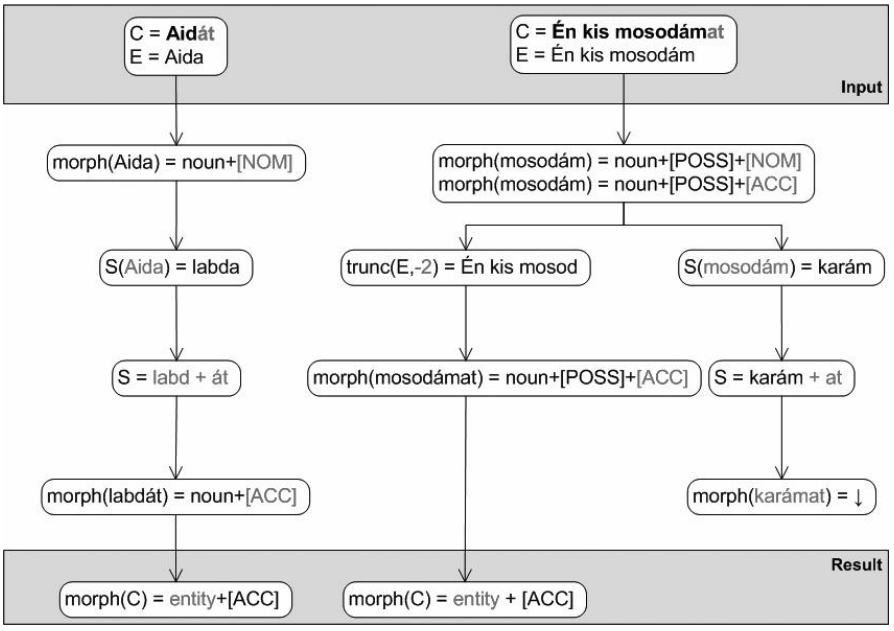
5. példa: Lásd 3. ábra

Hol játsszák az Aidát?

E = *Aida*. Ez a 2 (a) eset, mert last(*E*)-t nem ismeri fel a morfológiai elemző. Legyen S a *labda*, így a morfológiai elemzővel a *labdát* szöveget elemeztetjük, mely eredménye

labda[noun]+[ACC]

Így a névelem felismerés eredménye: $Aida_{névelem}$+[ACC].



3. ábra Illusztráció a 4-5. példákhoz

5 Összefoglalás

A fentiekben ismertettük annak a feladatnak a jelentőségét és nehézségeit, mely egy természetes magyar nyelvű kérdőmondatban a szótári névelemek összes előfordulásának megkeresése és morfológiai jegyekkel való ellátása.

Ismertettünk egy algoritmust, mely megoldás erre a feladatra, és hatékonyan végrehajtható.

6 Köszönetnyilvánítás

A cikk a Nemzeti Kutatási és Fejlesztési Pályázatok NKFP-0019/2002 jelű projektjének támogatásával készült.

Irodalomjegyzék

1. Bergman, M.K.: The deep web: surfacing hidden value. Journal of Electronic Publishing 7 (2001) http://www.press.umich.edu/jep/07-01/bergman.html.
2. Hunmorph: (2004) http://mokk.bme.hu/resources/hunmorph/
3. Katz, B., Yuret, D., Lin, J., Felshin, S., Schulman, R., Ilik, A.: Blitz: A preprocessor for detecting context-independent linguistic structures. In: Proc. of the 5th Pacific Rim Conference on Artificial Intelligence (PRICAI '98), Singapore (1998)
4. Tikk, D., Kardkovács, Zs.T., Andriska, Z., Magyar, G., Babarczy, A., Szakadát, I.: Natural language question processing for hungarian deep web searcher. In: Proc. of IEEE Int. Conf. on Computational Cybernetics (ICCC04), Wien, Austria (2004) 303–309.
5. Tikk, D. Kardkovács, Zs.T., Magyar, G.: Deep web searcher for Hungarian. International Journal of Information Technology 1(4) (2004) 191--197.
6. Winkler, H.: Suchmaschinen. metamedien im internet? In Becker, B., Paetau, M., eds.: Virtualisierung des Sozialen, Frankfurt/NY (1997) 185–202 (In German; English translation: http://www.uni-paderborn.de/~timwinkler/suchm_e.html).