Computational Benefits of a Totally Lexicalist Grammar

Kata Balogh – Judit Kleiber*
University of Pécs, Department of Linguistics
matrica@stud.btk.pte.hu, kleiberj@btk.pte.hu

Abstract

In this paper we demonstrate the computational benefits of a radically lexicalist generative grammar. We have developed a Prolog-parser on the basis of the new approach of Totally Lexicalist Morphology (TLM), which is developed out of Generative Argument Structure Grammar (GASG; [2]), a new and radical version of lexicalist generative grammar (in the spirit of e.g. Karttunen [5]). The parser decides the grammaticality of Hungarian sentences, and creates their (practically English) DRSs.

1 Introduction

GASG is based on lexical signs, whose inner structure is so rich that they can constrain features of their linguistic environment. The description of a lexical sign serves a double purpose: it determines (1) the potential environment of the given sign in grammatical sentences and (2) the sign itself. The characterization is formulated with constants (for the sign itself) and variables (for environmental material). In this way signs have mutual co-occurence requirements[1], on the basis of formal features which are suited for unification, and when these requirements are met, the respective semantic features are unified simultaneously. Case marking and agreement thus get a straightforward "totally lexicalist" treatment; word order, and especially adjacency relations between realizations of lexical signs, however, require an optimalistic treatment: "environmental requirements" in lexical signs are assigned ranks which permit the indirect satisfaction of requirements by the satisfaction of requirements of higher ranks. In the earlier model of GASG lexical signs were assigned to words, and the lexical description of morphologically complex words[2] was claimed to be calculable in a multiple lexical inheritance network. In this paper a better method – TLM – is proposed which suits the principle of total lexicalism more radically: *each single morpheme within words is considered a lexical unit.*

*We are grateful for the Hungarian National Scientific Fund (OTKA T038386).

[1]Similarly to the method of Link Grammar [6].

[2]Very frequent in morphologically rich languages like Hungarian, Georgian, Turkish etc.

2 The parser

2.1 Morphophonology

Although our work is still being developed, the version which is available now can parse uncompound neutral Hungarian sentences. In our implementation we insist on the theoretically clear principles of TLM but of course we have to make some technical changes according to the special features of Prolog programming.

The input for the parser is a simple string, and as the first step, the relevant *lexical items* (LI's) should be collected on the basis of morphemes in the given sentence.

(1) `lexi(m("","ül",""), labstem("sit", phonfst(1,1,1,2), 2, [["NOM","LOC"]])).` (verbal stem 'sit')
`lexi(m("t","A","t"), labinfl("cause", phonfsu(2,2,0.2,2), 2, ac(-1,0,1))).` (causal verbal suffix)

LI's (see (1)) belong to morphemes and consist of the "own word"[3] of the morpheme (`m("","ül","")`) and a "label" where we store: the predicate name e.g. `labstem("sit",...,...,...)`, phonological features e.g. `labstem(..., phonfst(1,1,1,2),...,...)`, category [4] e.g. `labstem(...,...,2,...)` and inherent syntactic conditions e.g. `labstem(...,...,...,[["NOM","LOC"]])` (here the argument structure).

2.1.1 Phonological Features

The phonetic form of words is divided into three parts, where capital letters refer to variables, expressing underspecified parts of the given morpheme. For example, in (1) the "own word" of the causal verbal suffix is `m("t","A","t")`, which can appear as *-tat* or *-tet*, and the frontness of the vowel depends on the frontness of the stem: *ül-tet* 'sit-cause', *fut-tat* 'run-cause' (vowel-harmony). On the phonological level two kinds of requirements have to be incorporated. The first type accounts for the choice of the possible realizations of the given morpheme and the second one determines the effect of lexical items on the phonological realizations of other lexical items in the same word.

On the basis of phonological features (which is technically stored in an array) the program checks *compatibility* between morphemes in the given word. In the phonological characterization of a stem (`phonfst(...)`) four phonetic properties potentially relevant to (the vowels of) the environment are provided: *frontness*[5], *roundness*[6], *vertical* position of tongue and "*lowering* property" peculiar to Hungarian[7]. The label of an inflection also shows four phonetic fea-

[3] Own word=how the LI appears in the sentence.
[4] 1=noun/suffix for nouns, 2=verb/suffix for verbs, 3=determiners, 4=adjunct
[5] E.g. fiú-*nak* 'boy-DAT' vs. őr-*nek* 'guard-DAT'
[6] E.g. űr-*höz* 'space-ALL' vs. hír-*hez* 'news-ALL'
[7] E.g. vár-*ak* 'castle-PL' vs. cár-*ok* 'czar-PL'

tures, whether it causes: *lengthening*[8], *shortening*[9], *epenthesis*[10] and *lowering*[11]. Derivative suffixes (e.g. *-tAt* above) require other labels.

The program generates the proper form of a word by identifying variables with constants and finding the relations between the lexical items. It "sees" all the possible lexical items of the given word at once and it can take into consideration all kinds of information simultaneously.

2.1.2 Morpheme order

Having checked all phonological features the program next checks the right morpheme order in the sentence according to rank parameters. Each morpheme has a rank parameter which says that the given morpheme wants to be adjacent to another one and it also says how strong this requirement is (1 is the strongest, 2 is weaker etc.). Every suffix would like to be adjacent to the stem, but these requirements are not equally strong. For example, in the case of verbs, the strongest one (rank 1) is the requirement of the suffix *-hat* ('can/able to'), on rank 3, there are two kinds of suffixes: tense and mood marking, and the agreement suffixes are on rank 4. In our definition, a requirement can be satisfied indirectly if it cannot be satisfied directly (if there are more suffixes which want to be adjecent to the stem). If a suffix A wants to be adjacent to the stem on rank α, and another suffix B wants to be adjacent to the stem on rank β, and $\alpha < \beta$ then the right morpheme order is: *stem-A-B*.

(2)

be-ül-tet-het-né-d	be-ül-tet-het-t-ed
into-sit-cause-can-cond-2sg	into-sit-cause-can-past-2sg
'you could/should make sy. to sit into sg.'	'you could/were able to make sy. to sit into sg.'

Each suffix wants to be adjecent to the stem ('sit') with the following ranks: cause-$\alpha = 1$, can-$\alpha = 2$, mood-$\alpha = 3$, tense-$\alpha = 3$, agr-$\alpha = 4$.

2.2 Syntax and Semantics

2.2.1 Grammatical relations

The checking and parsing demonstrated in 2.1 gives us a list of LI's (morphemes) that calls a new predicate, which provides the syntactic parsing according to the listed LI's. Each morpheme calls its own syntactic requirements that carry out a mutual search with the other morphemes. In this way the program creates a new list, the elements of which are ordered pairs: (1) lexical item of the morpheme and (2) the grammatical relations that this morpheme can establish in the given sentence. The representation of a grammatical relation is always an ordered septuple: `gr(X,Y,Z, N,M, K,L)`. In this expression X is the element that calls the relation, Y is the environmental element that the first one searches and Z is

[8] E.g. apa-*ként* 'father-FORM' (as a father) vs. apá-*t* 'father-ACC'
[9] E.g. kerék-*ként* 'wheel-FORM' (as a wheel) vs. kerek-*et* 'wheel-ACC'
[10] E.g. farok-*ként* 'tail-FORM' (as a tail) vs. fark-*at* 'tail-ACC'
[11] E.g. sors-*ot* 'fate-ACC' vs. sors-ok-*at* 'fate-PL-ACC'

the type of the relation. The other four elements refer to the morphemes that have the relations: N, M are the serial numbers of X and K, L are the serial numbers of Y.

In our system finite verbs look for the two pillars of their arguments[12]. An intransitive verbal stem searches the two pillars of its nominative argument: the determiner pillar for `gr("regent", "noun", "subj", ...)`[13] and the nominative argument for `gr("regent", "det", "subj", ...)`. In this way transitive verbal stems search four elements – two-two pillars of two arguments. Determiners look for a noun stem for `gr("det", "noun", "free", ...)`, and the stem of the finite verb for `gr("det", "regent", "unspec", ...)`. Common nouns without a case marking suffix search the finite verb for a *subject* relation, affixes search the stem for `gr("pref/suff", "stem", "free", ...)`, and an environmental morpheme for a grammatical relation[14]. In the case of argument relations the search must be mutual, otherwise the program could accept the illformed sentence **A fiú a lány ül* 'the boy the girl is sitting'. Mutual search means that members of a pair of morphemes in a grammatical relation must find each other but no further morphemes can be found for the same relation. So we have to check if every `gr(A,B,REL,X,_,Z,_)` finds `gr(B,A,REL,Z,_,X,_)` only once.

2.2.2 Word order

Word order can be checked by means of lexically determined rank parameters which determine the required strength of "immprec" ('immediate precedence') between two morphemes that have a grammatical relation[15]. Word order, thus, is also determined by indirect satisfaction. This type of satisfaction is defined as follows: a rank n immprec requirement between word A and word B is satisfied: directly if A immediately precedes B $(...AB...)$; or indirectly if there is an element X and element Y, and X immediately precedes Y, and X is an n-dependent of A and Y is an n-dependent of B $(...A...XY...B...)$. X is an n-dependent of A if X stands in a rank k immprec relation with A, $k \leq n$, or an arbitrary immprec relation holds between X and an n-dependent of A.

[12] For example, Hungarian agreement relations. The finite verb has agreement with the subject in number and person, which appears on the noun; and the verb has agreement with the object in definiteness, which appears on the determiner of the object.

[13] Regent=lexical head.

[14] For example, the ACC suffix *-(V)t* searches a transitive verbal stem or, if the verbal stem is intransitive, it searches a morpheme within the verb which is resposible for the accusative nature of the verb, the *-tAt* ('cause') morpheme.

[15] With these immprec requirements we can explain, for example, why free adverbs and arguments can be freely mixed in Hungarian but not in English, which is also true in the case of idioms: Unfortunately Peter loves my sister. – *Peter unfortunately loves my sister.
Sajnos Péter szereti a húgomat. 'unfortunately Peter love-2sg.defobj the sister-poss.1sg-ACC'
Péter sajnos szereti a húgomat. 'Peter unfortunately love-2sg.defobj the sister-poss.1sg-ACC'
Péter beadta tegnap a kulcsot. – *Peter kicked yesterday the bucket. The explanation is that the regent-adjacent relation has an immprec rank α and free adverb-finite verb relation has a rank β, and in Hungarian $\alpha{=}\beta$, but in English $\alpha{<}\beta$.

2.2.3 Semantic representation

If the parser operating on a sentence has the right morphosyntactic output the program turns to semantic selection, and if this is also successful, it can provide the semantic representation: a DRS. In our system each lexical item – so each morpheme – contributes a proto-DRS to the final DRS. In this way we first get a proto-DRS, which contains underspecified referents and the lexical descriptions determine the identification of the referents. According to DRT [3, 4], determiners (and proper names) provide referents, common nouns and adjuncts predicate something of them, and verbal stems and suffixes provide a situation referent besides predicating something (of other predicates).

(3) Egy fiú be-ül -tet -het-né a nagymamá-t a szék -em -be.
a boy in-sit-cause-can-would the grandma-ACC the chair-poss.1sg-INESS
'The boy can sit his/her grandma in my chair.'

For sentence (3), the proto-DRSs are: *egy* 'a': provide $[r_1]$, *fiú* 'boy': pred $[boy(r_2)]$, *be-* 'into': pred $[into(r_3, r_4)]$, *ül* 'sit': provide $[e_1]$, pred $[sit(e_1; r_5)]$, *-tet* 'cause': provide $[e_2]$, pred $[cause(e_2; r_6, e_3)]$, *-het* 'can': provide $[e_4]$, pred $[can(e_4; r_7, e_5)]$, *-né* 'cond': provide $[e_6]$, pred $[would(e_6; r_8, e_7)]$, *a* 'the': prov $[r_9]$, *nagyi* 'grandma': pred $[grandma(r_{10})]$, *-t* 'ACC': –, *a* 'the': prov $[r_{11}]$, *szék* 'chair': pred $[chair(r_{12})]$, *-em* 'poss.1sg': pred $[P(r_{13}, I)]$, *-be* 'INESS': –.

The description of the lexical item of a common noun stem "says" that the referent it says something about must be the same as the referent that was provided by its determiner. At this point we have to use the list in which we collected all the grammatical relations before. Similarly, the *-tAt* morpheme says that the situation it predicates something of must be the same situation as the one described by the "morpheme string" just before it (here: $e_3 = e_1$), and this is the same in the case of *-hAt*. The possessive suffix says that the owner is 1sg or 2sg etc. and the owned thing is the same referent as its stem was predicated something of. Finally the verbal prefix should be involved in the verbal stem: $sit\_into(e_1; r_5 = r_3, r_4)$. According to these constraints on the lexical item we have all the equivalence relations: $r_2 = r_1$, $r_3 = r_5$, $e_3 = e_1$, $e_5 = e_2$, $r_{10} = r_9$, $r_{12} = r_{11}$ and $r_{13} = r_{12}$. So we get the final DRS:

$r_1, r_2, r_3, e_1, e_2, e_3$
boy(r_1)
grandma(r_2)
chair(r_3,I)
sit_into($e_1; r_2, r_3$)
cause($e_2; r_1, e_1$)
can($e_3; r_1, e_2$)
would($e_4; r_1, e_3$)

In comparison with earlier approaches there are two important changes. First, in this version morphemes give the proto-DRSs, so that we can give more precise interpretation, and second we try to give a formulation which easily fits in the structure of a Lifelong DRS [1], which goes beyond sentence parsing. According to the formulation for Lifelong DRSs, each referent belongs to one world, and some morphemes can open new worlds (e.g. conditional). We have

to define which refrent was provided in which world and from which world it is available, and we also have to define the ordering between the worlds, which is represented by relations: $\sim$, $\leq$, $<$. So, the DRS for sentence (3) contains the following worlds and ordering: $v(r_1, r_2, r_3, e_3) \rightarrow w_1(e_1, e_2) \rightarrow w_2(e_4)$.

3 Conclusion and further work

In our paper we demonstrated a Prolog parser for Hungarian sentences. The principal aim of this paper is to legitimize a new sort of generative grammar (GASG). A successful implementation is the best evidence for the exactness and consistency of a formal system. In our system information extraction is based on a radically lexicalist generative grammar, GASG, from which even Function Application has been omitted. What has remained is lexical sign and unification as the engine of combining signs; which promises a staightforward implementation in Prolog. Our parser decides whether a sentence is grammatical and then creates the semantic representation: a DRS completed with information about its embedding interpreters' information state.

For further research we would like to work out a (sentence) translation system. According to our approach, the translation of sentences should be mediated by the meaning representation, the DRS ($language_1 \rightarrow DRS \rightarrow language_2$). At operation $language_1 \rightarrow DRS$ we must register which propositions copredicate and how. So we should extend the DRS with formulas that tell how these propositions copredicate: regent-argument relation (and which arguments) or free relation. These formulas do not give additional information to the DRS, but help to build the syntax for $language_2$. Our aim is to point out that, in spite of different input words, the network of copredication is the same.

References

[1] Alberti, G.: Lifelong Discourse Representation Structure. In Poesio - Traum (eds.): *Gothenburgh Papers in Computational Linguistics*, 00-5, 2000.

[2] Alberti, G., K. Balogh, J. Kleiber, A. Viszket: Total Lexicalism and GASGrammars: A Direct Way to Semantics, In A. Gelbukh (ed.): *Proceedings of CICLing2003*, Lecture Notes in Computer Science N2588, 37-48, Springer-Verlag, 2003.

[3] Kamp, H.: A theory of truth and semantic representation, In Groenendijk - Janssen - Stokhof (eds.): *Formal methods in the study of language*, Amsterdam, Math. Centre., 1981.

[4] Kamp, H., U. Reyle: *From Discourse to Logic*, Kluwer AP, 1993.

[5] Karttunen, L.: *Radical Lexicalism*, CSLI Report 68., Stanford, 1986.

[6] Sleator, D., D. Temperley: *Parsing English with a Link Grammar*, Technical Report CMU-CS-91-196, School of Comp. Science, Carnegie Mellon Univ., Pittsburg, 1991.