Processing as abduction+deduction: A sentence processing model

Shravan Vasishth[1] & Geert-Jan M. Kruijff[2]

[1] Department of Linguistics
The Ohio State University, Columbus OH

[2] Institute of Formal & Applied Linguistics, Faculty of Mathematics & Physics
Charles University, Prague Czech Republic

{vasishth@ling.ohio-state.edu, gj@ufal.ms.mff.cuni.cz}

This paper provides an account for the processing complexity of center-embedded constructions (CECs) in Japanese and Dutch. We present a model of sentence processing based on abductive and deductive inference, and show that the model makes the correct predictions for a wide array of data from these two languages. The model extends to CECs in German, Korean, and Hindi.

The structure of the paper is as follows. Section 1 introduces the problem posed by CECs for natural language; Section 2 outlines the main proposal: an algorithm, and a complexity metric; Section 3 illustrates the operation of the model by giving several derivations for the Japanese and Dutch facts; and Section 4 concludes the paper.

1 Introduction

Experiments with English (e.g., [Blu66]) have shown that CECs like (1a) take longer for humans to process than right-embedded constructions like (1b).

(1) (a) The salmon [that the man [that the dog chased] smoked] fell off the grill.
(b) The dog chased the man [that smoked the salmon [that fell off the grill]].

The problem with processing CECs is that the person hearing the sentence must temporarily store partial constituents of the sentence (e.g., in (1a)), so for example the NP *The salmon* must be stored while the embedded constituent is processed.

Miller [Mil62] and Miller and Chomsky [MC63] were the first to investigate the difficulty of comprehending English CECs like (1), and Chomsky [Cho59] proved that natural languages in general are not context-free because they have the property of arbitrary center-embedding (also see [HU79]).

Clearly some kind of constrained parsing comes into play in human sentence processing. The exact nature of this parsing mechanism has been hard to pin down, especially when facts about CECs from different languages are taken into consideration. In response to this issue of providing a cross-linguistically robust account of CEC processing, several experimental studies have targetted Dutch, German, Japanese, and Korean (e.g., [BBMW86], [DV97], [Lew98], [BG99], and [Ueh99], [HR97]), and as a result we now have a body of interesting, empirically determined facts about these languages.

Moreover, several theories that address these results have also been proposed. In particular, Babyonyshev and Gibson [BG99] present a theory based on integration cost and memory cost; and Lewis [Lew98] develops a similarity-based interference theory that relies on constraints on working memory during comprehension.[1]

The present model is a contribution to the theoretical side of the CEC processing question. We present a theory that (i) allows incremental (semantic and syntactic) processing (regarding the relevance of incremental processing, see [Mil92], [Mil94], and [Mil95], [Pic94], [Kaa97], [Kon96], [Ino91], and [Yam94], among others), (ii) has cross-linguistic coverage, and (iii) takes into account both working memory constraints and constraints on the parsing mechanism.

2 Processing as abduction+deduction: The main proposal

The *processing* model we propose for explaining the processing complexity of center-embedded constructions is based on a combination of *abduction* and *deduction*. And, as we are aiming at providing an explanation for processing complexity *as experienced by human language users*, the processing model provides an *incremental* perspective on sentence processing. The aim of this section is to explain what we understand by incrementality, abduction, deduction, and how these all fit together into a single model of human language processing.

The basic idea is as follows. We assume that we have a grammar, $\mathcal{G}$, for a particular natural language, $\mathcal{L}$. $\mathcal{G}$ defines what types of functional categories (or, alternatively, predicate-valency frames) we can encounter in $\mathcal{L}$, and how these functions can combine with their arguments. Throughout this paper we will assume that $\mathcal{G}$ is a categorial grammar, and that the basic types of functional categories can be derived directly from $\mathcal{G}$'s lexicon.

From $\mathcal{G}$'s lexicon we can extract the set of types of functional categories. Disregarding the specific words that each of these functional categories have been assigned to in the lexicon, we can consider this set essentially as providing us with a set of "templates" elucidating how words (of particular categories) can be combined. We regard this set as our set of *hypotheses*, $\mathcal{H}$. We employ $\mathcal{H}$ in the following way.

When we (humans) process a sentence, we do so by starting at the beginning of the sentence, and proceeding word by word towards the end of the sentence.[2] In most languages, it may happen that -when processing a sentence- we encounter a word that is obviously an argument to another word, before we have encountered the actual function into which the word-as-argument may fit.

The processing model we propose is based on the idea that, whenever we encounter such a word that is believed to be an argument of an as yet unseen function, we assume a *hypothetical* function that would *explain* the occurrence of that word as a (projected) argument. For example, if we encounter a noun in nominative case before we have encountered a verb, we assume a verbal category that would -at least- take a noun in nominative case as its argument. It is our set $\mathcal{H}$ that provides us with the possible candidates, since $\mathcal{H}$ includes all the (basic) types of functions that we can conceivably encounter, given $\mathcal{G}$.

Subsequently, whenever we do encounter a verb, the processing model stipulates that what we do is try to match a hypothesized function against the actual functional category of the verb. If there is a match, or if the verbal category subsumes the hypothesized function, then we can instantiate the hypothesis

[1] This is by no means an exhaustive list– we mention these theories because they are the two theories in the area with the largest empirical coverage.

[2] Computers do not necessarily have to do so - for example when using head-corner parsing algorithms.

as the encountered verbal category, and compose the verb with the noun as its argument.

Abduction, then, is here understood as the kind of reasoning we can use to advances a hypothetical function to explain the occurrence of an argument, acting on the assumption that we are trying to process a grammatical sentence. Deduction is used in the traditional (CG) sense, as the means to subsequently try to *compose* an actually encountered function and any available, suitable argument(s). The account of processing *complexity* arises from the number of hypotheses we have to assume, and how difficult it is to match them against the functional categories of observed words.

At this point, the attentive reader may want to raise two issues: How does the approach outlined above differ in any way from a standard CG approach using type-raising, and doesn't incrementality entail strictly right-branching structures? To start with the latter, we believe that the idea of strictly right-branching is not a fact but an *arte*fact of particular theories of syntax. We completely agree with Moortgat, Morrill, and Steedman who argue that syntactic structure is essentially dispensable, and that it does not constitute a level of representation (cf. [Ste00]).

As for the former, it is true that an account based on type-raising *could* account for some of the phenomena. Obviously, we can have the following derivation (in CCG):

$$\dfrac{\dfrac{\dfrac{[nom]N}{S/(S \setminus [nom]N)}\,{>}T \quad \dfrac{[acc]N}{(S \setminus [nom]N)/(S \setminus [nom]N) \setminus [acc]N}\,{>}T}{S/(S \setminus [nom]N) \setminus [acc]N}\,{>}B \quad \dfrac{S}{S \setminus [nom]N \setminus [acc]N}}{S}\,{>}B$$

However, Baldridge provides in [Bal98] a detailed account of why type-raising, when placed in the context of grammars for free word order languages, leads to overgeneration. Given the space restrictions we cannot repeat Baldridge's discussion here, but we take it to provide us with a sufficient argument against the idea that our entire approach could thus be "suitably" rephrased purely in terms of type-raising.

In the next subsections we discuss the notion of abduction in some more detail, and we present the processing model itself, including the complexity metric.

2.1 Abduction

The contemporary understanding of abduction, as a third form of logical reasoning next to deduction and induction, is usually traced back to its discussion by the American logician, Charles S. Peirce [Kru95, Kru98]. Peirce defined abduction as the following kind of inference:

> *A surprising phenomenon O is observed;*
> *but if H were to be the case, then O would follow as a matter of course.*
> *Therefore, there is reason to believe that H.*

Thus, whereas (intuitively speaking) deduction derives a consequence from given axioms, and induction establishes a rule or generalization, abduction proposes an *explanation* for a *surprising* observation.

The surprise is the key, here. Being surprised means that either (a) we did not predict to observe (anything like) O at all, or (b) we did predict to make some observation, but we did not expect to observe

O. On (a), our knowledge is incomplete, whereas on (b), our knowledge is in some way incorrect. Either way, we cannot rely (entirely) on our knowledge then to create a hypothesis explaining O, for that would otherwise negate the point that we were surprised - if we knew all along that O would happen, how come we got surprised?[3]

Peirce's claim was that only through abduction could we obtain genuinely new knowledge (on the assumption we would let ourselves be surprised, and acknowledge that fact).

Here, like in AI in general, we take in a substantially weaker (but more workable) position. We assume that we have at our disposal all hypotheses that could be possibly abduced.[4] For our application, the set is assumed to be the smallest set that is "necessary and sufficient" given a grammar/lexicon - the set $\mathcal{H}$ discussed above. We assume that $\mathcal{H}$ is finite and closed.[5]

When we are working with categorial grammars (as we do here), we obtain $\mathcal{H}$ as follows. $\mathcal{H}$ is created on the basis of a compilation of the lexicon - which, in a lexicalist approach like categorial grammar, determines how words can be put together. Structural rules, like the combinators in Combinatorial Categorial Grammar (CCG) [Ste00] or the meaning postulates in Multi-Modal Logical Grammar (MMLG) [Moo97], only vary the order in which words occur grammatically.

The compilation of the lexicon is based on a procedure proposed for linear logic in [Hep98], and extended in [Kru99] to cover a larger range of multiplicative resource logics used in MMLG. Originally, compilation was proposed for the purposes of efficient chart parsing with Lambek-style grammars, in order to overcome problems with earlier approaches (e.g., [Hep92] and [Kön90]). The result of the procedure is a set of first-order functions to represent categories (i.e., there are no higher-order formulas).

Once we have a compiled version of the lexicon, we abstract away from individual words, and retain the different functional categories that are defined. Taken together these functional categories make up $\mathcal{H}$.

2.2 Some definitions

Next we define some terms that we use in the proposed algorithm.

Abducible structure(s):

An ABDUCIBLE STRUCTURE is a hypothesis based on the information available so far; no more hypotheses are selected than are justified by the information available up to a certain point (cf. [Kon96]). New information results in replacement (or refinement) of previous hypotheses. The abduced functions f_i that are part of the abducible structures are taken from $\mathcal{H}$, and thus posit the presence of a word with a particular syntactic category. For example, in Japanese, if only a nominative N has appeared so far, the f_i in $f_i(N[nom])$ is a syntactic hypothesis that says: an intransitive verb with the nominative N will give a sentence.[6]

Note that although a nominative case marked N is in principle consistent with an infinite set of possible continuations, our model selects only that set of hypotheses from the hypothesis space $\mathcal{H}$ that are minimally consistent with the nominativeN. In the above example, $f_i([nom]N, x)$, where

[3]For that reason, Peirce advanced the idea that a hypothesis is created by a "guessing instinct" because we cannot rely on reasoning from our knowledge as such. In this context it is perhaps interesting to note that not only Peirce believed in there being a fundamental role for something like a "guessing instinct" in logic. Gödel also did– cf. [Par95], and the brief comparison between Gödel's ideas and Peirce's in [Kru97].

[4]Thus, what [Kru95] refers to as the "demise of surprise" in AI.

[5]This is not to confused with the fact that the set of sentences is infinite.

[6]The significance of the subscript will soon become clear.

x is a hypothesized N, is not minimally consistent with the information available so far (i.e., the nominative NP). By contrast, if the first nominative N is followed by a second nominative N, the minimally consistent hypotheses are now $f_i([nom]N1, [nom]N2)$, where f_i is a stative verb, and $f_{i+1}([nom]N1, f_{i+2}([nom]N2))$, i.e., a center embedded structure.

Matching:

A verb V MATCHES with a function f_i iff V has a functional category that is consistent with that of f_i.

An NP can match with a posited NP argument as long as its case marking, person, number, gender, and other information, is consistent with that of the posited argument.

2.3 The algorithm

The details of the interface of the performance metric discussed in this paper will be spelt out in a subsequent version of this paper. For the moment, we focus on the processing issue per se.

The processing algorithm works as follows. First of all, we distinguish the following steps:

Init: Set the structure $\mathfrak{S}$ to $\emptyset$, set the scanning pointer to position 0.

Scan: Scan the next word w_i, moving the pointer to the next position.

Lookup: Lookup the scanned word w_i in the lexicon of $\mathcal{G}$.

Process: If $\mathfrak{S} = \emptyset$, then check the category C of w_i. If C is a function category C, then $\mathfrak{S} = \mathfrak{S} \cup \{\mathsf{C}\}$ else $\mathfrak{S} =$ **abduce**$(\mathcal{H}, \mathsf{C}, \mathfrak{S})$. Else $\mathfrak{S} \neq \emptyset$, if the category of w_i is a function category C, then try $\mathfrak{S} =$ **deduce**$(\mathsf{C}, \mathfrak{S})$ or failing that $\mathfrak{S} = \mathfrak{S} \cup \{\mathsf{C}\}$. Else [$\mathsf{C}$ is not a functional category], try $\mathfrak{S} =$ **deduce**$(\mathsf{C}, \mathfrak{S})$ or failing that $\mathfrak{S} =$ **abduce**$(\mathcal{H}, \mathsf{C}, \mathfrak{S})$. If the latter step fails, or we arrive at the letter step and w_i is the last word in the sentence, then we FAIL.

Deduce: Given a category C and a structure $\mathfrak{S}$, if C is an argument then try to combine it with a hypothesis or function in $\mathfrak{S}$, starting with the outermost hypothesis/function first (FIFO). Else, C is a function, and try to match it against a hypothesis h in $\mathfrak{S}$, such that C is either equal to h or subsumes it. Failing all that, throw an exception stating that the word with category C cannot be combined with anything in the structure.

Abduce: Given a set of possible hypotheses $\mathcal{H}$, a category C, and a structure $\mathfrak{S}$, find the simplest hypothesis h in $\mathcal{H}$ that takes C as an argument, and which can be combined with $\mathfrak{S}$ either as an argument of a hypothesis/function in $\mathfrak{S}$, or as a function taking the outermost hypothesis/function in $\mathfrak{S}$ as its argument. If no such hypothesis h can be found, then FAIL. Otherwise, integrate C and h into $\mathfrak{S}$ and return the updated structure.

Processing then starts with **Init**. Subsequently, we make cycles **Scan**-**Lookup**-**Process**, either until we FAIL or until we arrive to the end of the sentence. If the structure $\mathfrak{S}$ contains no unmatched hypotheses, then the sentence is grammatical and can be assigned $\mathfrak{S}$; otherwise, the sentence is considered ungrammatical on $\mathcal{G}$.

To repeat an earlier example from Japanese: two nominative case marked NPs starting a sentence could be followed either by a stative predicate (that is, $f_2([nom]N1, [nom]N2)$), or a nested dependency construction with a single level of embedding $(f_3([nom]N1, f_4([nom]N2)))$– these are two hypotheses selected from $\mathcal{H}$. No other hypotheses are selected because these are the only two that are "justified", given the information so far. These hypotheses are based on the grammatical possibilities in Japanese, and since a single clause sentence has a simpler structure than a sentence with an embedded clause, the hypotheses are ordered as shown below: $f_2([nom]N1, [nom]N2)$, $f_3([nom]N1, f_4([nom]N2))$. Then, given the hypotheses $f_2([nom]N1, [nom]N2)$, $f_3([nom]N1, f_4([nom]N2))$, the appearance of an accusative case marked N will result in the next hypothesis being $f_5([nom]N1, f_6([nom]N2, [acc]N3))$, the previous ones being discarded.

2.3.1 The complexity metric

The complexity metric has three components: ABDUCTION COST, the total number of (incrementally) abduced functions (f_i), and MISMATCH COST, a cost associated with a mismatch between an encountered verb and abduced functions, and SIMILARITY BASED INTERFERENCE, a constraint on working memory (Lewis 1999).

Abduction cost: This reflects to the increasing processing load as sentence fragments appear incrementally. Since we discard or refine hypotheses as processing progresses, the hypotheses are assumed to take up resources, and therefore contribute to the total processing cost.[7]

Mismatch cost: We assume that the (queued) hypotheses are unanalyzable units at first, and that when a word appears, the hypotheses have to be examined as a whole– hence the left to right depth first search.

Similarity based interference: Following Lewis (in progress), we assume that two or more verbs with the same number of arguments will cause similarity based interference in working memory, adding a cost to processing.

The numerical value associated with the operation of each component in the metric is assumed to be 1. This is merely a convenience, and nothing crucial hinges on this assumption. In a fully developed version of this model, the unit costs associated with each component will probably have to be adjusted.

In the next section, we provide some illustrations of the empirical coverage of this processing model. Our data is from experiments conducted by others for Japanese and Dutch.

3 The empirical coverage

3.1 Japanese

Note that in the following discussion, the verbs in nested sentences are numbered in reverse order of occurrence, i.e., the matrix verb, which appears last, is number V1. The numbers do not reflect the verbs' valencies; this reverse numbering convention is merely in order to highlight the difference from Dutch (discussed later).

3.1.1 Gibson's (1998) data

Gibson [Gib98] has experimentally determined that (2a) is harder than (2b).

(2) (a) [NP1 [NP2 [NP3 NP4 V3] V2] V1]

[7] Maybe we need to incorporate abduced NPs etc. too in our count, but for the purposes of this paper we will ignore them.

[obasan-ga	*[bebiisitaa-ga*	*[ani-ga*	*imooto-o*	*izimeta-to]*	*itta-to]*
aunt-nom	babysitter-nom	brother-nom	sister-acc	teased-comp.	said-comp.

omotteiru]
thinks

'The aunt thinks that the babysitter said that the elder brother teased the younger sister.'

(b) [[NP1 [NP2 NP3 V3] V2] NP4 V1]

[[bebiisitaa-ga	*[ani-ga*	*imooto-o*	*izimeta-to]*	*itta-to]*	*obasan-ga*
babysttr.-nom	brother-nom	sister-acc	teased-comp.	said-comp.	aunt-nom

omotteiru]
thinks

'The aunt thinks that the babysitter said that the elder brother teased the younger sister.'

First, consider the application of the algorithm for (2a):

Step 1:

Input	Abduction/deduction	Cost
NP1-ga	f_1(NP1)	1+0+0

Here, given only the first NP (*obasan-ga*), a sentence with an intransitive verb (IV), denoted by f_1, is abduced. This contributes 1 to our cost so far (abduction cost; the other cost components are currently 0).

Step 2:

Input	Abduction/deduction	Cost
NP1-ga	f_1(NP1)	1+0+0
NP2-ga	f_2(NP1,NP2)	
	f_3(NP1,f_4(NP2))	4+0+0

Given the second NP (bebisitaa-ga), and given that both the NPs seen so far are nominative case marked, the abducible structures are: a stative predicate taking two nominative arguments (f_2(NP1,N2)), and a center embedded construction (f_3(N1,f_4(N2))). To our cost we add 1 for every function abduced, so the total cost so far is 4.

Step 3:

Input	Abduction/deduction	Cost
NP1-ga	f_1(NP1)	1+0+0
NP2-ga	f_2(NP1,NP2)	
	f_3(NP1,f_4(NP2))	4+0+0
NP3-ga	f_5(NP1,f_6(NP2,NP3))	
	f_7(NP1,f_8(NP2,f_9(NP3))	9+0+0

We now have three nominative NPs, and so we either have an embedded stative predicate, as in f_5(NP1,f_6(NP2,NP3)), or a series of center embeddings, as in f_7(NP1,f_8(NP2,f_9(NP3))). Since we added five more functions, our total cost so far is 9.

Step 4:

Input	Abduction/deduction	Cost
NP1-ga	f_1(NP1)	1+0+0
NP2-ga	f_2(NP1,NP2)	
	f_3(NP1,f_4(NP2))	4+0+0
NP3-ga	f_5(NP1,f_6(NP2,NP3))	
	f_7(NP1,f_8(NP2,f_9(NP3))	9+0+0
NP4-o	f_{10}(NP1,f_{11}(NP2,f_{12}(NP3,NP4)))	12+0+0

f_{10}(NP1,f_{11}(NP2,f_{12}(NP3,NP4))) is abduced because the fourth NP is marked with accusative case, and so there must be at least a double embedding with a transitive embedded verb. Since we introduced three more functions, the cost now is 12. Further, the number of hypotheses in working memory has been reduced from two to one, we predict a fall in reading time at NP4. We do not have reading time data for exactly the above kind of data yet, but the model also make a similar (correct) prediction for data in [Yam97], which we discuss further on.

Step 5:

Input	Abduction/deduction	Cost
NP1-ga	f_1(NP1)	1+0+0
NP2-ga	f_2(NP1,NP2)	
	f_3(NP1,f_4(NP2))	4+0+0
NP3-ga	f_5(NP1,f_6(NP2,NP3))	
	f_7(NP1,f_8(NP2,f_9(NP3))	9+0+0
NP4-o	f_{10}(NP1,f_{11}(NP2,f_{12}(NP3,NP4)))	12+0+0
V3	f_{10}(NP1,f_{11}(NP2,V3(N3,NP4)))	12+2+0

Here, the next word is *izimeta-to*, and the above deduction is effected in the following manner:

(i). V3 tries to match f_{10} in

f_{10}(NP1,f_{11}(NP2,f_{12}(NP3,NP4))) $\Rightarrow$ failure.

This matching attempt fails because the outermost function has a valency frame that doesn't match the actual verb's.

(ii). V3 tries to match f_{11} in

f_{10}(NP1,f_{11}(NP2,f_{12}(NP3,NP4))) $\Rightarrow$ failure.

Here, again, the failure occurs due to the valency frames of the verb not matching that of the next function.

(iii). V3 tries to match f_{12} in

f_{10}(NP1,f_{11}(NP2,f_{12}(NP3,NP4))) $\Rightarrow$ f_{10}(NP1,f_{11}(NP2,V3(N3,NP4)))

This succeeds because the valency frame of the verb matches that of the next function.

The cost now is the number of abduced functions so far (12) plus the number of failed matches (2): 14.

Step 6:

Input	Abduction/deduction	Cost
NP1-ga	f_1(NP1)	1+0+0
NP2-ga	f_2(NP1,NP2)	
	f_3(NP1,f_4(NP2))	4+0+0
NP3-ga	f_5(NP1,f_6(NP2,NP3))	
	f_7(NP1,f_8(NP2,f_9(NP3))	9+0+0
NP4-o	f_{10}(NP1,f_{11}(NP2,f_{12}(NP3,NP4)))	12+0+0
V3	f_{10}(NP1,f_{11}(NP2,V3(N3,NP4)))	12+2+0
V2	f_{10}(NP1,V2(NP2,V3(NP3,NP4)))	12+3+2

The deductive process goes as follows:

(i). V2 tries to match f_{10} in

f_{10}(NP1,f_{11}(NP2,V3(N3,NP4))) $\Rightarrow$ failure.

(ii). V2 tries to match f_{10} in

f_{10}(NP1,f_{11}(NP2,V3(N3,NP4))) $\Rightarrow$ f_{10}(NP1,V2(NP2,V3(NP3,NP4)))

V2 fails to match f_{10}, but successfully matches f_{11}. We add 1 to the previous cost (14) due to the failed match of V2 with f_{10}. The two similar verbs result in an increase of the cost by 2 due to interference–they have the same number of arguments.[8]

Step 7:

Input	Abduction/deduction	Cost
NP1-ga	f_1(NP1)	1+0+0
NP2-ga	f_2(NP1,NP2)	
	f_3(NP1,f_4(NP2))	4+0+0
NP3-ga	f_5(NP1,f_6(NP2,NP3))	
	f_7(NP1,f_8(NP2,f_9(NP3))	9+0+0
NP4-o	f_{10}(NP1,f_{11}(NP2,f_{12}(NP3,NP4)))	12+0+0
V3	f_{10}(NP1,f_{11}(NP2,V3(N3,NP4)))	12+2+0
V2	f_{10}(NP1,V2(NP2,V3(NP3,NP4)))	12+3+2
V1	V1(NP1,V2(NP2,V3(NP3,NP4)))	12+3+3

[8]As mentioned earlier, the numerical values of the costs are of secondary importance. We do not claim that each computational procedure always has a cost of one. Whatever these costs actually turn out to be for human computation, they will apply across the board to all derivations, so this does not affect the model.

The deduction in this case is immediate:

V1 tries to match f_{10} in

f_{10}(NP1,V2(NP2,V3(N3,NP4))) $\Rightarrow$ V1(NP1,V2(NP2,V3(NP3,NP4)))

Here, V1 matches the outermost abduced function f_{10} immediately, and the parse is completed. Interference by V3 contributes to the increase in total processing cost, and the total cost now is 18.

By contrast, (2b)'s processing yields a lower cost of 11:

Step	Input	Abduction/deduction	Cost
1	NP1-ga	f_1(NP1)	1+0+0
2	NP2-ga	f_2(NP1,NP2)	
		f_3(NP1,f_4(NP2))	4+0+0
3	NP3-o	f_5(NP1,f_6(NP2,NP3))	6+0+0
4	V3	f_5(NP1,V3(NP2,NP3))	6+1+0
5	V2	f_7(x,V2(NP1,V3(NP2,NP3)))	7+1+2
6	NP4-ga	f_7(NP4,V2(NP1,V3(NP2,NP3))	7+1+2
7	V1	V1(NP4,V2(NP1,V3(NP2,NP3))	7+1+3

Table 1: (2b)

Note that in Step 5 above, the appearance of an embedded verb results in an abduced hypothesis involving a matrix verb and a nominal argument. This is because V2 has the complementizer *-to*, which requires it to be an embedded verb.

3.1.2 Nakatani et al. (2000)

[NBG00] conducted several off-line questionnaire studies with Japanese, and their data may be summarized as follows:[9]

Fact 1: Double-nesting is harder than left branching

(3) (a) *[obasan-wa [bebiisitaa-ga [imooto-ga naita-to] itta-to] omotteiru]*
aunt-top babysitter-nom sister-nom cried-comp. said-comp. thinks
'The aunt thinks that the babysitter said that the younger sister cried.'

(b) *[imooto-ga naita-to] bebiisitaa-ga itta-to] obasan-wa omotteiru]*
sister-nom cried-comp. babysitter-nom said-comp. aunt-top thinks
'The aunt thinks that the babysitter said that the elder brother teased the younger sister.'

Our model makes the correct prediction about this set of examples, as the following two derivations show.

[9]Note: the English glosses are sometimes different from [NBG00].

Step	Input	Abduction/deduction	Cost
1	NP1	f_1(NP1)	1+0+0
2	NP2	f_2(NP1,NP2)	
		f_3(NP1,f_4(NP2))	4+0+0
3	NP3	f_5(NP1,f_6(NP2,NP3))	
		f_7(NP1,f_8(NP2,f_9(NP3)))	9+0+0
4	V3	f_7(NP1,f_8(NP2,V3(NP3)))	9+4+0
5	V2	f_7(NP1,V2(NP2,V3(NP3)))	9+5+0
6	V1	V1(NP1,V2(NP2,V3(NP3)))	9+5+2=16

Table 2: Double nesting, 3NPs (3a)[10]

Step	Input	Abduction/deduction	Cost
1	NP1-ga	f_1(NP1)	1+0+0
2	V3	f_2(V3(NP1),x)	2+0+0
3	NP2-ga	f_2(V3(NP1),NP2)	2+0+0
4	V2	f_3(V2(V3(NP1),NP2),y)	3+0+0
5	NP3-ga	f_3(V2(V3(NP1),NP2),NP3)	3+0+0
6	V1	V1(V2(V3(NP1),NP2),NP3)	3+0+2 = 5

Table 3: Left branching, 3NPs (3b)

Due to similarity based interference, we expect a slowdown at the last verb; this is borne out by Lewis and Nakayama's experiments for Japanese [LN99], and by [KV00] for Dutch.

Fact 2: In double embeddings intransitive V3 easier than transitive V3

These are examples like (4):

(4) (a) *haha-ga* *titi-ga* *fukigen-na* *akatyan-ga* *naita-to* *itta-to* *omotteiru*
mother-nom father-nom fussy baby-nom cried-comp. said-comp. thinks
'My mother thinks that my father said that the fussy baby cried.'

(b) *obasan-ga* *syoojiki-na* *bebisitaa-ga* *ani-ga* *imooto-o* *izimeta-to*
aunt-nom honest babysitter-nom brother-nom sister-acc teased-comp.
itta-to *omotteiru*
said-comp. thinks
'My aunt thinks that the honest babysitter said that my brother teased my sister.'

(4)a has cost 16 and (4)b has cost 18. See earlier derivations (Tables 2 and the full derivation for (2a)) respectively).

Note that this difference is by no means predictable from the fact that the easier one has an intransitive verb and the harder one a transitive verb– Hagstrom et al. [HR97] found in a pilot study that there was no such difference in the case of similar structures in Korean.

[10]In examples like (3a), we predict a fall in reading time at V3 due to a hypothesis being eliminated. We do not have any data yet to confirm or disconfirm this prediction.

Fact 3: There was no significant difference between Accusative V3 and Dative V3

Examples of sentences with V3 taking an accusative vs. dative NP argument are shown below:

(5) (a)

tuma-wa	*uranaisi-ga*	*otto-ga*	*kakarityoo-o*	*oiniku-to*
wife-top	fortune-teller-nom	husband-nom	chief-clerk-acc	surpass-comp.
tugeta-to	*ziman-sita*			
told-comp.	boasted			

‘The wife boasted that the fortune-teller said that the husband would surpass the chief clerk.’

(b)

tuma-wa	*uranaisi-ga*	*otto-ga*	*kakarityoo-ni*	*narabu-to*
wife-top	fortune-teller-nom	husband-nom	chief-clerk-dat	catch-up-comp.
tugeta-to	*ziman-sita*			
told-comp.	boasted			

‘The wife boasted that the fortune-teller said that the husband would catch up with the chief clerk.’

(5a) and (5b) both have costs 18 (see derivation for (2a)): in the *-ni* case, a verb with a *-ni* marked argument is posited, rather than a a verb with an *-o* marked argument.

Fact 4: Singly nested, 5 NP stack easier than doubly nested, 3-4 NP stacks

The relevant examples are given below and the derivation for (6a) is shown in Table 4.

(6) (a)

tuma-wa	*kakarityoo-ni*	*uranaisi-ga*	*otto-ni*	*seekoo-o*
wife-nom	chief-clerk-dat	fortune-teller-nom	husband-dat	success-acc
yakusoku-sita-to	*ziman-sita*			
promised-comp.	boasted			

‘The wife boasted to the chief clerk that the fortune-teller promised the husband that he’d succeed.’

(b)

haha-ga	*titi-ga*	*fukigen-na*	*akatyan-ga*	*naita-to*	*itta-to*	*omotteiru*
mother-nom	father-nom	fussy	baby-nom	cried-comp.	said-comp.	thinks

‘My mother thinks that my father said that the fussy baby cried.’

(c)

obasan-ga	*syoojiki-na*	*bebisitaa-ga*	*ani-ga*	*imooto-o*	*izimeta-to*
aunt-nom	honest	babysitter-nom	brother-nom	sister-acc	teased-comp.
itta-to	*omotteiru*				
said-comp.	thinks				

‘My aunt thinks that the honest babysitter said that my brother teased my sister.’

Input	Abduction/deduction	Cost
NP1	f_1(NP1)	1+0+0
NP2	f_2(NP1, NP2)	2+0+0
NP3	f_3(NP1, NP2,f_4(NP3))	4+0+0
NP4	f_5(NP1, NP2,f_6(NP3,NP4))	6+0+0
NP5	f_7(NP1, NP2,f_8(NP3,NP4,NP5))	8+0+0
V2	f_7(NP1, NP2,V2(NP3,NP4,NP5))	8+1+0
V1	V1(NP1, NP2,V2(NP3,NP4,NP5))	8+1+1

Table 4: (6a)

In contrast to (6a), which has cost 10, (6b) has cost 16 as shown in Table 2, (6c) has cost 18, as in the first derivation given earlier for (2a).

Fact 5: Singly nested, 5 NP stack with preceding adjunct clause easier than doubly nested, 3-4 NP stacks

Consider the examples below:

(7) (a) *syaintati-ga* *taisya-sita* *ato* *tuma-wa* *kakarityoo-ni* *uranaisi-ga*
employees-nom get-off-work after wife-top chief-cler-dat fortune-teller-nom
otto-ni *seekoo-o* *yakusoku-sita-to* *ziman-sita*
husband-dat success-acc promised-comp. boasted
'After the employees got off work, the wife boasted to the chief clerk that the fortune-teller promised the husband that he'd succeed.'

(b) *haha-ga* *titi-ga* *fukigen-na* *akatyan-ga* *naita-to* *itta-to* *omotteiru*
mother-nom father-nom fussy baby-nom cried-comp. said-comp. thinks
'My mother thinks that my father said that the fussy baby cried.'

(c) *obasan-ga* *syoojiki-na* *bebisitaa-ga* *ani-ga* *imooto-o* *izimeta-to*
aunt-nom honest babysitter-nom brother-nom sister-acc teased-comp.
itta-to *omotteiru*
said-comp. thinks
'My aunt thinks that the honest babysitter said that my brother teased my sister.'

(7a) has cost 10+1= 11 (one more than (6a; see Table 4) since it just involves the positing of one more function (compared to the case where no adjunct clause exists) which the verb in the adjunct clause immediates matches with (7b) has 16, (7c) has 18.

Fact 6: Adding an adverb doesn't affect complexity

(8) (a) *tuma-wa* *uranaisi-ga* *otto-ga* *syoosin-suru-to* *tugeta-to*
wife-nom fortune-teller-nom husband-nom be-promoted-comp. told-comp.
ziman-sita
boasted
'The wife boasted that the fortune-teller said that her husband would be promoted.'

(b) *tuma-wa* *uranaisi-ga* *otto-ga* *ikinari* *syoosin-suru-to*
wife-nom fortune-teller-nom husband-nom suddenly be-promoted-comp.
tugeta-to *ziman-sita*
told-comp. boasted
'The wife boasted that the fortune-teller said that her husband would be promoted suddenly.'

This model does not consider adverbs in the complexity metric, there is no predicted increase in complexity due to the presence of adverbs.

In sum, the model correctly predicts the experimentally determined differences in complexity, as the table below indicates. $X > Y$ means that X is harder than Y, etc.

	Data	Model
(1a) vs. (1b)	$>$	$>$
(2a) vs. (2b)	$>$	$>$
(3a) vs. (3b)	$<$	$<$
(4a) vs. (4b)	$=$	$=$
(5a) vs. (5b)	$<$	$<$
(5a) vs. (5c)	$<$	$<$
(6a) vs. (6b)	$<$	$<$
(6a) vs. (6c)	$<$	$<$
(7a) vs. (7b)	$=$	$=$

3.1.3 Yamashita (1997)

Yamashita [Yam97] investigated the effect of word order and case marking on the processing of Japanese. One of her experiments is a moving window task involving three conditions:

Condition A. Canonical order, with 4NPs and 2 verbs:

[NP1-nom NP2-dat [NP3-nom NP4-acc V2] V1]

Condition B. Same structure as in Condition A, but scrambled NP3 and NP4:

[NP1-nom NP2-dat [NP4-acc NP3-nom V2] V1]

Condition C. Same structure as in Condition A, but scrambled NP1, NP2, NP3 and NP4:

[NP2-dat NP1-nom [NP4-acc NP3-nom V2] V1]

The results for Condition A are interesting in the context of the present model;[11] consider the example below.

(9)

[denwa-de	*hansamu-na*	*gakusei-ga*	*sensei-ni*	*[tumetai*	*koibito-ga*	*nagai*
phone-on	handsome	student-nom	teacher-dat	cold	girlfriend-nom	long

tegami-o	*yabutta-to]*	*itta]*
letter-acc	tore-comp.	said

'On the phone, a handsome student told the teacher that the cold-hearted girlfriend had torn the letter.'

Yamashita found that reading times rose steadily in such examples till the accusative marked NP, and then fell at the accusative NP; then, with each incoming verb, reading times rose again.

The present model predicts this pattern, as shown below.

Step	Input	Abduction/deduction	Cost
1	NP1-ga	f_1(NP1)	1+0+0
2	NP2-ni	f_2(NP1, NP2)	2+0+0
3	NP3-ga	f_3(NP1, NP2,f_4(NP3))	
		f_5(NP1, f_6(NP2, NP3))	6+0+0
4	NP4-o	f_7(NP1, NP2,f_8(NP3,NP4))	8+0+0
5	V2	f_7(NP1, NP2,V2(NP3,NP4))	8+1+0
6	V1	V1(NP1, NP2, V2(NP3,NP4))	8+1+0

Table 5: (9)

[11] In this paper, due to space constraints, we do not discuss the effect of word order variation.

Before step 4, the reading time is predicted to rise steadily. At step 4, a fall in reading time is predicted since the number of hypotheses in working memory falls from two to one.[12] After that, reading time rises again steadily.

3.2 Dutch: Kaan et al. (2000)

Turning next to Dutch, Kaan and Vasić [KV00] conducted several moving window studies and found the following.

Fact 1: Double embeddings harder than single embeddings

Examples of each type are shown below:

(10) (a) *De leider heeft Paul Sonya het kompas helpen leren gebruiken tijdens de bergtocht*
the leader has Paul Sonya the compass help teach use during the hike
'The leader helped Paul teach Sonya to use the compass during the hike.'

(b) *Mit aanwijzingen van de leider heeft Paul Sonya het kompas helpen gebruiken tijdens de bergtocht*
with directions of the leader has Paul Sonya the compass teach use during the hike
'With the leader's directions Paul taught Sonya to use the compass during the hike.'

Double embeddings have a cost of 21:

Step	Input	Abduction/deduction	Cost
1	NP1	f_1(NP1)	1+0+0
2	NP2	f_2(NP1,NP2), f_3(NP1,f_4(NP2))	4+0+0
3	NP3	f_5(NP1,NP2,NP3))	
		f_6(NP1,f_7(NP2,NP3))	
		f_8(NP1,f_9(NP2,f_{10}(NP3)))	10+0+0
4	NP4	f_{11}(NP1,f_{12}(NP2,NP3,NP4))	
		f_{13}(NP1,f_{14}(NP2,f_{15}(NP3,NP4)))	15+0+0
5	V1	V1(NP1,f_{14}(NP2,f_{15}(NP3,NP4)))	15+3+0
6	V2	V1(NP1,V2(NP2,f_{15}(NP3,NP4)))	15+3+2
7	V3	V1(NP1,V2(NP2,V3(NP3,NP4)))	15+3+3

Table 6: (10a)

Single embeddings have a lower cost, 13.

[12] Although the abduction cost increases at step 4 from 6 to 8, this is offset by the reduction in the number of hypotheses in working memory.

Step	Input	Abduction/deduction	Cost
1	NP1	f_1(NP1)	1+0+0
2	NP2	f_2(NP1,NP2), f_3(NP1,f_4(NP2))	4+0+0
3	NP3	f_5(NP1,NP2,NP3))	
		f_6(NP1,f_7(NP2,NP3))	
		f_8(NP1,f_9(NP2,f_{10}(NP3)))	10+0+0
4	V1	V1(NP1,f_7(NP2,NP3))	10+1+0
5	V2	V1(NP1,V2(NP2,NP3))	10+1+2

Table 7: (10b)

Fact 2: Increased RTs with each incoming NP, increased RTs from first verb to last, especially in 3 Verb condition

Kaan and Vasić showed that reading times increase with NPs and fall at V1, and reading times rise with each subsequent verb. The model predict all these facts.

In the present model, the NP reading times are predicted to rise due to the increase in the number of abduced functions, and a fall in reading time is predicted at the first verb is due to the elimination of some hypotheses from working memory (see derivations above to see how this happens exactly). After that, reading times increase.

Data	0 + + + + - + +
Model	0 + + + + - + +

Table 8: Reading times and model's predictions (schematic)

4 Concluding remarks

In this paper, we presented a hybrid abductive/deductive model of human language processing, based on existing psycholinguistic results. An important observation is that all the mechanisms proposed have correlates in other theories. Our contribution is to identify a general perceptual mechanism– abduction– as an appropriate primitive for developing a processing account, next to deduction as an abstract way to conceive of how people combine words when analyzing a sentence, given a particular grammar $\mathcal{G}$ for the relevant language $\mathcal{L}$.

The model fares better than existing accounts for the data considered here. For example, none of the existing theories can currently account for the fall in reading times at the accusative verb in Japanese, and at the first verb in Dutch; and Gibson's model [Gib98] appears to make incorrect predictions for the rising reading times for verbs (see [KV00] for details). However, it remains to be seen whether the predictions it makes are all borne out. For example, the model predicts that there will be a fall in reading time when the number of abduced hypotheses is reduced in working memory to one as a result of new incoming information. This happens to be the correct prediction for Yamashita's Japanese data and Kaan and Vasić's Dutch data, but we do not have enough data yet to determine whether this is prediction is borne out (for example) for (2b).

Further, we currently do not have a precise account for the scrambling facts (e.g., those presented in [Yam97]). One reason that we hesitate to extend our model for scrambling is that word order variation is almost always correlated with a particular discourse context, and yet studies on scrambling and processing like Yamashita's [Yam97] assume that processing of a scrambled sentence presented to subjects out of the blue (i.e., without any discourse context) can be compared with unscrambled correlates. Pilot sentence completion studies using Hindi, conducted by the first author, indicate that subjects reject scrambled clauses when present without any preceding discourse context, perhaps due to an inability to imagine an appropriate context for these. We must therefore await further empirical work before any valid conclusions can be drawn about the processing of scrambled sentences.

Also, as it currently stands, the model does not immediately extend to other structures, such as the processing of Japanese relative clauses within sentential complements and vice versa [BG99]. The model can be generalized to account for these constructions, but we leave this task for future work. The basic idea here is to extend the abductive procedure to take into account structural rules, besides composition, to account for an observed word/wordgroup.

Finally, in relation to other, similar accounts, we contend that our account is a useful generalization over accounts like the ones based on pushdown automata [Jos90], or incremental processing by prediction of minimum valency as proposed in work by Scheepers et al. Implicit in all these treatments is the idea of abductive inference. Our proposal foregrounds abduction, and demonstrates the considerable predictive power such foregrounding makes available to us. In this sense, our model is less a challenge to existing accounts than a reformulation of these in more general (although very precise) terms. In future work, we hope to broaden the scope of comparison to work like Steedman's CCG [Ste00] and Milward's model of incremental parsing of CG [Mil92, Mil94, Mil95].

5 Acknowledgments

Thanks to Rick Lewis for his detailed feedback, and to Chris Brew, Takao Gunji, Martin Jansche, Keith Johnson, Mineharu Nakayama, Carl Pollard, Shari Speer, and Mark Steedman for comments. At the time of writing, the second author was a visitor at the Institute of Communicating and Collaborative Systems (ICCS), Division of Informatics, University of Edinburgh, and would therefore like to acknowledge here and Mark Steedman's and Bonnie Webber's hospitality.

This paper will be presented in Japan at the Japanese Cognitive Science Meeting, 2000, at the Kobe Shoin Graduate School Colloquium series, and at the ESSLLI 2000 Linguistic Theory and Grammar Implementation Workshop to be held in Birmingham, UK. The authors alone are responsible for any errors.

References

[Bal98] Jason Michael Baldridge. Local Scrambling and Syntactic Asymmetries in Tagalog. Master's thesis, University of Pennsylvania, Philadelphia, PA, 1998.

[BBMW86] E. Bach, C. Baker, and W. Marslen-Wilson. Crossed and nested dependencies in German and Dutch: A psycholinguistic study. *Language and Cognitive Processes*, 1(4):249–262, 1986.

[BG99] Maria Babyonyshev and Edward Gibson. The complexity of nested structures in Japanese. *Language*, 75(3):423–450, 1999.

[Blu66] A. L. Blumenthal. Observations with self-embedded sentences. *Psychonomic Science*, 6:453–454, 1966.

[Cho59] Noam Chomsky. On certain formal properties of grammar. *Information and Control*, 2:137–167, 1959.

[DV97] Michael Walsh Dickey and Wietske Vonk. Center-embedded structures in Dutch: An on-line study. ms., Northwestern University and Max Planck Institute for Psycholinguistics, 1997.

[Gib98] Edward Gibson. Linguistic complexity: locality of syntactic dependencies. *Cognition*, 68:1–76, 1998.

[Hep92] Mark Hepple. Chart parsing lambek grammars: Modal extensions and incrementality. In *Proc. COLING 1992*, 1992.

[Hep98] Mark Hepple. Memoisation for glue language deduction and categorial parsing. In *Proceedings of the COLING-ACL Joint Conference, Montreal Canada, August. Ninth International Congress on Linguistics*. ACL, 1998.

[HR97] Paul Hagstrom and Jaemin Rhee. Differences Between Korean and Japanese Processing Overload. *Journal of Psycholinguitic Research*, 26, 2:189–206, 1997.

[HU79] John E. Hopcroft and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, Reading, MA, 1979.

[Ino91] A. Inoue. *A comparative study of parsing in English and Japanese*. PhD thesis, University of Connecticut, Storrs, CT, 1991.

[Jos90] Aravind K. Joshi. Processing crossed and nested dependencies: An automaton perspective on the psycholinguistic results. *LCP*, 5(1):1–27, 1990.

[Kaa97] Edith Kaan. *Processing subject-object ambiguities in Dutch*. PhD thesis, University of Groningen, Groningen, the Netherlands, 1997.

[Kön90] Esther König. The complexity of parsing with extended categorial grammars. In *Proc. COLING 1990*, 1990.

[Kon96] Lars Konieczny. *Human sentence processing: A semantics-oriented parsing approach*. PhD thesis, University of Freiburg, Freiburg, Germany, 1996.

[Kru95] Geert-Jan M. Kruijff. The Unbearable Demise of Surprise: Reflections on abduction in Artificial Intelligence and Peirce's philosophy. Master's thesis, University of Twente, Enschede, The Netherlands, August 1995.

[Kru97] Geert-Jan M. Kruijff. Peirce's theory of abduction and Goedel's axioms of infinity - reflections on the mathematics of creativity. Presentation at LOGIKA'97, Liblice (Czech Republic). Available from http://kwetal.ms.mff.cuni.cz/˜gj/abduction.html, June 1997.

[Kru98] Geert-Jan M. Kruijff. Peirce's late theory of abduction: A comprehensive account. Available from http://kwetal.ms.mff.cuni.cz/˜gj/abduction.html, 1998.

[Kru99] Geert-Jan M. Kruijff. Resource logics, grammars, and processing. In Christian Retoré and Ed Stabler, editors, *Proceedings of the Workshop on Resource Logics and Minimalist Grammars*, Utrecht, 1999. ESSLLI.

[KV00] Edith Kaan and Nada Vasić. Cross-serial dependencies in Dutch: Evidence for a discourse-based metric? Poster presented at CUNY sentence processing conference, UCSD, 2000.

[Lew98] Richard L. Lewis. Interference in Working Memory: Retroactive and Proactive Interference in Parsing. Columbus, OH: OSU, MS, 1998.

[LN99] Richard Lewis and Mineharu Nakayama. Determinants of processing complexity in Japanese embeddings: new theory and data. In *Proceedings of the East Asian Psycholinguistics Workshop*, Columbus, OH, 1999.

[MC63] G. A. Miller and N. Chomsky. Finitary models of language users. In D. R. Luce, R. R. Bush, and E. Galanter, editors, *Handbook of Mathematical Psychology, Volume II*. John Wiley, New York, 1963.

[Mil62] G. A. Miller. Some psychological studies of grammar. *American Psychologist*, 17:748–762, 1962.

[Mil92] David Milward. Dynamics, dependency grammar, and incremental interpretation. In *COLING 1992*, 1992.

[Mil94] David Milward. Dynamic dependency grammar. *Linguistics & Philosophy*, 17(6):561–606, December 1994.

[Mil95] David Milward. Incremental interpretation of categorial grammar. In *EACL 1995*, 1995.

[Moo97] Michael Moortgat. Categorial type logics. In Johan van Benthem and Alice ter Meulen, editors, *Handbook of Logic and Language*, chapter 2. Elsevier, Amsterdam, 1997.

[NBG00] Kentaro Nakatani, Maria Babyonyshev, and Edward Gibson. The Complexity of Nested Structures in Japanese. Poster presented at CUNY sentence processing conference, UCSD, 2000.

[Par95] Charles Parsons. Platonism and mathematical intuition in Kurt Gödel's thought. *BSL*, 1(1):44–74, March 1995.

[Pic94] Martin J. Pickering. Processing Local and Unbounded Dependencies: A Unified Account. *Journal of Psycholinguistic Research*, 23(4):323–352, 1994.

[Ste00] Mark Steedman. *The Syntactic Process*. MIT Press, Cambridge, MA, 2000.

[Ueh99] Keiko Uehara. Center-Embedding Problem and The Contribution of Nominative Case Repetition. In *Proceedings of the Ninth Japanese/Korean Linguistics Conference, Columbus, OH*, 1999.

[Yam94] Hiroko Yamashita. *Processing of Japanese and Korean*. PhD thesis, Ohio State University, Columbus, OH, 1994.

[Yam97] Hiroko Yamashita. The Effects of Word Order and Case Marking Information on the Processing of Japanese. *Journal of Psycholinguistic Research*, 26(2):163–188, 1997.