Journal of Logic, Language and Information (2021) 30:407–428
https://doi.org/10.1007/s10849-019-09308-0

A "Maximal Exclusion" Approach to Structural Underspecification in Dynamic Syntax

Tohru Seraku[1]

Published online: 3 December 2019

Abstract
'Case' and 'grammatical relations' are central to syntactic theory, but rigorous treatments of these concepts in surface-oriented grammars such as Dynamic Syntax are pending. In this respect, Japanese is worthy of mention; in this language, the nominative case particle *ga*, which typically marks a subject, may mark an object in certain syntactic contexts, and more than one instance of *ga* may be present within a single clause. These patterns cannot be captured if we simply assume that *ga* marks a subject. In the present article, we aim to advance formal aspects of the framework, especially the mechanism of 'structural underspecification,' by proposing that the parse of a case particle maximally excludes potential landing sites of an unfixed node at the time of parsing the case particle, delaying the resolution of the unfixed node until a subsequent stage of structure building. This maximal exclusion approach to structural underspecification accounts for a range of case marking patterns and their connections with grammatical relations.

Keywords Case · Grammatical relation · Parsing · Incrementality · Japanese

1 Introduction

'Case' and 'grammatical relations' are central to syntactic theory, but rigorous accounts of these concepts are pending in surface-oriented grammars such as Dynamic Syntax (Cann et al. 2005; Kempson et al. 2001, 2011). In the present article, we model the relation between case and grammatical relations in formal grammar terms, with examples drawn from Japanese.

The case system of Japanese poses challenges for grammar modelling, but these issues have not previously been seriously tackled in Dynamic Syntax (Sects. 2–3). To give a concrete example, it is assumed that the nominative case particle *ga* resolves

✉ Tohru Seraku
seraku@hufs.ac.kr

1 Hankuk University of Foreign Studies, Yongin, Korea

an ‘unfixed node’(i.e. node whose structural position in the tree is not yet fixed) as the subject node. This assumption, though it works for a number of examples, encounters empirical problems; *ga* may be used to mark an object, and *ga* may appear more than once within a single clause. We account for these data by proposing a ‘maximal exclusion’ approach to structural underspecification and resolution (Sects. 4–5). The account is further applied to ‘complex predicate’ data (Sect. 6).

2 Empirical Background: Case and Grammatical Relation

In Japanese, case is designated by a post-NP particle. In (1), *ga* indicates that *Ken* has nominative case, and *o* indicates that *sushi* has accusative case.

(1)	*Ken-ga*	*sushi-o*	*tabe-ta*	(*koto*)
	K-NOM	sushi-ACC	eat-PAST	(COMP)
	‘Ken ate sushi.’[1]			

The notion of ‘grammatical relations’ refers to a syntactic relation between a predicate and its argument NP in a sentence; examples include ‘subject’ and ‘object.’ These are abstract concepts and are identified based on several tests in each language. The standard tests for subjecthood in Japanese relate to ‘honorification’ and ‘reflexives’ (Kishimoto 2004; Shibatani 1977). To take ‘honorification’ as an example, α is the subject of a sentence if α may be the target of honorification in the sentence. In (2), the honorific form *otabeninat* ‘eat’ elevates the referent of *sensei* ‘teacher.’ *Sensei* is thus said to be the subject of the sentence in (2).

(2)	*Sensei-ga*	*sushi-o*	*otabeninat-ta*	(*koto*)
	teacher-NOM	sushi-ACC	eat.HON-PAST	(COMP)
	‘The teacher ate sushi.’			

Some frameworks (e.g. Lexical-Functional Grammar; Dalrymple 2001) view grammatical relations as theoretically primitive concepts and express them through syntactic representations. In other syntactic theories (e.g. Chomsky 1995), such primitive concepts are not postulated, and grammatical relations are defined structurally. In Dynamic Syntax, grammatical relations are considered to be epiphenomena, and are structurally designated. Thus, the ‘subject node,’ for instance, corresponds to the argument node that is immediately dominated by the root node (see the discussion of (10) in Sect. 3.2 for details).

Within Dynamic Syntax, the issue of how case relates to grammatical relation has not been examined in detail, and it has been simply assumed that the nominative *ga* marks a subject (Cann et al. 2005; Kempson and Kiaer 2010). Whilst this stipulation holds true of (1) and (2), it is not sustainable due to the following facts (see Nambu et al. 2018 and references therein).

[1] When the subject of certain types of predicate is nominative-marked, it has an ‘exhaustive’ implication (Kuno 1973: 38). Unless an appropriate context is set out, sentences with such implications are degraded. Since exhaustive implications disappear in embedded clauses, *koto* ‘COMP’ is put at the end

- *Ga* may mark an object NP.
- *Ga* may occur several times within a single clause.
- A subject NP may be marked with the dative particle *ni*.

These properties are not observable with all verbs; for instance, *ga*-marking of an object is possible only with stative predicates such as predicates of competence (e.g. *joozuda* 'good at,' *dekiru* 'able to do'), predicates of feeling (e.g. *sukida* 'like,' *hoshii* 'want'), and *tai*-derivatives (e.g. *tabe-tai* 'want to eat') (Kuno 1973: 81–82); see Koizumi (2008: 147) for additional examples. The first two properties are illustrated in (3) (see Sect. 4.7 for discussion of the properties of *ni*).

(3)	*Ken-ga*	*sushi-ga*	*sukina*	(*koto*)
	K-NOM	sushi-NOM	like	(COMP)

'Ken likes sushi.'

This clause involves two occurrences of *ga*. The second NP *sushi*, though it is marked with *ga*, is regarded as an object based on syntactic tests for objecthood (Koizumi 2008: 142–145). Therefore, the simple correspondence between *ga* (case) and subject (grammatical relation) cannot deal with data such as (3). In Sect. 4, we shall consider more complex data involving scrambling, null arguments, and structural ambiguity, among other things. These data are directly relevant to surface-oriented grammars, where structure building proceeds on the basis of left-to-right linear parsing.[2]

3 Formal Background: Dynamic Syntax

3.1 Basic Machinery

Dynamic Syntax (DS) models the process whereby a string is parsed in a step-by-step fashion and the structure representing an interpretation of the string is progressively built up (Cann et al. 2005; Kempson et al. 2001, 2011). This structure building process dispenses with syntactic representations; the parse of a word directly updates a semantic structure. In this framework, *Dynamic* refers to online parsing, and *Syntax* refers to an abstract system that gradually maps a string onto a semantic structure.

Let us first clarify the representational aspect of the model. Suppose that the input to the parser is the string (4). The parse of the whole string derives the semantic tree (5).

[2] Japanese exhibits Major Subject Constructions (Kuroda 1992: 248), as illustrated in (i). Although (i) resembles (3), they should be distinguished. In (i), the first *ga*-NP and the second *ga*-NP stand in a possessor-possessum relation.

(i)	*Ken-ga*	*imouto-ga*	*yasashii*
	K-NOM	younger.sister-NOM	sweet

Lit. 'It is Ken$_i$ that his$_i$ younger sister is sweet.'

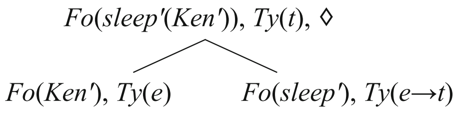
Nakamura et al. (2009) address Major Subject Constructions within Dynamic Syntax, but their account is formally illicit, as pointed out in Seraku (2016); see also Kiaer (2014). We assume that the first *ga* and the second *ga* are distinct in that only the latter concerns grammatical relations. The account of *ga*, developed in this article, is fully consistent with Seraku's (2016) analysis of Major Subject Constructions.

(4)

Ken-ga	*ne-ta*	(*koto*)
K-NOM	sleep-PAST	(COMP)

‘Ken slept.’

(5) Final state (ignoring tense)



The tree is binary-branching, the argument appears on the left and the functor on the right. Each node is decorated with statements. For instance, *Fo*(*Ken*′) states that the content on this node is *Ken*′. (The formula predicate *Fo* takes a content as an argument). The same node is decorated with *Ty*(*e*), a statement that the logical type of the content is *e* (i.e. entity type). (The type predicate *Ty* takes a logical type as an argument.) Examples of other logical types include *t* (i.e. a truth-evaluable type) and $e \rightarrow t$ (i.e. a type of functor which takes a type-e content and returns a type-t content). As will be shown below, in each tree-update step a single node is under development, and this node is designated by the pointer ◊.

Let us turn next to the structure-building aspect of the framework. The initial state is defined as the AXIOM (6).

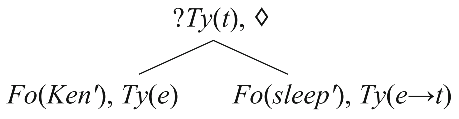
(6) AXIOM

?*Ty*(*t*), ◊

At the initial stage, there is only a root node. This is annotated with ?*Ty*(*t*), a requirement that the node will be decorated with some type-t content.[3] The structure building process proceeds through a combination of ‘computational,’ ‘lexical,’ and ‘pragmatic’ actions.

Computational actions: This type of action is not lexically triggered and is optionally run at the discretion of the parser. A paradigm example is ELIMINATION. Suppose that the parser has processed *Ken-ga ne* ‘Ken slept’ in (4). At this stage, the parser has constructed the tree (7).

(7) Parsing *Ken-ga ne*



The node under development, signalled by ◊, has two daughters, each of which is specified for both content and type. This structural state licenses ELIMINATION, whose effect amounts to a functional application and a type deduction. This computational action updates (7) to (5). In (5), ?*Ty*(*t*) has been satisfied because the root node is now decorated with *Ty*(*t*); the node is also decorated with the semantic content *Fo*(*sleep*′(*Ken*′)).

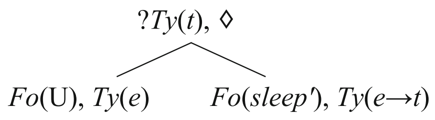
[3] There have recently been debates about the AXIOM. For instance, though it is generally assumed that the node set out by the AXIOM occupies the root position (Kempson et al. 2001: 299), this assumption is challenged by Seraku (2013: 73), who claims that the position of the node introduced by the AXIOM may be determined later in the course of structure building.

Lexical actions: Each lexical item encodes a set of actions for tree-update. Consider (8).

(8) *Ne-ta*
sleep-PAST
'(A contextually salient person) slept.'

Japanese is a 'pro-drop' language; thus, argument NPs may be covert as long as they are contextually retrievable. In DS, it is held that the parse of a predicate projects a propositional template. For instance, the parse of *ne* 'sleep' updates the initial state (6) to (9). The argument node is decorated with a metavariable U, a placeholder to be saturated with a type-e content.

(9) Parsing *ne*

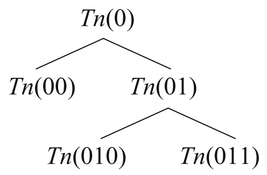


Pragmatic actions: The metavariable U in (9) is in need of saturation. If Ken is a salient individual in context, U is given the value *Ken′*. This process is an instance of the pragmatic action called SUBSTITUTION.

3.2 Structural Underspecification

Each node is assigned an address, with the 'tree-node' predicate *Tn*.

(10)

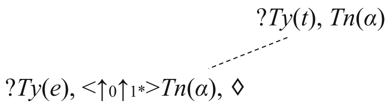


Tn takes a numeral as its value. If a node is assigned a numeral α, its left daughter is assigned α0 and its right daughter α1. As the root node is assigned 0, its left daughter receives 00 and its right daughter 01. In (10), the subject node corresponds to the *Tn*(00)-node, and the object node corresponds to the *Tn*(010)-node.

Not only can we describe node addresses using *Tn*-statements, we can also describe their relations by defining node-relation operators. 'Immediate dominance' is modelled with $<\uparrow_0>$ or $<\downarrow_0>$ (for argument daughters) and $<\uparrow_1>$ or $<\downarrow_1>$ (for functor daughters). To illustrate these with the diagram (10), the *Tn*(0)-node is referred to as $<\uparrow_0>Tn(0)$ from the perspective of the *Tn*(00)-node or $<\uparrow_1><\uparrow_1>Tn(0)$ from the perspective of the *Tn*(011)-node. For brevity reasons, multiple occurrences of the operator will be indicated through a single pair of angle brackets, as in $<\uparrow_1\uparrow_1>$, rather than $<\uparrow_1><\uparrow_1>$. See Blackburn and Meyer-Viol (1994) for further details.

Node labelling and node-relation descriptions are essential for structural underspecification, a device which leaves the position of a node unspecified and which allows the ‘unfixed node’ to be resolved subsequently. A computational action that introduces an unfixed node is Local *Adjunction. In (11), an unfixed relation is indicated visually through use of a dashed line.

(11) Local *Adjunction



An unfixed node introduced by this action must be resolved in a local structure. In $<\uparrow_0\uparrow_{1*}>$, where the Kleene star is used, 1* is an arbitrary succession of 1 (including none), as in $<\uparrow_0>$,$<\uparrow_0\uparrow_1>$, $<\uparrow_0\uparrow_1\uparrow_1>$, and so on. $<\uparrow_0\uparrow_{1*}>Tn(\alpha)$ means: if you go up from an argument node by one node (and optionally keep going up through functor nodes), you will reach the $Tn(\alpha)$-node. $<\uparrow_0\uparrow_{1*}>Tn(\alpha)$ thus conveys that the current node is at some argument position within a local propositional structure although the exact position is uncertain at this stage.

Structural underspecification may be resolved in two ways: (i) through the computational action of Merge (see Sect. 4.2) or (ii) through lexical actions encoded in a case particle. As for (ii), it has been assumed that the parse of a case particle immediately resolves an unfixed node (Cann et al. 2005; Kempson and Kiaer 2010). The nominative case particle *ga*, for instance, has been assumed to fix the position of an unfixed node at the subject node.

This analysis of case particles, however, results in the problems stated in Sect. 2. In the next section, we will propose an alternative approach to structural underspecification.

4 Proposal: A Maximal Exclusion Approach

4.1 Informal Sketch

In Cann et al. (2005) and Kempson and Kiaer (2010), a case particle is considered to uniquely determine the landing site for an unfixed node. In the present article, we offer an alternative view based on Seraku (2016), proposing that a case particle reduces the range of landing sites by maximally excluding potential sites modulo the restrictions imposed by each case particle.

(12) Proposal: General claim

a. A case particle excludes all landing sites for an unfixed node except for a few candidates.

b. Such candidates vary depending on the type of a case particle.

Under this view, a case particle may not immediately resolve an unfixed node (although when the number of potential landing sites is reduced to one, it amounts to an immediate resolution.)[4]

(12) states that the parse of a case particle leads to maximal exclusion of potential landing sites of an unfixed node, modulo the lexical constraints encoded in each case particle. For the nominative case particle *ga*, the 'lexical constraints' are specified as (13).

(13) Proposal: Nominative particle *ga*

a. *Ga* excludes all but the subject node and the object node.

b. If the above exclusion has already occurred, further exclusion occurs: exclude all but the subject node or the object node (not both).

The general proposal (12) and the lexical constraints (13) will be illustrated in Sects. 4.2–4.6. Other case particles than *ga* will be touched upon in Sect. 4.7. Illustrations in this section are semi-formal; they will be formalised in the next section (Sect. 5).

4.2 Nominative Particle (Part I)

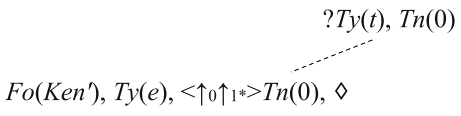
Let us start with the basic example (14).

(14)	*Ken-ga*	*ne-ta*	(*koto*)
	K-NOM	sleep-PAST	(COMP)
	'Ken slept.'		

As will be detailed in what follows, the parse of *ga* allows the unfixed node to be resolved only at the subject or the object node. In (14), however, *ne* 'sleep' is an intransitive verb, and the object node is not created. Thus, the unfixed node will be resolved at the subject node.

After LOCAL *ADJUNCTION is applied, the parse of *Ken* in (14) derives the tree-state (15).

(15) Parsing *Ken*



$<\uparrow_0\uparrow_{1*}>Tn(0)$ specifies the set of constraints (16).

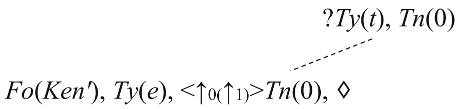
(16) $\{<\uparrow_0>Tn(0), <\uparrow_0\uparrow_1>Tn(0), <\uparrow_0\uparrow_1\uparrow_1>Tn(0), \ldots\}$

[4] (12)a may be comparable to search systems of library resources. Suppose that a library stores three books *Food and Health*, *History of Japan*, and *History of Money*. If one puts *History* in the search box, the system maximally excludes all book titles without *History*, in this case excluding *Food and Health*. If one puts *Food* in the search box, the system excludes all book titles without *Food*, excluding *History of Japan* and *History of Linguistics* (in which case the system happens to output the single title *Food and Health*.).

$<\uparrow_0>Tn(0)$ means: if you go up one node from the current argument node, you will reach the root node. $<\uparrow_0>Tn(0)$ refers to the subject node, $<\uparrow_0\uparrow_1>Tn(0)$ refers to the object node, and $<\uparrow_0\uparrow_1\uparrow_1>Tn(0)$ refers to the indirect-object node. In this way, (16) indicates that an unfixed node may be resolved at any argument position in the local propositional structure.

The next item to be parsed in (14) is *ga*. According to (13)a, *ga* excludes all but the subject node and the object node.

(17) Parsing *Ken-ga*



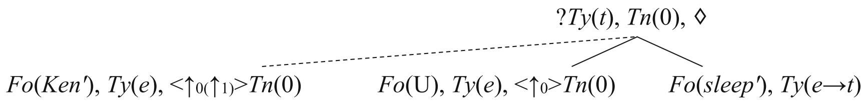
In $<\uparrow_{0(}\uparrow_{1)}>Tn(0)$, $(\uparrow_1)$ means that $<\uparrow_1>$ is optional, as delineated in (18).

(18) $\{<\uparrow_0>Tn(0), <\uparrow_0\uparrow_1>Tn(0)\}$

Unlike (16), (18) indicates that an unfixed node is resolvable at either the subject or the object node. Put differently, unlike the previous treatment of *ga*, which uniquely resolves an unfixed node at the $<\uparrow_0>Tn(0)$-position, the new analysis of *ga* still allows structural leeway.

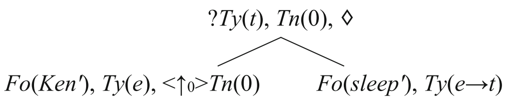
What comes next in (14) is the intransitive verb *ne* 'sleep,' which yields the tree-state (19). As has been illustrated in (9), *ne* 'sleep' projects a propositional template.

(19) Parsing *Ken-ga ne*



The propositional template built by *ne* 'sleep' contains the subject node, and it unifies with the unfixed node through the computational action of Merge (see Sect. 3.2)

(20) Merge



This unification process is legitimate because $<\uparrow_{0(}\uparrow_{1)}>Tn(0)$ in (19) dictates that the unfixed node is resolvable as the $<\uparrow_0>Tn(0)$-node or the $<\uparrow_0\uparrow_1>Tn(0)$-node.

Finally, Elimination outputs the final state (see (5) in Sect. 3.1). The *Fo*-statement at the root node represents the interpretation of the string parsed: 'Ken slept' (ignoring tense).

4.3 Nominative Particle (Part II)

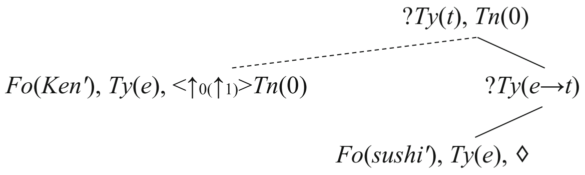
Let us turn to example (21).

(21) *Ken-ga* *sushi-o* *tabe-ta* (*koto*)
K-NOM sushi-ACC eat-PAST (COMP)
'Ken ate sushi.'

As we shall see, the parse of *ga* allows the unfixed node for *Ken* to be resolved only at the subject or the object node. In (21), however, the parse of the accusative *o* resolves the unfixed node for *sushi* as the object node, and the unfixed node for *Ken* will be resolvable only at the subject node.

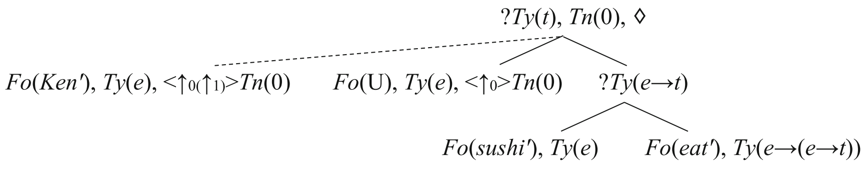
After *Ken-ga* is parsed (see (17)), the parse of *sushi-o* yields (22). (The parse of the accusative *o* resolves an unfixed node at the object position; see Sect. 4.7.)

(22) Parsing *Ken-ga sushi-o*



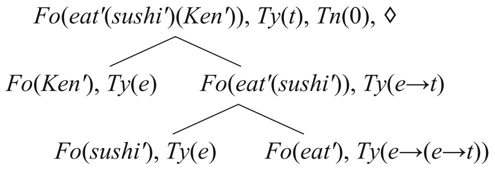
At this stage, there is no position where the unfixed node could be resolved. The unfixed node thus remains as it is. The parse of *tabe* 'eat' then projects a propositional structure, as in (23).

(23) Parsing *Ken-ga sushi-o tabe*



In the propositional template, the subject node is annotated with $<\uparrow_0>Tn(0)$. This tree-node description is compatible with $<\uparrow_{0}(\uparrow_{1})>Tn(0)$ of the unfixed node. The parser then executes MERGE, combining the description of the subject node with that of the unfixed node. Finally, ELIMINATION derives the final tree-state.

(24) MERGE + ELIMINATION


Fo(eat′(sushi′)(Ken′)), Ty(t), Tn(0), ◊
Fo(Ken′), Ty(e)
Fo(eat′(sushi′)), Ty(e→t)
Fo(sushi′), Ty(e)
Fo(eat′), Ty(e→(e→t))

The account straightforwardly carries over to the scrambling data (25).

(25)

Sushi-o	*Ken-ga*	*tabe-ta*	(*koto*)
sushi-ACC	K-NOM	eat-PAST	(COMP)

'Ken ate sushi.'

After LOCAL *ADJUNCTION is applied, *sushi* is parsed on an unfixed node, and it is resolved as the object node by the parse of *o*. The parser then runs LOCAL *ADJUNCTION once again to introduce an unfixed node to parse *Ken*. The parse of *ga* decorates this unfixed node with $<\uparrow_{0}(\uparrow_{1})>Tn(0)$. After the parse of *tabe* 'eat' creates a propositional template, the subject node in this propositional template unifies with the unfixed node decorated with $\langle\uparrow_{0}(\uparrow_{1})\rangle Tn(0)$ through MERGE. Finally, ELIMINATION derives a tree-state identical to (24).

4.4 Nominative Particle (Part III)

Let us then examine (26), where both the subject and the object are nominative-marked. (26) presents a case of structural ambiguity, with two distinct readings.

(26)	*Ken-ga*	*Naomi-ga*	*sukina*	(*koto*)
	K-NOM	N-NOM	like	(COMP)

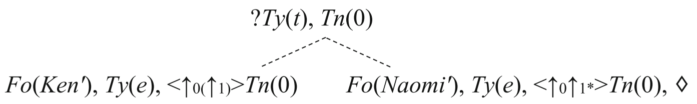
a. 'Ken likes Naomi.'

b. 'Naomi likes Ken.'

As will be illustrated below, the ambiguity is modelled in terms of the parser's decision to resolve the unfixed node for *Naomi* as the subject or the object node.

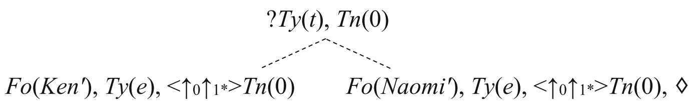
The parse of *Ken-ga* proceeds as usual (see (17)). The parser then runs LOCAL *ADJUNCTION to create an unfixed node, where the second NP *Naomi* is parsed.

(27) Parsing *Ken-ga Naomi*


$?Ty(t), Tn(0)$
$Fo(Ken'), Ty(e), \langle\uparrow_{0}(\uparrow_{1})\rangle Tn(0)$
$Fo(Naomi'), Ty(e), \langle\uparrow_{0}\uparrow_{1*}\rangle Tn(0), \Diamond$

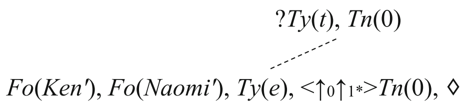
Let us make a brief digression here. In the tree logic adopted in DS (Blackburn and Meyer-Viol 1994), multiple unfixed nodes of the same type cannot be distinguished. Consider the hypothesised tree-state (28).

(28) Illegitimate tree (hypothesised)


$?Ty(t), Tn(0)$
$Fo(Ken'), Ty(e), \langle\uparrow_{0}\uparrow_{1*}\rangle Tn(0)$
$Fo(Naomi'), Ty(e), \langle\uparrow_{0}\uparrow_{1*}\rangle Tn(0), \Diamond$

It looks as though (28) contains two unfixed nodes, but there only exists a single unfixed node, as shown in (29). This is because the putative two unfixed nodes are indistinguishable in that they involve the same tree-node statement $\langle\uparrow_{0}\uparrow_{1*}\rangle Tn(0)$. Note that (28)–(29) are illicit since one and the same node is annotated with two inconsistent descriptions: $Fo(Ken')$ and $Fo(Naomi')$.

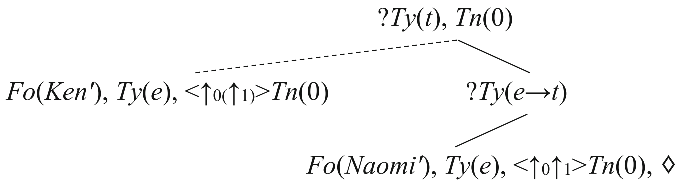
(29) Illegitimate tree (hypothesised)


$?Ty(t), Tn(0)$
$Fo(Ken'), Fo(Naomi'), Ty(e), \langle\uparrow_{0}\uparrow_{1*}\rangle Tn(0), \Diamond$

The problem of inconsistent description does not occur in (27) because the two unfixed nodes are of different types; one is decorated with $<\uparrow_{0}(\uparrow_{1})>Tn(0)$, and the other with $<\uparrow_{0}\uparrow_{1*}>Tn(0)$.[5]

In (27), the exclusion in (13)a has occurred for the node decorated with *Fo*(*Ken′*) due to the parse of the first instance of *ga*. According to (13)b, then, the parse of the second instance of *ga* excludes all potential landing sites for the unfixed node for *Naomi* with the exception of the subject or object node. If the parser chooses to exclude all but the **object** node, $<\uparrow_{0}\uparrow_{1}>Tn(0)$ is posited at the unfixed node for *Naomi*; the unfixed node is now identified as the object node.

(30) Parsing *Ken-ga Naomi-ga*


?Ty(t), Tn(0)
Fo(Ken′), Ty(e), <↑0(↑1)>Tn(0)
?Ty(e→t)
Fo(Naomi′), Ty(e), <↑0↑1>Tn(0), ◊

The rest of the parse process is as outlined in the last subsection. The parse of *sukina* 'like' projects a propositional template, and the newly created subject node unifies with the unfixed node through MERGE. The parse of *sukina* also creates the object node, but this node harmlessly collapses with the node for *Naomi* which is already present in (30). This parse process leads to the (26a) interpretation.

In parsing the second instance of *ga*, the parser could have excluded all but the **subject** node as landing sites for the unfixed node for *Naomi*. If this exclusion happens, the unfixed node for *Ken* is licensed only at the object node. This yields the interpretation in (26b).

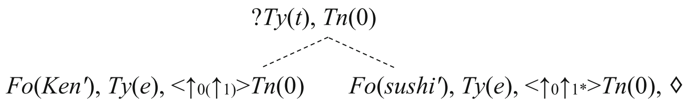
In some cases, structural ambiguity does not occur. Consider (31).

(31)	*Ken-ga*	*sushi-ga*	*sukina*	(*koto*)
	K-NOM	sushi-NOM	like	(COMP)

'Ken likes sushi.' (cf. *'Sushi likes Ken' unless a personifying reading is intended.)

The parse of *Ken-ga sushi* outputs (32).

(32) Parsing *Ken-ga sushi*


?Ty(t), Tn(0)
Fo(Ken′), Ty(e), <↑0(↑1)>Tn(0)
Fo(sushi′), Ty(e), <↑0↑1*>Tn(0), ◊

In parsing *ga* in *sushi-ga*, if the parser chooses to put $<\uparrow_{0}\uparrow_{1}>Tn(0)$ at the unfixed node for *sushi*, it is resolved as the object node. This leads to the interpretation mentioned in (31). If the parser chooses to put $<\uparrow_{0}>Tn(0)$ at the unfixed node for *sushi*, it is resolved

5 In a similar vein, Cann et al. (2005: 235) argue that an unfixed node introduced by LOCAL *ADJUNCTION is differentiated from an unfixed node introduced by another computational action *ADJUNCTION in that the tree-node statement for the former, $<\uparrow_{0}\uparrow_{1*}>Tn(\alpha)$, is distinct from that for the latter, $<\uparrow_{*}>Tn(\alpha)$.

as the subject node. This results in the interpretation ‘Sushi likes Ken.’ This tree-update is formally licit, but the resulting interpretation would be blocked on semantic grounds.

4.5 Nominative Particle (Part IV)

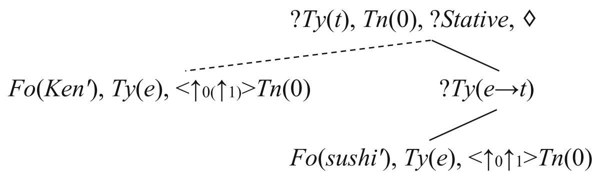
The account is still not complete. The *ga*-marking of an object NP is allowed only with stative predicates (see Sect. 2). Thus, (33), where *tabe* ‘eat’ is an action verb, is ungrammatical.

(33)	**Ken-ga*	*sushi-ga*	*tabe-ta*	(*koto*)
	K-NOM	sushi-NOM	eat-PAST	(COMP)
	Int. ‘Ken ate sushi.’			

Our account of case particles developed thus far does not rule (33) out since the possibility of *ga*-marking of an object NP is dependent upon the type of predicate. We will thus refine the account by introducing a requirement concerning the type of predicate.

We assume that when *ga* marks an object, it posits the requirement that the forthcoming predicate will be a stative predicate. This is satisfied when a stative predicate is subsequently parsed. In (33), the parse of *ga* in *sushi-ga* puts the requirement ?*Stative* at the root node.[6]

(34) Parsing *Ken-ga sushi-ga*


?Ty(t), Tn(0), ?Stative, ◊
Fo(Ken′), Ty(e), <↑0(↑1)>Tn(0)
?Ty(e→t)
Fo(sushi′), Ty(e), <↑0↑1>Tn(0)

?*Stative* is satisfied when the parse of a stative predicate introduces *Stative* (see Sect. 5 for formalisation). In (33), *tabe* ‘eat’ is not a stative predicate, and ?*Stative* remains in the tree. The string (33) is thus not mapped onto a well-formed tree, hence it is ungrammatical.

Note that the second instance of *ga* (which posits ?*Stative*, if it resolves the unfixed node as the object node) may be absent at the surface. This is because Japanese allows null arguments (see Sect. 3.1). Consider (35), with the intended meaning ‘Ken likes Naomi.’

(35)	*Ken-ga*	*sukina*	(*koto*)
	K-NOM	like	(COMP)
	Int. ‘Ken likes (Naomi).’		

In the literature on the nominative-object construction, data such as (35) have rarely been inspected because the nominative object NP itself is absent. For surface-oriented

6 Formally, ?*Stative* may be expressed as a requirement at a ‘situation’ node (Cann 2011).

grammars, examples such as (35) are important because any information encoded in a lexical item is not available if the lexical item has not been parsed.

In (35), the running of LOCAL *ADJUNCTION, followed by the parse of *Ken-ga*, outputs an unfixed node decorated with $<\uparrow_{0}(\uparrow_{1})>Tn(0)$. The parse of *sukina* 'like' builds a propositional template. The parse of *sukina* posits *Stative* at the ?*Ty*(*t*)-node; this usually satisfies ?*Stative*, but the requirement is not present in the tree for (35) because the nominative object NP (i.e. *Naomi-ga*) is absent at the surface. If the parser unifies the unfixed node for *Ken* with the **subject** node created by *sukina*, and saturates the meta-variable at the object node created by *sukina* as *Naomi'* (i.e. SUBSTITUTION), the interpretation 'Ken likes Naomi' emerges.

There is another reading of (35); in the appropriate context, it may mean 'Naomi likes Ken.' This interpretation also follows from our analysis. After *sukina* 'like' projects a propositional structure, the parser could identify the unfixed node for *Ken* as the **object** node because the unfixed node is decorated as $<\uparrow_{0}(\uparrow_{1})>Tn(0)$ and this is consistent with the address of the object node. The meta-variable at the subject node is given the value *Naomi′* by SUBSTITUTION.[7]

4.6 Nominative Particle (Part V)

Japanese exhibits the 'postposing' construction in casual speech (Kuno 1987: 67–80). This is shown in (36), where the final particle *yo* is indicative that the string is uttered in casual speech.

(36)	*Ken-ga*	*sukida-yo*	*sushi-ga*
	K-NOM	like-FP	sushi-NOM
	'Ken likes sushi.'		

(36) is acceptable, e.g., when the speaker responses to *In this class, who likes Japanese food?* with additional information about Ken's favorite Japanese food (which the speaker decides to convey after uttering *sukida* 'like').

The postposing construction raises two questions for our analysis. Firstly, an unfixed node is introduced by LOCAL *ADJUNCTION, but its application is conditional upon

[7] One reviewer wondered whether our account predicts that example (ii), where the non-stative verb *tatai* 'hit' is involved, allows the 'a'-reading alone. (Example (ii) was constructed by the author based on the reviewer's comment.)

(ii)	*Ken-ga*	*tatai-ta*	(*koto*)
	K-NOM	hit-PAST	(COMP)
	a. 'Ken hit (something).'		
	b. *'(Someone) hit Ken.'		

In our analysis, *ga* resolves the unfixed node for *Ken* as the subject or object node. If the unfixed node is resolved as the **subject** node, the 'a'-reading arises. If the unfixed node is resolved as the **object** node, the root node is decorated with ?*Stative* but this requirement cannot be met by the non-stative verb *tatai* 'hit.' Thus, the 'b'-reading is ruled out. (?*Stative* is posited only when the unfixed node is resolved as the object node; see Sect. 5.) I am grateful to the reviewer for bringing this issue to my attention.

the presence of ?*Ty*(*t*). In (36), however, the requirement has been satisfied by the time *sushi* is parsed. Secondly, according to the proposed analysis, the parse of *ga* in *sushi-ga* introduces ?*Stative*, which needs to be satisfied by the parse of a stative predicate. In (36), however, the stative predicate comes before *sushi-ga*.

As for the first issue, a DS analysis of Japanese postposing is found in Seraku and Ohtani (2016), who contend that the input condition *?Ty*(*t*) on LOCAL *ADJUNCTION is relaxed in the casual register (see Kempson et al. 2002: 24 for a similar proposal relating to FINAL *ADJUNCTION). The parser may thus introduce an unfixed node to parse *sushi* in (36).

As for the second issue, the order in which a *ga*-NP and a stative predicate are parsed does not affect the final tree-state. When *ga* in *sushi-ga* is parsed, ?*Stative* is introduced at the root node. This requirement is immediately satisfied by the presence of *Stative* at the same node, which has been introduced by the parse of the stative predicate *sukida* 'like.' The rest of the process is as usual; the parse of *ga* resolves the unfixed node for *sushi* as the one decorated with $<\uparrow_0\uparrow_1>Tn(0)$, which harmlessly collapses with the object node which has been created by the parse of *sukida*. The resulting final-state for the string (36) is identical to the final-state for the string (31) (setting aside the contribution of the final particle *yo* to the tree, which would be modelled by Purver et al.'s (2010) LINK analysis of non-truth conditional content).

Our account of *ga* is therefore consistent with casual speech, once a variant of LOCAL *ADJUNCTION is defined, reflecting register variations (Kempson et al. 2002: 24).

4.7 Other Case Particles: Some Provisional Considerations

According to our proposal (12), a case particle excludes all but a few candidates as landing sites for an unfixed node and candidates are differently specified in different case particles. In this subsection, we present a preliminary analysis of the accusative *o* and the dative *ni*.

The accusative *o* typically marks an NP bearing the semantic role of theme; see *sushi* in (21). The accusative *o* may also mark an NP bearing the semantic role of path (37) or that of departure site (38) (NKK 2009: 67–70).

(37)	*Ken-ga*	*sono*	*yama-o*	*koe-ta*	(*koto*)
	K-NOM	that	mountain-ACC	pass-PAST	(COMP)

'Ken passed that mountain.'

(38)	*Ken-ga*	*ie-o*	*de-ta*	(*koto*)
	K-NOM	house-ACC	leave-PAST	(COMP)

'Ken left a house.'

In light of the 'double-*o* constraint' (Harada 1973), Shibatani (1978: 289–292) shows that the *o*-marked NPs as in (37)–(38) are 'objects.'

Setting aside a number of more complex issues,[8] we hold that *o* excludes all landing sites but the object node.

(39) Proposal: Accusative particle *o*

O excludes all but the object node.

(39) amounts to immediately resolving an unfixed node as the object node. Thus, as long as *o* is concerned, our 'maximal exclusion' approach converges with the 'unique determination' approach in previous studies (Cann et al. 2005; Kempson and Kiaer 2010).

The dative *ni* typically marks an indirect object, as in (40).

(40)	*Ken-ga*	*ringo-o*	*Naomi-ni*	*age-ta*	(*koto*)
	K-NOM	apple-ACC	N-DAT	give-PAST	(COMP)
	'Ken gave apples to Naomi.'				

When a stative predicate is involved as in (41), *ni* may mark a subject; in this marking pattern, the object NP is obligatorily marked with *ga*. Besides, *ni*-marking of an object is possible with a limited number of verbs such as *au* 'meet' and *niru* 'resemble' (Kishimoto 2017: 455–456), as illustrated in (42).

(41)	*Ken-ni*	*eigo-ga/*o*	*wakaru*	(*koto*)
	K-DAT	English-NOM/ACC	understand	(COMP)
	'Ken understands English.'			

(42)	*Ken-ga*	*Naomi-ni*	*at-ta*	(*koto*)
	K-NOM	N-DAT	meet-PAST	(COMP)
	'Ken met Naomi.'			

From the maximal exclusion perspective, we tentatively assume (43).

(43) Proposal: Dative particle *ni*

a. *Ni* excludes all but the subject node, the object node, and the indirect-object node.

b. If there has been another unfixed node, exclude all landing sites of the current unfixed node except for the subject node or the object node or the indirect-object node.

The set of predicates allowing *ni*-subject is a proper subset of the set of predicates allowing *ga*-object (Kuno 1973: 88). We hold that if the unfixed node constrained by the parse of *ni* is resolved at the subject node, the root node is decorated with

[8] Kuno (1976) argues that the accusative *o* appears in raising constructions, but its 'raising' status remains contentious (Kishimoto 2017: 465–468).

$?Stative_{\mathrm{DAT}}$, a requirement which can be satisfied by only a stative predicate allowing *ni*-subject.[9]

Consider (41). The parse of *ni* decorates the unfixed node for *Ken* with $<\uparrow_{0}(\uparrow_{1})(\uparrow_{1})>Tn(0)$, allowing it to be resolved at the subject node or the object node or the indirect-object node. The parse of *ga* then decorates the unfixed node for *eigo* 'English' with $<\uparrow_{0}(\uparrow_{1})>Tn(0)$, which allows it to be resolved at the subject or the object node. After *wakaru* 'understand' projects a propositional structure, the unfixed node for *Ken*, annotated with $<\uparrow_{0}(\uparrow_{1})(\uparrow_{1})>Tn(0)$, may unify with the subject node. Also, the unfixed node for *eigo*, annotated with $<\uparrow_{0}(\uparrow_{1})>Tn(0)$, may unify with the object node. (The parser could unify the node for *Ken* with the object node, but this becomes illegitimate since *wakaru* is not a type of predicate allowing a *ni*-object, which is captured as lexically heterogeneous constraints; see footnote 9.)

It also follows from the analysis that *eigo* cannot be accusative-marked in (41). The parse of *o* is assumed to put *?Non-Stative* at the root node, but the node is decorated with $?Stative_{\mathrm{DAT}}$ due to the parse of *ni*. These requirements cannot be satisfied simultaneously.

Another illustrative example is (40). The parse of *ga* decorates the unfixed node for *Ken* with $<\uparrow_{0}(\uparrow_{1})>Tn(0)$, and the parse of *o* resolves the unfixed node for *ringo* 'apple' as the object node. In this tree-state, (43)b applies. If the parse of *ni* decorates the unfixed node for *Naomi* with $<\uparrow_{0}\uparrow_{1}\uparrow_{1}>Tn(0)$, it is resolved at the indirect-object node. This resolution process models the interpretation of (40).

Due to (43)b, *ni* in (40) could decorate the unfixed node for *Naomi* with $<\uparrow_{0}\uparrow_{1}>Tn(0)$, but it would become indistinguishable from the node for *ringo*. This, then, leads to the problem of inconsistent description (see (28)–(29)). Similarly, *ni* in (40) could decorate the unfixed node with $<\uparrow_{0}>Tn(0)$, but it would become indistinguishable from the unfixed node for *Ken* once the unfixed node for *Ken* is resolved as the subject node. (Note that the unfixed node for *Ken* cannot be resolved as the object node since it would become indistinguishable from the node for *ringo* 'apple.') In both cases, the problem of inconsistent description occurs. The upshot is that the analysis correctly identifies *Naomi* in (40) as the indirect-object (but neither as the subject nor the object).[10]

In a nutshell, though there are residual problems (see footnotes 8–10), we have suggested that the maximal exclusion approach is in principle applicable to other case particles than the nominative case particle *ga*.

[9] The formulation of $?Stative_{\mathrm{DAT}}$ is left for future work (see also footnote 6). The treatment of *ni*-object is more complicated because the predicates allowing *ni*-object constitute a heterogeneous set (e.g. 'meet,' 'resemble'). As the number of predicates allowing *ni*-object is small, it might be reasonable to encode relevant constraints heterogeneously for each predicate allowing *ni*-object.

[10] One reviewer wondered how to prevent the unfixed node for *Naomi* from being resolved at the subject node or the object node in (40) when *Ken* and *ringo* are **covert**. Firstly, our analysis correctly prevents the unfixed node for *Naomi* from being resolved as the subject node; if this happens, $?Stative_{\mathrm{DAT}}$ is posited at the root node but this cannot be satisfied by the non-stative verb *age* 'give.' Secondly, the analysis is in principle capable of preventing the unfixed node for *Naomi* from being resolved as the object node, if we postulate lexically heterogeneous constraints encoded in the dative *ni*, as suggested in footnote 9. I thank the reviewer for his/her constructive question.

5 Formalisation: Explicating Lexical Entries

We now formalise our account, proposed in the last section, by explicating the lexical entries for the nominative case particle *ga* as well as stative predicates.

The parse of each linguistic item triggers a set of actions for tree-update. A set of actions to be executed is encoded in each item in the conditional format: IF…, THEN…, ELSE… (Cann et al. 2005: 45). For an illustration, consider the lexical entries for the proper noun *Ken*.

(44) Entries for *Ken*

```
IF      $?Ty(e)$
THEN    put($Fo(Ken'), Ty(e), [\downarrow]\bot$)
ELSE    abort
```

To see how (44) works, consider (45), where the effect of LOCAL *ADJUNCTION is displayed more explicitly than (11). The unfixed node is decorated with $?\exists x.Tn(x)$, which requires that the node will be assigned a fixed node-address.

(45) LOCAL *ADJUNCTION

$$\begin{tikzcd}
 & ?Ty(t), Tn(\alpha) \arrow[dl, dashed, no head] \\
?Ty(e), ?\exists x.Tn(x), <\uparrow_0\uparrow_{1*}>Tn(\alpha), \Diamond &
\end{tikzcd}$$

The current node (indicated by $\Diamond$) is decorated with $?Ty(e)$. This satisfies the IF-line in (44), and the parser follows put($Fo(Ken'), Ty(e), [\downarrow]\bot$). $[\downarrow]\bot$, which was disregarded in Sect. 4, ensures that no further node will be constructed below the current node. The action stated in put($Fo(Ken'), Ty(e), [\downarrow]\bot$) updates (45) to (46).

(46) Parsing *Ken*

$$\begin{tikzcd}
 & ?Ty(t), Tn(0) \arrow[dl, dashed, no head] \\
Fo(Ken'), Ty(e), ?\exists x.Tn(x), <\uparrow_0\uparrow_{1*}>Tn(0), [\downarrow]\bot, \Diamond &
\end{tikzcd}$$

If the IF-line is not met, the parser would run abort in the ELSE-line. This action has the effect of quitting further tree-updates, in which case the tree-update crashes.

Now that the format of lexical entries is set out, let us specify the entries for *ga*.

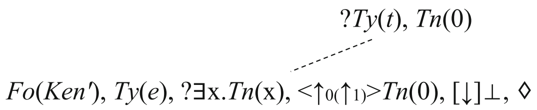
(47) Entries for the nominative particle *ga*

```
IF      $Ty(e), <\uparrow_0\uparrow_{1*}>(Tn(\alpha)\wedge ?Ty(t))$
THEN    IF      $<\uparrow_0\uparrow_{1*}><\downarrow_{1*}\downarrow_0>(?\exists x.Tn(x)\wedge <\uparrow_{0(}\uparrow_{1)}>Tn(\alpha))$
        THEN    put($<\uparrow_0>Tn(\alpha)/<\uparrow_0\uparrow_1>Tn(\alpha)$); gofirst$_\uparrow$($?Ty(t)$); put($?Stative$)
        ELSE    put($<\uparrow_{0(}\uparrow_{1)}>Tn(\alpha)$)
ELSE    abort
```

The outer IF-line holds in (46); the $\Diamond$-marked node is decorated with Ty(e), and if you go up through the path $<\uparrow_0\uparrow_{1*}>$, you will find $?Ty(t)$ and $Tn(0)$. ($Tn(\alpha)$ is instantiated as $Tn(0)$.) The parser then looks at the THEN-line, which embeds conditional statements.

The embedded IF-line does not hold in (46); the tree currently contains no node annotated with $?\exists x.Tn(x)$ and $<\uparrow_{0(}\uparrow_{1)}>Tn(0)$. Since the embedded IF-line is not satisfied, the parser runs the action in the embedded ELSE-line; that is, `put`($<\uparrow_{0(}\uparrow_{1)}>Tn(\alpha)$) decorates the node with $<\uparrow_{0(}\uparrow_{1)}>Tn(0)$. This lexical action updates (46) to (48).

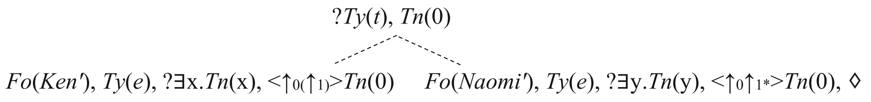
(48) Parsing *Ken-ga*


$?Ty(t), Tn(0)$
$Fo(Ken'), Ty(e), ?\exists x.Tn(x), <\uparrow_{0(}\uparrow_{1)}>Tn(0), [\downarrow]\bot, \Diamond$

As claimed in Sect. 4.2, the unfixed node is not resolved at this stage, but a tighter constraint is imposed; the unfixed node is resolvable only at either the subject or the object node.

The embedded IF-line in (47) is met in (32), repeated here as (49) (with amendments).

(49) Parsing *Ken-ga Naomi*


$?Ty(t), Tn(0)$
$Fo(Ken'), Ty(e), ?\exists x.Tn(x), <\uparrow_{0(}\uparrow_{1)}>Tn(0)$
$Fo(Naomi'), Ty(e), ?\exists y.Tn(y), <\uparrow_{0}\uparrow_{1*}>Tn(0), \Diamond$

If the parser goes up to the $Tn(0)$-node and goes down to the $Fo(Ken')$-node, the description in the inner-IF line holds at this node. The parser, thus, follows `put`($<\uparrow_0>Tn(\alpha)/<\uparrow_0\uparrow_1>Tn(\alpha)$), decorating the node with $<\uparrow_0>Tn(0)$ or $<\uparrow_0\uparrow_1>Tn(0)$. The parser may choose either option, reflecting the structural ambiguity (see Sect. 4.4). Next, $\texttt{gofirst}_{\uparrow}(?Ty(t))$ moves the pointer ($\Diamond$) up to the closest $?Ty(t)$-node, and `put`($?Stative$) decorates this node with $?Stative$.

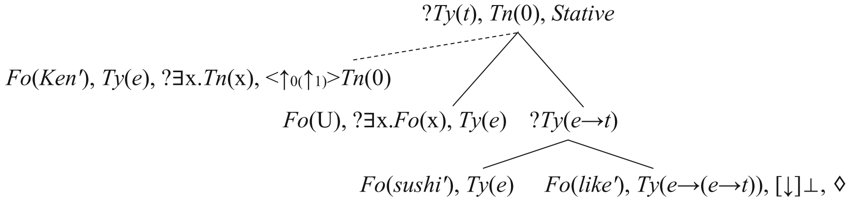
In Sect. 4.5, it was stated that $?Stative$ is satisfied when a stative predicate is parsed. We present the entries for the stative predicate *sukina* 'like' in (50).

(50) Entries for *sukina* 'like'

```
IF      ?Ty(t)
THEN    put(Stative);
        make(<↓0>); go(<↓0>); put(Fo(U), ?∃x.Fo(x), Ty(e)); go(<↑0>);
        make(<↓1>); go(<↓1>); put(?Ty(e→t));
        make(<↓0>); go(<↓0>); put(Fo(V), ?∃y.Fo(y), Ty(e)); go(<↑0>);
        make(<↓1>); go(<↓1>); put(Fo(like'), Ty(e→(e→t)), [↓]⊥)
ELSE    abort
```

Let us illustrate (50) with the tree-state (34). The current node is decorated with $?Ty(t)$ and the IF-line thus holds. The parse of *sukina* updates the tree-state (34) to (51). (In (51), the object node introduced by *sukina* has harmlessly collapsed with the node for *sushi*.)

(51) Parsing *Ken-ga sushi-ga sukina*


?*Ty*(*t*), *Tn*(0), *Stative*
Fo(*Ken'*), *Ty*(*e*), ?∃x.*Tn*(x), <↑0(↑1)>*Tn*(0)
Fo(U), ?∃x.*Fo*(x), *Ty*(*e*)
?*Ty*(*e*→*t*)
Fo(*sushi'*), *Ty*(*e*)
Fo(*like'*), *Ty*(*e*→(*e*→*t*)), [↓]⊥, ◊

?*Stative* has been met by the presence of *Stative*, provided by *sukina* 'like.' The unfixed node unifies with the subject node through MERGE, and ELIMINATION yields the final state.

We have formalised our 'maximal exclusion' approach to structural underspecification by explicating the entries for the nominative particle *ga* and stative predicates.

6 Extension: Preliminary Analysis of Complex Predicates

As pointed out in Kuno (1973: 85), some stative predicates allow the object NP to be marked either by the nominative *ga* or the accusative *o*.

(52)	*Ken-ga*	*furansugo-ga/o*	*yom-eru*	(*koto*)
	K-NOM	French-NOM/ACC	read-can	(COMP)
	'Ken can read French.'			

Whilst the *ga/o* alternation is affected by various factors such as style (Noda 1996: 264–265), it is generally perceived that the alternation is possible only with stative 'complex' predicates such as *yom-eru* 'can read' and *yomi-tai* 'want to read' (Takano 2003). In all of the examples surveyed, the stative predicates are not 'complex,' and the object can be only *ga*-marked.

To account for the *ga/o* alternation, we propose the following:

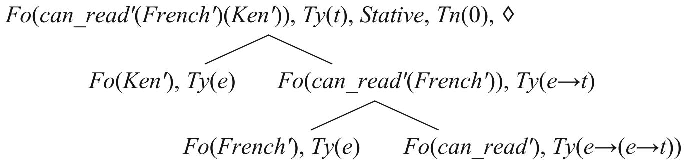
(53) a. The parse of the accusative *o* decorates the ?*Ty*(*t*)-node with ?*Non-Stative*.

b. The verb may be (i) the one-word *yomeru* or (ii) the combination of *yom* and *eru*.

c. In option (i), *yomeru* 'can read' posits *Stative*. In option (ii), *yom* 'read' is an action verb and it posits *Non-Stative*; the potential suffix *eru* posits *Stative* at a higher node.

If we follow option (i), the final tree-state is (54).

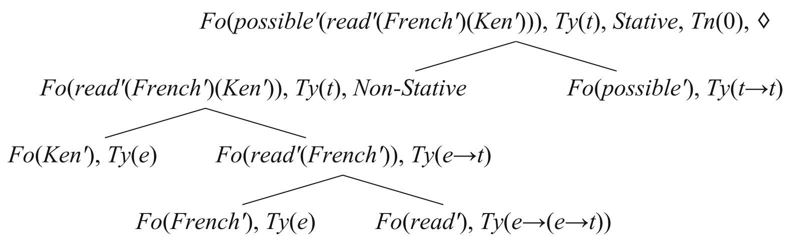
(54) Parsing (52) with option (i)


Fo(*can_read'*(*French'*)(*Ken'*)), *Ty*(*t*), *Stative*, *Tn*(0), ◊
Fo(*Ken'*), *Ty*(*e*)
Fo(*can_read'*(*French'*)), *Ty*(*e*→*t*)
Fo(*French'*), *Ty*(*e*)
Fo(*can_read'*), *Ty*(*e*→(*e*→*t*))

In option (i), *furansugo-o* is disallowed; the parse of *o* posits ?*Non-Stative* but it is not met by the parse of *yomeru* 'can read,' which posits *Stative*. On the other hand, *furansugo-ga* is licit; *ga* in *furansugo-ga* posits ?*Stative*, which is later satisfied by the parse of *yomeru*.

If we follow option (ii), the final output of the tree-update is (55).

(55) Parsing (52) with option (ii)


Fo(possible′(read′(French′)(Ken′))), Ty(t), Stative, Tn(0), ◊
Fo(read′(French′)(Ken′)), Ty(t), Non-Stative
Fo(possible′), Ty(t→t)
Fo(Ken′), Ty(e)
Fo(read′(French′)), Ty(e→t)
Fo(French′), Ty(e)
Fo(read′), Ty(e→(e→t))

The content of *eru*, notated as *possible'*, selects propositional content as an argument and adds a potential meaning to the proposition. In the embedded structure, the action verb *yom* 'read' introduces *Non-Stative*. This is consistent with *furansugo-o* (where *o* posits ?*Non-Stative*), but not *furansugo-ga* (where *ga* posits ?*Stative*).[11]

To sum up, the object only receives *ga*-marking in option (i), and the object only receives *o*-marking in option (ii). The proposed account thus handles the case-marking pattern in (52), but there are residual problems. In particular, we assume that the verbal element is ambiguous between *yomeru* and *yom-eru*, but this assumption has not been motivated by independent evidence.[12] Still, apart from the stipulation about the availability of mono-structure and bi-structure, the analysis is based on the standard machinery of the framework, together with the proposed mechanism of the 'maximal exclusion' process.

7 Conclusion

We have presented a 'maximal exclusion' approach to structural underspecification. In this approach, the parse of a case particle does not immediately resolve an unfixed node but posits further constraints on the potential range of landing sites where the unfixed node is resolved, delaying the resolution process until a later point in the structure building process. This account conforms to the architecture of the framework, where a structure is gradually built up with requirements being added incrementally, driving but also constraining further structure building. It is an empirical matter whether the proposed approach is applicable to comparable data in other languages (see Aikhenvald

[11] In (55), *Ken-ga furansugo-o yom* is parsed in the embedded structure. To this end, the parser needs to run GENERALISED ADJUNCTION before parsing *Ken*. This action introduces an unfixed ?*Ty*(*t*)-node, which may be embedded in an arbitrary depth. See Cann et al. (2005: 242) for details.

[12] It might be expected that the *o*-marked object takes scope under 'can,' whilst the *ga*-marked object takes scope over 'can,' provided that the node for 'can' appears in a higher structure only when the object is *o*-marked. Tada (1992) indeed observes this scope pattern. It has been pointed out, however, that despite a strong tendency for this scope pattern, it is no more than a tendency (Koizumi 2008).

et al. 2001). This and other issues mentioned in the preceding sections will, we hope, stimulate explorations of case and grammatical relations in formal grammar research.

Acknowledgements My sincere gratitude goes to Ruth Kempson, who provided me with constructive and stimulating comments on an earlier version of this article. I appreciate the anonymous reviewers for their valuable questions and suggestions.

References

Aikhenvald, A., Dixon, R., & Ohnishi, M. (Eds.). (2001). *Non-canonical marking of subjects and objects*. Amsterdam: John Benjamins.

Blackburn, P., & Meyer-Viol, W. (1994). Linguistics, logic and finite trees. *Logic Journal of the IGPL, 2,* 3–29.

Cann, R. (2011). Towards an account of the auxiliary system in English. In R. Kempson, et al. (Eds.), *The dynamics of lexical interfaces* (pp. 279–317). Stanford: CSLI Publications.

Cann, R., Kempson, R., & Marten, L. (2005). *The dynamics of language*. Oxford: Elsevier.

Chomsky, N. (1995). *Minimalist program*. Cambridge: MIT Press.

Dalrymple, M. (2001). *Lexical functional grammar*. New York: Academic Press.

Harada, S.-I. (1973). Counter equi-NP deletion. *Annual Bulletin Research Institute of Logopaedics and Phoniatrics, 7,* 113–148.

Kempson, R., Cann, R., & Otsuka, M. (2002). *On left and right dislocation*. Edinburgh: Ms., University of Edinburgh.

Kempson, R., Gregoromichelaki, E., & Howes, C. (Eds.). (2011). *The dynamics of lexical interfaces*. Stanford: CSLI Publications.

Kempson, R., & Kiaer, J. (2010). Multiple long-distance scrambling. *Journal of Linguistics, 46,* 127–192.

Kempson, R., Meyer-Viol, W., & Gabbay, D. (2001). *Dynamic syntax*. Oxford: Wiley-Blackwell.

Kiaer, J. (2014). *Pragmatic syntax*. London: Bloomsbury Publishing.

Kishimoto, H. (2004). Transitivity of ergative-marking predicates in Japanese. *Studies in Language, 28,* 105–136.

Kishimoto, H. (2017). Case marking. In M. Shibatani, et al. (Eds.), *The handbook of Japanese syntax* (pp. 447–495). Berlin: De Gruyter Mouton.

Koizumi, M. (2008). Nominative object. In S. Miyagawa & M. Saito (Eds.), *The Oxford handbook of Japanese linguistics* (pp. 141–164). Oxford: Oxford University Press.

Kuno, S. (1973). *The structure of the Japanese language*. Cambridge: MIT Press.

Kuno, S. (1976). Subject raising. In M. Shibatani (Ed.), *Syntax and semantics* (Vol. 5, pp. 17–41). New York: Academic Press.

Kuno, S. (1987). *Danwa-no bunpou. (Grammar of discourse)*. Tokyo: Taishukan Publishing Company.

Kuroda, S.-Y. (1992). What can Japanese say about government and binding? In S.-Y. Kuroda (Ed.), *Japanese syntax and semantics* (pp. 40–52). Dordrecht: Kluwer.

Nakamura, H., Yoshimoto, K., Mori, Y., & Kobayashi, M. (2009). Multiple subject construction in Japanese. In H. Hattori, et al. (Eds.), *New frontiers in artificial intelligence* (Vol. 5447, pp. 103–118). Dordrecht: Springer.

Nambu, S., Hwang, H.-K., Oshima, D., & Nomura, M. (2018). The nominative/accusative alternation in Japanese and information structure. *Journal of East Asian Linguistics, 27,* 141–171.

NKK (Nihongo Kijutsubunpou Kenkyuukai). (2009). *Gendai nihongo bunpou. (The grammar of modern Japanese)* (Vol. 2). Tokyo: Kuroshio Publishers.

Noda, T. (1996). *"Wa" to "ga". ("Wa" and "ga")*. Tokyo: Kuroshio Publishers.

Purver, M., Gregoromichelaki, E., Meyer-Viol, W., & Cann, R. (2010). Splitting the 'I's and crossing the 'you's. In P. Łupkowski & M. Purver (Eds.), *Aspects of semantics and pragmatics of dialogue* (pp. 43–50). Poznań: Polish Society for Cognitive Science.

Seraku, T. (2013). *Clefts, relatives, and language dynamics*. Ph.D. diss., the University of Oxford.

Seraku, T. (2016). A "maximal exclusion" approach to structural uncertainty in dynamic syntax. In *Proceedings of the 30th Pacific Asia conference on language, information, and computation* (pp. 39–47).

Seraku, T., & Ohtani, A. (2016). *Wh*-licensing in Japanese right dislocations. In C. Piñón (Ed.), *Empirical issues in syntax and semantics* (Vol. 11, pp. 124–132). Paris: CSSP.

Shibatani, M. (1977). Grammatical relations and surface cases. *Language, 53,* 789–809.
Shibatani, M. (1978). *Nihongo-no bunseki. (The analysis of Japanese)*. Tokyo: Taishukan Publishing Company.
Tada, H. (1992). Nominative objects in Japanese. *Journal of Japanese Linguistics, 14,* 91–108.
Takano, Y. (2003). Nominative objects in Japanese complex predicate constructions. *Natural Language & Linguistic Theory, 21,* 779–834.