Unambiguous paths and the theory of syntactic working memory

Manuscript, 2022

Abstract. A mathematical reformulation of Kayne's (1983, 1984) unambiguous path hypothesis is proposed as a unifying concept for capturing grammatical dependencies involved in control, selection, EPP, adjunct attachment, binding, Ā operator scope, Ā/A reconstruction, extraposition, agreement (Agree) and case assignment. The proposal is formalized as an algorithm and tested with a large corpus by deductive calculation. A further hypothesis is then considered that identifies the notion of unambiguous path with syntactic working memory and interprets the proposed algorithm as a memory scanning operation. The resulting synthesis could help integrate formal-computational linguistic theories with a broader cognitive theory of language. Data from English, Italian and Finnish is considered.

Keywords: Unambiguous paths; control; binding; agreement; case assignment; operators; adjuncts; syntactic working memory

1 INTRODUCTION

This article argues that Richard Kayne's (1983, 1984) notion of unambiguous path provides a possible avenue for unifying a number of linguistic phenomena. The core idea of Kayne's hypothesis is to substitute the standard top-down c-command (more recently, probe-goal) model with a bottom-up path that runs from the target element to an antecedent via upward dominance. Anaphora, for example, are linked with their antecedents by unambiguous paths that connect the anaphor to its antecedent(s) by following dominance relations (see Kayne 1984, principle (3), p. 131). Here we propose a mathematically precise reformulation of Kayne's original analysis such that it can be implemented as a feasible algorithm over minimalist bare phrase structure and tested by calculation. Let us assume that a head, called the *probe*, can initiate a grammatical operation targeting some head we call the *goal*. To establish a probe-goal dependency, we form a *path* from the probe to the goal (1) over a binary branching bare phrase structure.

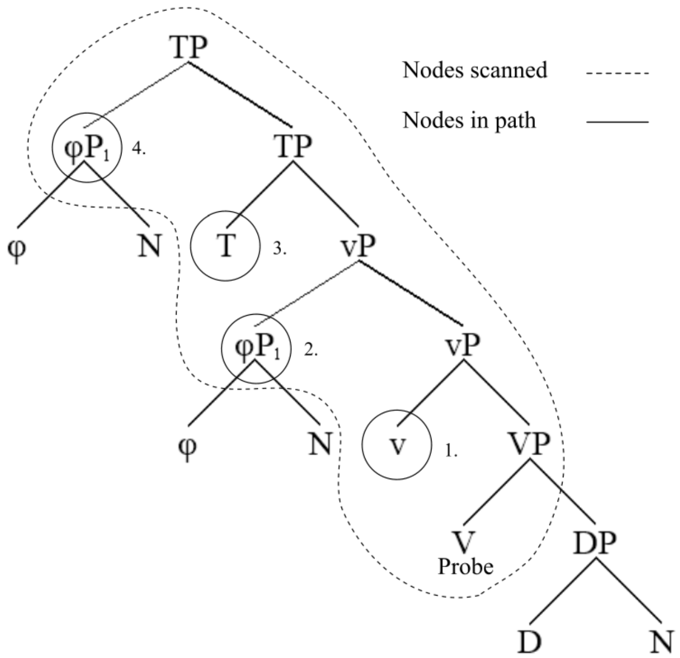
(1) *Unambiguous path (informal version)*

a. Path is an ordered list L of daughters D(C) of constituents C such that C dominates the probe while D(C) does not;

b. L is well-ordered by *search* that terminates if and only if (i) an intervention occurs or (ii) there is no more structure.

Condition (1) defines a linearly organized set of phrase structure objects that are “accessible” from the probe and contains the goal. The idea is illustrated in (2).

(2)



The nodes contained within the “nodes scanned” are the ones that are visited by the algorithm (C in the definition of (1)), whereas the nodes “in path” are contained in the final list $L = D(C)_1, \ldots D(C)_n$ where they appear in the order generated by the search algorithm. All paths are unambiguous because there is at most one mother-of relation for any head and the daughter that was used to reach the mother is not included in the path per (1). All complex constituents are binary branching, so the unambiguous case $[_{\alpha P}$ X Y Z…] never arises. Thus, the system can be implemented by a trivial algorithm that does not consume time or memory. Search can end in one of two ways: either there is no more structure or an intervention occurs. Intervention is defined by a well-defined grammatical property, often by probe's lexical features. The actual computer code defining (1) is discussed in the technical appendix.[*] The intuitive interpretation of (1) is that it represents a "memory scanning" operation locating goals from the active working memory. I will argue that left branches and right adjuncts, both invisible for (1), are not inside the same active syntactic working memory as the probe.[1] This makes it possible to propose that the notion of upward path coincides or is identical with the contents of active syntactic working memory. Finally, the notion of *edge* will be used frequently in this article. It refers to the part of the path that resides inside the probe's own projection.

First we reduce several grammatical principles and mechanism to (1). This step is summarized in Table 1. The leftmost column contains the names of the grammatical mechanisms we target for reduction. The next two columns tell what the probe and goal will be in the proposed end point, and the rightmost column contains a brief comment on how the reduction will be accomplished.

Table 1. Unification of grammatical principles by Kayne's upward path.

#	GRAMMATICAL PHENOMENON	PROBE	GOAL	UNIFICATION BY (1)
1	Head-spec selection (several mechanism)	Head	Specifier	Calculates if a specifier selection feature of a head can be satisfied by an element inside upward path containing only head's own projection (=edge)
2	θ-marking	Head	Thematic specifier	Verifies if a head theta-marks a phrase inside upward path containing only head's own projection (=edge)
3	Selector	Selectee	Selector	Closest primitive head in upward path
4	XP^{max}	X	Highest element inside XP^{max}	Derivative of the notion of edge
5	Case assignment	Case assignee	Case assigner	Closest complete case feature match inside upward path
6	A-chain creation	Head	Noncanonical phrase at specifier	Establishes a dependency between head and a phrase inside upward path containing only head's own projection (=edge)

[*] The principle is on lines 330-346 in the source code file https://we.tl/t-wcKw0zFktd. These files are provided for review purposes and they expire in 12 months.

7	Head chains	Head being reconstructed	Selector	Determines the nature of the selecting head, needed in some instances of head chain creation
8	Agreement	Head requiring agreement	Phrase providing agreement	Closest agreeing element inside upward path containing only head's own projection (=edge)
9	Ā-chain creation	Head with criterial feature	Noncanonical Phrase at specifier	Establishes a dependency between head and a criterial phrase inside upward path containing only head's own projection (=edge), such as [*who* [C(*wh*) …]]
10	Operator binding	Operator	Scope marker	Closest suitable scope marker (or several if ambiguous) inside upward path
11	Control	Null argument	Antecedent	Closest possible antecedent inside upward path
12	Binding	Referential expression	Antecedent	Calculate relevant referential expressions considered as possible antecedents inside upward path (cf. c-command), with search limited by locality factor (e.g., *him*, *himself*)
13	Adjunct processing	Adjunct head	Specifier	Several mechanisms
14	Locality boundary	Head	Boundary	Closest head with relevant features (e.g., finiteness, tense) inside upward path
15	Information structure, marked word order	Dislocated projection	Starting point	Examine if the starting point is inside upward path to determine information structural interpretation (focus, topic)
16	Extraposition	Bottom head	Extraposition point	Closest extraposition point inside upward path

We prove that (1) can unify the grammatical phenomena in Table 1 by establishing that it deduces them. One way to do this is to formalize (1) together with other theoretically motivated grammatical operations (e.g., Merge, Move, Agree) and have the aggregate system generate expressions that are matched with native speaker intuitions in terms of grammaticality, syntactic analyses and semantic interpretations. This amounts to an enumerative grammar. A formally equivalent way is to map input sentences into grammaticality judgments, syntactic analyses and semantic interpretations. Under this scheme, lexical elements arrive from the input in a well-defined order and are processed by the proposed grammatical principles and a simple parser handling ambiguities. This corresponds to a recognition grammar. Enumerative and recognition grammars are generative grammars: they define a set of grammatical sentences in a way that can satisfy observational, descriptive and explanatory adequacy. I use a recognition grammar in what follows.

2 METHODS

2.1 Stimuli

We test (1) against sentences from English, Italian and Finnish that exhibit the linguistic properties listed in Table 1. The input contains 2211 sentences, representing approximately the same amount of construction types.[2] The original file is available online.[†] The contents of the test corpus are summarized in Table 2.

Table 2. Summary of the test corpus, with group numbers (left), sentence identifiers (middle) and descriptions (right column).

GROUP	#	DESCRIPTION
0	1-32	**Sentences cited in this article**
1	33-263	**Basic construction types**
1.1		Basic verb classes
1.1.1		Intransitive verbs
1.1.2		Transitive verbs
1.1.3		Ditransitive verbs
1.1.4		Ditransitive verbs plus PP argument
1.1.5		Two PP arguments
1.2		Special finite elements
1.2.1		Auxiliary-like negation
1.2.2		Modal constructions, Neg + Modal + V
1.2.3		Pure tensed auxiliaries
1.3		Clausal infinitivals
1.3.1		Clausal infinitivals in English
1.3.2		English OC-constructions
1.3.3		Finnish clausal infinitivals (A/inf, VA/inf)
1.4		Nominals
1.4.1		Basic DP constructions
1.4.2		N + clausal infinitival
1.5		Adpositions
1.5.1		Prepositions, postpositions
1.6		Embedded *that*-clauses
1.6.1		Embedded *that*-clauses in Finnish and English
2	264-306	**Adjuncts and adjunction structures**
2.1		PP adjuncts
2.1.1		Postverbal PP adjunct constructions
2.2		Adjectives
2.2.1		DP-internal adjectives (Finnish, English)

[†] Corpus file for review purposes: https://we.tl/t-axBYZ77dcu.

11.2.6	Possessive pronouns
11.2.7	Pronouns and conversation
11.2.8	Human vs. nonhuman pronouns
11.2.9	C-command tests
11.2.10	Pronouns and null subject sentences
11.3	Reflexive pronouns (Condition A)
11.3.1	Grammatical, assignment possible
11.3.2	*Ungrammatical, gender mismatch, assignment not possible
11.3.3	*Ungrammatical, human mismatch
11.3.4	*Ungrammatical, subject reflexives, assignment not possible
11.3.5	Reflexives in conversation
11.3.6	C-command condition
11.3.7	Reflexives and control
11.3.8	Ungrammatical reflexive binding from that-clause
11.3.9	Reflexives and null subjects

Input sentences were lists of linearly ordered unanalyzed phonological words without morphosyntactic tagging. Morphological analyses and decompositions were done by the algorithm. In some cases the word was disambiguated in order to avoid superfluous calculations, but disambiguation had no impact on the results or analyses.

2.2 Procedure

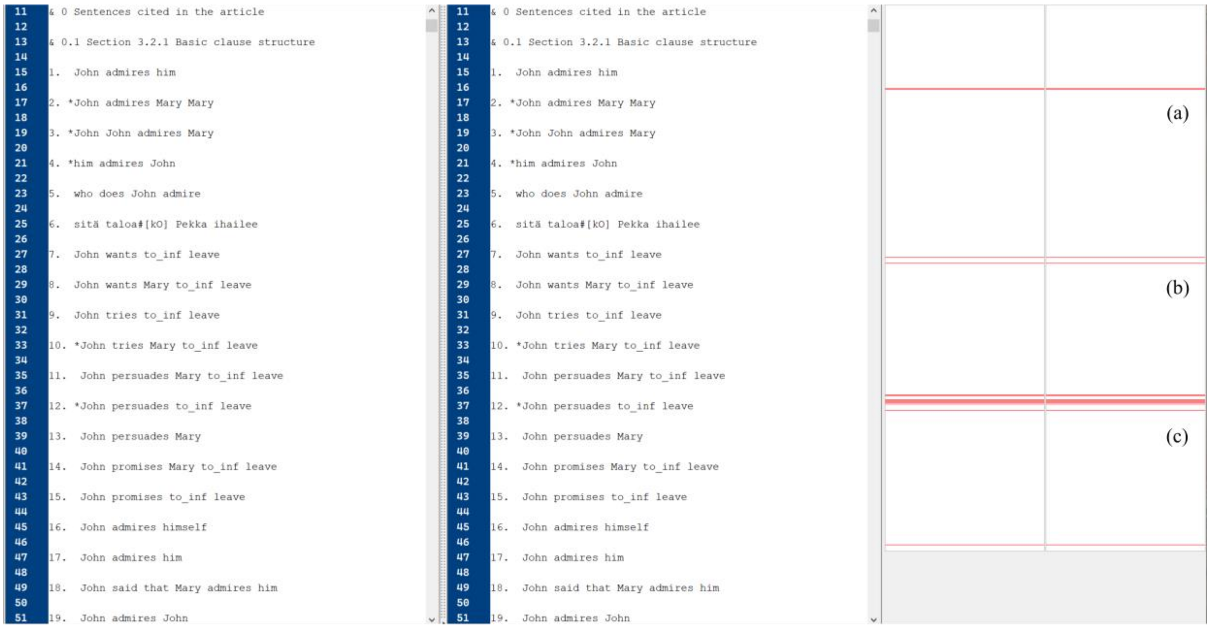
A simple recursive minimalist sentence processor implemented in Python (a general-purpose programming language) mapped input sentences into sets of binary branching bare phrase structure parses that were submitted for semantic interpretation (Brattico, 2019). Grammatical operations listed in Table 1 were then defined by (1). The model processed all sentences from the input corpus without human intervention and linked each with an output containing calculated grammaticality judgments, syntactic analyses, and semantic interpretations.

The plausibility of the research hypothesis was verified by examining the correctness and plausibility of the calculated output, for example, whether binding, case and control configurations calculated by the model were correct and/or at least plausible. Observational adequacy was assessed by comparing grammaticality judgments calculated by the model with those provided by a native speaker, by using an automatic file-comparison tool. Descriptive adequacy was assessed by looking at the syntactic and semantic analyses provided by the algorithm and assessing to what extent the former are plausible from the point of view of a linguistic theory and to what extent the latter agree with native speaker intuitions. I will ignore psycholinguistic adequacy and the performance metrics; the latter are available online in raw format.[‡] I will evaluate the output within the context of a fairly standard minimalist grammar (Chomsky, 2000, 2001, 2008). The output files are available online.[§]

3 RESULTS

3.1 Observational adequacy

Observational adequacy was assessed by comparing the grammaticality judgments provided by the model[**] with a file where grammaticality was assessed by a native speaker (author)[††]. Comparison was performed with a mechanical file-comparison tool. The results are shown in Figure 1.



A. Grammaticality judgments calculated by the model

B. Grammaticality judgments by native speaker

Discrepancies between A and B

[‡] https://we.tl/t-R0Q9Tt3oZL

[§] Grammaticality judgments by the model (https://we.tl/t-co70fDjlyH), results of the calculations (https://we.tl/t-tWmKI9hOvf); derivational log file, compressed (https://we.tl/t-aBhpZWeR7b); resource metrics (https://we.tl/t-R0Q9Tt3oZL); grammaticality judgments by native speaker (https://we.tl/t-Cl0p6XPVRP).

[**] https://we.tl/t-co70fDjlyH

[††] https://we.tl/t-Cl0p6XPVRP

Figure 1. Comparison between model and native speaker outputs, with differences shown by the lines inside the rightmost column. The image is a screenshot from the program where the mechanical comparison was done. Notice that the two leftmost columns contain only a tiny slice of the whole test corpus file (19 sentences); the rightmost column covers the whole file. Three error clusters (a-c) emerged, with 13 errors in total. Clusters (a) and (c) contained sentences involving successive-cyclic A-chains in connection with genitive-marked referential arguments in Finnish. Cluster (b) contained phi-concord errors.

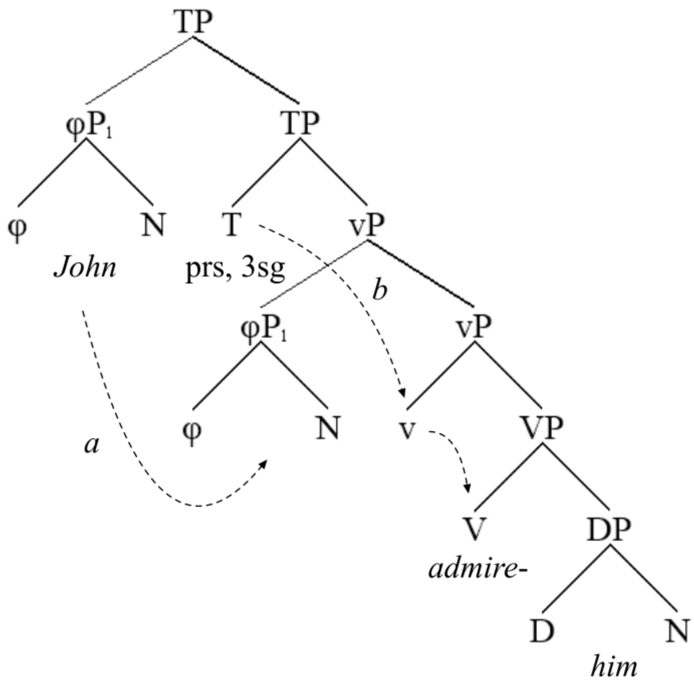
The model reached 99.4 % accuracy over this dataset (13 misjudgments out of 2179). Therefore, the model is sufficiently on target to merit evaluation of descriptive adequacy. Problems (a-c), which are irrelevant to the main hypothesis, are discussed in the technical supplementary document.

3.2 Descriptive adequacy

Descriptive adequacy was assessed by examining the calculated output within the context of a relatively standard minimalist theory and native speaker intuitions concerning semantic interpretation. The whole calculated output exists in several separate files associated with this study (see note §) and can be examined independently. This is important because verification cannot be automatized. There is also a technical supplementary document that goes into details that cannot be included into the main article for space reasons, including instructions on how to read the output files. I will use the format #n when referring to the numerical identifiers of the test sentences appearing in the calculated output.

We begin by examining how the upward path mechanism operates in connection with simple canonical finite sentences. The calculated analysis for a simple English transitive sentence *John admires him* (sentences #1, 91 in the test corpus; intransitive constructions are sentences #33-77) is shown in (3).

(3)



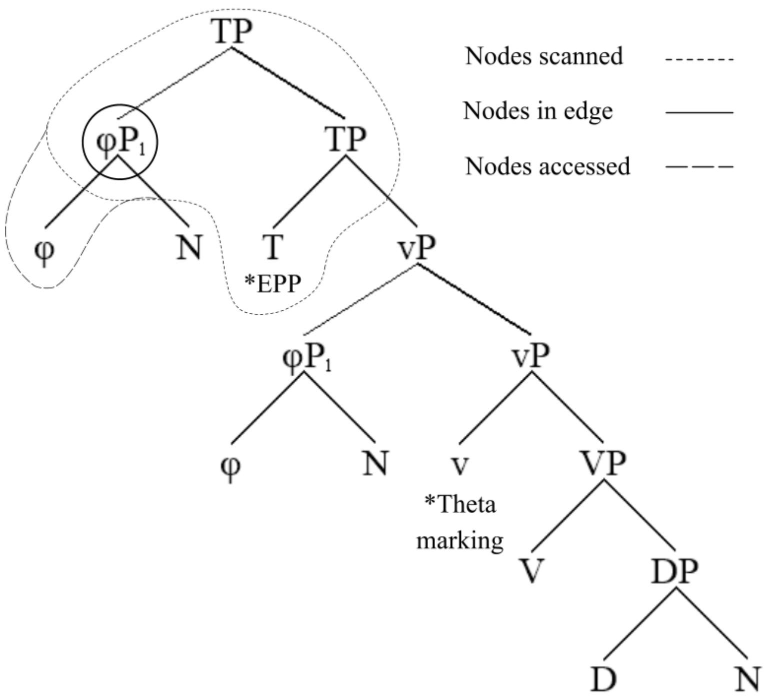
Before examining the output further it is perhaps useful to recapitulate that the underlying algorithm implements a recognition grammar that generates these outputs from linear sequences of phonological words, therefore the chains are produced by reconstruction rather than movement. For most purposes reconstruction can be thought of as reverse-engineered movement; I will therefore ignore the difference and use the neutral term “chain.” A chain is a set of member constituents and has no direction.

The phrase structure geometry and the labels shown in this image were generated by the algorithm, thus they constitute raw data[‡‡]; the rest (chain arrows, words) were added by the author for illustration. Two chains are generated: an A-chain connecting the grammatical subject with the thematic position SpecvP (a) and a head chain relating T, v and V of the finite verb *admires* (b). The detailed computational steps involved in the creation of these chains, ignored here, can be found from

[‡‡] All 1775 phrase structure analyses are available in image (PNG) format in one compressed (zipped) file https://we.tl/t-2liKVSTQPe. There are 1775 phrase structure analyses in this batch because it excludes ungrammatical sentences but provides several analyses for ambiguous sentences.

lines 44265-44282 in the derivational log file[§§]. The two specifier member positions of the A-chain, SpecTP and SpecvP, are calculated by relying on the upward path mechanism such that the lexical features of T and v are the probes, the heads of referential φPs the goals (the results would be the same were φPs replaced with DPs[3]). In the case of T, the probe is a generalized EPP feature; in the case of v, it is a thematic 'agent' specifier selection feature. When the probe activates the respective grammatical mechanisms, the edge was scanned for the goal.[4] Thus, v can only see inside vP; T inside TP, and so forth. These assumptions are illustrated in (4) in connection with T.

(4)



The fact that scanning terminates at maximal projections (e.g., TP for T) constitutes an intervention factor in the sense of (1b). The reason I include the head of the referential argument *John* into the "nodes accessed" is because the chain creation algorithm (not the path mechanism itself) needs this information in order to calculate correct dependencies (vP for example, cannot check the English EPP feature). The underlying phrase structure model provides a unique and unambiguous head/label for all

---

§§ https://we.tl/t-aBhpZWeR7b

phrases. All head-specifier dependencies were reduced. There are several examples of A-chains in the test corpus, including successive-cyclic A-chains (see, e.g., sentences #91-96, 155-161 and sentence (8) in this article).

The head chain (T, v, V) relies on the upward path mechanism when it computes local head selection dependencies. The system needs to determine if two local heads establish a legitimate grammatical configuration X^0–Y^0 (e.g. want + to leave) and does this by locating the closest X^0 (goal) inside the upward path from Y^0 (probe) and by verifying that X^0 can or must select Y(P). Both were implemented by using paths.[5] Head chains are tested systematically by sentences #1661-1922. As shown by the large number of sentences included into this batch, the tests were rather exhaustive. Some constructions involved variations of T-to-C head chains, illustrated below (with ungrammatical variations #1694-1761)[6]:

(5) a. Istu-u-ko$_1$ Pekka __$_1$! (T-to-C, #1661-1668, 1679-1680)
sit-PRS.3SG.Q Pekka.NOM
’Does Pekka sit!’

b. Ei-kö$_1$ Pekka __$_1$ ihaile Merja-a. (Neg/Aux-to-C, #1669-1676, 1681-1683)
not-Q Pekka-NOM admire Merja-PAR
‘Pekka does not admire Merja?’

c. Täytyy-kö$_1$ Peka-n __$_1$ pestä vaatteet? (Modal-to-C, #1677-1678)
must-Q Pekka.GEN wash clothes
’Must Pekka wash the clothes?’

Head chains were also tested with several left peripheral operator features and their combinations (#1878-1922). The details were left for the technical supplementary.

English transitive clauses included into this study are sentences #91-93, #123-126. Also ditransitive #127-135 and intransitive + PP constructions #136-143 were tested. Sentences #144-146 tested finite clauses with two PP constituents. The *projection principle* is of special interest. It ensures that all referential arguments are assigned a thematic role. The system relies on functions checking that referential arguments occur under appropriate theta-marking configurations. These mechanisms rule out (6) and many others like it (#43-77, 97-126).

(6) a. *John admires Mary Mary (#2)
b. *John John admires Mary (#3)
c. *Sleeps. (#73)
d. *John sleeps Mary. (#75)
e. *John John sleeps. (#76)
f. *John gave the key to. (#131)

There are two legitimate theta-marking configurations: a head-complement configuration and head-specifier configuration. The latter makes use of the edge mechanism, as elucidated above; the head-complement dependency relies on the notion of maximal projection that is defined by relying on the notion of edge.[7] Thus, the projection principle relies on the upward path mechanism. Calculations that involve head-complement dependencies were implemented in the original algorithm by relying on minimal search in the sense of Chomsky (2008). The original minimal search algorithm was not modified in this study.

Case assignment makes use of the upward path mechanism by requiring that the case assignee (probe) links with an appropriate case assigner (goal). Mainly Finnish is used in this study to test these mechanisms (sentences #877-984), since the empirical footprint of Finnish case assignment is considerable and thus useful for our purposes.[8] In the case of *John admires him* (#1, 17) , the pronoun has accusative case (probe) that is checked against v (goal)(7). Nominative case is checked against T. This mechanism rules out sentences such as **Him admires John* in the dataset (#4, 2094-2101).

(7) [$_{TP}$ *John* [$_{TP}$ T [$_{vP}$ (*John*) [$_{vP}$v [$_{VP}$ V *him*]]]]
G ← P G ← P

The operation is triggered by the unchecked case feature. The closest potential case assigner limits the search, so that the nominative case cannot be checked by finite T from CompVP position over the small verb v. If a case assignee cannot check case, the derivation crashes. Overall, since the model was observationally adequate, it filters out ungrammatical case configurations while letting the grammatical ones pass. This supports the claim that the upward path mechanism can handle structural case assignment.

Also agreement makes use of the upward path mechanism, although in a more limited way. In most instances, a head with unvalued phi-features agrees with a referential phrase inside its sister, making reduction into (1) impossible. There are, however, examples in the dataset which require a head to agree with an argument at its edge, such as clitic-participle agreement in Italian (#1936-2001) and some others discussed below. The edge mechanism was again used. Ideally all head-specifier agreement should be eliminated, but I was not able to accomplish this.[9] Grammatical and ungrammatical agreement configurations were tested in Finnish (#985-990, 1006-1035), English (#991-998, 1036-1037) and Italian (#999-1005, 1038-1041). Agreement was also tested against noncanonical word orders (#1042-1257).

The test corpus contains transitive and intransitive negative sentences with and without auxiliary verbs (#147-165, #185-188). Modal constructions were sentences #169-179, and combinations of modals with negation and auxiliary are #180-184. When the sentence contains a verb sequence, a successive-cyclic A-chain relating the grammatical subject to its thematic position (8) is typically generated.

(8) [Loro$_1$ [__$_1$ hanno [__$_1$ volute [__$_1$ lavare Luisa]]]] (#160)
they have wanted to.wash Luisa

Let us consider operators next. Several aspects of Ā-chains and operator constructions were reduced to the upward path mechanism. First, operators are linked with their thematic positions by Ā-chains (9).

(9) [Who$_1$ [does [John admire __$_1$?]]] (#5, for Finnish see #307-324)

The interrogative phrase at SpecCP is targeted by the *wh* probe at C hosting the auxiliary via the specifier-head dependency that relies on the path mechanism. The thematic landing site is determined by specifier-head and head-complement relations, the former which relies on specifier-head dependencies and hence upward paths. For several variations in English, see #325-335. Constructions of this type are interpreted at the LF interface by linking the canonically positioned operator with a scope marker at closest finite head with the same operator feature, where the operator and the scope marker are linked by an upward path (10).[10]

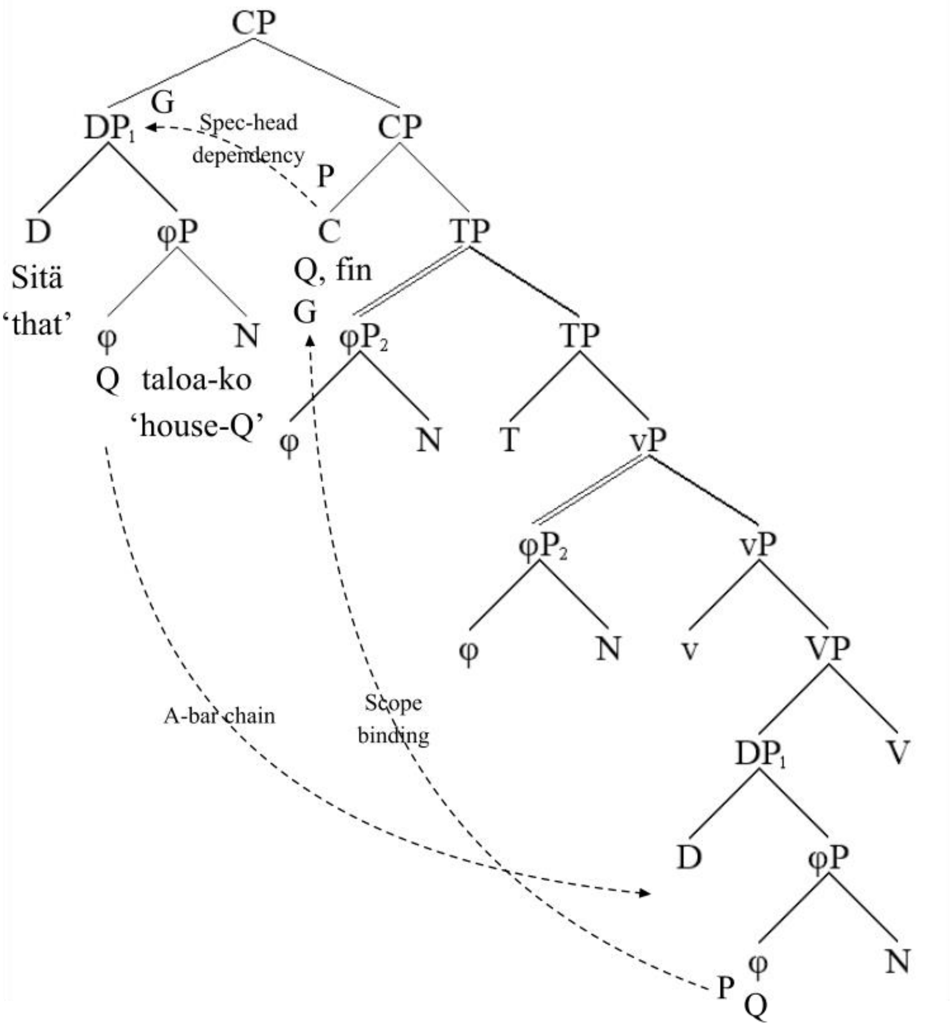
(10) [Who [does John admire (who)?]]
wh, fin wh
G P
‘Which x: John admires x?’

This dependency is shown in the output, where it is reported that the pronoun containing the *wh*-operator is bound by a finite head containing a copy of the same operator (lines 60-61 in the results file). It represents scope information. The system calculates also pied-piping (#336-338, 490-583, 631-662). This requires an additional function that “scans” criterial features from phrases. This means that the specifier-head dependency [$_{CP}$ who [$_{CP}$ C(*wh*)…]] illustrated in (9) holds between C and a phrase inside its edge, whereas the criterial feature matching requires a special scanning mechanism. This is illustrated by (11)(#6), where the criterial feature, a yes/no particle -*ko*, is embedded inside the dislocated phrase.

(11) [$_{CP}$[$_{DP}$ Sitä taloa-**ko**] [$_{CP}$ C [Pekka ihailee __?]] (#6, Finnish)
that house-Q C(Q) Pekka admire.PRS.3SG]]
‘Was it that house (and not that car) that Pekka bought?’

The analysis calculated by the model for (11) is (12).

(12)



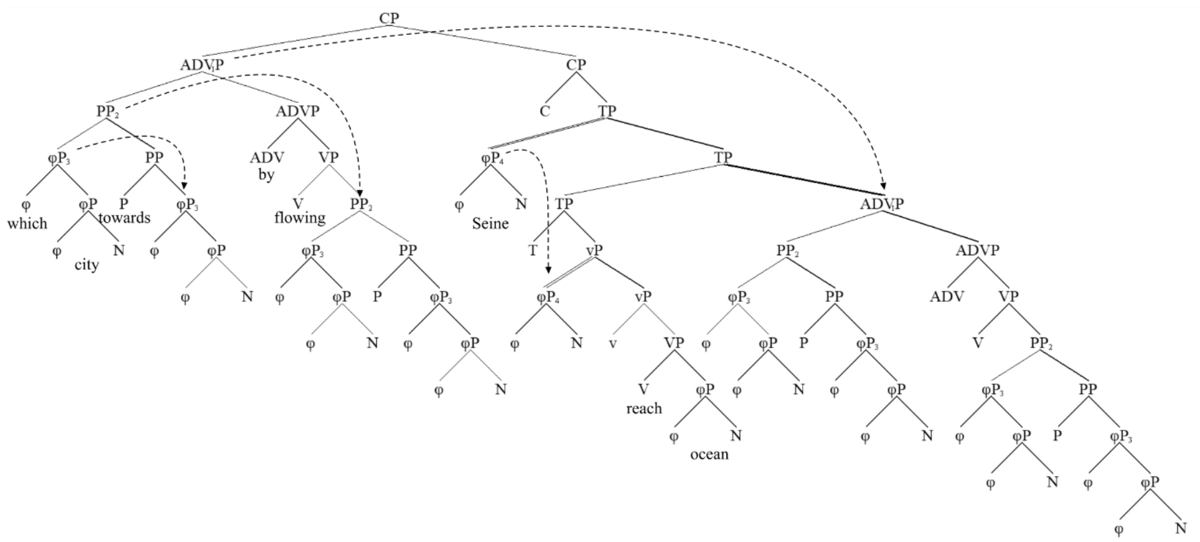
This is not particularly controversial, but it requires we access left branches even deeper than the head. The DP hosting the Q operator is bound to the scope marker C(Q) at its canonical position at the LF interface by an upward path ("scope binding", see note 10). Ā-chain constructions were collected into Group 3 (#307-876) in the test corpus which tested basic interrogatives (#307-327), several ungrammatical variations (#328-335), pied-piping in connection with secondary Ā-chains (#336-338), echo-interrogatives and in situ contrastive focus operator sentences (#339-342, 848-876), islands (#343-361), different left peripheral operator features in Finnish (#362-489), variations of pied-piping (#490-562), sentences where the operator features were scattered among several words (#563-583), operators in connection with noncanonical word order (#584-662), embedded interrogatives and operators with and without pied-piping and noncanonical word order (#663-820) and selection tests

(#821-847). The model can also derive long-distance wh-movement (e.g., #352). Finnish secondary Ā-chain formation (13) merits special mention.

(13) [[Mitä kaupunki-a kohti] virtaamalla] Seine saavuttaa valtamere-n? (#338)
what.PAR city-PAR towards by.flowing Seine reaches ocean-ACC
'By flowing towards which city does Seine reach the ocean?'

Notice the inverted word order when compared to English. A declarative Finnish clause uses the same order as the English sentence *Seine reaches ocean by flowing towards Paris.* Word order inversion is due Heck's edge generalization (Heck, 2008, 2009) which applies to Finnish (Huhmarniemi, 2012) and requires wh-elements to occupy the edge positions of pied-piped phrases. If the wh-element does not occupy the edge position canonically, an Ā-chain is generated (Huhmarniemi and Brattico, 2013). Thus, sentences like (13) involve several embedded internal Ā-chains that the model calculates correctly (14).

(14)



Even if the head carrying the wh-feature ('which') is always moved to the edge, it does not appear at the head of the phrase (AdvP) that occurs at the final landing position, once more showing that deeper access is required. The model calculates secondary Ā-chains correctly because left branches constitute phrases, thus the secondary chain [$_{PP}$ *which city*$_1$ *towards* __$_1$] is generated when the PP is processed as a phase, and then the same procedure is applied to the next layer [$_{\text{AdvP}}$ PP$_1$ *by.flowing* __$_1$] and so on (Bianchi and Chesi, 2006; Huhmarniemi, 2012; Brattico and Chesi, 2020).

Calculations involved in control were reduced to the upward path mechanism by assuming that predicates (as defined by argument placeholders corresponding to PRO elements) are linked with antecedents (goals) at the LF interface such that the antecedents occur inside probe's upward path, and such that the closest antecedent is targeted, a proposal that goes back to Rosenbaum (1967). Some of the resulting control dependencies are shown in (15)-(17).

(15) a. John$_1$ wants to leave$_1$. (#7, 1293; ungrammatical variants #1295-1297)
b. John$_2$ wants Mary$_2$ to leave$_2$. (#8, 1294)

(16) a. John$_1$ tries to leave$_1$. (#9, 1299)
b. *John tries Mary to leave. (#10, 1302)

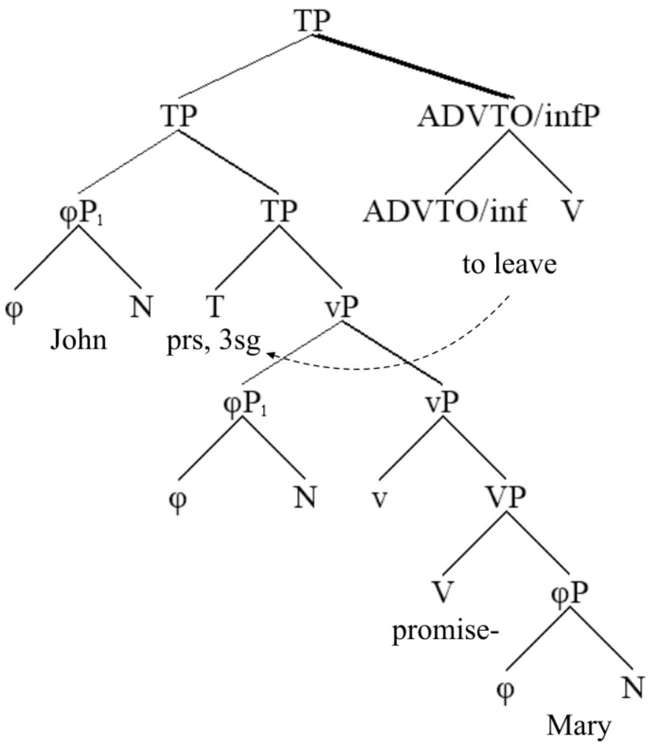
(17) a. John$_1$ persuades Mary$_2$ to leave$_2$. (#11, 1305, 1311)
b. *John persuades to leave. (#12, 1307, 1312)
c. John persuades Mary. (#13)

The model analyses these sentences as embedded clause constructions where the embedded *to*-clause is complemented to the main verb, e.g. [*John wants* [$_{\text{InfP}}$ *to* V]] (see #189-200). The embedded subject participates in a two-member chain [*John wants* [$_{\text{InfP}}$ *Mary*$_1$ [$_{\text{InfP}}$ *to* __$_1$ V]]]. The selecting main verb controls the head-specifier behavior of the embedded infinitival head (expressed as *to* in English), which results in the profiles (15)-(17): the *want*-class accepts both options (15)(for Finnish, see #1298), OC-constructions exclude embedded subjects (16), while the *persuade* class requires them (17). We rely on the upward path mechanism to connect the infinitival with the selecting predicate (e.g., *persuade*/*want*/*try* + *to leave*) when the appropriate profile is selected. Consider (18) next.

(18) a. John$_1$ promises Mary$_2$ to leave$_{1,*2}$. (#14, 1310)
b. John$_1$ promises to leave$_1$. (#15, 1300)

The model calculates correct control dependencies shown above, but the reason is nontrivial. The infinitival clause *to leave* is adjoined to a higher position in the clause, *Mary* remains at the direct object position (19).

(19)



Due to its high position in the clause, the infinitival cannot see *Mary* inside its upward path and cannot select it as an antecedent; it sees referential agreement features at T that are listed as its antecedent in the output. This mechanism is required on independent grounds in order to connect predicates to pro-subjects in languages that exhibit pro-drop. An alternative would be to adjoin the infinitival to the vP. Adjunction is discussed later. Two functions calculate these data, the first creates the path and observes an intervention factor, the second picks up the first suitable antecedent from the result.[11] Control constructions are sentences #1290-1317 in the test corpus testing the four verb classes discussed above plus finite control in Finnish (discussed further below) and control sentences involved in generic interpretation and generic null subjects. Finnish infinitival control is somewhat understudied at present and controversial (Vainikka, 1989; Koskinen, 1998; Brattico, 2017; Kiparsky, 2019); Finnish analyses generated by the model can be seen from #201-211 and #1303-1304, 1306, 1308, with #212-225 showing infinitivals combined with modals and negation. Control into adverbial phrases are #1309-1310. Control was also tested against word order variations (#1444-1583, 1586-1619). There were three problems. First, the syntactic background model assumes that also complement arguments are linked to their predicates by the generalized control mechanism. Since complements are not inside upward paths, this special condition had to be stipulated. The second complication has to do with long-distance finite control that appears in the dataset, discussed later in this section. Third, the model calculates properties of one generic object control construction incorrectly (#1313). The original sentence *Pekka pyytää lähtemään* ‘Pekka asks (people) to leave’ was analysed wrongly as an OC-construction ‘Pekka asks (himself) to leave.’

Binding principles regulate referential assignments connecting referential expressions to their denotations in the language-external global discourse inventory representing what the transient conversation “is about.” Referential expressions contain lexical features which guide assignment management by restricting the search space. Phi-features, for example, direct denotations into semantic objects that have compatible properties (e.g., [first person] ~ ‘speaker’, [masculine] ~ ‘male’). It was assumed in the existing model that R-expressions, pronouns and reflexives contain lexicalized assignment management features which restrict the search space in accordance with the binding principles A-C (Chomsky, 1981). This mechanism was reduced in this study to the upward path system, following closely Kayne’s original idea. Admissible coreference relations for R-expressions, pronouns and reflexives are computed on the basis of referential expressions visible in the probe’s upward paths. A probe-goal dependency of this type is illustrated by (20).

(20) a. John$_1$ admires himself$_{1,*2}$. (#16)
G P

The reflexive pronoun contains a lexical feature which imposes a coreference requirement with the closest referential expression inside the upward path during assignment generation. In the case of pronouns, the closest possible binder is excluded:

(21) a. John$_1$ admires him$_{*1,2}$. (#17)

b. John$_1$ said that Mary$_2$ admires him$_{1,3}$. (#18)

c. He$_1$ admires him$_{*1,2}$. (#2088)

R-expressions must be free, so that all coreference readings available within the upward path are correctly excluded (22).

(22) a. John$_1$ admires John$_{*1,2}$. (#19)

b. John$_1$ said that Mary$_{*1,2}$ admires Bill$_{*1, *2, 3}$. (#20)

The c-command condition follows, since non-c-commanding expressions are excluded from the paths:

(23) a. [John's$_1$ sister]$_2$ admires him$_{1,*2}$. (#20)

b. [John's$_1$ sister]$_2$ admires herself$_{*1, 2}$. (#21)

Locality properties are determined by lexical features which restrict upward paths and create an intervention factor. The reflexive, for example, cannot be bound by a nonlocal antecedent.

(24) a. John$_1$ wants Mary$_2$ to admire *himself/herself$_{*1, 2}$. (#22, 23)

b. *John said that Mary admires himself. (#24)

c. John$_1$ wants to admire himself$_{1,*2}$. (#25)

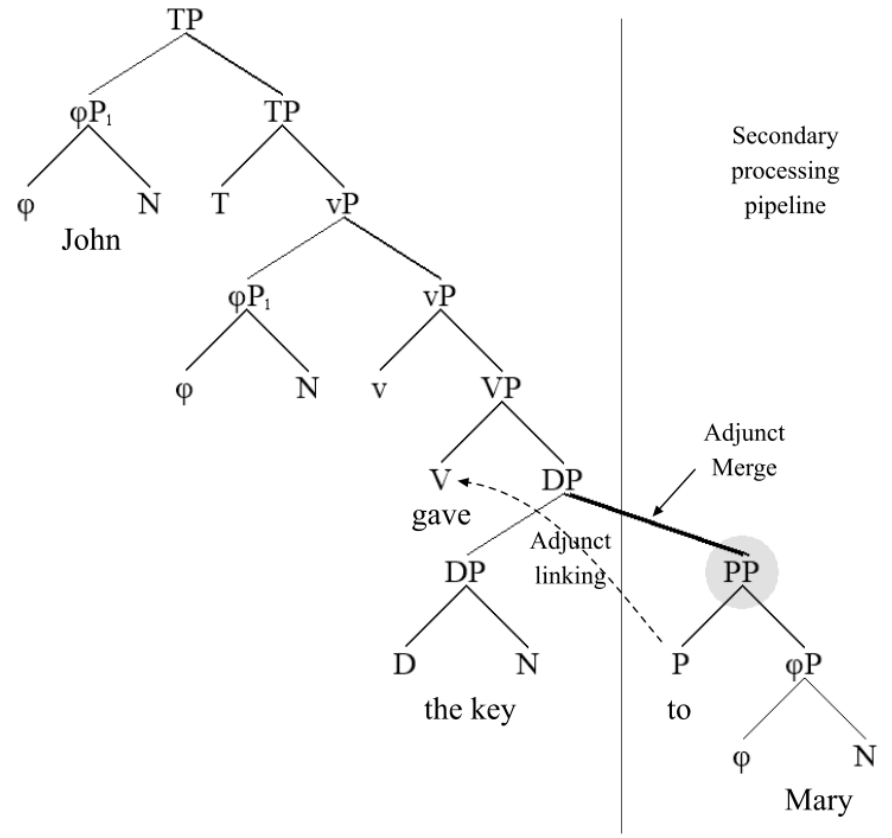
Binding configurations were tested systematically by sentences #2042-2211. They tested binding conditions A-C, the c-command condition, locality, phi-feature matches and mismatches, binding in infinitival constructions (#2052-2053, 2112-2125, 2189-2207), binding in null subject sentences (#2151-2154, 2208-2211) plus few examples of whole conversations (#2063-2066, #2181-2186, 2129-2140) where pronouns were allowed to denote objects projected into existence by prior sentences. While this does not demonstrate that the implementation of the binding theory adopted here is necessary more elegant than any other existing model, it does demonstrate that it is possible to unify binding and the path mechanism. Moreover, we could apply the path analysis to a binding theory that begins from different assumptions.[12]

The syntactic background theory assumes that adjuncts (in contrast to ordinary constituents) are geometrical constituents of the host structure but processed inside parallel processing pipelines as independent phases in the sense of (Chomsky, 2000, 2001). We imagine them being “pulled out” into a parallel working space while still remaining inside the host structure as daughters. They are invisible for grammatical relations such as sisterhood, labelling and selection, and are excluded from all upward paths generated inside the host structure, becoming invisible from the point of view of an active probe.[13] Legitimate adjunct positions are determined by features of the adjunct heads which must be linked to designated heads in the host structure. For example, a VP-adverbial must be linked to V, TP-adverbial to T, and so on. Preposition phrases are typically analysed as V-adjuncts as shown in (25).

(25) John [gave [$_{DP}$ [the key] ⟨$_{PP}$ to Mary⟩] (#26, with several variations #264-270)

Because the adjunct ⟨$_{PP}$ *to Mary*⟩ is invisible to labelling, sisterhood and complement computations inside the host structure, the complement of the main verb will be the DP = *the key*. The position of the PP *to Mary* is determined by a feature which requires that an upward path can be established from the preposition (probe) into the lexical element with feature V (goal), in this case the main verb *gave*. This prevents the preposition phrase from being attached into a too high position. These properties of (25) are illustrated in (26).

(26)



Dependencies between adjuncts and the lexical heads inside their host structures were established in this study by relying on paths. Locality properties were captured by assuming that only the closest goal head is considered. In this case, the PP adjunct can only be linked with the closest V. Finally, if an adjunct is found at a wrong position in the surface structure such that the required probe-goal dependency fails, it will be reconstructed into a legitimate position (if any) by creating an adjunct chain. Adjunct chains are created by locating the closest finite tense boundary and searching for suitable location downwards; “closest finite tense boundary” was again defined by the path mechanism.[14] One way to illustrate adjunct chains is to consider the Finnish free word order signature and its relation to information structure. In Finnish, arguments of a finite clause can occur in almost any logically possible combination, two of which are shown in (27).

(27) a.	Pekka	ihaile-e	Merja-a. (SVO) (#27)
	Pekka.NOM	admire-PRS.3SG	Merja-PAR

‘Pekka admires Merja.’

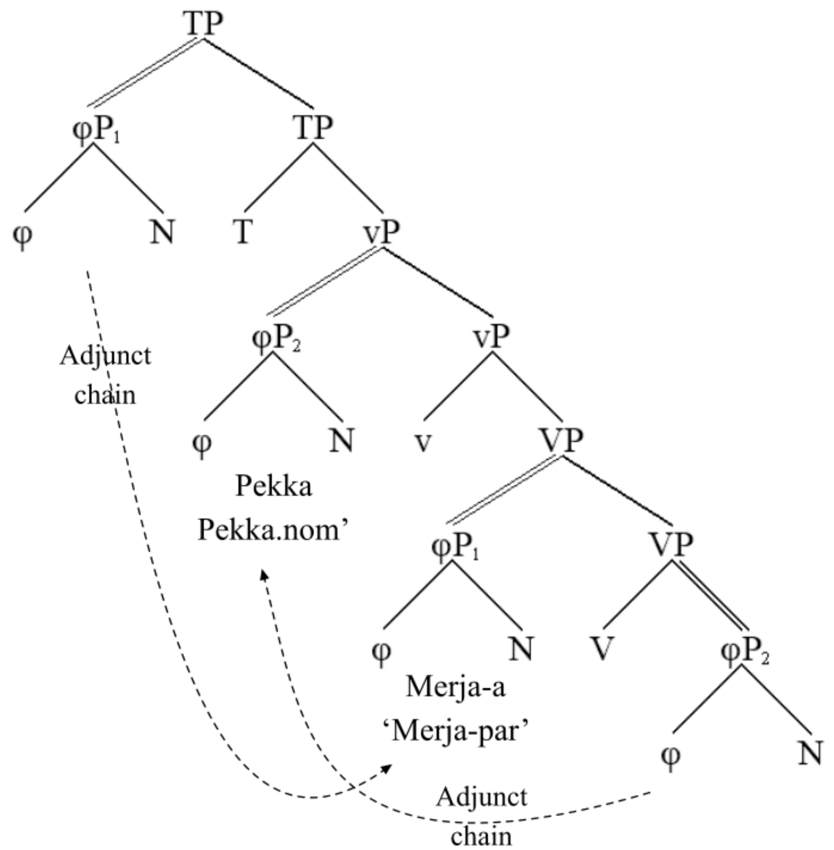
b. Merja-a ihaile-e Pekka. (OVS) (#28)

Merja-PAR admire-PRS.3SG Pekka.NOM

‘Pekka (information focus) admires Merja (topic).’

The syntactic background theory assumes that richly case marked arguments, such as those appearing in the example above, are attached to the structure as adjuncts (Jelinek, 1984; Baker, 1996; Brattico, 2019). This explains why word order is relatively free: referential arguments are excluded from selection, sisterhood and labelling inside the host structure. They are reconstructed to the canonical positions via adjunct chains by using case information. Thus, in the example (27)b, the partitive-marked preverbal subject will be reconstructed into the direct object position, while the postverbal nominative argument will be reconstructed into the thematic subject (agent) position. This will generate correct thematic interpretations for sentences of this type.

(28)



The thematic landing positions illustrated above are those where these elements can check their case features (nominative by T, partitive by v), by the case checking mechanism elucidated earlier. These adjunct chains are connected to information structure. The fronted direct object will be interpreted as the marked topic, the postverbal subject as information focus. This interpretation can be found from the output files (line 787 in the results file). See also the translations in (27). Notice how right adjunct attachments affect the computations inside the host structure: the postverbal nominative subject will not be the sister of the V; rather, the reconstructed argument is. Finnish word order permutations were tested systematically by sentences #1324-1354, 1379-1402, 1427-1443, 1586-1619 among other examples. These were usually contrasted with similar permutations in English, which represents a fixed word order language. Also adjectives (#271-282) and Finnish adverbial phrases (#283-306) were tested.

Both Italian and Finnish exhibit the pro-drop profile, where the subject pronoun of a finite clause can be left unpronounced. Since there were pro-drop sentences of this type in the dataset (#1258-1289, 1290-1292, 1314-1317, 2151-2154, 2208-2211), the phenomenon had to be taken into account. The main concern raised by these data in the present context is that pro subjects can satisfy specifier selection requirements (i.e., EPP), participate in finite control, and must be taken into account when handling agreement, suggesting that they interact with the path mechanism.

The syntactic background theory regards pro-subjects as virtual constituents residing inside heads with sufficiently rich phi-sets (i.e. T_{pro}). They are otherwise treated as ordinary referential elements. A tensed verb T_{pro}, for example, is regarded as a tensed predicate that “contains its own argument.” Several grammatical mechanisms were defined in the original model in such a way that they were sensitive to the presence of these elements. For example, a pro-element can satisfy the finite EPP requirement in Finnish. These assumptions, when translated into the present context, suggest that pro-elements should be present in upward paths. This led to problems, however. Their presence inside the paths gave wrong results and the pro had to be ignored explicitly in the calculations. Many of these problems stemmed from the fact that pro-elements are not equivalent to overt pronouns. Finnish third person pro-elements, for example, have an idiosyncratic behavioural profile (discussed below).

Analysing these issues further showed that there were only two cases where pro-elements had to be examined. They were removed on such grounds from the generalized upward path mechanism and processed in connection with these exceptional circumstances. These exceptional circumstances concern certain head-specifier dependencies and special situations involving agreement, analysed below. These two cases are furthermore unified by the curious fact that the pro-elements are seemingly only needed as a "last resort" if overt phrasal elements are missing, which suggest that they are special objects rather than regular constituents. Let us examine these issues in some more detail.

Example (29)a shows a standard pro-drop construction in Finnish, (b) the analysis calculated by the model. The pro-element is generated on the basis of the first person singular agreement features (1SG). Italian pro-drop clauses are analysed in the same way.

(29) a. (Minä) ihaile-n Merja-a. (#29)
I.NOM admire-PRS.1SG Merja-PAR
'I admire Merja.'

b. [$_{TP}$ T$_{pro}$ [$_{vP}$ v [$_{VP}$ admire Merja]]]

Finnish canonical finite clauses exhibit a strong topic-related EPP phenomenon, which is satisfied uncontroversially by the pro-element in (29)(Vainikka and Levy, 1999; Holmberg and Nikanne, 2002). This is handled by assuming that the mechanism that picks up the closest edge element for T returns a pro-element (if available) if nothing else is present,[15] which then allows the pro-element to check the EPP. Notice that, at least in the context of the present dataset, the pro-element is always irrelevant for the purposes of local specifier processing if a phrasal specifier is present: in such cases the pro-element is either redundant, in essence just "doubling" agreement information already present in the pronoun, or leads into an agreement violation detected by the agreement mechanism. The latter shows that pro-elements cannot be ignored when calculating agreement (for agreement tests, see sentences #985-1257). In the current model, unvalued phi-features of predicates can be valued by local phrases inside the sister or the edge. In the case of (29), they are valued by the pro-element, since an overt phrase is available neither inside the sister nor the edge, which is implemented again by assuming that the pro-element is exceptionally present when the edge is scanned for a potential agreement partner.[16] This is another reason why agreement had to rely on the head-specifier mechanism. In sum, the model operates with pro-elements in two exceptional circumstances. The required code was factored into two specialized functions rather than into the generalized upward path mechanism.

A major issue with pro-drop concerns Finnish third person pro-elements which exhibit finite control and cannot satisfy the EPP. The standard assumption in the literature is that Finnish third person agreement features are too “weak” to sustain interpretation and require (finite) control (Vainikka, 1989; Vainikka and Levy, 1999; Holmberg, 2005; Holmberg, Nayudu and Sheehan, 2009; Holmberg and Sheehan, 2010). The data is shown in (30).

(30)	a.	*Ihaile-e	Merja-a. (#30)			
		admire-PRS.3SG	Merja-PAR			
		Intended: ’(He) admires Merja.’				
	b.	Pekka	sanoi	että	ihaile-e	Merja-a. (#31)
		Pekka	said	that	admire-PRS.3SG	Merja-PAR
		’Pekka said that he (=Pekka) admires Merja.’				

This was handled by assuming, following Holmberg and Sheehan (2010), that the third person agreement features are unable to value the uninterpretable D-feature of T, which triggers the long-distance control relation shown in (30)b. Importantly, I was unable to formulate a completely general control mechanism handling this special case plus the general case, thus these mechanisms had to be added as a special clause to the control theory. Working out a completely general control theory, embedding both finite control and standard infinitival control, was left for future. For a useful discussion of finite control in the context of standard infinitival control, see (Landau, 2004, 2013). Standard finite complement clauses, such as (30)b, were tested separately (#247-263). One problem was found: the model wrongly accepted a derivation where a referential argument was merged above the high complementizer in Finnish (#263).

The dataset contains long head movement (LHM) constructions where a head moves to the beginning of the clause over a finite element (#1762-1782, 1789-1877, 1878-1922)(31).

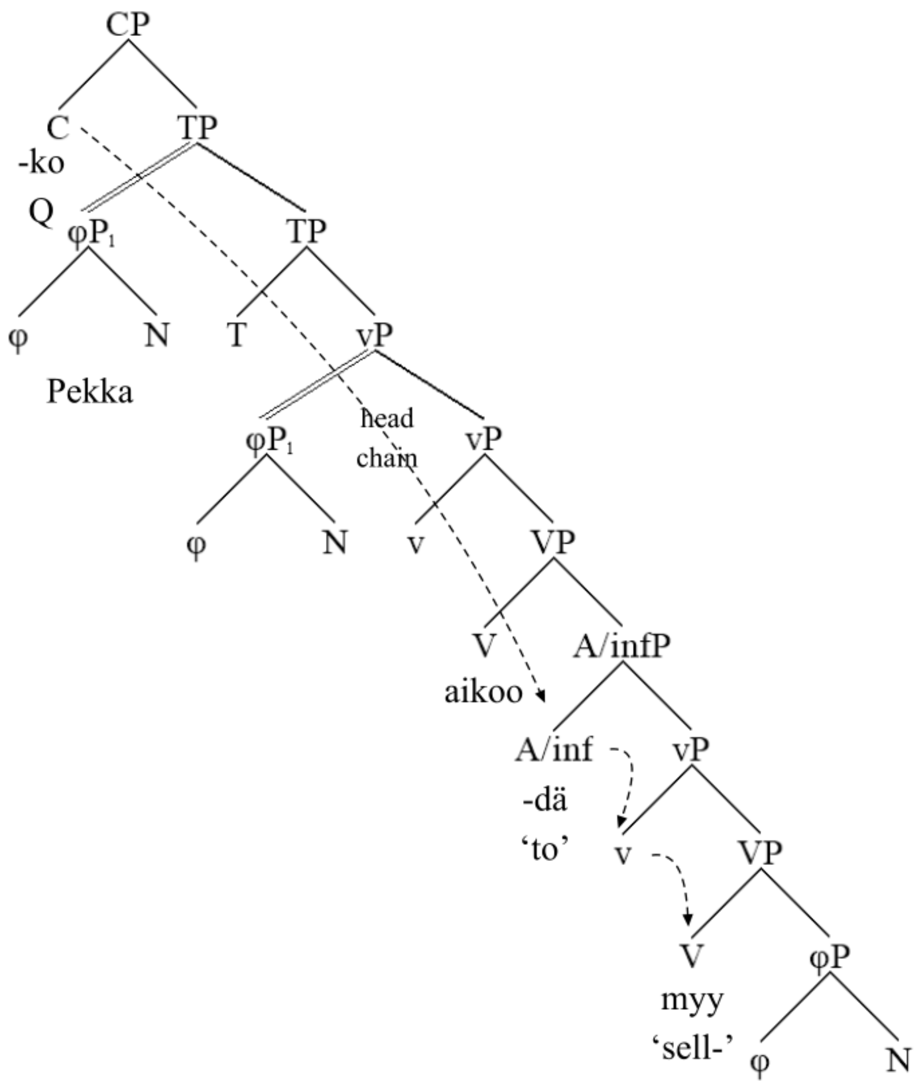
(31)	Myy-dä-kö	Pekka	aiko-o	__	kaiken	omaisuutensa? (#32)
	sell-A/INF-Q	Pekka	plan-PRS.3SG		all	possessions

‘Did Pekka plan to sell all his possessions?’

‘Was it selling that Pekka planned to do with all his possessions?’

The model calculates properties of these constructions by generating nonlocal head chains. In this particular case, the calculated analysis is (32).

(32)



The thematic gap position for the fronted predicate is calculated by relying on selection, as already mentioned. In this case, the A-infinitival can be selected by the verb ‘plan’ to create an interpretation ‘plan + to sell’. The reconstructed head contains an operator feature that is linked with the scope marker, which creates the predicate cleft interpretation ‘was it selling…’ shown in the translation (see line 886 in the results file). Hypothetical LHM constructions in English were correctly judged ungrammatical (#1783-1788); a large number of ungrammatical LHM constructions in Finnish were collected into data batches #1789-1821 and #1824-1877.

Several additional issues were tested in this study, but will not be commented in detail in this article. Nominal phrases (#226-240) were correctly derived with the exception of some constructions where implausible analyses were generated for DPs containing several adjectives (see, e.g., #231). Since adjectives were not in the focus in this study, I did not attempt to correct this problem. Adpositions are sentences #241-246. Word order principles were tested by using canonical sentences and often all logically possible permutations of them (#1318-1660), as illustrated by (33).

(33) a. John admires Mary.(#1318) / Mary admires John. (#1319)
b. *John Mary admires. (#1320)
c. *admires John Mary. (#1321)
d. *Mary John admires. (#1323)

Finnish equivalents are #1324-1330, most of which are grammatical due to the relatively free word order profile of this language. The same experiment was repeated for ditransitives (#1331-1378), negated clauses (#1379-1402) and sentences with auxiliary verbs (#1403-1426). Ungrammatical head permutations (#1427-1443), infinitival word orders (#1444-1583, 1586-1619) and topicalization (#1620-1660) were tested. One erroneous analysis of an English sentence was found (#1584), while the Finnish batch contained several problems that are visible in Figure 1 and which have to do with the creation of A-chains for genitive arguments. Italian clitic constructions were also included (#1923-2041).

4 DISCUSSION AND CONCLUSIONS

Kayne's upward path mechanism was proposed as a possible avenue for unifying a number of grammatical phenomena such as selection, control, binding, adjunct placement, chain creation, case assignment and others. The model was observationally adequate and arguably not completely implausible from the point of view of descriptive adequacy. The latter claim merits a brief comment. Descriptive adequacy requires that the output of the model is not implausible from the point of view of *an* empirically justified linguistic theory. It does not require that the output is plausible in the context of some specific theory, or that it is unproblematic. These are too strong requirements. The ultimate purpose of a model of this type is to invite a better model that satisfies observational adequacy but is more accurate, simple, or covers a larger test corpus, or perhaps has all these characteristics.

Beyond the technical implementation, an interesting possibility opens up when we consider that the upward path mechanism and the notion of an "active syntactic working memory" coincide over an interesting range of cases. The syntactic background model and the underlying recognition grammar assume that left branches and right adjuncts constitute phases in the sense of Chomsky (2000, 2001).[17] A phase is a syntactic segment that is processed (transferred to the LF interface and interpreted) independently during the derivation. Thus, being processed independently, it does not need to reside in the active syntactic working memory containing the phrase structure segment that is being processed together with an active probe element. It is of some interest to observe that the path analysis specifically ignores both left branches and right adjuncts and therefore seems to scan what resides in an active syntactic working memory. We could therefore consider that the probe-goal mechanism represents "memory scanning." Specifically, we could assume that all stages of the derivation are associated with an active working memory that holds the elements that are currently under processing such that they have not been transferred out as a phase. Grammatical mechanisms establish probe-goal dependencies between lexical items in the active syntactic working memory. Unification of grammatical mechanisms with a working memory model could be useful in integrating theories of syntax with broader theories of linguistic cognition.

The upward path mechanism was designed to substitute c-command (probe-goal) based models. Although I did not argue explicitly against this alternative here, there were several reasons Kayne's approach was adopted. One is that in order to deduce all the data discussed here (Table 1) by using a c-command based framework, a computationally complex and resource-costly algorithm is called for that must search the phrase structure tree exhaustively and recursively. This is because the data requires upward paths reaching from any branch (i.e., left branches, right adjuncts, complements) into a head; the reverse dependency therefore requires branching. Kayne's path mechanism is unambiguous, simple, has a very advantageous complexity profile and could have intuitive appeal as a theoretical construct. The second reason has to do with what I mentioned above, namely that doing the model in this way allows one to seek further reduction by relying on the notion of syntactic working memory. This seems impossible if we begin from c-command. Third, the upward path mechanism works seamlessly with a recognition grammar where lexical elements are merged into the structure in a left-to-right order (Phillips, 1996, 2003; Chesi, 2012). Thus, an active probe can always "see" the structure to its left, while the right side remain in the derivation's future. Finally, the direction matters for the implementation of the intervention factor which restricts the search space.

This work used a Python-based formalization of a linguistic theory for its calculations. Such methods are routinely used in the more advanced sciences, but the issue is sometimes misunderstood in linguistics. Any scientific hypothesis in any field must ultimately be put into a rigorous test by showing that the data follows from it in a literal sense. A computer is a device that can perform the required calculations. Moreover, they are extremely useful in linguistics, because hypotheses in this field tend to be logically deep in the sense that the computational path from the hypothesis to the data is long.[18]

References (automatically generated for submission)

Baker, M. (1996) *The Polysynthesis Parameter*. Oxford: Oxford University Press.

Bianchi, V. and Chesi, C. (2006) "Phases, left branch islands, and computational islands.," *University of Pennsylvania Working Papers in Linguistics (PWPL)*, 12(1), pp. 15–28.

Brattico, P. (2017) “Null subjects and control are governed by morphosyntax in Finnish,” *Finno-Ugric Languages and Linguistics*, 6, pp. 2–37.

Brattico, P. (2019) *A computational implementation of a linear phase parser. Framework and technical documentation (version 9.0)*. Pavia.

Brattico, P. and Chesi, C. (2020) “A top-down, parser-friendly approach to operator movement and pied-piping,” *Lingua*, 233, p. 102760. doi:10.1016/j.lingua.2019.102760.

Chesi, C. (2012) *Competence and Computation: toward a processing friendly minimalist Grammar*. Padova: Unipress.

Chomsky, N. (1977) “On Wh-movement,” in Culicover, P., Wasow, T., and Akmajian, A. (eds) *Formal Syntax*. New York: Academic Press, pp. 71–132.

Chomsky, N. (1981) *Lectures in Government and Binding: The Pisa Lectures*. Dordrecht: Foris.

Chomsky, N. (2000) “Minimalist Inquiries: The Framework,” in Martin, R., Michaels, D., and Uriagereka, J. (eds) *Step by Step: Essays on Minimalist Syntax in Honor of Howard Lasnik*. Cambridge, MA.: MIT Press, pp. 89–156.

Chomsky, N. (2001) “Derivation by Phase,” in Kenstowicz, M. (ed.) *Ken Hale: A Life in Language*. Cambridge, MA.: MIT Press, pp. 1–37.

Chomsky, N. (2008) “On Phases,” in Otero, C., Freidin, R., and Zubizarreta, M.-L. (eds) *Foundational Issues in Linguistic Theory: Essays in Honor of Jean-Roger Vergnaud*. Cambridge, MA.: MIT Press, pp. 133–166.

Chomsky, N. and Lasnik, H. (1977) “Filters and Control,” *Linguistic Inquiry*, 8, pp. 425–504.

Dechaine, R. and Wiltschko, M. (2002) “Decomposing pronouns,” *Linguistic Inquiry*, 33(3), pp. 409–422.

Déchaine, R.-M. and Wiltschko, M. (2017) “A Formal Typology of Reflexives,” *Studia Linguistica*, 71(1–2), pp. 60–106. doi:10.1111/STUL.12072.

Fiengo, R. and May, R. (1994) *Indices and identity*. Cambridge, MA.: MIT Press.

Gibson, E. (1998) “Linguistic complexity: locality of syntactic dependencies,” *Cognition*, 68, pp. 1–76.

Gordon, P., Hendrick, R. and Johnson, M. (2001) “Memory interference during language processing,” *Journal of Experimental Psychology. Learning, Memory, and Cognition*, 27, pp. 1411–1423.

Heck, F. (2008) *On Pied-Piping: Wh-Movement and Beyond*. Amsterdam: Mouton de Gruyter.

Heck, F. (2009) “On Certain Properties of Pied-Piping,” *Linguistic Inquiry*, 40(1), pp. 75–111.

Holmberg, A. (2005) “Is There a Little pro? Evidence from Finnish,” *Linguistic Inquiry*, 36(4), pp. 533–564.

Holmberg, A., Nayudu, A. and Sheehan, M. (2009) “Three partial null-subject languages: a comparison of Brazilian Portuguese, Finnish and Marathi,” *Studia Linguistica*, 63, pp. 59–97.

Holmberg, A. and Nikanne, U. (2002) “Expletives, subjects and topics in Finnish,” in Svenonius, P. (ed.) *Subjects, Expletives, and the EPP*. Oxford: Oxford University Press, pp. 71–106.

Holmberg, A. and Sheehan, M. (2010) “Control into finite clauses in partial null-subject languages,” in Biberauer, T. et al. (eds) *Parametric Variation: Null Subjects in Minimalist Theory*. Cambridge: Cambridge University Press, pp. 125–152.

Hornstein, N. (2001) *Move! A Minimalist Theory of Construal*. Malden, USA: Blackwell.

Huhmarniemi, S. (2012) *Finnish A’-movement: Edges and Islands*. University of Helsinki.

Huhmarniemi, S. and Brattico, P. (2013) “On primary and secondary movement,” *Acta Linguistica Hungarica*, 60(2), pp. 173–216.

Jelinek, E. (1984) “Empty categories, case and configurationality,” *Natural Language & Linguistic Theory*, 2, pp. 39–76.

Kayne, R. (1983) “Connectedness,” *Linguistic Inquiry*, 14.2, pp. 223–249.

Kayne, R. (1984) *Connectedness and Binary Branching*. Foris.

Kimball, J. (1973) “Seven principles of surface structure parsing in natural language,” *Cognition*, 2(1), pp. 15–47. doi:10.1016/0010-0277(72)90028-5.

Kiparsky, P. (2019) “Notes on nonfinite clauses in Finnish,” in Condoravdi, C. and King, T. (eds) *Tokens of Meaning: Essays in Honor of Lauri Karttunen*. Stanford, CA: CSLI Publications.

Koskinen, P. (1998) *Features and categories: Non-finite constructions in Finnish*. University of Toronto.

Landau, I. (2004) “The scale of finiteness and the calculus of control,” *Natural Language & Linguistic Theory*, 22, pp. 811–77.

Landau, I. (2013) *Control in Generative Grammar A Research Companion*. Cambridge: Cambridge University Press.

Lewis, Richard, L. and Vasishth, S. (2005) “An Activation-Based Model of Sentence Processingas Skilled Memory Retrieval,” *Cognitive Science*, 29, pp. 375–419.

Miller, G. and Chomsky, N. (1963) “Finitary models of language users,” in *Handbook of mathematical psychology*. New York: John Wiley & Sons Inc., pp. 419–491.

Phillips, C. (1996) *Order and structure, MIT*.

Phillips, C. (2003) “Linear order and constituency,” *Linguistic Inquiry*, 34, pp. 37–90.

Reuland, E. (2001) “Primitives of Binding,” *Linguistic Inquiry*, 32, pp. 439–492.

Reuland, E. (2006) “Agreeing to bind,” in *Organizing Grammar*. Berlin: de Gruyter Mouton, pp. 505–513.

Rooryck, J. and Wyngaerd, G. vanden (2011) *Dissolving Binding Theory*, *Dissolving Binding Theory*. Oxford: Oxford University Press. doi:10.1093/acprof:oso/9780199691326.001.0001.

Rosenbaum, P. (1967) *The Grammar of English Predicate Complement Constructions*. Cambridge, MA.: MIT Press.

van Steenbergen, M. (1987) *Binding Relations in Finnish*. Univ. of Groningen.

van Steenbergen, M. (1991) “Long-distance binding in Finnish,” in Koster, J. and Reuland, E.J. (eds) *Long-distance anaphors*. Cambridge University Press, pp. 231–244.

Vainikka, A. (1989) *Deriving Syntactic Representations in Finnish*. University of Massachusetts Amherst.

Vainikka, A. and Levy, Y. (1999) “Empty subjects in Finnish and Hebrew,” *Natural Language & Linguistic Theory*, 17(3), pp. 613–671.

Warren, T. and Gibson, E. (2002) “The influence of referential processing on sentence complexity,” *Cognition*, 85, pp. 79–112.

Warren, T. and Gibson, E. (2005) “Effects of NP type in reading cleft sentences in English,” *Language and Cognitive Processes*, 85, pp. 751–767.

---

[1] The idea that linguistic processing utilizes a working memory is not new (Miller and Chomsky, 1963; Kimball, 1973; Gibson, 1998; Gordon, Hendrick and Johnson, 2001; Warren and Gibson, 2002,

2005; Lewis, Richard and Vasishth, 2005). Kimball's seminal work on cognitive principles of parsing was of particular relevance to the present work.

[2] There are few redundancies in the corpus. For example, canonical transitive finite clauses occur several times due to the fact that they exhibit several grammatical phenomena such as projection principle, thematic roles, case assignment and A-chains.

[3] Referential arguments can be either φPs or DPs. D occurs when the input contains an overt determiner. This distinction is motivated on the grounds that there are languages, such as Finnish, where overt determiners are not obligatory and hence do not appear in the input. φPs are analyzed as "minimal" referential arguments. This is not a new idea (Déchaine and Wiltschko 2017; Dechaine and Wiltschko 2002; van Steenbergen 1987; 1991).

[4] The Python definition for the notion of edge is `list(takewhile(lambda x:x.mother.head() == probe, probe.working_memory_path()))` where we take all elements inside the active working memory such that they are contained within the projection from the probe, as specified by the lambda function. The function `head()` returns the head/label of any phrase structure object.

[5] The Python definition for the closest primitive head is `next((const for const in self.working_memory_path() if const.is_primitive()), None)` which takes the first ("next") primitive constituent from the active working memory and returns nothing if there is none (which occurs if the probe is the highest projection in the active phrase structure).

[6] The following abbreviations were used in glossing: 1, 2, 3 = first, second and third person; ACC = accusative case; A/INF = A-infinitival in Finnish, corresponding loosely to English to-infinitival; GEN = genitive case; NOM = nominative case; PAR = partitive case; PRS = present tense; Q = yes/no question particle in Finnish; SG = singular.

[7] The Python definition for maximal projection is `[probe, *probe.edge()][-1].mother` which designates the mother of the highest element at the edge of the probe.

[8] The test corpus contains Finnish case assignment constructions exhibiting all structural case forms in connection with canonical intransitive, transitive and intransitive clauses with nominative, accusative and partitive case (#877-892); effects of aspect (#893-902), polarity (#903-907) and agreement (#908-912) on the accusative case; long-distance case assignment (#913-923); genitive and partitive in connection with adpositions (#924-928); genitive and partitive in connection with infinitivals (#929-939); possessive constructions (#940-941); numeral constructions (#942-951); direct object case marking inside adverbials, including DP-adverbials (#952-967). Some special constructions, such as psych verbs (#968-973), impersonal passives (#974-979), copular constructions (#980-984) were also tested. Discussion of this batch was left for the supplementary document.

[9] The agreement function therefore contains two separate operations, one implementing agreement with elements inside sisters (e.g., T-DP), another with elements inside the edge (e.g., T-SpecTP).

[10] The Python definition for operator binding is `next((node for node in head.working_memory_path() if {operator_feature, 'FIN'}.issubset(node.features)), None)` which targets the closest ("next") finite operator head inside the upward path.

[11] The Python definitions are `working_memory = list(takewhile(self.recovery_termination, probe.working_memory_path()))` and `antecedent = next((const for const in working_memory if self.is_possible_antecedent(const, probe)), None)`, respectively.

[12] An alternative is to bind inside syntax (e.g., Chomsky, 1977, 1981; Chomsky and Lasnik, 1977; Fiengo and May, 1994; Hornstein, 2001; Reuland, 2001, 2006; Rooryck and Wyngaerd, 2011). The upward path mechanism by itself does not rule out syntax-based analyses of binding, but the matter must be addressed in a separate study.

[13] Currently these properties are "forced" on the structure by stipulative definitions, thus they must be regarded as descriptions of a more fundamental system that remains to be implemented.

[14] The Python definition is `next((node.mother for node in ps.working_memory_path() if {'T/fin', 'FORCE'} & node.features), ps.top())` which returns the mother of the first node that is either finite tense or force head, or returns to top node if no such head exists.

[15] The Python definition is `next((const for const in h.edge()), h.extract_pro())`, which returns the first phrasal constituent from the edge and extracts a pro-element if nothing is found.

[16] The Python definition is `next(((const.head(), sorted({f for f in const.head().features if phi(f) and valued(f)})) for const in [const for const in head.edge()] + [head.extract_pro()] if const and self.agreement_condition(head, const)), (None, {}))` where the extra pro-condition is the function `head.extract_pro()`.

[17] When it comes to how the derivation can be sliced into several independent packages, the direction of the derivation matters. vPs and CPs were not assumed to be phases due to this; it seems impossible to slice a left-to-right derivation in this way. The matter is nontrivial and not discussed in this article.

[18] Verifying the current hypothesis against the present test corpus required close to 350000 elementary Merge operations for the grammatical sentences alone, making paper-and-pencil methods useless. To get a sense of how many computational operations of each type (e.g., Merge, Agree, Move) are consumed while processing grammatical sentences in the present test corpus see the resource file https://we.tl/t-R0Q9Tt3oZL.