Supplementary document to the article “Null arguments and the inverse problem” published in *Glossa*

Algorithm and methodology

Pauli Brattico

2020

Contents

1 Introduction

This supplementary document provides some of the details concerning the algorithm and methodology used in the main article “Null arguments and the inverse problem” published to *Glossa*. It begins with a relatively nontechnical introduction to the problem itself, followed by a description of the algorithm, and concludes with a section explaining the methodology. Methodological aspects, including the results of the study, are elucidated in considerable detail. The purpose of this document is to explain how the analysis was implemented and tested, not the analysis itself. It will also try to elucidate problems and issues that were observed during the study and which should be seen as areas of improvement and possible points of criticism. Full formalization is available in the source code and its documentation, both in the public domain.

2 The inverse problem

The inverse problem addressed in the main paper is how to infer syntactic and semantic properties of null arguments from a linear string of phonological words, thus ultimately from the sensory input, where they are absent. We also assume that the model must satisfy certain elementary criteria of cognitive plausibility, such as observational and descriptive adequacy, and that it is also explanatory and therefore takes language acquisition into account (Chomsky 1965). Stated in this way, we are neither committed to any particular syntactic framework, such as the generative grammar, LFG or a connectionist model, nor do we assume that there are null arguments or, if we assume, what kind of objects they have to be. Instead, we want to derive their syntactic and semantic properties in whichever way they are ultimately represented in an algorithm that, we hope, solves the problem itself. Perhaps we can hope that the problem itself provides some guidance in selecting appropriate computations and representations.

Solving the inverse problem is motivated by two concerns. The first is that since null arguments, or their observed semantic effects, are not present in the sensory input, we hope that by examining how the brain reconstructs them we could learn something useful about language comprehension and the cognitive processes and representations that are involved in such processes, for example, to what extent they are performance factors or innate properties of the language faculty (UG) itself (most likely a mixture of both). The second concern is that by solving the inverse problem in some feasible way hope to provide knowledge that could help researchers to come up with a more comprehensive and computationally explicit (hence also rigorous, and testable) theory of human natural language parsing. One could potentially adopt features of the present analysis into any other computationally explicit approach to parsing and language comprehension. Although any particular solution to the inverse problem might not satisfy these needs, it does provide a possible starting point.

We require the analysis to be observationally adequate. This means that it must be able to partition to set of possible inputs correctly into those which are judged as grammatical and those which evaluate as ungrammatical by a native speaker. We can think of the native speaker as constituting a gold standard against which the model is evaluated. An analysis that is not observationally adequate does not count as a successful scientific explanation – it does not replicate and derive observations. It must be emphasized here, however, that many computational approaches to language processing (e.g., practical parsers, engineering solutions) do not assume this condition. Being able to recognize ungrammatical sentences as ungrammatical serves no immediate practical purpose, as such sentences are almost never used in real communication and even when they do, they serve no useful purpose. We cannot ignore them here, however.

Descriptive adequacy means that the theory should only posit computations and representations that can be taken as realistic descriptions of the human language faculty (UG) in the light of independent evidence. What is taken as a realistic description of the language faculty depends on the theoretical framework, and many other explicit and implicit assumptions, but we can approach this property from a behavioral point of view and require that the analysis captures as much independent linguistic and other behavioral data as possible. In addition, we can assume that alternative models are compared against each other in terms of how much independent data they explain. Suppose we have a number of sentences in an observation set (corpus) that we want to algorithm to analyze. A trivial solution to this problem consists in a table-lookup that maps each input directly into a specific output. This model, although observationally adequate, cannot be considered descriptively adequate. As soon as any change is introduced to any of the test sentences, no matter how small, the model fails. Purely statistical, correlative analyses that only map a finite number of clauses into a finite number of outputs, popular as these approaches are, might be considered irrelevant as well on such grounds.

Explanatory adequacy means that the computations and representations assumed in the algorithm should be consistent with what is known from language acquisition and language universals. In the main article I discussed null argument data from three languages, English, Italian and Finnish, and found similarities and differences. Insofar as some property is universal or close to one, we do not want to model it by a theory or algorithm which assumes that the property is acquired from the surrounding linguistic culture or input and would therefore share properties among languages only by mere coincidence; instead, we assume that it derives from the biologically determined architecture of the language faculty. Differences must be captured by something that can be assumed to be realistically determined by the surrounding linguistic context during (mostly unsupervised) language acquisition, such as properties of lexical items, parameters, and other factors. These choices are informed by what we know about the properties in question. It might be that a property that looks language-specific turns out to be less so, and vice versa.

We can envision further criteria. One important and certainly relevant criteria is psycholinguistic adequacy, which refers to a requirement that the algorithm does not violate properties that are independently known from psycholinguistic research. If we want the algorithm to capture real language comprehension, it must satisfy this condition. For example, the algorithm should not enter into a garden-path when processing any particular input sentence unless real speakers are likely to do that as well. A psycholinguistically adequate model would be one in which the amount of computational resources consumed as a function of the complexity and the type of input approximates the cognitive resources spend during online processing of the same item. There are many additional psycholinguistic factors that one might be interested in capturing, such as errors and failures to find implausible parses, lexical-semantic factors that affect the parsing solutions selected, and many others. Psycholinguistic adequacy was not specifically addressed in this study, although the condition does play an important (but implicit) role.

It is assumed that the algorithm works with a sensory input that is viewed abstractly as a linear string of phonological words. We therefore ignore and presuppose phonetic and phonological preprocessing. Each phonological word is represented as an orthographic sequence of phonemes (an idealization) together with word boundaries, represented by spaces, again an idealization. No syntactic decompositions, annotations or other properties that cannot be assumed to be part of the sensory input is accepted, however. No null arguments are marked into the input; rather, the inverse algorithm must find their properties from the information available at a sensory input, realistically constructed. No contextual help is allowed. Although context affects linguistic processing, speakers can reconstruct null argument sentences correctly even if they are presented completely out of the blue. Hence, context is not necessary, but might certainly facilitate (and disrupt) online processing.

In some linguistic literature, especially in the literature representing modern generative grammar, one finds considerable amount of skepticism towards the usefulness of an approach of this type. It could be viewed as an irrelevant performance theory. This skepticism is legitimate, but the matter can only be decided by empirical inquiry. Philosophical arguments either way will accomplish little. There is, however, a sense in which one can take this approach without entering this debate at all. An observationally adequate grammatical theory must capture a set of observations. A set of observations can be captured in two ways, by (1) generating the set or (2) recognizing the set. The former method uses the grammatical theory to generate all possible expressions (or observations), and this set is then compared with the attested corpus of observations. The two should match. If they do not match, the theory is either undergeneralizing or overgeneralizing and requires revision. The latter recognition method works by taking expressions as input and by judging (i.e. "recognizing") their properties, such as whether the input is grammatical or ungrammatical. These methods are mathematically equivalent under very weak assumptions because they define a set, and whatever the set may be, either method can be used to define that *same* set. Hence, there is no difference. In other words, one can consider that the inverse problem is just an alternative way of testing a theory of grammatical competence. Still, the emphasis should be on the word "can": it is equally possible to assume that the theory describes real properties of language comprehension in the human brain. Whether one assumes this or not depends on what requirements we put on the theory, as elucidated above.

3 The model

3.1 General properties

The algorithm is a Python program that reads an input sentence and provides a set of phrase structures and corresponding semantic interpretations (e.g., antecedent relations, binding relations, generic readings) as output. If no such object is generated, the input is judged ungrammatical. It therefore implements a function from linearly organized inputs into sets of structured syntax/semantics pairs. Its purpose is to model language comprehension.

The first step in the processing pipeline is phonological and morphological preprocessing. This step is responsible for morphological decomposition. Derivational and inflectional affixes are separated from each other and fed into the syntactic component either as individual lexical items or as features of lexical items, and in a linear order determined by their order inside the phonological words. The lexical stream is acquired from the surface vocabulary, which maps phonological words into morphological decompositions. For example, a complex Finnish word such as *juoks-isi-n-ko-han* 'run-cond-1sg-q-han' or 'should I perhaps run?' will be retrieved from the surface vocabulary, after which its constituents will be "steamed" into syntax as a linear order of individual lexical items and features C(HAN) * C(Q) * [1SG] * COND/T * V. The lexical input stream will be used to construct phrase structure representations for the input.

Word-internal morpheme order is reversed following Baker's Mirror Principle, which means that the mirror principle is assumed to be morphological, not syntactic operation (the decision is relatively irrelevant for this study). The decomposition itself is generated from the phonological input by using one of two possible procedures: either a direct mapping via a surface vocabulary, in which phonological words are mapped directly into their decompositions and thus to the ultimate lexical items, or via morphological parser, which accomplishes the mapping if the entry does not exist in the vocabulary. Morphological parser was not modeled in this study, but its output can be simulated by providing the input in a decomposed form, here 'run#cond#1sg#q#han'. This is also the form we find from the surface lexicon (/juoksisinkohan/ → run#cond#1sg#q#han). Inflectional features are inserted inside adjacent (following) lexical items. Thus, the first-person singular affix '1sg' is stored inside the conditional tense morpheme T/cond, which follows it in the inversed order. Inflectional morphemes, like all morphemes, can be provided via two routes: lexical and morphological.

The lexical stream is fed into syntax. It is used to construct a bare binary phrase structure tree representing a first pass parse. Because the first pass parse is generated directly form the linear input, it is related transparently to the sensory input and can be thought of representing a language-specific "sensorimotoric plan" for that expression. I will call the resulting structure as the *spellout structure*, a term used in the standard theory in a sense which captures its role in the present analysis quite well. If we think of the derivation in the opposite way, from meaning into spellout, this representation constitutes a structure that can be spelled out by a left-to-right, depth-first linearization.

Lexical stream is read from left-to-right, hence the syntactic structure is created in left-to-right order, following (Phillips 1996). Suppose the first word was α and the incoming lexical item is β. Because there are no other units in the syntactic working space, the system will merge α and β together to form a complex asymmetric unit [α β] which substitutes α (α is the *left constituent*, β is the *right constituent*). This implements a mapping α * β → [α β], with α and β arriving from the input stream and being preprocessed by morphology, as detailed above. The operation is similar to minimalist Merge, but due to some differences I call it Merge-1 ("inverse" of Merge) to avoid any confusion. Suppose a next word γ arrives from the lexical stream. There are several possible ways γ could be merged-1 to [α β], among them [[α γ] β], [α [β γ]] and [[α β] γ] and their inverse versions, in which γ is merged-1 to left. But because the spellout structure maps transparently into the sensorimotoric input/object by a left-to-right/depth-first linearization, as elucidated above, incoming words, which are linearized to the right of anything that was consumed before, must be merged-1 to the *right edge* of the existing structure, "right edge" containing of the highest node and any of its right daughters, granddaughters, and so on. The two options available are therefore (1) [[α β] γ] and (2) [α [β γ]]. All solutions that are available in principle are ranked and explored recursively by the algorithm by following the rankings. So the system implements a function α * β *. . .* γ → $\langle SYN_1, \ldots, SYN_n \rangle$, where SYN1,...,SYNn can be thought of as possible parses for the linear surface string explored in determinate order defined by the recursively applied rankings. We can further think of the rankings as creating a "psycholinguistic plausibility metric" over possible parses.

The SYN objects are language-specific because they mirror the sensorimotoric input/output. In a polysynthetic language like Finnish, where word order is relatively free, SYN objects are organized differently from, say, English. Yet, the propositional content of the respective sentences could be same or at the very least highly similar. It is fairly common in linguistics and cognitive science to think that the human conceptual system is universal, with very little language-based variation.[1] To solve this issue, the algorithm contains a *transfer* function which normalizes spellout structures in such a way that what arrives at the system responsible for semantic interpretation is close to being universal. It eliminates language-specific noise from the representation. In this way, we do not need to assume that the human conceptual system responsible for understanding linguistic communication differs radically from language to language; we assume, rather, that it is the language that differs. Transfer can be thought of as an inverse version of Move, call it Move-1, as it reconstructs misplaced constituents into their canonical, thematic positions. It forms chains but does it in reverse order when compared to the standard bottom-up theory.

Syntax and semantic interpretation interact at the *LF-interface*. A normalized phrase structure is checked for LF-legibility at the LF-interface, so that what is passed on is at least in principle semantically interpretable. If it is not, next syntactic object is constructed from the recursive ranked list and normalized via transfer. And so on (this scenario corresponds to a garden-path). Normalized SYN objects are binary branching bare phrase structure objects but they may contain *chains* which maintain a record of the normalization process. A chain is a representation in which one lexical element/phrase appears in several geometrical locations inside the phrase structure object. The model is summarized in Figure 1.

---

[1] Lexical items (words) may differ from language to language, while the conceptual system remains close to identical. Variation is considered possible, but the null hypothesis, based on the observation that it is possible to translate sentences in one language into another, is that there is no significant variation. We appear to communicate the same "world experience."

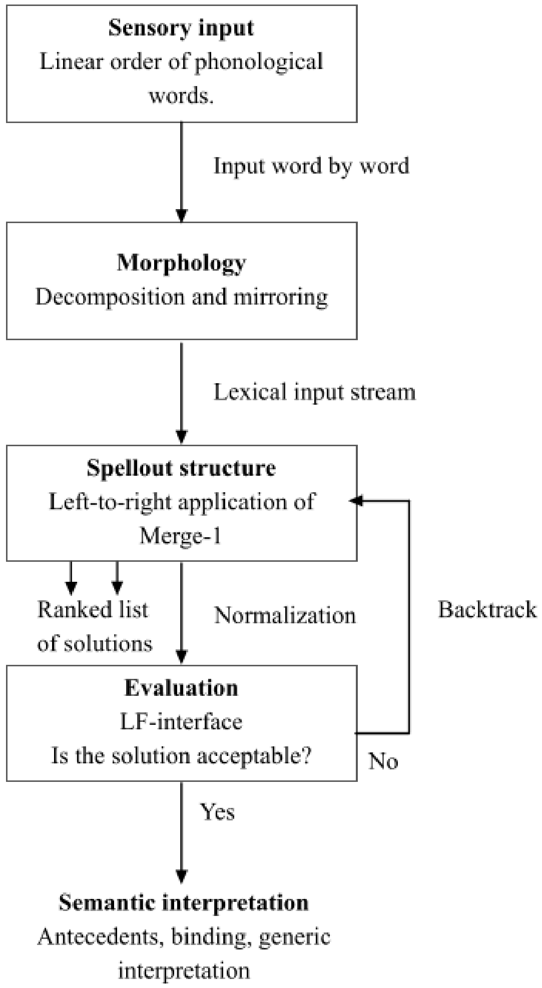

Figure 1. The general architecture of the language comprehension model. See the main text for explanation.

The above description should be regarded as a summary of the main properties of the model. A full documentation and the algorithm itself is available in public domain. In addition, what matters for the argument presented in the main article, the solution for the inverse problem, is the output, not the implementation; thus, the model should be evaluated in relation to the structural analyses it produces. There are three ways how a model of this kind could be improved. First, and most obviously, there were several problems that were left for future research, and these should be addressed; second, one can show that it is possible to compute the same input-output function with a much more elegant formalization, with elegance defined as code length; and third, one can enlarge the test corpus by considering additional evidence and build a model that applies to a larger dataset. The rest does not matter.

3.2 Null arguments and the architecture of the UG

In this section I recapitulate the general properties of null arguments and their reconstruction in the comprehension model. Most of these properties were discussed in the main article, but some details, elucidated here, were omitted.

Null arguments are not present in the sensory input, overt agreement suffixes are. Overt agreement suffixes are interpreted as inflectional features and are stored inside adjacent lexical items. A lexical stream T * 3sg will be represented as a lexical item T^0 = {T, PHI:NUM:3, PHI:PER:3, ...} at spellout structure (see Figure 1), with the phi-features being inside it. Insertion of the phi-features inside the lexical item is done in the syntactic component. The morphological system creates lexical input streams in which they are still separate entities. If a derivational morpheme is followed by several inflectional features in the sensory input, they are all inserted inside the following lexical item. Figure 2 illustrates this process.

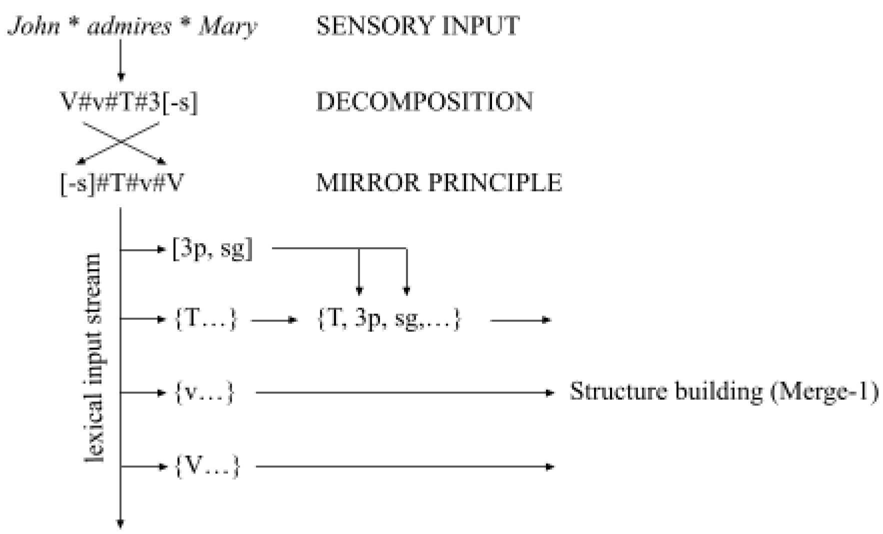

Figure 2. Morphological preprocessing. Derivational affixes are mapped to lexical items, whereas inflectional affixes (e.g., 3p, sg) are stored inside adjacent lexical items in the lexical input stream.

The next step relevant to null arguments takes place during transfer. A lexical item with unvalued phi-features φ_ and positive marking for morphosyntactic valuation +VAL, both features retrieved from the lexicon during morphological preprocessing and carried into syntax, will attempt to value these features by Agree-1 from properties available at the phrase structure available at the time the operation is triggered (during the processing pipeline). The input to Agree-1 constitutes the surface structure created by Merge-1 that has undergone head and phrasal reconstruction during transfer (Move-1 operations), hence we are targeting something close to "d-structure" in the standard theory. Agree-1 tries to value unvalued phi-features of head H locally from the phrase structure by examining its complement, DP specifiers inside of its complement, and its own specifier or edge, as shown in (1). It does this independent of whether the head contains valued phi-features from the sensory input.

(1) [DP_2 [T [$_{vP}$ DP_1 [v XP]]]

(3.) ← φ_ → (1.) (2.)

Various formulations of Agree-1 create different outcomes that must be tested by means of simulation. The crucial properties are the order in which the agreement domain is explored; whether the operation is triggered before or after phrasal reconstruction and whether it targets all members of a chain or only some; what counts as a specifier and specifier of a complement; and how deep we allow the operation to look inside the sister (i.e. long-distance phi-agreement phenomenon). A crucial assumption in the explanation of the pro-drop phenomenon is that non-conflicting (i.e. coherent) pronominal elements inside the head H will count as specifiers in the case no phrasal specifiers are found (technically the notion of specifier is redefined as "edge"). A non-conflicting pronoun is generated from the valued phi-features inside the head. "Non-conflicting" means that there may not occur phi features which share the type but not the value. For example, if the head contains two features 'number:sg' and 'number:pl', then no matter what other features there are, no coherent pronominal element is created. On the same token, if there is only one valued phi-feature inside the head, a consistent (but abstract) pronominal element is automatically generated corresponding to that feature.

Agree-1 will react to feature mismatches. If the input is **we admires*, then the features valued from the grammatical subject conflict with the features extracted from the input, and a mark is left to the lexical item signaling feature conflict which then crashes the derivation at LF. However, this mismatch does not affect parsing, and a parse is still generated.[2] This presupposes, as was explained above, that unvalued phi-features are valued from overt grammatical subjects even if some valued features already exist due to overt agreement suffixes in the input. In such situations Agree-1 performs phi-feature checking.

If a head contains an unvalued phi-feature when it is evaluated by LF-interface, then an antecedent recovery is triggered at LF which tries to locate a c-commanding antecedent.[3] LF-recovery looks first at the sister of the triggering head and, if the phrase does not constitute a suitable antecedent, applies the same test to the mother, iteratively. We can imagine LF-recovery as a process that establishes a connection, or a "path," between the triggering feature and the antecedent. This means that the reason lexical items have unvalued phi-features must be because they are semantically unsaturated in the Fregean sense, containing argument placeholders that must be provided with a referential argument, either a phrasal DP or minimally a set of valued phi-features.

4 The simulation experiment

4.1 Stimuli

The model was tested by feeding it with sentences from a test corpus and verifying the output. Test sentences were selected on the basis of the research agenda. Syntactic construction variables were selected, based on the research agenda, and they were crossed to generate a set of construction types and by populating the resulting construction types in a nonredundant way with the test sentences. Null arguments were explained on the basis of agreement reconstruction and LF-recovery, which suggests that we need to model at least agreement, pro-drop and several types of control sentences that have predicates that fail to satisfy their unvalued phi-features by Agree-1. These variables provided the starting point. To examine Finnish finite control in connection with third person pro-drop, *that*-clause embedding was added as an additional variable. It has also proven useful to include noncanonical word orders, because an approach that begins with a string of input words is always extremely sensitive to slight adjustments in word order. Noncanonical word order variations will quickly reveal if the model over- or under-generates, as we will see also

---

[2] This raises the interesting question of whether overt agreement features have any role in parsing, i.e. whether they affect the plausibility metric. No agreement-based rule was part of the algorithm reported in this study, but only because none was needed.
[3] Sisters of the mother-to-mother upward path, ignoring right adjuncts.

in this case. Word order variation was added into the list of variables for this reason. Finally, three languages were used: English, Finnish and Italian. Adverb attachment, wh-movement and radical pro-drop were added as separate items. Table 1 lists the variables used in generating the test corpus and the number identifiers of the sentences belonging to each group. 2512 test sentences were included in this study.

Table 1. Variables crossed in this study to create the test suite corpus.

Sentences	Pro-drop	Agreement	Word order	Embedding	Control	Comment
1-82	n/a	n/a	n/a	n/a	n/a	Sentences cited in the main article
83-103	No	Grammatical	Canonical	No	No	Canonical agreement [a]
104-130	No	Grammatical	Canonical	No	Yes	Canonical control [b]
131-144	No	Grammatical	Canonical	Yes	No	Canonical embedding
145-171	No	Grammatical	Canonical	Yes	Yes	Control under embedding
172-246	No	Grammatical	Noncanonical	No	No	Noncanonical word order
247-1186	No	Grammatical	Noncanonical	No	Yes	Control with noncanonical order
1186-1261	No	Grammatical	Noncanonical	Yes	No	Embedding and noncanonical word order
1262-1503	No	Grammatical	Noncanonical	Yes	Yes	Embedding, control and order
1504-1541	No	Ungrammatical	Canonical	No	No	Agreement errors
1542-1568	No	Ungrammatical	Canonical	No	Yes	Control with agreement errors
1569-1600	No	Ungrammatical	Canonical	Yes	No	Agreement errors and embedding
1601-1627	No	Ungrammatical	Canonical	Yes	Yes	Control with agreement errors and embedding
1628-1807	No	Ungrammatical	Noncanonical	No	No	Noncanonical order with agreement errors
1808-2051	No	Ungrammatical	Noncanonical	No	Yes	Control, agreement and noncanonical order
2052-2081	No	Ungrammatical	Noncanonical	Yes	No	Embedding, order and agreement errors
2082-2099	No	Ungrammatical	Noncanonical	Yes	Yes	Control, agreement, order and embedding
2100-2116	Yes	Grammatical	Canonical	No	No	Basic pro-drop [b]
2117-2143	Yes	Grammatical	Canonical	No	Yes	Pro-drop with control
2144-2151	Yes	Grammatical	Canonical	Yes	No	Pro-drop with embedding (partial pro-drop) [a, c]
2152-2178	Yes	Grammatical	Canonical	Yes	Yes	Pro-drop with embedding (partial pro-drop) [a, c]
2179-2194	Yes	Grammatical	Noncanonical	No	No	Pro-drop with noncanonical word order
2195-2260	Yes	Grammatical	Noncanonical	No	Yes	Pro-drop, word order, and control
2261-2266	Yes	Grammatical	Noncanonical	Yes	No	Embedding, word order and pro-drop
2267-2510	Yes	Grammatical	Noncanonical	Yes	Yes	Pro-drop, order, control and embedding
-	Yes	Ungrammatical	Canonical	No	No	n/a (i.e. pro-drop and agreement error)

a. This category is important in demonstrating how Agree-1 works.

b. Contains the basic control and pro-drop sentences discussed in the main article.

c. Relevant for the analysis of Finnish partial pro-drop and its antecedent recovery.

In addition to the sentences in Table 1, all sentences mentioned in the main article were inserted as first items in the input corpus, in the order of their presentation. They are items number #1-82 in the test corpus.

The input corpus was disambiguated and normalized. Disambiguation is not necessary, but it removes irrelevant data from the output files. Normalization means that all input words are presented in the same form, thus, sentence-initial capitalization is removed, there are no extra spaces, punctuation is removed, Scandinavian special letters ä and ö are replaced with a and o, and so on.[4] Same lexical items were used when possible to avoid irrelevant errors, as each lexical item must be specifically coded as a separate item. When all sentences use the same lexical items (when possible), fixing that one item generalizes automatically to all sentences that contains it. This is also the reason why some test sentences can be judged as pragmatically odd; lexical variation was minimized.

A test corpus that contains natural language sentences in some language presupposes that there is a *lexicon*. A lexicon is provided by means of three external files, which were (1) *language specific lexicon*, containing individual items and their properties insofar as they were specific to certain language (here, Finnish, English, Italian, plus the hypothetical 'radical pro-drop English' mentioned in the main article); (2) *universal lexicon*, which hosts lexical items that are assumed to be universal (e.g., T, v, C, P); and (3) *lexical redundancy rules* which describe the universal and general dependencies between words (e.g. that prepositions take DP complements). If a language-specific rule violates a lexical redundancy rule, then the former wins; a lexical redundancy rule constitutes an "elsewhere category" that is assumed unless another rule overrules it. The lexical architecture is illustrated in Figure 3.

[4] This is not necessary, but lead to occasional problems and bugs in reading, processing and storing text files.

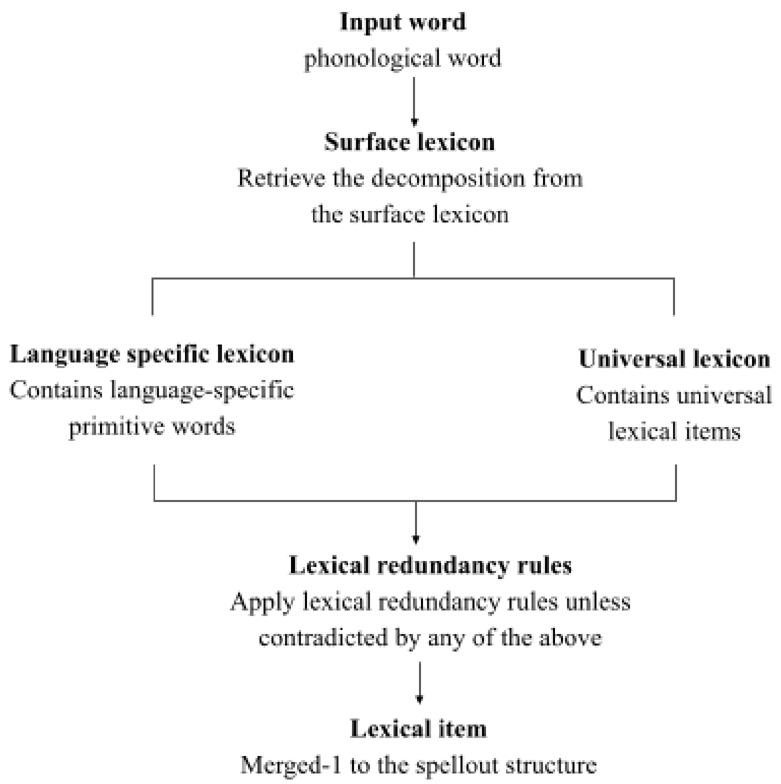

Figure 3. The lexical architecture assumed in this study. A phonological word is first matched with an entry in the surface lexicon, which retrieves its morphological structure. Each morpheme piece is then matched with the language-specific and universal lexicon, and the entry is further processed through lexical redundancy rules. The surface lexicon and the language-specific lexicon are in the same external file (lexicon.txt), but this does not necessarily reflect the empirical reality. Universal lexical elements are in their own external file (ug_morphemes.txt), but again only for convenience. Redundancy rules are also in an external file (redundancy rules.txt).

4.2 Procedure

The experiment is run by starting a main script which reads the three lexicon files (previous section, Figure 3) and the test suite corpus, prepares the parser, feeds each sentence from the test corpus into the analysis component, one by one, and then records the results, provided by the parser component, into two output files. The whole study is run at once, so that all input and output files can be associated unambiguously with each other.[5] See Figure 4.

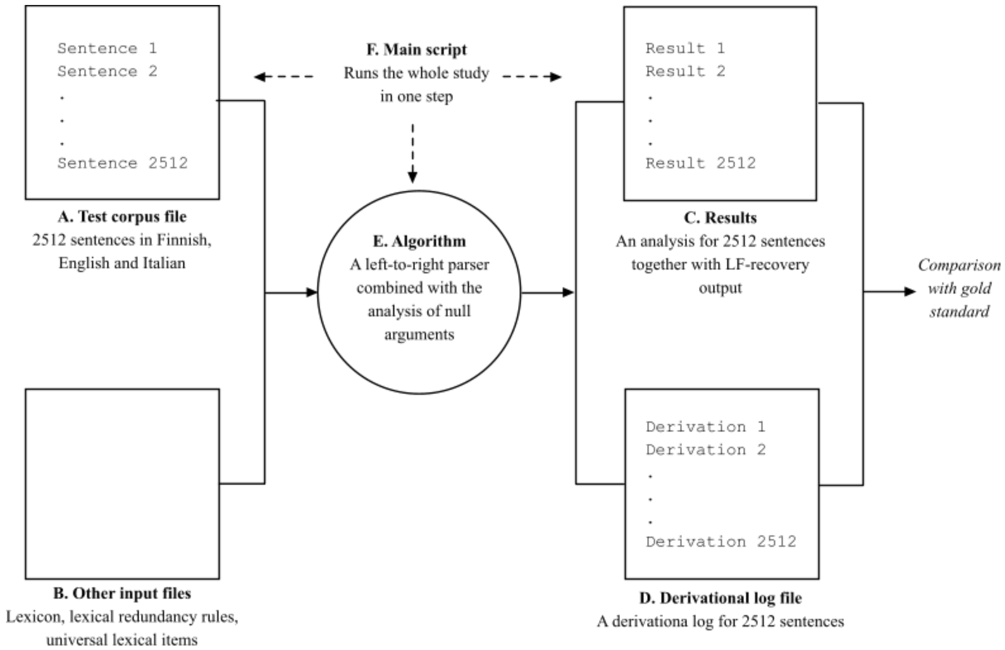

Figure 4. Design of the study.

The three input files, the source code of the Python program, and the two output files are all kept in the same folder which is stored for replication purposes. Thus, one should be able to replicate the study by starting the main script and having all the required files, unaltered, in the same folder. The results file is verified by a native speaker (in this case, the author), who should be a linguist with sufficient skill to evaluate the plausibility and correctness of the output. The algorithm provides each input sentence with a unique number identifier that can be used to identify it from the output files.

In this study the script processed 2512 sentences, which took 3 minutes and 10 seconds for a relatively powerful desktop computer with the standard Python interpreter and without parallel processing.

[5] The downside is that after each change, no matter how small, the whole study must be run and verified anew (this tends to put a practical limit on how big the test corpus can be).

4.3 Results

4.3.1 How to interpret the output

The results can be examined in the following way. A sentence of interest is identified from the test corpus and/or from the results file. This operation is shown in Figure 5.

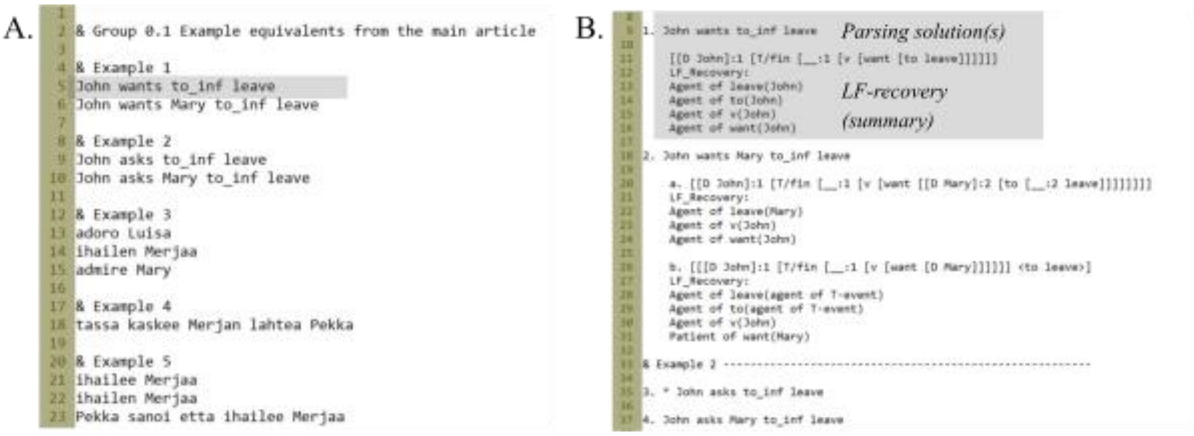

Figure 5. A screenshot of the test corpus (panel A, left), as opened in a standard text editor, and a corresponding results file (pane B, right). File B was produced by the algorithm when it read sentences from file A.

When the algorithm processes sentences from the input corpus (left, Figure 4), it will assign each sentence a unique identifier (here #1), which can be found also from the results file. The results file contains a summary analysis (first line) and a list of antecedents provided by LF-recovery. These output fields were selected due to their relevance to the subject matter of this study, but they can be in principle anything. LF-recovery pairs heads with unvalued phi-features with antecedent constituents; the output is simplified, and the thematic roles "agent of" and "patient of" are provided heuristically (i.e. they appear here for readability reasons and should not be interpreted as part of any explicit theory). The model contains neither an explicit thematic theory nor compositional semantics.

Suppose we want to understand the properties of the derivation in more detail, say because we suspect that the analysis is wrong, or that it delivers a wrong antecedent under some circumstances. We will first examine the log file and find the sentence with the same numerical identifier #1. Figure 6 illustrates part of what the log file contains in the case of sentence #1.

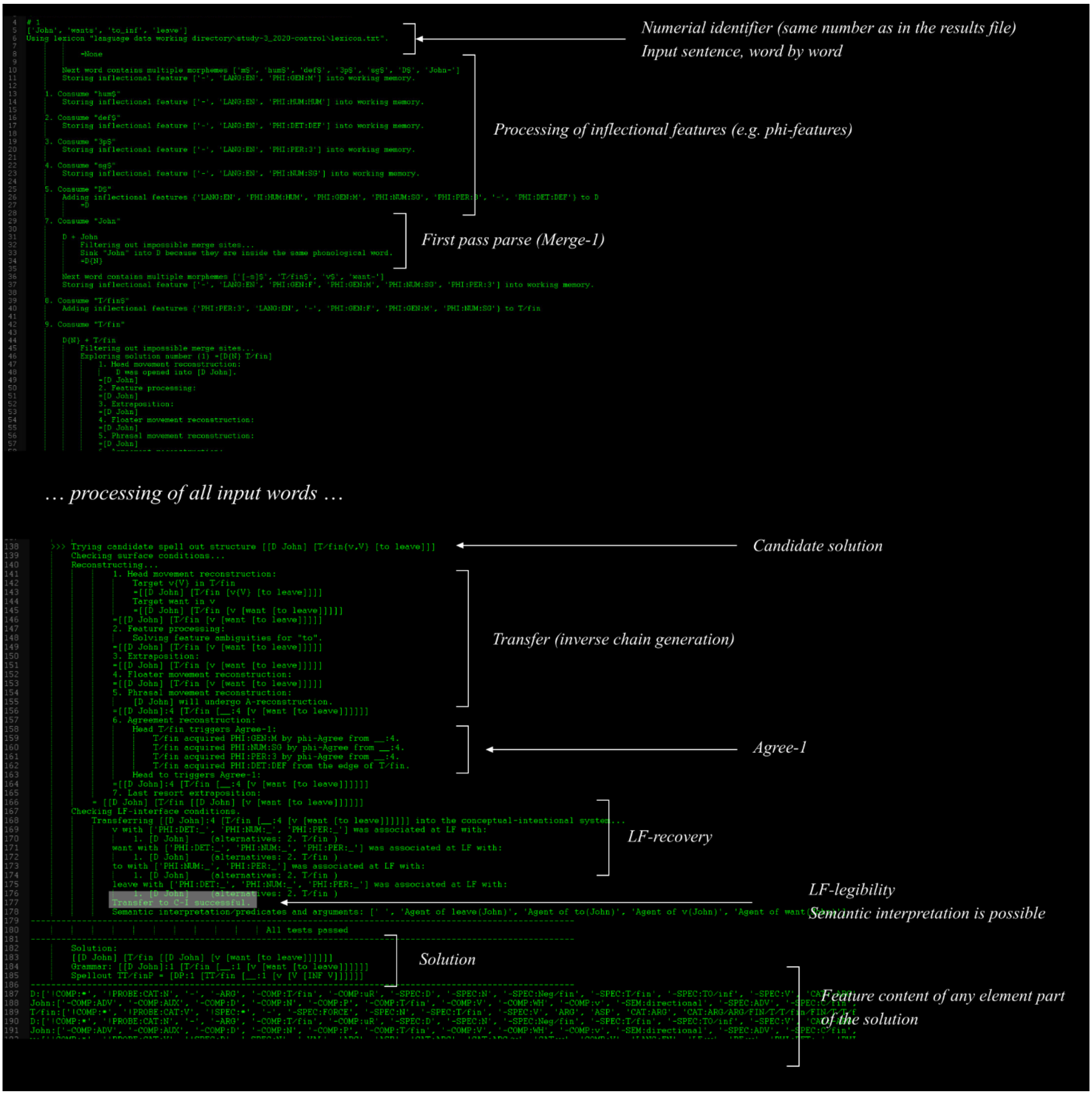

Figure 6. A screenshot of the log file. A log file is produced each time the algorithm processes the test corpus. Because the log file is generated mechanically, interpreting it requires a moderate understanding of the internal working of the algorithm, but the main features should be clear and the figure itself contains few explanatory remarks.

This shows how the model arrived to the output provided in the results file. This information can used if the output is in suspect, or if it is unclear how it was derived. If something is still left unclear, then the next step is to look at the source code that contains the formalization. Each of the operations visible in the log file correspond with a piece of code in the source code, as they were generated by that code while the sentence was processed. We can examine the operation more carefully by leaving ad hoc logging entries into the code and by running the same sentence or the whole study anew. Suppose we suspect that there is an error in how Agree-1 works. We can navigate to the function that performs Agree-1, which is in its own Python file (Python files are ordinary text files that can be opened with any standard text editor). A snapshot of this text file is shown in Figure 7.

Figure 7. A screen capture from the Python file defining Agree-1.

Notice the two logging commands embedded inside the function. If the error is not obvious from the code itself, we can add ad hoc logging commands anywhere in the code after which they will produce corresponding reports into the log files. In this way we can examine the derivation at any level of detail. Extremely detailed log files become unreadable and unusable, however, so it is preferable to work with a more austere format first and add ad hoc entries only when more details are called for.

4.3.2 Main results

Table 3 lists the variable combinations elucidated earlier (Table 1) together with a qualitative assessment of the model output. The observed problems and issues are then discussed in detail in subsequent sections. Three problem categories were used: problems, which involve clear problems involved in the computation of null arguments and their properties, such as wrong antecedents; issues, which involve problems elsewhere in the analysis, such as adverbial scope problems, and finally comments, which are meant as clarifications and explanations of solutions that I regard as nontrivial but still plausible or not in erroneous in any obvious sense, not at least within the generative background theory assumed as a starting point. Section 4.3.4 summarizes the results.

Table 3. Qualitative results of the computer simulation

#	Pro	Agr	Ord	Emb.	PRO	**PROBLEMS, *ISSUES AND COMMENTS	Section	Examples in the corpus #
83-103	–	+	+	–	–	**Canonical agreement**		
						Comment: analysis of case-marked postverbal direct object	4.3.3.1	
						Comment: conflicting φ-features inside grammatical heads at LF	4.3.3.2	
104-130	–	+	+	–	+	**Canonical control**		
						Comment: clausal antecedents and LF-recovery	4.3.3.3	
						Comment: analysis of Finnish infinitival complements	4.3.3.4	
						Comment: ambiguity in English *John wants Mary to leave*	4.3.3.5	
131-144	–	+	+	+	–	**Canonical that-embedding**		
						*Issue: postverbal direct object scope ambiguities	4.3.3.6	1265e, 1277d-e, 1325d-e, 1337e-f, 1381d-e, 1387d, 1399d, 1447d, 1459e, 1503d, 2270d-e, 2282d-e, 2330d-e, 2342e-f, 2386d-e, 2392d, 2404d, 2452d, 2464e, 2508d
145-171	–	+	+	+	+	**Embedded control clauses**		
						Comment: scope issues in adverbial *to leave* in English	4.3.3.7	
172-246	–	+	–	–	–	**Noncanonical word orders in otherwise regular finite clauses**		
						*Issue: double subject agreement in Finish	4.3.3.8	
						Comment: English accusative pronouns	4.3.3.9	1237, 1242
247-1186	–	+	–	–	+	**Control with noncanonical word orders**		
						Comment: Finnish free word order phenomenon	4.3.3.10	
						**Problem: Spurious solution in Finnish (*käsken minä lähteä*)	4.3.3.11	425, 1318, 2323, 2445, 2180, 2182
						*Issue: Spurious solution **to leave wants John*	4.3.3.12	508, 531, 580, 591, 1059, 1108, 1119, 1132, 1133, 1138, 1180,
						**Problem: *John wants* [*to Mary leave*]	4.3.3.13	517, 571, 604, 605, 610
1186-1261	–	+	–	+	–	**Embedded noncanonical word orders**		
						Comment: see §4.3.3.6	4.3.3.6	
1262-1503	–	+	–	+	+	**Embedded noncanonical control**		
						Comment: see §4.3.3.6	4.3.3.6	
1504-1541	–	–	+	–	–	**Agreement errors**	-	
1542-1568	–	–	+	–	+	**Agreement errors and control**	-	
1569-1600	–	–	+	+	–	**Agreement errors, embedding and order**	-	
1601-1627	–	–	+	+	+	**Agreement error, embedding and control**	-	
1628-1807	–	–	–	–	–	**Agreement error and noncanonical order**	-	
1808-2051	–	–	–	–	+	**Agreement error, control and order**	-	
2052-2081	–	–	–	+	–	**Agreement error and embedding**	-	
2082-2099	–	–	–	+	+	**Agreement error , control and embedding**	-	
2100-2116	+	+	+	–	–	**Canonical pro-drop**	-	
2117-2143	+	+	+	–	+	**Pro-drop and control**	-	
2144-2151	+	+	+	+	–	**Pro-drop in an embedded clause**	-	
2152-2178	+	+	+	+	+	**Pro-drop and control in an embedded clause**	-	
2179-2194	+	+	–	–	–	**Pro-drop with noncanonical word order**		
						*Issue: variation of 4.3.3.8	4.3.3.14	2180, 2184, 2186, 2188
2195-2260	+	+	–	–	+	**Pro-drop, control and order**	-	
2261-2266	+	+	–	+	–	**Embedded pro-drop and order**		
						*Issue 4.3.3.8	4.3.3.8	2261-2266
2267-2510	+	+	–	+	+	**Embedding, pro-drop, control and order**	-	
	+	–	+	–	–	n/a		

Abbreviations: Pro = whether the sentence involve pro-drop (+/–); Agr = whether the sentence contains grammatical (+) or ungrammatical (–) SV agreement patterns; Ord = whether word order is canonical (+) or uncanonical (–); Emb = whether the sentence is embedded (+) or not (–); PRO = whether the sentence involves LF-recovery and thus control (+) or not (–).

4.3.3 Problems, issues and comments

4.3.3.1 Analysis of case-marked postverbal direct objects

The model finds two solutions for the Finnish transitive clause *Minä ihailen Merja-a* 'I.nom admire.1sg Merja-par' (#83), a primary or standard solution (#83a) in which the direct object is merged to the complement of V, and another (#83b), in which it is initially right-adjoined and then reconstructed to the complement position. This solution corresponds to "extraposition" in the standard theory and is made possible by the polysynthetic free word property attested in Finnish. Furthermore, the reconstructed object appears to be in SpecVP in solution (#83b), but that is illusory because the "complement" is an adjunct and hence it is not part of the primary syntactic working space; the labeling provides that the argument is in a shared complement and specifier position in (#83b). This solution can be blocked by knocking off the polysynthetic profile from Finnish. Thus, it does not appear in the analysis of English or Italian (#89-103).

4.3.3.2 Valued φ-features inside grammatical heads

Results of Agree-1 can be seen in the log files and appear to be correct. It is important to note, however, that lexical, valued and conflicting φ-features that are present in a language such as English, and which can be seen from the lists of features associated with T/fin in the log file, still remain in the solution and appear at LF. Thus, Agree-1 does not remove φ-feature conflicts. In a sentence *They admire Mary* the verb does not exhibit "third person plural agreement"; the verb still has the conflicting φ-set which it acquired from the lexicon and which can be interpreted as licensing (and requiring a specification for) the third person plural subject. It is, of course, possible to rewrite Agree-1 in such a way that it removes conflicts when an overt subject is present.

4.3.3.3 Clausal antecedents and LF-recovery

A predicate with a complete unvalued φ-set (triggering "standard control") may take a clausal antecedent. When this happens, the result file lists either the clause as such as an argument (e.g., #109) or writes simply "clausal argument" (e.g., #105). The difference depends on the nature of the label of the clause (label = head provided by the labeling algorithm). If it contains valued phi-features, LF-recovery treats it (paradoxically, wrongly?) as a kind of DP because it is viewed as potentially providing values for the unvalued φ-set. This applied to the Finnish VA-infinitival, which exhibits these properties (overt phi-agreement). If the label does not have nominal features, then the recovery algorithm, as it stands now, cannot analyze it further and simply lists "clause argument" (e.g. #105). This does not apply to the situation in which only D remains unvalued: then a legitimate antecedent must be a DP. This correctly blocks Finnish third person null subject from having a clausal antecedent.

4.3.3.4 Analysis of Finnish infinitival complements

Most Finish infinitival complement clause constructions are analyzed as whole complement sentences with the embedded thematic subject inside the clause. This happens even if raw semantic intuition would suggest that the embedded subject should be a patient of the main clause (controlling a null argument inside the complement clause). The reason for this analysis is unambiguous and unequivocal syntactic evidence showing that the thematic subject is inside the complement clause. These construction contrast with ECM-style alternatives by all direct object tests, and the matter cannot be regarded as controversial. The implication is that raw semantic intuition cannot be used as grounds for structural analyses in such cases, and that a full thematic interpretation/theory must be able to finds (second order) "patients" also inside complement clauses, as shown in (2).

(2) Pekka käski [Merja-n_1 A/inf^0 $\_\_{}_1$ lähte-]
Pekka order Merja-gen -A leave-
--------->

Notice that because the algorithm uses the bare phrase structure, the embedded argument is still highly "visible" to the selecting verbs: there is no boundary of any kind, no grammatical head or specific phrasal features, between them.

4.3.3.5 Ambiguity in English control clause

The analysis of sentence #125 = *John wants Mary to leave* provides two solutions (a-b), one in which Mary's leaving is the object to John's wanting (a), and another in which Mary is the object of John's wanting while *to leave* expressed the purpose, analogously to *John wants the key [to open the door]* or *John wants the car [to leave]* (b). The model is insensitive to pragmatic plausibility calculations and does not see that (b) is very unlikely in this case. In the latter case, the antecedent for *to leave* is marked as "agent of T-event" (=John). This is because the solution is produced by right-adjoining *to leave*, and adjuncts (adverbials) are interpreted as providing properties of things denoted by grammatical heads with which they are linked with (by a mechanism not specifically addressed in this study but which is part of the algorithm, see full documentation). For example, a VP-adverbial is interpreted at LF/conceptual-intentional system as providing a property for the events denoted by V. The adverbial *to leave* is bound to T, and its antecedent will be the argument of T, which is what is reported in the results. But there is no rigorous compositional semantics, and this should be regarded as a heuristic estimation of what such a rigorous system should do.

Solution (b) is always available if two conditions are met: (1) the main verb can select for a DP-argument, which leaves the infinitival fragment hanging and causes the algorithm to find the right-adjoined solution, and (2) the infinitival fragment can be interpreted as an adjunct-adverbial in the first place. Whether some category is "adjoinable," thus not an adjunct by lexical specification, is a stipulated property that has to do with which categories can be interpreted as adjuncts at LF. In addition, the whole process of converting fragments of linguistics structure into adjuncts is a rather complex operation. It involves several nontrivial hypotheses and assumptions, such as what is an adjunct, how they are processed, how they are transferred to LF, how they are linked with other heads, how they are linearized, how much surrounding structure must be involved in the conversion, can only a single head be an adjunct, can the whole structure be an adjunct, and so on. I regard this matter controversial and subject to possibly major revision once a large and representative set of adjuncts (various adverbials, adjectives) are included into a test corpus.

Finally, in a sentence #129 = *John promises Mary to leave*, the verb takes a DP-complement, leaving the right-adjoined parse the only solution.

4.3.3.6 Postverbal object ambiguities (Finnish)

The ambiguity/extraposition issue raised in §4.3.3.1 is rather pathological here due to the number of possible adjunction sites. There is a general underlying scope issue that will be discussed later in this document. The model produces a curious successive-cyclic derivation in which an embedded object climbs to SpecCP and then is right adjoined to the main clause (#131c-d) to obtain "wide scope" at the spellout structure. Although such derivations are possible according to the standard theory I regard them implausible or at the very least something one might want to attend further in future research.

4.3.3.7 Ambiguities in to leave

The model again finds wide scope interpretations for *to leave* when it adopts the adjunction solution (e.g., #170b). This generates parses corresponding to (3)

(3) $John_1$ said that Bill wants Mary to resign, PRO_1 to convince everybody of Bill's evil intentions.

The interpretation is marginal, perhaps impossible, but I did not stipulate anything to rule them out, since they might still be possible. The wide scope issue remains unaddressed.

4.3.3.8 Double subject agreement in Finnish

The test revealed a problem in the analysis that occurs in a Finnish double subject construction of the type illustrated by (4)(#176-7).

(4) Merjaa minä ihaile-n.
Merja-par I.nom admire-1sg
'When it comes to Merja and me, I admire her.'

The model judges the sentence wrongly as ungrammatical, but still produces a candidate solution. It analyses (4) as a double subject construction in which both 'Merja-par' and 'I.nom' are inside the SpecT/finP. The analysis itself is plausible, in my view (I ignore the reasons). The nominative subject is reconstructed to SpecvP, where it will correctly Agree-1 with T/fin and value/check the first-person singular feature. But the derivation looks also to the edge for a valued D-feature. This would not be necessary a problem, because a copy of 'I.nom' remains. The problem is, rather, that the edge is examined in a top-down order, which targets 'Merja-par' which causes a phi-feature conflict (first person vs. third person). If we adopt the bottom-up order, then the same problem surfaces in the case of *Minä Merjaa ihailen* 'I.nom Merja-par admire' and in #177. I leave this issue for future and mark it as a problem in the formalization of Agree-1.

4.3.3.9 English accusative pronouns

The model does not react to the accusative case in English and therefore accepts sentences such as **John admires he* (e.g., #222) with the interpretation 'John admires him'. The issue would be trivial to fix for this particular example but fixing it by stipulation or ad hoc would mask the problem rather than solve anything. Masking it would have potential negative consequences, as the required stipulations would be carried over to further versions of the algorithm and the issue would remain hidden and cause possible unexpected consequences elsewhere (i.e. Italian, Finnish case assignment). The problem must be solved, instead, by developing a more general and empirically tested, explicit, formal and documented analysis for case assignment, including English case assignment.

4.3.3.10 Finnish free word order phenomenon

The main purpose of trying all logically possible word orders is to detect possible spurious parses and false positives, not to capture word order principles as such. The majority of noncanonical word orders are correctly judged as ungrammatical. Most verb-initial third person clauses in Finnish are ungrammatical because they are interpreted T-VP structures and hence there is nothing to satisfy the EPP. Verb-initial clauses are grammatical if the verb is interpreted as being in the C-position, in which case it should also carry phonological stress. This configuration was not included into the present study. Notice that if the verb exhibits full non-third person agreement, the model judges it grammatical because the pro-element satisfies the EPP (e.g., #371-372, 381, 383, 384). Sentences that have the genitive thematic argument in a noncanonical position are judged ungrammatical because in this study A-movement was assumed to be strictly local, and genitive arguments were reconstructed by A-movement. If the conditions for A-movement are loosened, then most of these sentences come out correct. The matter – proper definition for A-movement and the role of Finnish genitive arguments in it -- is controversial and was left unsolved.

4.3.3.11 Spurious solution (#425)

This problem was left into the study not only because it is nontrivial to correct but also because it illustrates the reason why all word orders must be tested. The problem involves the following pair, both which should be ungrammatical, but with (b) judged wrongly as grammatical by the model.

(5)

a. *Minä käsken lähte-ä. (Canonical word order, #423)
   I.nom order-1sg leave-A/inf
   '*I order to leave.'

b. Käsken minä lähte-ä. (Noncanonical word order, #425) order-1sg I.nom leave-A/inf

Comes out with interpretation 'I order one to leave.'

The model judges (5)a correctly as ungrammatical, because the main verb 'to order' requires that the complement VP contains an independent thematic argument. Because the argument is missing, (5)a is correctly judged as ungrammatical. What happens in (5)b, then, is that the noncanonically positioned grammatical subject is merged into SpecVP on the basis of its position in the linear order where it "accidentally" satisfies the SPEC-requirement of the A-infinitival head before being reconstructed into the thematic position SpecvP (6). The reason the grammatical subject must reconstruct in this way is because a word order in which the subject occurs in a right-ward position in relation to its canonical locality is generally possible in Finnish. In other words, this word order should be judged grammatical, the problem is that the combination of the main verb and the infinitival is not possible.

(6) [T/fin^0 [DP$_1$ [v V* [__$_1$ [A/inf^0 V$_{\varphi_=generic}$]]]]]

SEM:ext +ARG

The antecedent for the infinitival V comes out as "generic" because the LF-recovery cannot look past v* and does not see copies (__$_1$), i.e. positions in which the constituent had been earlier in the derivation.

The root of this problem is that here one and the same argument satisfies two conditions (thematic condition at SpecvP and EPP at SpecA/infP) which should involve two different constituents, and the solution must involve some extra mechanism that ensures that outcome. Notice that SpecA/infP is not a thematic position, so we cannot rely on the projection principle or some variation thereof (which is part of the current model). The chain (__$_1$, DP$_1$) is also legitimate, and the nominative subject is interpreted "as if" it indeed was in the SpecvP position into which it is reconstructed during the derivation. Although upward reconstruction (rightward movement in the standard theory) is possible in Finnish, it not possible to satisfy an EPP condition at the lower position (i.e. move rightward to satisfy any given head's EPP). I suspect that the problem comes to this latter issue, but how to capture it looks very nontrivial.

4.3.3.12 Spurious solution to leave wants John *in English*

The model accepts the sentence *to leave wants John* as grammatical with the meaning 'the event of one's leaving wants John', analogously to 'to leave would be a mistake'. It is clear, however, that the event of one's leaving is not or cannot be interpreted as a sentient being and cannot want anything. There are a variety of ways to rule them out, including lexical features, but not trivially because infinitivals can occur at SpecTP.

4.3.3.13 *Spurious solution* John wants to Mary leave (#517)

This sentence is analysed as grammatical with the structure [John wants [to [Mary leave]]] and with the interpretation 'John wants Mary to leave'. The reason this input is wrongly judged as grammatical is because *want* occurs both in -ARG and +ARG environments, so that the infinitival 'to' cannot have either marking. Here, then, the surface word order puts Mary directly in the correct thematic position, and the result is of course accepted at LF-interface. This problem is possible related to misalignment between syntactic structure and thematic role assignment, observed in Finnish. In Finnish, syntactic evidence shows, without doubt, that in many constructions of this type the embedded thematic subject is inside the infinitival clause and is not part of the main clause, yet semantic intuition suggests that it is thematically visible to the main clause as well and behaves like an independent patient. In the sentence *John wants to Mary leave* the embedded subject could be embedded too deeply to be visible for main clause thematic role assignment. But this is only one possibility and, if true, requires an implementation of the thematic theory. Lacking any obvious solution, I decided to leave this issue unsolved. Notice that the control properties are computed correctly.

4.3.3.14 Noncanonical subjects

The model accepts sentences such as *Merjaa ihailee* 'Merja-par admire-3sg' due to the problem with double subject agreement (§4.3.3.8). The model thinks, wrongly, that *Merjaa* 'Merja-par' counts as a third person subject.

4.3.4 Summary of the results

Two problems and three issues were left for future research. The two problems were spurious parsing solutions that were generated from a noncanonical word order both in English and Finnish. In English, the model accepts sentences such as *John wants* [*to Mary leave*], with Mary generated directly to the canonical thematic position from the PF-input. An interpretation was 'John wants Mary to leave'. It is not obvious what the solution is, and the problem itself depends on many auxiliary assumptions (analysis of the infinitival, and so on). Notice that any candidate solution must solve the inverse problem as well, which makes the issue nontrivial even if there were a solution available that works on paper. In Finnish, the model accepts sentence *Käskin minä lähteä* 'order.1sg I.nom to.leave', with a curious analysis in which 'I' satisfies the SEM:external property of the main verb 'order' inside the embedded structure and is them reconstructed into SpecvP inside the main clause. The sentence is interpreted as generic 'I order one to leave.' I could not formulate an immediate solution for this problem but suspect that it occurs because the thematic component, as formulated in the current model, is insufficient.

The three issues were the following. First, the algorithm does not understand English accusative case and accepts *John admires he* with the meaning 'John admires him'. This was ignored here because it requires a systematic approach to structural case assignment, something the model still lacks. Second, the formalization of Agree-1 is still insufficient as it does not handle exceptional Finnish double subject constructions correctly, picking a wrong constituent to agree. These are constructions that have two subject-like elements at SpecTP position. The construction, and its analysis, is controversial, so the matter was left for future research. Third, the algorithm accepts sentences such as *to leave wants John*, with the meaning in which 'one's leaving is wanting John'.

References

Chomsky, Noam. 1965. *Aspects of the Theory of Syntax*. Cambridge, MA.: MIT Press.

Phillips, Colin. 1996. *Order and Structure*. Ph.D. Dissertation. MIT