Helyesírás.hu – Nyelvtechnológiai megoldások automatikus helyesírási tanácsadó rendszerben

Miháltz Márton[1], Hussami Péter[2], Ludányi Zsófia[1], Mittelholcz Iván[1], Nagy Ágoston[3], Oravecz Csaba[1], Pintér Tibor[1], Takács Dávid[1]

[1] MTA Nyelvtudományi Intézet, Nyelvtechnológiai Osztály, 1068 Budapest, Benczúr u. 33.
{mihaltz.marton, ludanyi.zsofia, mittelholcz.ivan, oravecz.csaba, pinter.tibor}
@nytud.mta.hu,takdavid@gmail.com

[2] Rényi Alfréd Matematikai Kutatóintézet, 1053 Budapest, Reáltanoda utca 13–15.
hussami@renyi.hu

[3] Szegedi Tudományegyetem, Bölcsészettudományi Kar, Francia Nyelvi és Irodalmi Tanszék, 6722 Szeged, Egyetem u. 2.
nagyagoston@lit.u-szeged.hu

Kivonat: A helyesiras.hu projekt célkitűzése egy nyelvtechnológiai eszközökkel támogatott magyar helyesírási tanácsadó portál kifejlesztése, mely 7 különböző területen próbál interaktív segítséget nyújtani: külön- és egybeírás, helyesírás-ajánló, elválasztás, tulajdonnevek írása, számnevek helyesírása, keltezés, betűrendbe sorolás. A cikkben szeretnénk bemutatni a webalkalmazások kifejlesztésében alkalmazott nyelvtechnológiai erőforrásokat és algoritmusokat, valamint a leginkább problémás kérdést jelentő témakör, a különírás-egybeírás fejlesztése során felmerült kihívásokat. A magyar helyesírás rendszere nagymértékben épít a nyelvhasználók értelmezési képességeire, így ahol lehet, a rendszer többféle választási lehetőséggel, visszakérdezésekkel próbálja a hiányzó egyértelműsítő információkat beszerezni.

1 Bevezetés

A magyar helyesírás számítógépes, illetve formális igényű feldolgozása nem új keletű a magyar nyelvészetben [1], [3], [10]. A helyesírási szabályok komplexitása, „túlszabályozottsága" és inkonzisztenciája miatt azonban számítógéppel feldolgozható teljes formális leírást nem sikerült kidolgozni. Az MTA Nyelvtudományi Intézete 2009 óta dolgozik egy olyan szakértői rendszeren, amelyben nyelvtechnológiai eljárások alkalmazásával lehetővé válik a helyesírási kérdésekre automatikus választ adni – esetenként a kérdezők aktív bevonásával (a kezdeti lépésekről, tervekről, illetve a megvalósítás nehézségeiről bővebben lásd [6].) A *helyesírás.hu* portál célja a tudatos és kevésbé tudatos helyesírók hasznos segédeszközévé válni: egyrészt a válaszok gyorsaságával, másrészt megbízhatóságával. A magyar helyesírás túlszabályozottságát és kiterjedt ismeretigényét tekintve nyilvánvaló, hogy a rendszer nem tud minden szabálypontot tökéletesen kezelni. Az azonban elvárható, hogy tisztában legyen a „gyenge pontja-

ival", azaz csak azokra a kérdésekre válaszoljon, amelyekre valós találatot és szabálypontot talál – ellenkező esetben a rendszer visszakérdez, bevonva a kérdezőt a helyes alak kiválasztásába (intuitív döntések meghozatala), illetve végső esetben felkínálja a humán szakértői segítség lehetőségét.

A portál gerincét alkotó dinamikus alkalmazások mellett hangsúlyt kapnak a statikus tartalmak is, melyek segítik a helyesírásban tájékozódni vágyókat. A portálra látogatók megtalálják az Akadémiai Kiadó Magyar helyesírás szabályainak (AkH.) [7] mindenkori legújabb kiadását, valamint többféle megközelítésben böngészhetik az MTA Nyelvtudományi Intézetének nyelvi tanácsadó szolgálatán gyűjtött kérdéseket és az azokra adott válaszokat (kategóriacímkék és szakmai besorolás címkék alapján történő böngészés, valamint különböző kategóriák szerint kezelhető szabadszavas keresés formájában).

A dolgozat részletesen beszámol az AkH. (azaz az akadémiai helyesírás) alkalmas szabálypontjainak formalizálásáról, a rendszer egyes moduljainak működéséről, a modulokat működtető szabályrendszerek felépítéséről, valamint kitér a rendszer mögött álló speciális lexikonok kapcsolódásának és integrálásának módjára.

2 A rendszer általános felépítése

A felhasználói kérdések hatékony és automatikus kezelhetőségének érdekében két alapvető és egymással ellentétes cél között kell ergonomikus kompromisszumot találni: (1) kényelmes, a felhasználótól a lehető legegyszerűbb bemenetet igénylő felület; (2) a várható inputról minden lehetséges forrásból a lehető legtöbb információt begyűjtő rendszer. A minden megszorítás nélküli, gyakorlatilag tetszőleges szöveg bevitelét megengedő bemenet automatikus feldolgozása lehetetlen vállalkozás, ezért első lépésben feltétlen szükséges minimálisan a kérdéskörök, problémakategóriák egyértelmű meghatározása, és ennek alapján a választ előállító feldolgozási lépések kiválasztása.

Az aktuális helyesírási problémakört (kategóriát) a felhasználó választja ki a felületen megjelenő kötelező típusválasztó segítségével. A kiválasztás függvényében rendelődik a bemenethez egy meghatározott feldolgozó modul, mely a választ előállítja. Ezek a modulok alapvetően két csoportba sorolhatók:

1. A kategóriaválasztás alapján olyan meghatározott problémakört kezelő modulok, melyek csak specifikus, megszorított bemenetet fogadnak és erre egyértelmű választ adnak. Ilyenek a számnevek helyesírása, keltezés, betűrendbe sorolás (3., 4. és 5. rész), tulajdonnevek írása (6. rész) és az elválasztás (7. rész) kategóriákat kezelő modulok.

2. Gyakorlatilag szabadszöveges (bár a lehetőségekhez mérten szűrt) bemenetet kezelő feldolgozó modulok: a helyesírás-ajánló (8. rész) és a külön- és egybeírást kezelő modul (9. rész). Ez utóbbi a legkomplexebb, egyben a legaktívabb felhasználói interakciót is igénylő komponens, ahol az elérhető információ maximalizálása alapvető fontosságú.

A helyesírási kérdések jelentős részének megválaszolása egy bemeneti karaktersorozatban rendelkezésre álló információ mellett további, a közvetlen felszíni jellemző-

kön messze túlmutató ismereteket is igényel. Ezeket vagy erőforrásként a rendszerhez kell csatolni, vagy adott esetben a felhasználótól megszerezni. A rendszer erőforrásai egyrészt a **Humor** morfológiai elemző ([5], [8]), másrészt olyan lexikális adatbázisok, melyek a helyesírási szempontból speciálisan viselkedő lexikális elemek lehetőleg kimerítő felsorolását tartalmazzák (színnevek, anyagnevek, mindig egybeírandó előtagok, egybeírt alanyos/tárgyas összetételek stb.). Ezeket az erőforrásokat jórészt a 2. típusú feldolgozó modulok használják, illetve a morfológiai elemzőkre támaszkodik még az elválasztást kezelő komponens is.

Az alábbi fejezetekben részletesen bemutatjuk az egyes témaköröket megvalósító webalkalmazásokat működtető modulok célját és felépítését.

3 Számnevek helyesírása

Felhasználói bemeneteként előjel, számjegyek, tizedesvessző és törtvonal valamilyen értelmes kombinációja szolgál. A modul kezeli a tőszámneveket, a sorszámneveket, tizedes- és egyszerű törteket és a felhasználónak visszaadja a megadott szám betűkkel való lehetséges átírásait, kiegészítve ezt az AkH. vonatkozó pontjaira mutató hivatkozásokkal és az esetleges további megjegyzésekkel.

Ahol több helyes átirat lehetséges, ott a rendszer magyarázatokkal igyekszik a felhasználók segítségére lenni. Az egyik ilyen eset a törtek egybe-, illetve különírása. A *2/3* például jelzői értelemben egybe írandó (*kétharmad csésze liszt*), minden egyéb használatban viszont külön (*két harmad nagyobb, mint egy harmad*).

Ehhez hasonló eset a *kettő* és a *két* megfelelő használata. Jelzői értelemben a *két* használata az elfogadott (*két pár zokni*), míg a *kettő* csak nem-jelzői értelemben felel meg a standard nyelv- és íráshasználatnak (*egy meg egy az kettő*). Ennek megfelelően a *kettő*-nek egy számnév elején vagy a belsejében történő használata egyértelműen szubsztenderd. Így a *harminckettő* elfogadott alaknak számít, szemben a *kettőezer*-rel, vagy a *háromezer-kettőszáz*-zal. A modul megadja az összes lehetséges, helyesen írt alakot (*kétezer* és *kettőezer*), ugyanakkor megjegyzésben hívja fel a felhasználó figyelmét a standard nyelvváltozattól való eltérésre.

4 Keltezés

A keltezés modul (*Dátumok*) *éééé-hh-nn* formátumú bemenetet dolgoz fel. A beolvasás során a formátumnak való megfelelésen túl ellenőrzi a dátum hozzávetőleges helyességét is. Így például kiszűri a március 50-ét éppúgy, mint az április 31-ét. Az ellenőrzés azonban nem teljes körű, nem terjed ki a szökőévek kezelésére, de az olyan történeti eseményeket sem veszi számításba, mint amilyenek a Gergely-naptár bevezetésekor kimaradó napok. Az ellenőrzés ilyen korlátozottsága azonban nem ok nélküli, hiszen ezeket a napokat akkor is le kell tudnunk helyesen írni, ha éppen azt akarjuk kifejezni, hogy nem is léteztek (pl. „*1582. október 10*. egy nem létező dátumot/napot jelöl.”).

A modul két módon segíti a felhasználót. (1) Felsorolja a dátumnak az AkH. által elfogadott írásmódokat (pl. *1582. október 10., 1582. okt. 10., 1582. X. 10.* stb.). (2) Példákat ad a dátum különböző (pl. toldalékolt) használataira: *1582. október 10-én, 1582. október 10-e óta, 1582 októberében* stb.

A megvalósítás során hónapok és számok (napok) listáját használja. Hangrendjük szerint csoportokra bontva ezekből generálja az általa nyújtotta ajánlásokat. Mint a többi modul esetében, a kimenet mellett az AkH. vonatkozó bekezdéseit is visszaadja.

5 Betűrendbe sorolás

A modul célja a felhasználó által megadott latin betűs (de nem feltétlenül csak a magyarban használt betűket tartalmazó) tételeknek a szabályzat szerinti betűrendbe sorolása (az AkH. 14–15. pontjai szerint. A 16. pont által említett kivételes betűrendbe sorolási eseteket, továbbá a számokat is tartalmazó tételeket, és más ábécék szerinti rendezést itt nem kezeljük).

Az általánosan használt szoftverekben többnyire a klasszikus „lexikális” rendezési algoritmus használatos: a két sztring összehasonlítása karakterenként (írásjelekkel stb. együtt), balról kezdve, és az első különböző karakterpár összehasonlítása adja a két sztring összehasonlításának eredményét. Ha az egyik sztring teljes egészében a másik elejét alkotja, akkor ez utóbbit tekintjük a másodiknak.

A szabályzat szerinti rendezés implementálásához több előfeldolgozási lépés is szükséges.

1) A szövegből ki kell válogatni az összehasonlítható (azaz szóalkotó) karaktereket, és a továbbiakban csak ezzel kell dolgozni (AkH. 14. e.).

2) A magyarban nem használt betűket normalizálni kell, hogy azokkal összehasonlíthatók legyenek. Ezeket a magyarban használt egyjegyű betűkre kell leképezni, mert a megfeleltetettjeikkel azonos súllyal veendők figyelembe a rendezéskor (AkH. 15.).

3) A hosszú magánhangzókat a rövid megfelelőjükkel kell azonos értékűnek tekinteni, ugyanakkor máshogy kell kezelni a szóalakokat, ha alakjuk ékezettelenítve megegyezik (AkH. 14. d.). Ugyanez vonatkozik az idegen mellékjeles betűkre is (AkH. 15.).

4) Az összetett (több karakterből álló, mássalhangzókat jelölő) betűket azonosítani kell, mert ezek egy egységként kezelendők, a hosszúakat pedig két rövid megfelelőjükkel kell helyettesíteni (AkH. 14. c.). Ennek a feladatnak a korrekt megoldása szótárat, illetve morfológiai elemzőt igényel, hiszen sok esetben a többjegyű betűnek is tekinthető karaktersorban morfémahatár van.

Az így előállított betűsorokra már alkalmazható a klasszikus rendezési algoritmus.

6 Tulajdonnevek helyesírása

A tulajdonnevek helyesírásának ellenőrzését a Névkereső modul támogatja. A tulajdonnevek közötti böngészést azok nagy száma miatt (jelenleg több mint kétszázezer) nem tesszük lehetővé, de a nevek kereshetők, és a szűkített találati listákat már megje-

lenítjük. Amint a felhasználó elkezdi begépelni a tulajdonnév első néhány karakterét, a modul prediktíven megjeleníti az illeszkedő tulajdonnevek listáját. Az egyes tulajdonnevek mellett azok besorolása is megjelenik, úgymint pl. földrajzi név, személynév stb.

A személynevek és a földrajzi nevek esetében a névalakok mellett további jegyeket is megjelenítünk (vezetéknév/keresztnév, férfi/női vagy becenév, illetve településnév; közterület neve, magyarországi vagy nem).

Az adatbázis legfontosabb forrásai: a *hunmorph*, a *huntag* és a *Hunspell*[1] szabadon letölthető erőforrásai, a Magyar Posta nyilvános listái, a publikusan hozzáférhető telefonkönyvi adatok, továbbá a FÖMI és az egyes minisztériumok által közölt névtárak és listák. A földrajzi nevek tára elsősorban a magyar vonatkozású neveket tartalmazza. Ez az adathalmaz így is több mint százezer nevet tartalmaz. A vezetéknevek száma szintén több mint százezer elemű, amhez több mint hatezer kereszt- és becenév járul. Ez az adathalmaz is elsősorban a magyar névtárakban és korpuszokban előforduló neveket foglalja magába.

Terveink között szerepel az oktatási, kulturális, civil és egyházi szervezetek neveinek, továbbá városrészek, településrészek neveinek felvétele az adatbázisba úgy, hogy egyes elemeik szerint is és egészükben is kereshetők és listázhatóak legyenek.

Speciális esetet képeznek a cégnevek. A hivatalos cégjegyzékben szereplő cégnevek közt nagy számban szerepelnek hibás alakok is – a nyilvánvaló elgépeléstől kezdve az egyes szavak jellegzetes helyesírási hibáin át az értelemzavaró, tévesztéses hibákig. Hivatalos kontextusban a cégneveket a bejegyzett alakjukban kell használni, akkor is, ha egyébként hibásnak minősülnek, tehát a javított alakok használata nem javasolható egyértelműen. Ezt a diszkrepanciát elkerülendő, jelenleg a cégneveket nem tartalmazza a rendszer mögött álló adatbázis.

7 Elválasztás

Az elválasztást leginkább tipográfiai okok miatt alkalmazzák. Célja, hogy egy szöveg minden sorában a sorok (kézírás esetén) és a szóközök egyenletes nagyságúak, utóbbiak minél rövidebbek legyenek.

Mivel számos szövegszerkesztő alkalmazás használ elválasztómodult, kézenfekvő volt a *helyesírás.hu* projektben is ilyenre támaszkodni, mégpedig az *OpenOffice/LibreOffice* irodai programhoz létrehozott **huhyphn-**re.

A huhyphn egy nyelvspecifikus karaktersorozatokat tartalmazó fájlt használ, melyben a tiltott és a lehetséges elválasztási helyek számokkal vannak jelölve: az előbbiek párossal, az utóbbiak páratlannal. A hosszabb karaktersorozat és a nagyobb szám felülírja a kisebbet.

A magyarban például az *en1d2ő* szabály lehetővé teszi az elválasztást az *n* és a *d* között és tiltja a *d* és az *ő* között, így lesz *ken-dő* vagy *kerülen-dő*, de sosem *kend-ő*. Ezt a szabályt felülírja a *ren2d3őr*, ami által a *rendőr* szó elválasztása *rend-őr* lesz (lévén összetett szó), miközben a *ken-dő* továbbra sem változik [4].

1 http://mokk.bme.hu/en/eszkozok/

A huhyphn-t sok esetben módosítani kellett, mivel főleg tipográfiai célokra készült, így nem engedélyez például olyan elválasztásokat, mint *a-pa-i*, mivel egy karakter leválasztása nyomdai szövegben nem esztétikus. Ezeket a tiltásokat a már meglévő szabályokban írtuk át: az adott helyen a páros számot kicseréltük egy páratlannal vagy új szabályt vettünk fel.

A huhyphn egyik tipikus tulajdonsága, hogy a többféleképpen elválasztható, többjelentésű szavakat nem, vagy csak az egyik lehetséges módon választja el. Ilyen szó például a *megint*, amelyet határozóként *me-gint*, igekötős igeként *meg-int* alakban lehet elválasztani. Az AkH. 233-238. szerint a szóösszetételi határok mentén kell elválasztani, de mivel a huhyphn szóegyértelműsítést nem végez, illetve rontani sem szeretne, ezért az ilyen típusú szavakat nem választja el. Ugyanígy a tanárok szót is csak *ta-ná-rok* alakban választja el, holott ezt *tan-á-rok* formában is lehetséges.

Ezen esetek kiküszöbölésére alkalmaztuk a *Humor* morfológiai elemzőt [5], [8]. Ezzel az alkalmazással minden input szóról el tudjuk dönteni, hogy az összetett-e: ha igen, akkor a szóösszetételeket külön-külön választjuk el, majd azokat egyesítjük, és azok közé |- jelet teszünk, például *kis*|*-a-u-tó* vagy *tan*|*-á-rok*. Ha egy szónak több morfológiai elemzése van, akkor mindegyiket számba vesszük, így lesz az *altest* szóból *al*|*-test* vagy *alt*|*-est*. Végül a szóösszetétel-elemzés után a kimenetet egyesítjük a huhyphn standard kimenetével: például akkor, ha a morfológiai elemzést követően az elválasztás *tan*|*-á-rok*, azonban a huhyphn szerint *ta-ná-rok*, akkor a modul mindkettőt visszaadja.

Az alkalmazást jelenleg egy egymillió szavas listán (MNSz gyakorisági lista) tesztteljük: ha a program olyan szótagot talál, amelyben a magánhangzók száma nem egy, akkor azt jelzi, és ha szükséges, azt kézzel javítjuk (a *Mar-seille* esetében például nem kell).

8 Helyesírás-ajánló

A különálló szavak helyesírásának ellenőrzése modul (*Helyes-e így?*) a felhasználó által megadott szóalakok létezését vizsgálja, helytelen alakok esetében javaslatot tesz a leginkább hasonló helyes alakokra. Ebben a modulban kombináljuk a nyílt forrású **Hunspell**[2] helyesírás-ellenőrző és a MorphoLogic **Humor** morfológiai elemzőjére [5], [8] építő helyesírás-ellenőrző motorok kimenetét.

A Hunspell 1.3.2-es verziója a *MySpell* motorjára épül, de támogatja a szóösszetételeket és a gazdag morfológiájú nyelveket is, a szabadon hozzáférhető **Magyar Ispell**[3] szótár 1.6.1-es verziójával.

A MorphoLogic helyesírás-ellenőrzője azokat a szóalakokat fogadja el helyesként, amelyeket a benne működő Humor morfológiai elemző képes a tárolt morfémákból a nyelvi szabályoknak eleget téve összerakni (kb. 100 000 alapszó több mint kétmilliárd szóalakja).

Az ismeretlen szavak nagy részéhez mindkét motor képes javítási javaslatok listáját visszaadni, ilyenkor ezek unióját közvetítjük a felhasználó számára. Belső tesztjeink

2 http://hunspell.sourceforge.net/

3 http://magyarispell.sourceforge.net/

alapján ismeretlen szóalakok esetén a konzervatívabb, de pontosabb Humor elemző eredményét vesszük alapul, így a szóalakot ismeretlennek tüntetjük fel akkor is, ha a Hunspell szerint ismert volt, de a Humor szerint nem.

9 Különírás-egybeírás

A Különírás-egybeírás modul (*Külön vagy egybe?*) ellenőrzi a megadott – akár helytelenül írt – (összetett) szót, vagy szavakat, illetve visszaadja a szabályok, illetve a rendelkezésre álló eszközök által biztosan megállapíthatóan helyesen (külön-, egybe-, illetve kis- vagy nagykötőjellel) írt alakokat, kiegészítve magyarázatokkal és hivatkozásokkal az AkH. megfelelő pontjaira.

A modul jelenleg nem fedi le az AkH. összes, különírással-egybeírással kapcsolatos rendelkezését. A szabályzat nyelvtechnológiai eszközökkel kezelhető területei közül jelenleg az alábbiakat valósítottuk meg:

- jelölt és jelöletlen alárendelői összetételek/szintagmák,
- a szótagszámlálási (6:3-as) szabály,
- mozgószabályok,
- rövidítéseket és mozaikszókat tartalmazó összetételek,
- néhány speciálisabb szabály, például a színnévi összetételek, anyagnévi összetételek stb.

A modul csak részben képes kezelni a jelentéssűrítő, illetve a szervetlen szóösszetételeket azok algoritmizálhatatlansága miatt. A mellérendelő (ikerszók, álikerszók stb.), valamint a morfológiai típusú összetételekkel foglalkozó szabálypontok – hasonló okokból – sincsenek beépítve.

A különírás-egybeírás helyesírási szabályrendszere felfogásunkban modellezhető egy generatív nyelvtani rendszerrel, hiszen a szabályzat rendelkezik arról, hogy milyen szóelemek milyen feltételek mellett, milyen írásmóddal (egybe-, külön, kötőjellel írás) vonhatók össze összetett szavakká, illetve szószerkezetekké (frázisokká). A szabályok egy része alkalmazható rekurzívan, illetve bizonyos szabályok láncokban egymásra is épülhetnek, így kézenfekvő egy formális nyelvtanra leképezni őket. Minden lehetséges nyelvtani levezetés (elemzési fa) megfeleltethető egy-egy értelmezésnek, illetve helyes írásmódnak. Attribútumok és értékadások segítségével az elemzési fák kiszámíthatják a bemeneti szóelemek közötti elválasztó karaktereket is (üres sztring – egybeírás esetén –, szóköz, kötőjel vagy nagykötőjel), melyekkel megadható, hogyan kell az adott értelmezés szerint helyesen leírni az összetett szót vagy kifejezést. Ha a fák felépítésekor feljegyezzük azt is, hogy egy adott összevonás (új csomópont létrehozása) melyik újraíró szabály alkalmazásával jött létre, a kész fa bottom-up bejárásával, a szabályokhoz rendelt magyarázó szövegek felhasználásával részletes segítséget tudunk generálni a felhasználó számára arról is, hogy milyen helyesírási szabályok alkalmazásával jött ki az adott megoldás és milyen értelmezési feltételek kapcsolódnak hozzá.

Célunk a fentieknek megfelelő formális nyelvtan kidolgozása, illetve egy, az ennek megfelelő kifejezéseket elfogadó, belső szerkezetüket feltáró nyelvtani elemző (parser) kifejlesztése volt. A környezetfüggetlen kifejezésnyelvtan újraíró szabályai a

morfológiai elemző által megadott szófaji, alaktani, szótagszámot, szóelemek számát tartalmazó, illetve az adatbázisainkból rendelkezésre álló szemantikai tulajdonságokból generált attribútum-érték szerkezetekkel operálnak. Utóbbiak biztosítják, hogy a szavak különböző fogalmi csoportjaira (anyagnevek, színnevek stb.) építő helyesírási szabályokat alkalmazni tudjuk.

Az AkH. a szabályokhoz illeszkedő analogikus alakok mellett nagymértékben tartalmaz kivételeket is. Igyekeztünk az összes, általunk implementált helyesírási részterületben megtalálható kivételes esetet azonosítani és felvenni őket a lexikai adatbázisba. A kivételek ellenőrzéséről és kezeléséről külön mechanizmusok gondoskodnak (l. később).

A következő részekben részletesen ismertetjük azokat a lépéseket, amelyek a felhasználói bemenetből a nyelvtani elemző számára feldolgozható elemeket állítanak elő, meghívják a kifejezésnyelvtanra épülő elemzőt, végül az elemzési fákból generálják a helyes írásmódokat és a hozzájuk tartozó magyarázó szövegeket. A modul futtatásának fő lépései vázlatosan a következők:

1. Előfeldolgozás:
   - A felhasználói bemenet egyszerű ellenőrzése,
   - A bemenet szegmentálása elemi tokenekre,
   - A tokenlista különböző írásmódjainak ellenőrzése kivételszótárainkban,
   - A tokenek felcímkézése morfológiai és szemantikai tulajdonságaikkal.
2. Elemzés: elemzési fák előállítása a kifejezésnyelvtan és az elemző (parser) segítségével
3. Kimenet: az elemzési fákból lehetséges helyes írásmódok, természetes nyelvű magyarázó szövegek generálása a felhasználó számára.

9.1 Előfeldolgozás

A modul számára érkező felhasználói bemenetet néhány egyszerű ellenőrzés után (karakterszám, ismétlődő karakterek ellenőrzése stb.) megkíséreljük tokenekre felbontani.

A felhasználó által megadott (normatív szempontból felesleges) kötőjeleket és egyéb írásjeleket szóközökre cseréljük, majd az így kapott elemeket megpróbáljuk további összetételi tagokra bontani. Ezek lehetnek akár önmagukban helyes, de a többi szó kontextusában, más szabályok szerint akár más módon (külön, kötőjellel stb.) is írható szóösszetételek tagjai (pl. „almafa” vö. „birsalma-fa”), akár helytelenül egy szóba írt kifejezések elemei is (pl. „pirosalma”). Az összetételi szabályokat ellenőrző kifejezésnyelvtan szabályait ezekre az atomi szinten szétválasztott tokenekre írtuk. Ehhez a művelethez a Humor morfológiai elemző egy speciális üzemmódját használjuk, amely képes a helyesírási szabályokkal nem konform összetett alakokhoz is elemzéseket előállítani.

Amennyiben nem tudtunk minden tokent a morfológiai elemzővel azonosítani, illetve tovább bontani, jelezzük ezt a felhasználó számára, és megkérjük, hogy próbálja meg kérdését, amennyire csak lehetséges, szavakra tagolva megismételni. Ha ez a bemenet sem értelmezhető, a folyamat hibaüzenet jelzésével véget ér, illetve az oldal átirányítja a felhasználót a *Helyes-e így*? és a *Névkereső* modulokhoz.

Sikeresen felbontott és morfológiailag elemzett bemenet esetén először ellenőrizzük, hogy a tokenek valamilyen lehetséges írásmódja nem szerepel-e kivétellistáink egyikében. N darab tokenből álló input esetén összesen $k^{(n-1)}$ írásmód lehetséges, ahol k a két szomszédos token közötti lehetséges elválasztó szimbólumok száma: k = |{egybeírás, szóköz, kötőjel, nagykötőjel}|. Az összes bemeneti tokent tartalmazó, általunk ismert kivétel azonosítása esetén a folyamat – a kivételes írásmód és megfelelő magyarázat jelzésével – véget ér.

Ha a bemeneti tokenek nem képeznek kivételt, sor kerülhet szószerkezet-nyelvtani elemzésük előkészítésére. Ehhez a már azonosított szófaji, morfológiai, szótagszámmal és összetételi tagok számával kapcsolatos információkon túl az adatbázis segítségével kikeressük a tokenekhez rendelhető szemantikai kategóriákat is.

Az egyes szabályok működését támogató adatbázis részét képezik a színnevek, foglalkozások és rangok, számnevek, földrajzi jellegű jelzők és köznevek, közterületek nevei, keresztnevek, népek és nyelvek nevei, rövidítések, közszói betűszavak, elő- és utótagok, a helyesírási szabályzatban az egyes szabályokban hivatkozott további kategóriák és különösen az egyes kivételek listája, mely jelenleg több mint 2100 szóból áll. Ezek a speciális tulajdonságok egy-egy szemantikai, illetve grammatikai jegyként vannak tárolva a szóalakok metaadatai között, és a szabályok számára közvetlenül hozzáférhetők.

9.2 A nyelvtan

A modul alapját képező környezetfüggetlen, jegystruktúrás nyelvtan formális leírása független a modul programkódjától, így könnyen karbantartható, fejleszthető. A nyelvtan a – jelenleg mintegy 160db – újraíró szabály megadásán kívül az alábbi elemeket tartalmazza:

- a szabály egyedi azonosítóját,
- a szabály alkalmazásának magyarázatát a felhasználó számára,
- hivatkozást az AkH. megfelelő szabálypontjaira és/vagy az Osiris-helyesírás [2] releváns témaköreire,
- példákat a szabály alkalmazására (az automatizált teszteléshez),
- valamint azokat a szabályokat, amelyek a szabályalkalmazó algoritmus futtatásakor konkurensek lehetnek az adott szabályra nézve; ilyenkor ezeket a szabályokat letiltjuk az alkalmazását.

Az újraíró szabályok a következőképpen néznek ki: *X(a=v, ...) + ... == Y(a=v, ...)*, ahol *X* bal oldali szimbólum, *Y* jobb oldali szimbólum, *a* egy attribútum neve, *v* ennek értéke. (A szabályokban, a konvenciótól eltérően a bal oldalon szerepelnek azok a szimbólumok, amelyekből egy elemzési lépésben összevonást végzünk a jobb oldalon megadott szimbólumba.) A bal oldali szimbólumokban az attribútum-érték párok az inputra érvényes megszorításokat, a jobb oldali szimbólumokban értékadásokat jelentenek.

A leíró nyelvtan szimbólumai megállapodás szerint az angol szófaji kategóriák kezdőbetűi vagy -betűcsoportjai: N (főnév), A (melléknév), V (ige), Adv (határozószó), Num (számnév). A szabályok bal oldalán a következő attribútumok állhatnak:

- Szemantikai jegyek listája (**sem**); a külön- vagy egybeírás kérdése (a szerkezetet alkotó szavak összetételi tagjainak számán kívül) bizonyos esetekben ezen dől el, pl.: *arany + gyűrű = aranygyűrű* (egybe), *fehérarany + gyűrű = fehérarany gyűrű* (külön) stb. Ebben az esetben feltétlenül szükséges az a többlettudás az *arany* szóról, hogy anyagnévről van szó.
- A **match** attribútum értéke egy reguláris kifejezés, amely illeszkedik a morfológiai elemző által előállított címkesorozatra. Például az alanyos vagy tárgyas viszonyt kifejező birtokos jelzői alárendelések (genitivus obiectivus/subiectivus) esetében a második tag mindig egy -ás/-és képzős ige: *match= "IGE,_IK,NOM"*, ahol *_IK* az -ás/-és képzőt jelöli.
- A bemenet felszíni alakja (**wordform**), illetve annak töve (**stem**).
- Az **ncomparts** attribútum azt mondja meg, hogy pontosan hány összetételi tagból áll az adott szimbólumnak megfelelő token-(rész)sorozat, az **ncompartsx** ennek alulról korlátos megfelelője.
- Az **nsylls** attribútum az adott szó szótagjainak számát adja meg (erre a szótagszámlálási szabálynak [más néven 6:3-as szabálynak] van szüksége).
- A **join1**, **join2** attribútumok a kivételes (nem formalizálható) írásmódú összetételek kezelésére szolgálnak. Az előfeldolgozás során, ha a tokenek felszíni alakjai valamilyen kombinációban szerepeltek a kivételszótárban, megkapják értékül a kivétel kategóriáját (pl. *Jelentessurito*), így az adott kivételeket kezelő szabályok érvényesek lesznek rájuk.

A jobb oldali értékadásban csak a tokenek közé kerülő elválasztó jeleket kódoló *sep* attribútum, illetve az összetett alak tagjainak számát megadó *ncompartsx* attribútum kerül.

A fej (a jobb oldalon az utolsó szimbólum) bizonyos jegyeit automatikusan megörökli a szabály jobb oldalán álló szimbólum, ha értékük specifikálva van (pl. *sem* attribútum: ha a fej például egy színnév, akkor a szabály által generált szimbólum is egy színnév lesz.)

9.3 A parser

A parser egy hagyományos bottom-up modellt valósít meg, a terminálisok többféle lehetséges értelmezéséből eredő összes értelmes feldolgozási fáját előállítja. A terminálisok többértelműségét az algoritmus csak akkor oldja fel, ha az adott értelmezés a szabályok megfelelő alkalmazási sorrendje mellett teljes fává összeáll. Ez a többértelműség korai feloldásánál nagyobb számítási igénnyel jár ugyan, de pontosabb (lásd pl. [9]), valamint sztochasztikus tényezőktől mentesen garantálja, hogy a szabályok megfelelő fedése esetén a végeredményül kapott fák halmaza tartalmazza a helyes eredményt is.

A terminálisok többértelműségén túl az alkalmazható szabályok halmaza és sorrendje sem egyértelmű. Bottom-up megközelítésről lévén szó, a helyes sorrend legenerálását csak kipróbálás útján lehet megtalálni.

Formálisan:

1. Legyen $F_1..F_k$ k db izolált fa, azaz k-komponensű erdő, amelyek levélpontjainak halmaza pontosan egybeesik a terminálisok halmazával, úgy, hogy az egyes fák levelei összefüggő szövegrészeket fednek le, és az F_i-k indexelési sorrendje egybeesik a terminálisokéval. A lehetséges szabályalkalmazások helye ezen fák $V_1..V_k$ gyökérpontjain lesznek.

2. Legyen H a lehetséges szabályalkalmazások halmaza, ahol H egy tetszőleges m-argumentumú h elemére jellemző, hogy az $V_a..V_{a+m-1}$ csomópontokra illeszkedik, és létrehoz feléjük egy új, G_h csomópontot. Ez m-1-gyel csökkenti a komponensek számát. Az algoritmus H minden elemére lemásolja az F_i erdőt, és a másolatokon sorra alkalmazza H szabályait.

3. Ha H üres volt, az erdőn nincs szabályalkalmazás, ami azt jelenti, hogy a rendszer szabályai szerint a terminálisok a jelen behelyettesítési értékük mellett nem állnak össze.

4. Ha az erdő egyetlen fává összeállt, az jó megoldás.

5. Amennyiben a szabályalkalmazás hatására még nem állt össze fává az erdő, a jelenlegi állapottal újra elindul az 1. lépéstől.

Az algoritmus futási ideje a terminálisok lehetséges értelmezéseinek számától ($t_1..t_n$), valamint az egy csomópontsorozatra illeszkedő szabályok maximális számától (m) függ. Felső becslés a legrosszabb esetű lépésszámra:

$$L = O(m)O(n^2)O(\prod_{i=1}^{n} t_i) = O(mn^2t^n)$$

ahol t a t_i-k maximuma. Ez n-ben hiperexponenciális, de csak irreális feltételek mellett valósulhat meg. A fenti implementációval az átlagos eset n-ben még mindig exponenciális, de nagy bemenet mellett található olyan heurisztika, amely segítségével a tényleges lépésszám n-ben csak polinom lesz.

9.4 Kimenet

A kifejezésnyelvtan és az elemző segítségével előállított elemzési fák tartalmaznak minden olyan információt, melyek segítségével a bemenethez megadható összes lehetséges helyes írásmód előállítható és ezekhez megfelelő magyarázatok fűzhetők. Az alábbiakban egy példán keresztül szemléltetjük a feldolgozás egyes lépéseit és a felhasználó számára megadott kimenetet.

A felhasználói bemenet legyen az alábbi:

```
sötétnarancssárga
```

Az előfeldolgozás során ezt a következő tokenekre választjuk szét:

```
sötét narancs sárga
```

A tokenekből a morfológiai elemző és a szemantikai jegyek adatbázisa segítségével az alábbi terminális szimbólumokat és attribútum-érték struktrúrákat állítjuk elő:

```
1.
N(wordform="sötét", stem="sötét", match="FN,NOM", sem=['Color3'],
ncomparts="1", ncompartsx="1+", nsylls="2")
A(wordform="sötét", stem="sötét", match="MN,NOM", sem=['Color3'],
ncomparts="1", ncompartsx="1+", nsylls="2")
```

```
2.
N(wordform="narancs", stem="narancs", match="FN,NOM", sem=['Color1'],
ncomparts="1", ncompartsx="1+", nsylls="2", join1=['Color1'])

3.
A(wordform="sárga", stem="sárga", match="MN,NOM", sem=['Color1'],
ncomparts="1", ncompartsx="1+", nsylls="2", join2=['Color1'])
```

Látható, hogy az első tokennek két különböző szófajú elemzése is van (főnév, melléknév). A `Color1` és `Color2` szemantikai jegyek a színnevek, illetve a színárnyalatok kategóriáit jelentik.

A parser segítségével a terminális struktúrákból első lépésben az alábbi elemzési fák építhetők:

```
1.
A(sep=[' '], ncompartsx="2+", sem=['Color1']) : M_EK_SZIN_3
    A(stem="sötét", sem=['Color3'])
    A(sep=[''], ncompartsx="2+", sem=['Color1']) : M_EK_SZIN_1_2
        N(stem="narancs", sem=['Color1'])
        A(stem="sárga", sem=['Color1'])

2.
A(sep=[' '], ncompartsx="2+", sem=['Color1']) : M_EK_MINOSEG_1_2
    A(stem="sötét", sem=['Color3'])
    A(sep=[''], ncompartsx="2+", sem=['Color1']) : M_EK_SZIN_1_2
        N(stem="narancs", sem=['Color1']
        A(stem="sárga", sem=['Color1'])
```

A fenti (egyszerűsített, nem az összes attribútumot megjelenítő) fákban a nemterminális szimbólumok után, kettősponttal elválasztva az őket létrehozó szabályok azonosítója olvasható. Az eltérés a fenti – helyesírás szempontjából egyforma végeredményt adó – két elemzés között az, hogy az `M_EK_SZIN_1_2` szabály (*Összetett színnevek képzése*) alapján egybe írt *narancssárga* tokent a *sötét* jelzővel 2 különböző szabállyal is összevonhatjuk: `M_EK_SZIN_2` (*Színárnyalat és színnév összetétele*) és `M_EK_MINOSEG_1_2` (*Minőségjelzős szerkezetek*).

A többértelműségek csökkentése érdekében a nyelvtan szabályai között részleges rendezési relációt definiáltunk, ez a gyakorlatban egyes szabályok más szabályokra vonatkozó letiltásával valósul meg (l. 9.2 rész). Ebben az esetben a specifikusabb `M_EK_SZIN_2` szabály tartalmaz egy tiltó utasítást az általánosabb `M_EK_MINOSEG_1_2` szabályra nézve, ennek eredményeképpen a parser kimenetében csak az 1. elemzési fa fog megjelenni.

Az utolsó lépésben az elemzési fák alulról-felfelé bejárásával, a csomópontokhoz rendelt szabályazonosítók, a **sep** attribútum (a külön- vagy egybeírást kódoló szimbólumok) és a terminális kategóriák segítségével, előre megadott sablonok kitöltésével generáljuk a felhasználó számára az elemzéseknek megfelelő természetes nyelvű magyarázatokat. Az alábbiakban bemutatjuk az 1. elemzési fából generált kimenetet:

```
Javasolt alak: „sötét narancssárga”

Magyarázat:

  1.  A „narancs” főnév és a „sárga” melléknév az alábbi szabály alap-
      ján egybeírandó:
      Az összetett színneveket egybeírjuk.
```

```
2. A „sötét” melléknév és a „narancssárga” melléknév az alábbi sza-
   bály alapján különírandó:
   A színnévi alaptag összetett, ezért különírjuk jelzőjétől (AkH.
   110.)
```

10 Összefoglalás

A dolgozatban bemutattuk a *helyesírás.hu* helyesírási tanácsadó portál hátterét, nyelvtechnológiai támogatással működő webalkalmazásainak részletes működését. Természetesen csak a valós használat során lehet majd felmérni, hogy a jelen formájában már működőképes, de lehetséges problémáknak csak korlátozott körét kezelni képes rendszer milyen felhasználói elégedettségi mutatókra számíthat, mint ahogy a mindennapi használatban felmerülő kérdések határozzák meg azt is, hogy a további fejlesztéseknek milyen területekre kell fókuszálni.

Hivatkozások

1. Kis Á.: Az akadémiai helyesírási szabályzat és a számítógép. Magyar Nyelvőr, Vol. 123, No. 2 (1999)
2. Laczkó K., Mártonfi A.: Helyesírás. Osiris Kiadó, Budapest (2005)
3. Naszódi M.: Nyelvhelyesség-ellenőrzés számítógéppel (Parciális szintaxis). In: VII. Országos Alkalmazott Nyelvészeti Konferencia, I. kötet. Külkereskedelmi Főiskola, Budapest (1997) 256–260
4. Németh, L.: Automatic non-standard hyphenation in OpenOffice.org. TUGboat, Vol. 27, No. 2 (2006) 750–755
5. Novák A., M. Pintér T.: Milyen a még jobb Humor?. In: Alexin Z., Csendes D. (szerk.): A 4. Magyar Számítógépes Nyelvészeti Konferencia előadásai. Szegedi Tudományegyetem (2006) 60–69
6. Pintér T., Oravecz C., Mártonfi A.: Online helyesírási szótár és megvalósítási nehézségei. In: Tanács A., Szauter D., Vincze V. (szerk.): MSZNY 2009. Magyar Számítógépes Nyelvészeti Konferencia JATEPress, Szeged (2009) 172–182
7. Pomázi Gy. (szerk.): A Magyar Helyesírás Szabályai. Tizenegyedik kiadás, tizenkettedik (példaanyagában átdolgozott) lenyomat. Akadémiai Kiadó, Budapest (2009)
8. Prószéky, G., Kis, B.: Morpho-syntactic Parsing of Agglutinative and Other (Highly) Inectional Languages. In: Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics. College Park, Maryland, USA (1999) 261–268
9. Yoshida, K., Tsuruoka, Y., Miyao, Y.: Ambiguous Part-of-Speech Tagging for Improving Accuracy and Domain Portability of Syntactic Parsers. In: Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (2007)
10. Varasdi K., Rebrus P.: A helyesírás mint default öröklődési hálózat. Előadás A mai magyar nyelv leírásának újabb módszerei VI. konferencián. Szeged (2003)