Studies in Second Language Acquisition (2025), 1–42
doi:10.1017/S0272263125101009

METHODS FORUM

Formal computational modelling in second language sentence processing research

Hiroki Fujita

Department of Linguistics, University of Potsdam, Potsdam, Germany
Email: hiroki.fujita@uni-potsdam.de

(Received 12 March 2024; Revised 23 April 2025; Accepted 17 June 2025)

Abstract

Various theories have been proposed in the field of second language (L2) sentence processing research and have significantly advanced our understanding of the mechanisms underlying L2 sentence interpretation processes. However, many existing theories have only been formulated verbally, and little progress has been made towards formal modelling. Formal modelling offers several advantages, including enhancing the clarity and verifiability of theoretical claims. This paper aims to address this gap in the literature by introducing formal computational modelling and demonstrating its application in L2 sentence processing research. Through practical demonstrations, the paper also emphasises the importance of formal modelling in the formulation and development of theory.

Keywords: sentence processing; formal computational modelling; computational model; second language sentence processing; language comprehension; psycholinguistics

Introduction

When we hear or read well-formed sentences, our brains generally extract meaning from them rapidly and incrementally. A central goal of sentence processing research is to understand this real-time sentence interpretation process.

A standard assumption about real-time sentence interpretation processes is that the brain is equipped with a processing system (*processor*) that computes hierarchical syntactic structures from input (i.e., the processor combines elements mapped from strings or sounds into larger units) and derives meaning from these structures. Much research on second language (L2) sentence processing has investigated potential differences in the properties of this processing system between first language (L1) speakers and adult L2 speakers who learn their L2 later in life (e.g., Cheng, Cunnings, Miller & Rothman, 2022; Clahsen & Felser, 2006b; Cunnings & Fujita, 2021, 2023; Dekydtspotter & Seo, 2017; Felser, Roberts, Marinis & Gross, 2003; Felser, Sato & Bertenshaw, 2009; Fernández Cuenca & Jegerski, 2023; Frenck-Mestre & Pynte, 1997; Fujita & Cunnings, 2020, 2021a, 2021b, 2022, 2024; Hopp, 2015, 2022; Juffs &

Harrington, 1995; Kim & Grüter, 2021; Lago & Felser, 2018; Marinis, Roberts, Felser & Clahsen. 2005; Omaki & Schulz, 2011; Rah & Adone, 2010; Roberts & Felser, 2011).

A growing number of studies have investigated L2 sentence processing, and various theories have been proposed. While these theories have significantly advanced our understanding of the mechanisms underlying L2 sentence interpretation processes, many have only been formulated verbally, and there has been limited progress in formal computational modelling (Frank, 2021). Verbal expressions are useful for making theories accessible to a wide audience. However, they often introduce ambiguity, making it difficult to formulate theories without interpretive issues, which can lead to misunderstandings about their claims and predictions. Indeed, instances of such misunderstandings have been documented in the L2 sentence processing literature (Clahsen & Felser, 2018). Formal computational modelling, with its explicitness, helps to avoid misunderstandings and oversights, as will become clear later in this paper.

One potential barrier to incorporating formal computational modelling into L2 sentence processing research is that it requires specific knowledge and techniques. Unfortunately, there are currently no introductory textbooks or methodological papers dedicated to this topic. To address this gap in the literature, this paper provides an overview of formal computational modelling and demonstrates how it can be incorporated into L2 sentence processing research. In the following sections, I will first discuss what formal computational modelling is. I will then introduce two formal computational models and demonstrate how formal computational modelling can be applied in L2 sentence processing research using these models.

Formal computational modelling in sentence processing research

In order to understand formal computational modelling in the context of human sentence processing, it is necessary to understand what the terms *formal*, *computational*, and *model* mean. In what follows, I will first introduce the concept of computation from the traditional perspective of cognitive psychology.

The study of computation spans many disciplines. In cognitive psychology, it refers to the cognitive processes that underlie human behaviour. In other words, behaviour is a direct result of computation. Broadly speaking, computation is defined in terms of input-output mappings and algorithms (finite, step-by-step procedures) used to produce outputs. That is, a system is defined as a computing system if it produces an output from an input according to a particular procedure. Additionally, researchers often assume certain constraints (e.g., J. A. Fodor, 1975), in part to distinguish between computing and non-computing systems (see Colombo & Piccinini, 2023). Under the standard assumption in sentence processing research, the processor is viewed as a computing system: It produces syntactic structure (output) from strings or sounds (input) and meaning (output) from syntactic structure (input), and this computational process follows certain procedures. This cognitive process is thought to causally explain certain behaviours observed in activities involving linguistic information (e.g., reading a sentence takes a long time because there are factors that make it difficult for the processor to compute the syntactic structure of the sentence).

A model is a simplified representation of an object of interest (model ≠ copy), expressed in a way that allows us to understand that object. In the context of sentence processing, the primary objects of interest are the algorithms used by the processor and its input and output. Additionally, models of sentence processing often consider the factors that may influence computational processes, such as memory capacity and attentional resources. Models can be expressed in a variety of communicative forms,

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

including diagrams, verbal expressions, and mathematical equations (Bailer-Jones, 2009). Models that are expressed in mathematical form are called *formal models*. In this paper, I use expressions such as "model X," where X is an object of interest, and "formalise Y," where Y is a verbally formulated hypothesis. The former means building a model of X, and the latter means translating Y into a formal description.

In summary, from the traditional perspective of cognitive psychology and in the context of human sentence processing, formal computational modelling means expressing a hypothesis about the algorithms used by the processor and its input and output (along with the factors that may influence the computational processes) in mathematical form that facilitates our understanding of sentence interpretation processes. Building on this foundation, I will present an approach to incorporating formal computational modelling into L2 sentence processing. This approach will be demonstrated by formalising two verbally proposed hypotheses about L2 sentence processing. The next section will introduce two formal approaches that will be used to formalise these hypotheses.

Formal computational models

As discussed in the previous section, formal modelling means expressing a hypothesis about an object of interest in mathematical form. In sentence processing research, the primary objects of interest are the algorithms used by the processor and its input and output. In real-time sentence processing, there are potentially several different processes involved at different levels of expression, and it is impractical to cover all these aspects in a single tutorial paper. Therefore, this paper focuses on two key processes: the computation of syntactic structure and memory retrieval. These processes have been selected based on the L2 processing hypotheses that will be formalised later in the paper. These hypotheses propose that potential differences between L1 and L2 processing stem from one of these processes. In the following subsections, I will first present a formal model of syntactic structure proposed by a theory of grammar. Based on this model, we will formally model the syntactic structures computed by the L2 processor and compare them with those computed by the L1 processor. I will then present a formal model of memory retrieval.

The structure-building system

In language, sentences are associated with particular meanings (i.e., sentences are pairings of form and meaning). For example, in the sentence (i) "Emily will visit John," "Emily" is the one doing the visiting and "John" is the one being visited. Their roles are not understood the other way around. Linguists usually assume that such form-meaning pairs are established because our brains possess a computational system (*structure-building system*) that generates hierarchical structures pairing form and meaning. In hierarchical structures, some elements are positioned higher or lower than others (this will be explained in more detail). There are several reasons for assuming the existence of hierarchical structures. Some of the reasons are that form does not seem to map directly to meaning and that hierarchical structure allows us to derive meaning systematically from form. For example, in the sentence (ii) "The man behind Emily will visit John," "The man…" is the visitor and "John" is the visitee. "Emily" is not understood as the visitor despite the fact that it is closer to "visit" than "The man" in the surface order and that the part (<…>) of (ii) "The man behind <Emily will visit John>" has exactly the same form as (i) where "Emily" is understood as the visitor.

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

The form-meaning pair in (ii) can be established in such a way that, in the corresponding hierarchical structure, "The man…" and "visit John" are related in the same way as "Emily" and "visit John" in (i), enabling a systematic derivation of meaning.

It is generally assumed that the structure-building system is distinct from the processor that computes syntactic structures (i.e., *the parsing system*).[1] The structure-building system generates syntactic structures in an unbounded way, following a set of rules, and these structures pair form and meaning. The processor computes syntactic structures in real time (using the structure-building system) as it receives strings or sounds, in order to derive meaning. Assuming that the structure-building system and the L1 processor output the same or similar syntactic expressions, we will formally model the syntactic structures computed during real-time sentence processing based on the structure-building system. If the L1 and L2 processors compute different syntactic structures, the L2 syntactic structures may not adhere to the constraints followed by the L1 syntactic structures.

This paper adopts the structure-building system assumed within the framework of *principles and parameters* (Chomsky, 1981; Haegeman, 1994; Hornstein, Nunes & Grohmann, 2005; van Riemsdijk & Williams, 1986). In this framework, syntactic structures take the form of rooted binary trees, as shown in Figure 1, where the top node (X1) is the root. Trees are mathematical objects consisting of a set of *nodes* carrying specific information (e.g., X1–X7 in Figure 1) and a set of *edges* connecting these nodes (e.g., the path connecting X1 and X2 in Figure 1). A tree is a rooted binary tree if each node has an outdegree of at most two (i.e., if each node has at most two children; see footnote 2 for the definition of *children* in the context of trees) and all nodes other than the root are directed away from the root. The structure of a rooted binary tree is hierarchical, with the root in the highest position and other nodes considered to be lower in position as they move away from the root (i.e., the more edges you traverse from the root to a node, the lower that node is positioned). For example, in the syntactic structure in Figure 1, X2 is in a higher position than X4 because only one edge needs to be traversed from X1 to X2, whereas two edges need to be traversed from X1 to X4.

Within the framework of principles and parameters, syntactic structures are constructed as each head node—a grammatical category mapped from a word (e.g., a V node mapped from the verb "bake")—combines with other nodes. This structure building follows the *X-bar schema* (Chomsky, 1981; Kornai & Pullum, 1990; Stowell, 1981). The X-bar schema can be expressed as a set of production rules, where each rule serves as a conditional statement connecting nodes. These rules follow the format $X \rightarrow Y$, where X and Y are nodes and the arrow represents a directed edge from X to Y.[2] The set of production rules is shown in (1) below, and the corresponding syntactic structure is shown in Figure 2.

[1] In sentence processing research, the term *structure building* is often used to refer to parsing. In this paper, structure building is used only for the system that generates syntactic structures in an unbounded way to pair form and meaning.

[2] This relation is sometimes called *immediate dominance* (i.e., X immediately dominates Y). Dominance is a transitive relation between two nodes: Node a dominates node b if and only if the trace from a to b follows a strictly downward trajectory. Dominance relations are often assumed to be reflexive (i.e., each node dominates itself), but in this paper, they are irreflexive. When there is a directed edge from a to b, a is called the *parent* of b, and b is called a *child* of a.

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

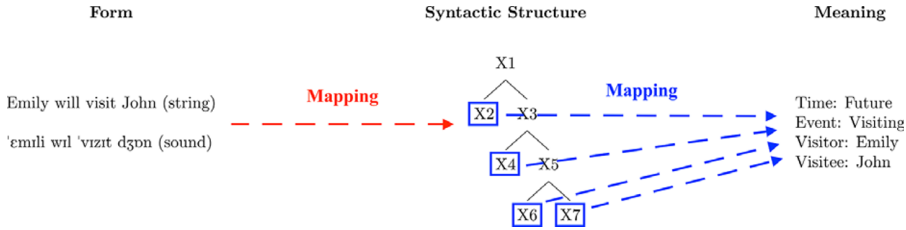


Figure 1. Mappings from form to syntactic structure and from syntactic structure to meaning.

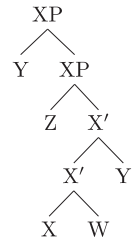


Figure 2. The X-bar schema.

(1) a. XP → Y XP
b. XP → Z X′
c. X′ → X′ Y
d. X′ → X W

In (1), W, X, Y, and Z are variables. In the X-bar schema, there are generally three levels of projection. X represents a minimal projection, a head node (i.e., a grammatical category mapped from a word). A projection is defined as a structural unit containing this head node. The prime symbol (′) denotes an intermediate projection (second level of projection), and P denotes a maximal projection (third level of projection). W, Y, and Z are optional nodes, each of which is a maximal projection (but not a projection of X). These nodes are called the *complement* (COMP) of X, an *adjunct* of XP, and the *specifier* (SPEC) of XP, respectively. Explicit rules for generating the syntactic structure of the sentence "Emily will visit John" are given below as an example.

(2) a. TP → NP T′
b. NP → (SPEC) N′
c. N′ → N (COMP)
d. T′ → T VP
e. VP → (SPEC) V′
f. V′ → V NP

In (2), the parentheses indicate positions without nodes. N stands for noun ("Emily," "John"), V for verb ("visit"), and T for tense ("will"). Figure 3 shows the computation of

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

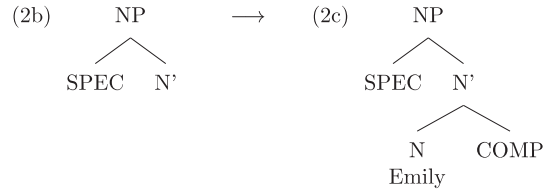


Figure 3. The computation of the subject NP. In this example, the computation runs top down.

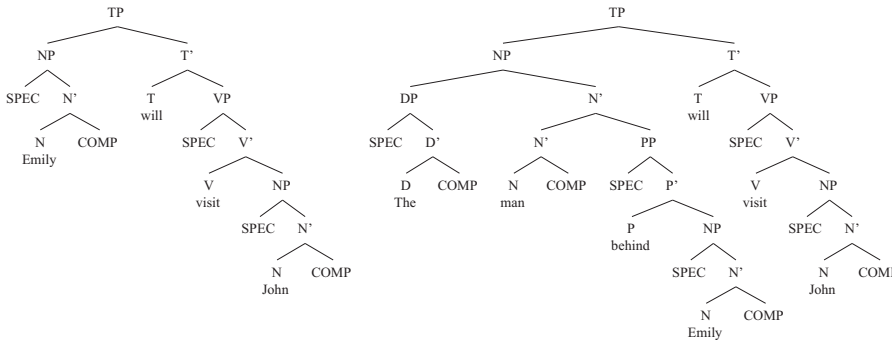


Figure 4. The syntactic structures of the sentences "Emily will visit John" and "The man behind Emily will visit John." [$_{NP}$ Emily] in the left tree and [$_{NP}$ The man…] in the right tree are related to [$_{VP}$ visit John] in the same way: [$_{VP}$ visit John] is dominated by the TP node that immediately dominates [$_{NP}$ Emily]/[$_{NP}$ The man…]. In the right tree, there is no such relation between [$_{NP}$ Emily] and [$_{VP}$ visit John].[3,4]

the subject NP structure, where (2b) is applied first and then (2c). Figure 4 shows the whole sentence structure, together with the syntactic structure of the sentence "The man behind Emily will visit John."

Memory retrieval

Memory retrieval is the process of bringing back information stored in memory. This process is often assumed to play an important role in the interpretation of nodes in relation to other nodes to which they are grammatically related during real-time sentence processing (e.g., Dillon, Mishler, Sloggett & Phillips, 2013; Fujita, 2025; González Alonso, Cunnings, Fujita, Miller & Rothman, 2021; Jäger, Mertzen, Van Dyke & Vasishth, 2020; Wagers, Lau & Phillips, 2009). Such grammatical relations are often called *dependency relations* (e.g., Chomsky, 1956; J. D. Fodor, 1978). For example,

[3]In the framework of principles and parameters, it is standardly assumed that a copy of the subject NP is present at the specifier of the VP (for modern analyses of verbal projections, see D'Alessandro, Franco & Gallego, 2017). In this configuration, the meaning of the sentence with regard to the relation between the subject NP, the verb, and the object is established within the same verbal domain.

[4]In the framework of principles and parameters, when it is assumed that determiners (D or Det) occupy the specifier position of NP (*NP-hypothesis*; see a special selection edited by Blümel & Holler, 2022), D is often assumed not to project (Abney, 1987). In this paper, we assume that D does project, as do other head nodes.

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

Table 1. Key assumptions about sentence processing in the activation model

A1	Declarative memory serves as the long-term memory for words and as the short-term working memory for the syntactic structures computed during sentence processing. Words and syntactic structures are stored as bundles of features (e.g., singular, NP).
A2	Each word and node are assigned an activation value.
A3	When attention is drawn to a word during sentence processing, information about that word (i.e., its features) is retrieved from long-term memory.
A4	The activation values of words fluctuate due to this retrieval and temporal decay.
A5	Lexical and structural features are used as retrieval cues to retrieve nodes for parsing and memory retrieval. The activation values of nodes fluctuate depending on how well they match the retrieval cues.
A6	Procedural memory serves as a repository for production rules (including grammatical knowledge) that are used for memory retrieval and parsing.

in (3) below, the reflexive NP [$_{NP}$ herself] is referentially related to the subject NP of the embedded clause [$_{NP}$ the girl].[5]

(3) John said that the girl introduced herself to the new teacher.

It is often assumed that during real-time sentence processing, the reflexive NP is interpreted by retrieving information about the embedded subject NP from memory and associating this information with [$_{NP}$ herself] (Dillon et al., 2013; Fujita & Yoshida, 2024; Jäger et al., 2020). Therefore, memory retrieval is often considered to be part of, or to subserve, the computational processes involved in real-time sentence interpretation.

To formally model memory retrieval, we will use the *activation-based model of memory retrieval* (activation model; Fujita, 2024a; Lewis & Vasishth, 2005; Vasishth & Engelmann, 2021). This model is based on the Adaptive Control of Thought–Rational (ACT-R) architecture (Anderson, 2007) and makes several assumptions about sentence processing. These assumptions are listed in Table 1 (adapted in part from Lewis & Vasishth, 2005).

Of these assumptions, A5 can be better understood through an example. Thus, let us consider memory retrieval at "herself" in (3) again. Focusing on lexical information, the critical features associated with this reflexive are grammatical category (N), number (singular), and gender (feminine). Roughly speaking, a feature is a fundamental building block of a syntactic object (e.g., the features "N," "singular," and "feminine" are constituents of the element mapped from "herself").[6] Suppose that the grammatical category, number, and gender features are used as retrieval cues. In (3), only [$_{NP}$ the girl] matches all of these cues.[7] Consequently, the activation model predicts that [$_{NP}$ the girl] receives the highest activation during the antecedent retrieval if we only consider

[5]Even if we replace "John" with "Mary" in (3), as in "Mary said that the girl introduced herself…," [$_{NP}$ Mary] cannot be understood as coreferential with [$_{NP}$ herself] because the NP in the subject position of the matrix clause is not structurally accessible for coreference with the reflexive (Chomsky, 1981). The activation-based model of memory retrieval treats such structural information as features, but there is debate about how it can be transformed into features (see Kush, 2013). For the purpose of demonstrating the formal modelling of memory retrieval, this paper assumes that structural information is encoded as features.

[6]If structural information is encoded as a feature, then this definition needs to be modified, or a distinction needs to be made between lexical and structure-based features.

[7]Technically, given that a coreference relation is established between NPs, we may need to assume that (a) NP is used as a retrieval cue, (b) the head category is retrieved first and then its maximal projection (or the highest projection representing the nominal) is retrieved, or (c) some other nominal feature is used.

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

Table 2. Equations for the calculation of activation values and retrieval times in the activation model

Component	Equation	Description
TA_i (total activation)	$BA_i + SA_i + MP_i + N_i$	total activation for *i*
BA_i (base-level activation)	$b_i + ln\left(\sum_{k=1}^{n} t_k{}^{-d}\right)$	*t* = time *k* = retrieval *d* = decay rate *b* = resting level of activation
SA_i (spreading activation)	$\sum_{j=1}^{n} W_j S_{ji}$	W_j = weight S_{ji} = associative strength between *j* and *i*
S_{ji} (associative strength)	$m + ln(P(i\|j))$	*m* = maximum associative strength
MP_i (mismatch penalty)	$\sum_{j=1}^{n} pM_{ji}$	*p* = scaling parameter *M* = similarity between *j* and *i*
N_i (noise)	$Normal\left(0, \sqrt{\frac{\pi^2}{3}x^2}\right)$	x^2 = scaling parameter
RT_i (retrieval time)	$Fe^{-(f \cdot TA_i)}$	*F*, *f* = scaling parameters

Note: i = element, j = cue.

matches with these retrieval cues. During memory retrieval, the element (node) with the highest activation that reaches a retrieval threshold is retrieved, with higher activation leading to shorter retrieval times. The calculation of activation values and retrieval times follows the equations summarised in Table 2.

In the equations, many letters serve as symbolic representations of numerical values, just as numbers do. Some symbols are constants, meaning their values are fixed (e.g., $\pi \approx 3.141593$). Many other symbols are variables, meaning their values are not fixed. The values of these variables are determined in different ways. For some variables, the values are determined by some equations. For others, the values depend on the underlying assumptions. Input values are also sometimes estimated from the data that the model is used to predict, by fitting it to the data with different values and finding the values that fit best. While this data-driven estimation is useful when there is no clear basis for determining input values, it does not allow the model to fully fulfil its role (i.e., to facilitate our understanding of the object of interest). Therefore, although the model under development may rely on data-driven estimation, there should ultimately be a basis for each input value. The choice of input values affects the output of the model. The bullet points below provide detailed descriptions of the equations and explain how different input values affect the model output.

- *Total activation* is calculated by summing base-level activation, spreading activation, mismatch penalty, and noise. Retrieval times are calculated using this total activation.
- *Base-level activation* is calculated by summing *n* retrievals of the element. In the equation, *t* represents the time elapsed since the *k*th retrieval of the element and is raised to the power of the negated decay rate *d*. The parameter *b* represents a resting level of activation. The power function reflects the assumption that forgetting is initially fast but slows down over time (*power law of forgetting*; Wickelgren, 1974). Thus, the activation of each element in memory undergoes exponential decay. The decay rate *d* controls the rate of forgetting. Figure 5 illustrates the fluctuations in the base-level activation of [$_{\text{NP}}$ the girl] in (3) over time.[8]

[8]Technically, different base-level activations are calculated for "the" and "girl." In this paper, we only consider the base-level activation of the word whose lexical features (see footnote 6) match retrieval cues. For

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

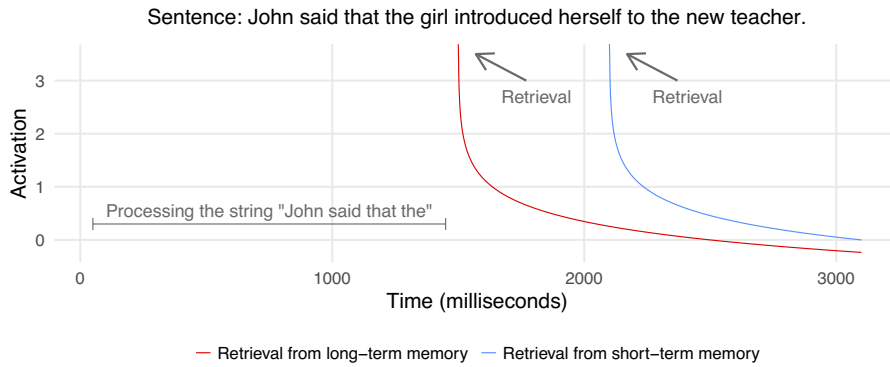


Figure 5. Fluctuation of the base-level activation of [$_{NP}$ the girl] in (3) over time (see footnote 8). Two peaks correspond to retrievals at "girl" (from long-term memory) and "herself" (from short-term memory). *Note*: b = 0, d = .5.

- *Spreading activation* is calculated by summing the products of weights and associative strengths between retrieval cues and elements. Weights refer to the relative influence of cues; a cue with a higher weight has a greater impact on activation. Weights are calculated using the equation G/n, where G is the amount of activation divided by the number of cues. By default, G is set to 1, and all cues are given the same weight. For example, if there are three cues, A, B, and C, each cue will have a weight of $1/3$. If one of the cues (e.g., A) is deemed more influential, it can be given a greater weight (e.g., $A = 3/5$, $B = 1/5$, $C = 1/5$).
- *Associative strength* indicates the degree of association between a cue and an element, reflecting the reliability of the cue in retrieving the element from memory. As the number of elements in memory that match a given cue increases, the degree of association between the cue and each element decreases. This is known as the *fan effect* (Anderson, 1974). The strength of association is expressed as the conditional probability of the occurrence of a particular element i given the presence of a particular cue j. The standard assumption in sentence processing research is that all elements associated with a cue are equally likely when the cue is present. Under this assumption, $P(i|j)$ is the reciprocal of the number of elements in memory that match the cue. For example, considering the number cue at the reflexive in (3), both [$_{NP}$ the girl] (*target*) and [$_{NP}$ John] (a *distractor*) match this cue (i.e., they are both singular).[9] Therefore, for $S_{number,target}$, $P(i|j) = 1/2$, resulting in a reduced associative strength compared to when the distractor does not match the number cue (in which case $P(i|j) = 1$). The associative strength is calculated by summing this value and the maximum associative strength (maximum activation that an element could have). The fan effect is illustrated in Figure 6.

example, in the retrieval of [$_{NP}$ the girl] at "herself," it is "girl" whose features match the retrieval cues (i.e., grammatical category, gender, and number). Therefore, only the base-level activation of "girl" is considered for the memory retrieval at "herself." We also assume that base-level activations are assigned to nodes (e.g., the base-level activation of "girl" is assigned to [$_{NP}$ the girl]).

[9]The target is the element that is structurally accessible for memory retrieval (or the element intended to be related to the element that triggers memory retrieval). Other elements (usually from the same grammatical category as the target) are called distractors.

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

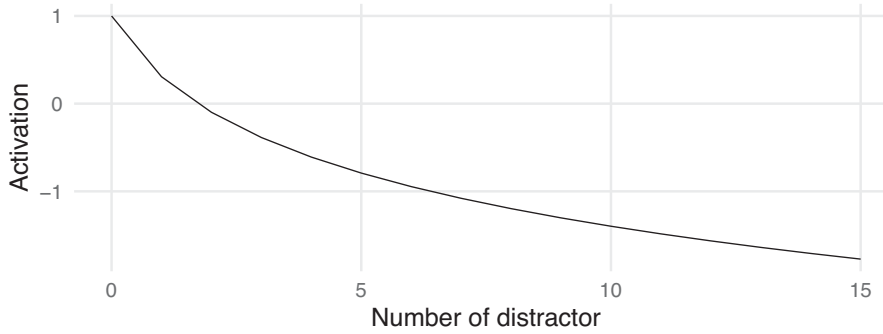


Figure 6. The fan effect.

- *Mismatch penalty* penalises elements that mismatch a set of retrieval cues. By default, the similarity (M_{ji}) ranges from 0 (identity) to −1 (maximum difference). For example, if the gender cue is used for antecedent retrieval at "herself" in (3), [$_{NP}$ John] incurs a mismatch penalty because its conventionally assigned gender (masculine) does not match the gender of the reflexive (feminine). The parameter p is used to scale the mismatch penalty. Scaling is a way of adjusting the output of the model. For example, increasing the value assigned to p results in a larger mismatch penalty.
- *Noise* is included in the activation model to consider the variability of human behaviour. Human performance is not perfectly accurate and can be affected by factors such as distraction and background noise. The noise function is used to reflect this variability. It generates random values within a certain range, with 0 being the most probable value and the probability decreasing as the value moves away from 0. The exact range depends on the scaling parameter x^2. This probability distribution (*normal distribution*) is shown in Figure 7.
- *Retrieval times* are calculated as the product of a scaling parameter and a negative exponential function, where the exponent is determined by the scaled total activation. Because of this negative exponential function, higher activation leads to shorter retrieval times, as illustrated in Figure 8.

In this section, we have reviewed the structure-building system and the activation model. In the following sections, we will use these to formalise two hypotheses about L2

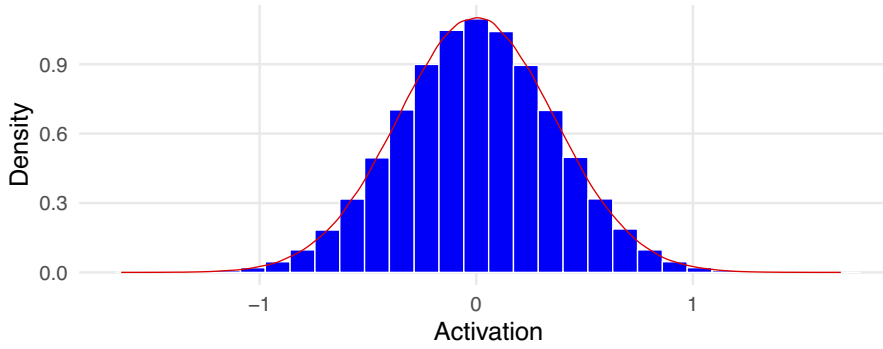


Figure 7. Stochastic noise.
Note: x = .2.

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

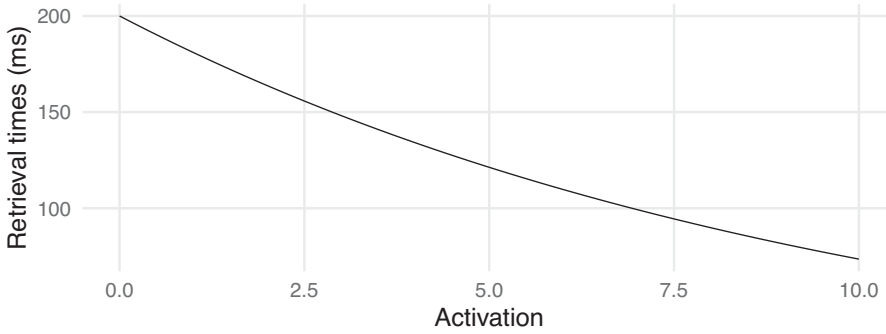


Figure 8. The retrieval time function.
Note: F = .2, f = .1.

sentence processing: the *shallow structure hypothesis* (SSH; Clahsen & Felser, 2006b) and the *interference-based hypothesis* (IBH; Cunnings, 2017b). These hypotheses have been proposed to explain the differences observed in previous studies between L1 and L2 processors (*L1/L2 differences*). The SSH proposes that L1/L2 differences arise because the L1 and L2 processors compute different syntactic structures in certain contexts and rely on non-structural information differently. The IBH claims that L1/L2 differences arise because the L1 and L2 processors weight retrieval cues differently when retrieving an element from memory during sentence processing. The SSH and the IBH can be seen as competing hypotheses because they attempt to explain L1/L2 differences in different ways. However, they can also be considered complementary if the source of L1/L2 differences relates to both parsing and memory retrieval. For each hypothesis, we will first model the syntactic structures computed by the processor and then the memory retrieval processes. We will start with the formalisation of the SSH.

The shallow structure hypothesis

The SSH occupies an important place in L2 sentence processing research and has been the subject of extensive investigation. The proponents of the SSH argue that its claims are often misunderstood (Clahsen & Felser, 2018). While misunderstandings are not uncommon for extensively studied theories, the SSH appears to be misunderstood, at least in part, due to the absence of formal modelling. In order to adequately formalise the SSH, it is essential to understand its claims. Therefore, in what follows, we will first review them.

Verbal descriptions of the SSH

The SSH proposes that divergent processing patterns emerge between the L1 and L2 processors because, under certain circumstances, they compute different syntactic structures during real-time processing. This proposition is consistently expressed in the papers dedicated to the SSH:

> [T]he syntactic representations adult L2 learners compute during comprehension are shallower and less detailed than those of native speakers (Clahsen & Felser, 2006b, p. 3).

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

> [A]dult learners … [compute] representations for language comprehension that lack syntactic detail (Clahsen & Felser, 2006b, p. 34).

> The SSH claims that during L2 processing, learners compute grammatical representations that lack complex hierarchical structure and abstract, configurationally determined elements such as movement traces (Clahsen & Felser, 2006a, p. 111).

The SSH also claims that L2 sentence processing is strongly influenced by non-structural information:

> L2 learners establish long filler-gap dependencies using direct lexical association (Clahsen & Felser, 2006b, p. 26).

> Adult L2 learners … rely more on lexical–semantic cues to interpretation (Clahsen & Felser, 2006a, p. 107).

> [E]ven highly proficient L2 speakers … are guided more strongly than native speakers by semantic, pragmatic, probabilistic, or surface-level information (Clahsen & Felser, 2018, p. 694).

However, the SSH argues that the L2 processor does compute hierarchical syntactic structures during real-time processing:

> Even in its weaker forms such as "NNSs [non-native speakers] cannot deploy syntactic representations in the task of sentence processing" … or "L2 sentence parsing involves only shallow syntactic structures" … this kind of statement clearly overstates the SSH. Without syntax and the ability to build hierarchical representations we would not be able to comprehend simple sentences … and this is certainly not what the SSH would predict even for learners at lower proficiency levels (Clahsen & Felser, 2018, p. 695).

In summary, the SSH claims that hierarchical syntactic structures are computed during real-time L2 sentence processing. However, these structures lack certain properties, such as movement traces (to be illustrated briefly), that are present in the syntactic structures computed by the L1 processor. Consequently, the L2 processor relies heavily on non-structural information for sentence interpretation.

The verbal descriptions above effectively convey the claims of the SSH. However, the specific syntactic structures computed by the L2 processor are unclear. This lack of clarity may have led to misunderstandings of the SSH in previous research. For example, Omaki and Schulz (2011) interpret the SSH as predicting that "they [L2 learners] should not respect the RC [relative clause] island constraint [a structure-relevant constraint] because there is no RC representation in their analysis" (Omaki & Schulz, 2011, p. 570). Clahsen and Felser (2018) argue that the SSH does not make such a claim. However, Omaki and Schulz's interpretation does not appear to be entirely inconsistent with some of the descriptions above. This misinterpretation could have been avoided if the SSH had presented the specific syntactic structures computed by the L2 processor.

In the following subsections, we will first formally model the syntactic structures computed by the L2 processor based on the claims of the SSH. We will then model the L2 memory retrieval processes, calculate the processing times, and compare them with the actual data used to support the SSH.

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

The SSH: Formal modelling of syntactic structures

For the formal modelling of syntactic structures, we consider the relative clauses in (4a/b) below.

(4) a. We photographed the fans of the actress who were looking happy.
b. We photographed the fans of the actress who was looking happy.

In (4a/b), "who," which marks the beginning of the relative clause, is locally ambiguous because it can be analysed locally as modifying either the plural nominal [fans] (NP1 modification) or the singular nominal [actress] (NP2 modification). In (4a), only the NP1 modification becomes grammatical at "were" because [fans], but not [actress], matches this verb in number. In (4b), on the other hand, only the NP2 modification becomes grammatical at "was" because this verb forms a dependency relation with a singular NP. The following production rules generate the relative clause structures computed by the L1 processor (the vertical bar "|" denotes alternatives)[10]:

(5) a. NP → DP N′
b. DP → (SPEC) D′
c. D′ → D (COMP)
d. N′ → N′ CP | N PP | N (COMP)
e. PP → P NP
f. CP → NP C′
g. C′ → C TP
h. TP → NP T′
i. T′ → T VP

In (5), D stands for determiner ("the"), the category "P" for preposition ("of"), and C for complementiser. To formally model the relative clause structures, we use the Natural Language Toolkit (NLTK) Python package (for installation instructions, visit https://www.nltk.org/install.html). Python is a programming language and can be downloaded from https://www.python.org/downloads/. Although NLTK is a Python package, this paper uses it in the R environment, given its widespread use in L2 research. All the code used in this paper can be found on the Open Science Framework (OSF) website (https://doi.org/10.17605/OSF.IO/9VCX7) and in the R notebook on Colab (https://colab.research.google.com/drive/1zjl1Mc7Sn8d-IfeeB340bErt_fXJ-lJH?usp=sharing). Python code can be executed in R by loading the reticulate package into R[11,12]:

```
library(reticulate)
```

[10]These relative clause structures are based on Clahsen and Felser (2006b), who adopt the classical analysis of relative clauses within the framework of principles and parameters. For modern analyses of relative clauses, see Salzmann (2019).

[11]If the reticulate package is not already installed, it can be installed using the following code: install.packages("reticulate").

[12]The R and Python code used in this paper is explained in detail. However, if readers are completely unfamiliar with these programming languages and wish to learn more, there are many free online resources available (e.g., https://r4ds.hadley.nz/, https://allendowney.github.io/ThinkPython/).

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

The reticulate package provides an interface between R and Python. In the NLTK package, the *nltk.Tree* module with its *.fromstring()* method allows us to compute tree structures. For example, the tree structure of the production rule (5a) can be computed as "[NP DP N′]":

```
import nltk

T1 = nltk.Tree.fromstring("[NP DP N’]", brackets = "[]")
```

The `import nltk` code imports the NLTK library. The `nltk.Tree.fromstring()` function outputs a tree structure from `[NP DP N′]`. The argument `brackets = "[]"` tells Python that square brackets are used to specify the tree structure. Python is an object-oriented programming language, where an object needs to be created to store a tree. In the code above, the equals sign **"="** directs Python to assign the tree to the object `T1`.

In the bracketing rule used, the parent node (i.e., NP) appears first, followed by the two children (i.e., DP and N′), which are enclosed in the same bracket (see footnote 2 for definitions of *parent* and *child* in the context of trees). The left child (DP) precedes the right child (N′) according to the production rule (5a). The content of the object can be checked by typing in the name of the object:

```
T1

## Tree('NP', ['DP', 'N’'])
```

In R (and Python), anything written after a hash symbol on the same line is ignored when the code is executed. The double hashes above indicate the code's output. The following code prints a visually formatted tree[13]:

```
T1.pretty_print(unicodelines = True)

##       NP
##   ┌────┴────┐
##  DP        N’
```

In this code, the object to be visualised is specified at the beginning, and the `pretty_print()` function is called on it to print a visually formatted tree. The following is another example where an NP structure is displayed:

[13] A more visually formatted tree, often seen in linguistic papers, can be viewed and downloaded by saving it as a Scalable Vector Graphics file. For details, see the SyntacticStructure file on the OSF website.

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

```
T2 = nltk.Tree.fromstring("[NP [DP [SPEC ε] [D' [D the] [COMP ε]]] [N' [N
fans] [COMP ε]]]", brackets = "[]")

T2.pretty_print(unicodelines = True)
```

```
##                NP
##        ┌────────┴──────────┐
##       DP                   │
##   ┌────┴────┐              │
##   │         D'             N'
##   │    ┌────┴────┐    ┌────┴────┐
## SPEC  D        COMP  N         COMP
##   │   │         │    │          │
##   ε  the        ε   fans        ε
```

Although the bracket structure in this code may look complicated, it follows the same pattern as the previous example. The ε symbols denote empty strings. To improve clarity and ease of visualisation, empty strings and positions not occupied by nodes will be omitted where possible. For the same reason, minimal and intermediate projections will also be omitted where possible. We now formally model the syntactic structures for the NP1 and NP2 modifications, which are assumed to be computed by the L1 processor (see footnote 10).

NP1 modification:

```
T3 = nltk.Tree.fromstring("[NP [DP the] [N' [N' [N fans] [PP [P of] [NP [D
P the] [N' [N actress]]]]] [CP [NP1 who] [C' [C ε] [TP [NP(t1)] [T' [T wer
e] [VP ...]]]]]]]", brackets = "[]")

T3.pretty_print(unicodelines = True)
```

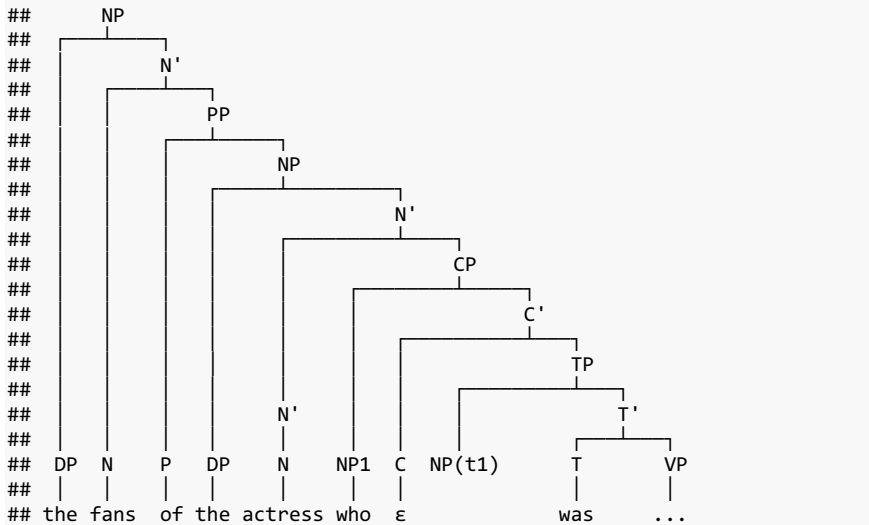
```
##                                NP
##   ┌──────────────────────────────┴──────────────┐
##   │                               N'            │
##   │                  ┌────────────┴──────────┐  │
##   │                 N'                       CP
##   │          ┌──────┴──────┐          ┌──────┴──────┐
##   │          │            PP         │            C'
##   │          │       ┌─────┴──────┐  │       ┌─────┴──────┐
##   │          │       │           NP  │       │           TP
##   │          │       │      ┌─────┴──────┐  │       ┌─────┴──────┐
##   │          │       │      │           N'  │       │           T'
##   │          │       │      │            │  │       │      ┌─────┴──────┐
##  DP  N       P       DP     N           NP1 C  NP(t1)      T            VP
##   │  │       │       │      │            │  │             │            │
## the fans     of     the   actress        who ε            were         ...
```

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

NP2 modification:

```
T4 = nltk.Tree.fromstring("[NP [DP the] [N' [N fans]  [PP [P of] [NP [DP t
he] [N' [N' [N actress]] [CP [NP1 who] [C' [C ε] [TP [NP(t1)] [T' [T was]
[VP ...]]]]]]]]]", brackets = "[]")

T4.pretty_print(unicodelines = True)
```



In these syntactic structures, the relative clause [$_{CP}$ who was/were…] adjoins to either the projection of the higher nominal (NP1 modification) or the projection of the lower nominal (NP2 modification). Differences in modification are thus expressed by different positions to which the relative clause adjoints. The symbol t (t1), called a *trace*, denotes an unpronounced copy of an element in the syntactic structure. Within the framework of principles and parameters, traces are used to explain the phenomenon that expressions are interpreted as if they were in different positions from where they overtly appear.[14] Traces are generated when elements move during structure building. For example, in the relative clause structures above, [$_{NP1}$ who] moves from the specifier of the TP to the specifier of the CP, and a trace is left in the former position. [$_{NP}$ who] and the trace are co-indexed by 1 to identify the trace. These elements are referentially related to either [fans] (NP1 modification) or [actress] (NP2 modification).

[14]For example, in the sentence "Which cake did John eat?" [$_{NP}$ Which cake] is interpreted as the direct object of [eat], even though it is not in the complement position of the verb. This phenomenon is explained by assuming that a trace denoting an unpronounced copy of [$_{NP}$ Which cake] occupies the complement position ([$_{NP1}$ Which cake] did John eat t1?).

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

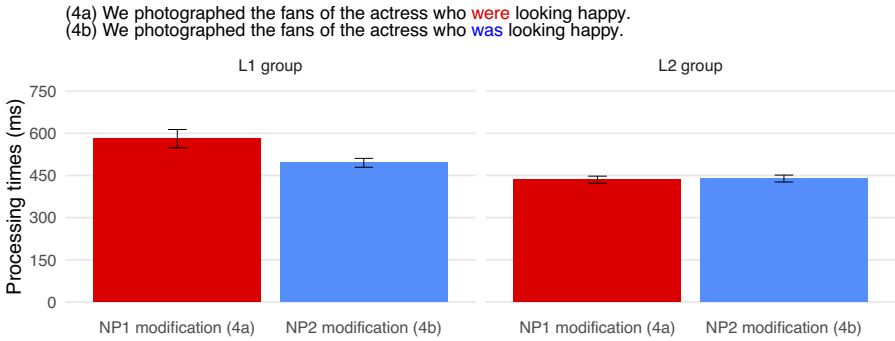


Figure 9. Processing times at the auxiliary verb in (4a/b) as reported in Felser et al. (2003). The L2 data are based on Experiment 2. Error bars are approximate standard errors.

Felser et al. (2003) used self-paced reading tasks to investigate whether L1 and L2 speakers prefer the NP1 or NP2 modifications at the point of local ambiguity. In (4a/b), if there is a preference for the NP1 modification, the processing time at the auxiliary verb is expected to be longer in (4b) than in (4a) due to the number mismatch between [fans] and [was] (*mismatch effects*; Dillon et al., 2013; Fujita & Cunnings, 2023; Wagers et al., 2009). In contrast, preferences for the NP2 modification are expected to lead to mismatch effects in (4a). For L1 participants, Felser et al. observed mismatch effects in (4a), suggesting that L1 speakers locally prefer the NP2 modification. However, L2 participants showed similar processing times between (4a) and (4b), suggesting a lack of specific preferences. These different processing patterns are illustrated in Figure 9.

The SSH interprets these findings as follows: The L1 processor employs a parsing strategy whereby it analyses incoming material (e.g., "who") as related to the structure of the most recently encountered word (e.g., the projection of the lower nominal; see Frazier, 1979; Fujita, 2021, 2024b; Kimball, 1973; Phillips & Gibson, 1997). In contrast, the L2 processor does not employ such a strategy due to the absence of fully detailed syntactic structures:

> If learners segment sentences from the L2 according to their thematic structure, then complex NPs such as *the servant of the actress* are likely to be treated as a single chunk … their apparent inability to apply any structure-based ambiguity resolution strategies … is expected if their representations of NP of NP complexes lack the relevant structural detail (Clahsen & Felser, 2006b, p. 33).

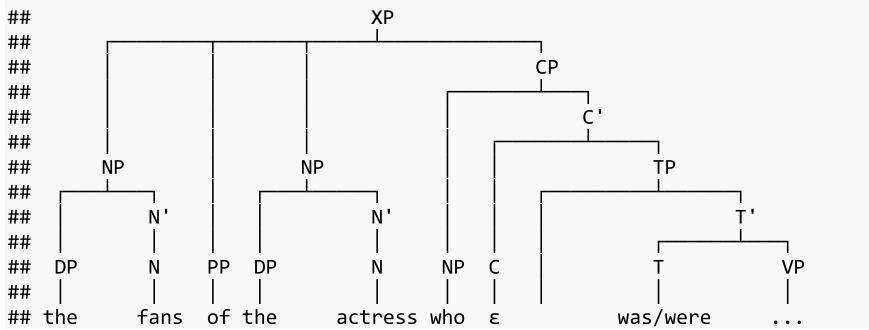
In this quotation, the term "chunk" introduces an ambiguity that allows for different interpretations. One possible syntactic structure computed by the L2 processor, which is not endorsed by the SSH but aligns with its verbal descriptions, is shown below:

(6) XP → NP PP NP CP

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

```
T5 = nltk.Tree.fromstring("[XP [NP [DP the] [N' [N fans]]] [PP of] [NP [DP
 the] [N' [N actress]]] [CP [NP who] [C' [C ε] [TP [] [T' [T was/were] [V
P]]]]]]", brackets = "[]")

T5.pretty_print(unicodelines = True)
```



In (6), the symbol X is an arbitrary node. The syntactic structure of T5 is hierarchical, as claimed by the SSH. However, the relative clause does not adjoin to either the higher or the lower nominal, and there is no trace. Regarding the parsing process, we can assume the following: "the fans of the actress" is first computed as a ternary tree, such as [$_{XP}$ [$_{NP}$ the fans] [$_{PP}$ of] [$_{NP}$ the actress]], forming a single chunk, as claimed by the SSH. When "who" appears, the relative clause structure is analysed as another child of XP, resulting in the quaternary tree.[15] As noted above, this structure is not necessarily what the SSH claims the L2 processor computes. It is presented here as a hypothetical structure to demonstrate formal modelling, following the descriptions of the SSH. In the following subsection, we will model L2 memory retrieval processes, assuming that T5 is the relative clause structure computed by the L2 processor within the SSH framework.

The SSH: Formal modelling of memory retrieval processes

We formally model L2 memory retrieval processes at the auxiliary verb in (4a/b) using the activation model. Recall that this model assumes that lexical and structural information are used as retrieval cues. In (4a/b), we can assume that the auxiliary verb triggers the memory retrieval of the subject NP, given that these elements are grammatically related.[16]

[15]In the ternary tree, the lower nominal seems to be available for the modification of the relative clause, so this tree does not properly formalise the following idea: "their [L2 learners'] apparent inability to apply any structure-based ambiguity resolution strategies … is expected if their representations of NP of NP complexes lack the relevant structural detail" (Clahsen & Felser, 2006b, p. 33). It seems that the SSH assumes the presence of a single NP structure that is not a projection of "fans" or "actress." Without clarification from the proponents of the SSH, it is difficult to formalise this idea.

[16]Within the SSH framework, there is no such subject-verb dependency for the L2 processor in (4a/b) because there is no element in the subject position of the relative clause (i.e., there is no element at the specifier of the TP; see T5). However, we can assume that memory retrieval still occurs at the auxiliary verb, since the L2 processor requires an element to link it to the verb in order to interpret the sentence.

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

In this memory retrieval process, we focus on two cues: the number cue and the (grammatical) category cue. The number cue is based on the fact that the verb agrees with the subject NP in number. In (4a/b), the auxiliary verbs "was" and "were" are marked as singular and plural, respectively. We assume that this number feature is used as a retrieval cue (Wagers et al., 2009). The grammatical category is a possible cue, given that an auxiliary verb often forms a subject-verb dependency with an NP. Note that a subject-verb dependency is established within a local configuration in the syntactic structure. While this structural information is crucial for the L1 processor, within the SSH framework, we can hypothesise that it is not important for the L2 processor in relative clauses such as those in (4a/b). Therefore, we do not consider the structure-based cue for L2 memory retrieval.

In the activation model, each element has an activation value, determined by base-level activation, spreading activation, mismatch penalty, and noise. To simplify the calculation of activation values, we make two assumptions. Firstly, each word in long-term memory has an activation value of 0 prior to the first retrieval. Secondly, the L2 processor takes 300 ms to process each word up to the auxiliary verb. Thus, in (4a/b), we assume that the L2 processor takes 1200 ms to reach the auxiliary verb from "fans" and 300 ms from "actress."

With these assumptions, we first calculate the base-level activation. Recall that base-level activation is related to previous retrievals and time-dependent decay and is expressed by the equation $b_i + ln\left(\sum_{k=1}^{n} t_k^{-d}\right)$. For our modelling, the decay rate (d) is set to the default value of .5. The default value is used because we know little about how quickly information about encountered words decays during real-time sentence processing.[17] The parameter t, representing the time elapsed since the kth retrieval of the element, is set to 1.2 for NP1 and .3 for NP2 (see footnote 8). The resting level of activation b is set to 0. With these values, the base-level activations of NP1 and NP2 can be calculated in R as follows[18]:

```
# Sum all retrievals
Sum_NP1 <- sum(c(1.2^-0.5))
Sum_NP2 <- sum(c(0.3^-0.5))

# Resting level of activation
b <- 0

# Calculate base-level activation
BA_NP1 <- b + log(Sum_NP1)
BA_NP2 <- b + log(Sum_NP2)
```

Like Python, R is an object-oriented programming language. In the above code, the arrow operator <- assigns a value (e.g., `0`) on the right-hand side to the variable on the left-hand side (e.g., `b`). The `sum()` function sums the values of all retrievals for each

[17]In the ACT-R model, default values are often set based on applications of the model to data. For example, previous studies have shown that the model generally fits data well with a decay rate of around .5 (e.g., Pavlik & Anderson, 2005). Following these studies, the default value is used for the decay rate. However, as mentioned earlier in this paper, there must ultimately be a basis for each input value.

[18]Some of the R code used in this paper for the formal modelling of memory retrieval is based on the code used in Engelmann, Jäger & Vasishth (2019).

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

Table 3. Cue matches and mismatches with NP1 and NP2 in L2 sentence processing

	Number cue	Category cue
(4a)		
NP1	Match	Match
NP2	Mismatch	Match
Fan effect	No	Yes
(4b)		
NP1	Mismatch	Match
NP2	Match	Match
Fan effect	No	Yes

NP. The input to this function is the time elapsed since the *k*th retrieval of the element, raised to the power of the decay rate. There is only one input value because there is only one retrieval before the auxiliary verb appears. The output of the `sum()` function is then passed to the `log()` function to obtain its natural logarithm. Recall that in R, anything written after a hash symbol on the same line is ignored when the code is executed. In this paper, a single hash is used to provide a brief description of code.

Next, we calculate the associative strength. The associative strength is influenced by whether the element matches the cue, and by how many matching elements there are. Cue matches and mismatches with NP1 and NP2 are summarised in Table 3.

In our model, a match is assigned 1, and a mismatch is assigned 0. The fan effect occurs for the category cue because NP1 and NP2 belong to the same grammatical category. For (4a), the following code encodes this cue match and mismatch information:

```
# Cue match = 1, cue mismatch = 0
Cue_Mx_4a <- matrix(c(1, 0, 1, 1), ncol = 2)

# Show the matrix
Cue_Mx_4a
##      [,1] [,2]
## [1,]    1    1
## [2,]    0    1
```

Here, the values `1, 0, 1, 1`, representing either a cue match or a cue mismatch, are combined into a vector using `c()` to create a matrix. The argument `ncol = 2` specifies that the matrix should have two columns. These matrix values are then used to quantify the fan effect and compute the associative strength ($ln(P(i|j))$) for (4a):

```
# Fan effect and associative strength
P_NP1_4a <- log(Cue_Mx_4a[1,] / (Cue_Mx_4a[1,] + Cue_Mx_4a[2,]) + 0.0001)
P_NP2_4a <- log(Cue_Mx_4a[2,] / (Cue_Mx_4a[1,] + Cue_Mx_4a[2,]) + 0.0001)
```

The code `Cue_Mx_4a[ 1,]` indicates the first row of the matrix (NP1's values; 1 1), while `Cue_Mx_4a[ 2,]` indicates the second row (NP2's values; 0 1). The / symbol represents a division. The code in the `log()` functions therefore calculates the proportion of the values in either the first or second row in relation to the sum of

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

the values in both rows. The fan effect occurs when both NP1 and NP2 are 1. A small value (`0.0001`) is added to avoid taking the logarithm of zero ($log(x)$ is only defined for $x > 0$). The resulting values are then combined with the maximum associative strength ($m + \ln(P(i|j))$):

```
# Maximum associative strength
m <- 4

# Calculate associative strength
AS_NP1_4a <- m + P_NP1_4a
AS_NP2_4a <- m + P_NP2_4a
```

The maximum associative strength is set to a high value, as NP1 and NP2 potentially match other cues, which are not used in this modelling for expository reasons. Next, we calculate the spreading activation ($\sum_{j=1}^{n} W_j S_{ji}$):

```
# Cue weight
G <- 1
CueWeight <- c(0.5, 0.5)
NumberOfCue <- length(CueWeight)

W <- matrix(G * CueWeight, ncol = NumberOfCue)

# Calculate spreading activation
CW_AS_NP1_4a <- rowSums(W * matrix(AS_NP1_4a, nrow = 1))
CW_AS_NP2_4a <- rowSums(W * matrix(AS_NP2_4a, nrow = 1))
```

Here, the amount of activation (`G`) and the distribution ratio (`0.5, 0.5`) are first set. The `length()` function is then used to return the number of elements in the `CueWeight` vector to set the number of cues. In the first `matrix()` function, the amount of activation is multiplied by each element in the `CueWeight` vector to set the value of W. The `rowSums()` functions calculate the sum of the element-wise product of the cue weight and the associative strength. We then calculate the mismatch penalty (i.e., $\sum_{j=1}^{n} pM_{ji}$):

```
# Scaling parameter
pm <- 0.2

# Calculate mismatch penalty
PM_NP1_4a <- rowSums(matrix(pm * (Cue_Mx_4a[1,] - 1), nrow = 1))
PM_NP2_4a <- rowSums(matrix(pm * (Cue_Mx_4a[2,] - 1), nrow = 1))
```

In this calculation, matched combinations are assigned 0, and mismatched combinations are assigned -1. This is done by subtracting 1 from the values in the matrix representing the cue matches (1) or mismatches (0). We now combine the base-level

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

activation, the spreading activation, and the mismatch penalty to calculate the total activation without noise:

```
TA_NP1_4a <- BA_NP1 + CW_AS_NP1_4a + PM_NP1_4a
TA_NP2_4a <- BA_NP2 + CW_AS_NP2_4a + PM_NP2_4a
```

The noise component is a normally distributed random variable (i.e., $Normal\left(0, \sqrt{\frac{\pi^2}{3}x^2}\right)$) that randomly affects the total activation. If one wants to look at the fixed output of the model, the total activation without noise can be used. We can also approximate this by generating many data points from the noise function and calculating the mean retrieval time. This approach allows us to consider variability in human behaviour. In R, the `rnorm()` function can be used to generate random numbers from a normal distribution:

```
# Scaling parameter
x <- 0.2

# Noise function
noise_NP1_4a <- rnorm(5000, 0, sqrt(pi^2 / 3 * x^2))
noise_NP2_4a <- rnorm(5000, 0, sqrt(pi^2 / 3 * x^2))

head(noise_NP1_4a)
## [1] -0.587228764 -0.243879548  0.065909245  0.005726552 -0.028061719
## [6] -0.239806528

head(noise_NP2_4a)
## [1] -0.82140380 -0.55057262  0.17897066  0.12159180 -0.06242845  0.0606
2152
```

In the `rnorm()` functions, `5000` is the number of data points to generate, `0` is the mean of the distribution, and `sqrt(pi^2 / 3 * x^2)` is the standard deviation of the distribution. The `head()` functions display the first six data points. The data points for NP1 and NP2 differ because the `rnorm()` function generates random numbers. The following code combines the total activations with each data point from the `rnorm()` functions:

```
TA_NP1_4aNoise <- TA_NP1_4a + noise_NP1_4a
TA_NP2_4aNoise <- TA_NP2_4a + noise_NP2_4a
```

We now have 5,000 activation values for each of NP1 and NP2 in (4a). Let us first check which NP has higher activation values:

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

```
Comparison_Greater <- sum(TA_NP1_4aNoise > TA_NP2_4aNoise)
Comparison_Equal <- sum(TA_NP1_4aNoise == TA_NP2_4aNoise)
TrialN <- length(TA_NP1_4aNoise)

cat("Out of", TrialN ,"trials, NP1 has a higher activation than NP2", Comp
arison_Greater, "times.")
## Out of 5000 trials, NP1 has a higher activation than NP2 5000 times.

cat("Out of", TrialN ,"trials, NP1 and NP2 have the same activation", Comp
arison_Equal, "times.")
## Out of 5000 trials, NP1 and NP2 have the same activation 0 times.
```

The first line of this code calculates the number of times `TA_NP1_4aNoise` is greater than (>) `TA_NP2_4aNoise`, and the second line calculates the number of times they are equal (==). The third line uses the `length()` function to get the total number of trials (i.e., 5,000). The `cat()` functions tell us how many times `TA_NP1_4aNoise` is greater than or equal to `TA_NP2_4aNoise`. Since these functions show that NP1 always has a higher activation than NP2, we focus on NP1.

Next, we examine the retrieval status of NP1. Recall that in the activation model, successful retrieval only occurs when the element with the highest activation reaches a retrieval threshold. The following code checks the retrieval status of NP1:

```
# Retrieval threshold
reth <- -1.5

# Extract trials with higher activation
TA_4aNoise <- pmax(TA_NP1_4aNoise, TA_NP2_4aNoise)

Retrieval_Status <- sum(TA_4aNoise >= reth)

cat("Out of", TrialN ,"trials, NP1 is retrieved", Retrieval_Status, "times.
")
## Out of 5000 trials, NP1 is retrieved 5000 times.
```

The retrieval threshold is set to the default value of –1.5. The default value is used because little is known about what threshold is appropriate for memory retrieval during sentence processing. The `pmax()` function compares the two input vectors (i.e., `TA_NP1_4aNoise` and `TA_NP2_4aNoise`), each containing 5,000 activation values of either NP1 or NP2 in (4a), and returns the maximum value at each position. As NP1 always has a higher activation value, the output of this function only contains the activation values of NP1. In the `sum()` function, the >= operator is used to check whether each activation value in `TA_NP1_4aNoise` is greater than or equal to the retrieval threshold. The `cat()` function shows that NP1 is always retrieved (i.e., its activation value is always greater than or equal to the retrieval threshold). Finally, we calculate the retrieval times of the 5,000 trials using the equation $Fe^{-(f \cdot TA_i)}$:

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

```
# Scaling parameters
lf <- 0.2
le <- 0.1

# Calculate retrieval times
RT_4a <- lf * exp(-le * TA_4aNoise) * 1000

mean(RT_4a)
## [1] 140.1636
```

The `exp()` function calculates the value of *e* raised to the power of (`-le * TA_4aNoise`). The resulting value is multiplied by `1000` for interpretive purposes. The `mean()` function shows that the average retrieval time at the auxiliary verb in (4a) is approximately 140 ms.

We have formally modelled L2 memory retrieval processes in (4a). The next step is to do the same for (4b) and for L1 memory retrieval processes. The code for (4b) is identical except for cue matches and mismatches (see Table 3). The formal modelling of L1 memory retrieval processes needs to take into account the following factors: a preference for the NP2 modification, the presence of the trace, and the structure-based cue. The details are omitted here for space reasons, but the code is available on the OSF website and in the R notebook on Colab.

In this subsection, we have focused on memory retrieval processes. However, as mentioned earlier in this paper, sentence processing potentially involves several other processes. To take these processes into account without actually modelling them, we simply assume the time taken by them. The activation model assumes that L1 sentence interpretation processes other than memory retrieval, such as parsing and attending to a word, take approximately 150 ms. Since L2 sentence processing usually takes longer than L1 sentence processing, we add 50 ms to this processing time for L2 sentence processing:

```
RTplus_4a <- 200 + RT_4a
```

The final processing times are shown in Figure 10, juxtaposed with the data from Felser et al. (2003) and the results of the formal modelling of L1 memory retrieval processes.

Overall, our model shows shorter processing times than the original data. It also shows longer processing times for the L2 group than for the L1 group, but this is a common finding in L2 sentence processing research. Importantly, our model shows the processing patterns that are largely consistent with the original data in terms of the differences between (4a) and (4b) for each language group.

In this section, we have formally modelled the relative clause structures and the memory retrieval processes in order to formalise the SSH and to demonstrate how formal modelling can be incorporated into L2 sentence processing research. In the following section, we will formalise the IBH, another hypothesis about L2 sentence processing.

The interference-based hypothesis

The IBH is a relatively recent hypothesis, proposed by Cunnings (2017b). As in the case of the SSH, we first review its claims.

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

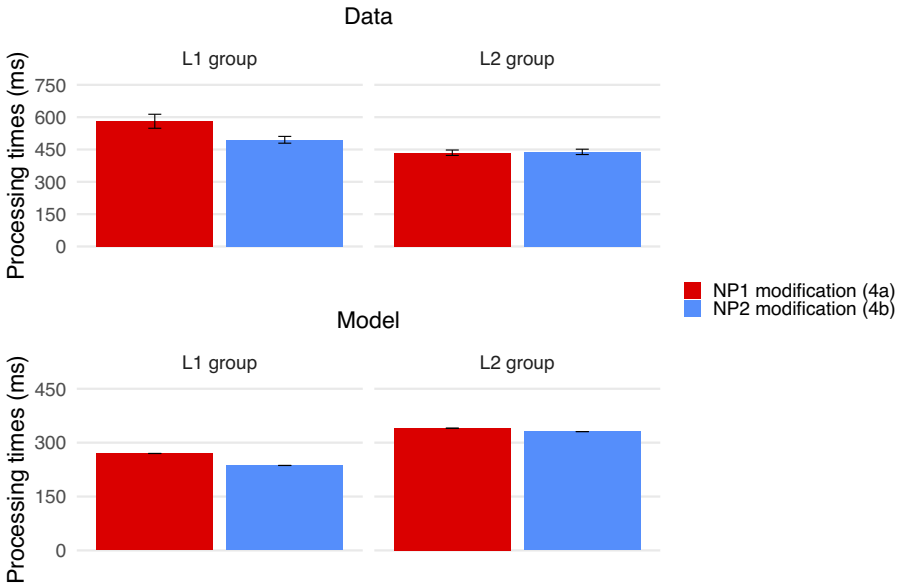


Figure 10. The data from Felser et al. (2003) and the model outputs. Error bars are standard errors.

Verbal descriptions of the IBH

The IBH argues that the source of L1/L2 differences processing lies in memory retrieval processes:

> [T]he primary source for L1/L2 differences relates to the ability to successfully retrieve information that has been constructed during sentence processing from memory (Cunnings, 2017b, p. 662).

According to the IBH, the L2 processor is more susceptible to interference (e.g., the fan effect) than the L1 processor during memory retrieval:

> I argue that L2 speakers are more susceptible to retrieval interference when successful comprehension requires access to information from memory (Cunnings, 2017b, p. 659).

The IBH claims that the L2 processor relies heavily on discourse-based information during memory retrieval:

> L2 learners may rely more heavily on discourse-based cues to memory retrieval than L1 speakers (Cunnings, 2017b, p. 663).
>
> For L2 speakers, the preference for a subject/topic seems to be a more heavily weighted cue than in L1 speakers (Cunnings, 2017b, p. 669).

Unlike the SSH, the IBH argues that the L2 processor computes fully detailed hierarchical structures:

> I will argue that existing research indicates that L2 learners do construct fully-specified syntactic parses (Cunnings, 2017b, p. 662).

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

However, the IBH claims that the L2 processor may be less reliant on the structure-based cue during memory retrieval than the L1 processor:

> Even if L2ers compute elaborate sentence structure, they may not implement retrieval cues drawn from this structure in the same way as L1ers (Cunnings, 2017a, p. 713).

In summary, the IBH claims that the L2 processor computes fully detailed syntactic structures as does the L1 processor during real-time processing. However, the L2 processor is more susceptible to interference during memory retrieval than the L1 processor. This increased susceptibility to interference arises because the L2 processor relies heavily on discourse-based information and underweights structure-based information during memory retrieval compared to the L1 processor. This claim can be formalised using the activation model. There are, however, two approaches to incorporating the heavy weighting of discourse-based information into the activation model. The first approach, proposed by Cunnings (2017b), is to assume the existence of a retrieval cue derived from topic information on which the L2 processor relies heavily (Cunnings, 2017b, p. 670).[19] The second approach is to assume that information about discourse prominent elements fades slowly from memory, or that such elements have a constant activation value. The former can be expressed by using cue weights, whereas the latter can be expressed by using the base-level activation equation. There are several issues with the former approach (Dillon, 2017; Jacob, Lago & Patterson, 2017). For example, an element does not necessarily become a topic when it is first encoded, and although discourse-based information may influence sentence processing, it does not necessarily constrain dependency relations. Despite these issues, we adopt the former approach to properly formalise the IBH by making the following assumptions:

(a) The L2 processor analyses the first NP it encounters as a topic, encodes this information as a feature alongside other information about the NP, and updates this topic feature in some way as needed during real-time processing.
(b) The L2 processor always uses topic as a retrieval cue.[20]

While these assumptions are not endorsed by the IBH, they are not inconsistent with its claims and help to address the issues raised.

Cunnings (2017b) discusses various studies to support the IBH. Among these is Felser, Sato and Bertenshaw (2009), who conducted an experiment to investigate real-time reflexive resolution using sentences such as the following:

(7) Introductory sentence: John/Jane and Richard were very worried in the kitchen of the expensive restaurant.

(a) [$_{NP1}$ John] noticed that [$_{NP2}$ Richard] had cut **himself** with a very sharp knife.
(b) [$_{NP1}$ Jane] noticed that [$_{NP2}$ Richard] had cut **himself** with a very sharp knife.

In principle, a reflexive corefers with an NP at a specific position in the syntactic structure (Chomsky, 1981). Due to this structural constraint, in (7a/b), only NP2 is structurally accessible for coreference with the reflexive. Therefore, NP2 is the target for

[19]Roughly speaking, topic refers to aboutness (what a sentence [or sentences] is about; for a discussion, see Reinhart, 1981).

[20]Later in this paper, we will assume that the L2 processor does not use the topic cue during re-retrieval. Therefore, this second assumption is confined to the initial retrieval.

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

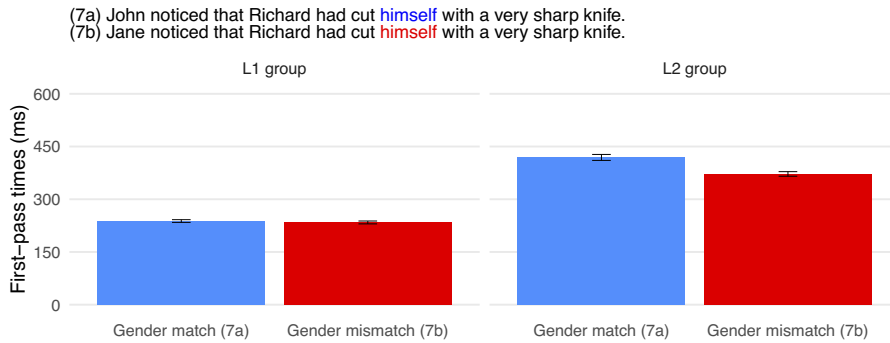


Figure 11. First-pass times at the reflexive in (7a/b) reported in Felser et al. (2009). Error bars are approximate standard errors.

antecedent retrieval at the reflexive, and NP1 is a distractor (see footnote 9 for definitions of *target* and *distractor* in the context of memory retrieval). The conventionally assigned gender of NP1 either matches ("John") or mismatches ("Jane") the gender of the reflexive. In an eye-movement-during-reading task, Felser et al. (2009) observed longer first-pass times at the reflexive in (7a) than in (7b) for L2 speakers, but not for L1 speakers. These processing patterns are shown in Figure 11.

Cunnings (2017b) interprets these findings as indicating an increased susceptibility of the L2 processor to interference from the gender-matching discourse prominent NP1 (assuming that NP1 is a topic).

The IBH: Formal modelling of syntactic structures

We now formalise the IBH, focusing on the reflexive resolution in (7a/b). As in the case of the SSH, we first model syntactic structures and then memory retrieval processes. In (7a/b), there are two finite clauses, with one embedded within the other:

```
T6 = nltk.Tree.fromstring("[TP [NP John/Jane] [T' [T PAST] [VP [V noticed]
 [CP [C that] [TP [NP Richard] [T' [T had] [VP [V cut] [NP himself]]]]]]]]
", brackets = "[]")

T6.pretty_print(unicodelines = True)
##              TP
##      ┌────────┴──────┐
##      │              T'
##      │       ┌───────┴──────┐
##      │       │             VP
##      │       │       ┌──────┴──────┐
##      │       │       │            CP
##      │       │       │       ┌─────┴──────┐
##      │       │       │       │           TP
##      │       │       │       │      ┌─────┴──────┐
##      │       │       │       │      │           T'
##      │       │       │       │      │      ┌─────┴──────┐
##      │       │       │       │      │      │           VP
##      │       │       │       │      │      │      ┌─────┴──────┐
##     NP      T       V       C      NP      T      V            NP
##      │       │       │       │      │      │      │            │
## John/Jane PAST noticed that Richard had     cut     himself
```

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

Since the IBH claims that the L2 processor computes fully detailed syntactic structures like the L1 processor, we assume that the L2 processor computes T6 when given the sentences (up to the reflexive) in (7a/b).

The IBH: Formal modelling of memory retrieval processes

Assuming T6, we formally model L2 memory retrieval processes at the reflexive. The first step is to calculate the base-level activation. As in the case of the SSH, we make the following assumptions about sentence processing: Each word in long-term memory has an activation value of 0, and each word up to the reflexive takes 300 ms to process. Therefore, our L2 processor spends 1,500 ms between "John/Jane" and "himself" and 600 ms between "Richard" and "himself." We also assume that a finite verb does not trigger the retrieval of the corresponding subject NP for expository purposes. The decay rate is identical to that used for the SSH:

```
Sum_NP1 <- sum(c(1.5^-0.5))
Sum_NP2 <- sum(c(0.6^-0.5))
b <- 0

BA_NP1 <- b + log(Sum_NP1)
BA_NP2 <- b + log(Sum_NP2)
```

Next, we calculate the associative strength. To formalise the IBH, we need to consider three retrieval cues: structure, gender, and topic cues. According to the IBH, the L2 processor relies heavily on the topic cue and does not give the same weight to the structure cue as the L1 processor. There is ambiguity here about the specific weighting of these cues. In our model, we assume a weighting distribution of structure cue = 2/10, gender cue = 3/10, and topic cue = 5/10.[21] Table 4 summarises matches and mismatches of the cues with NP1 and NP2 as well as the cue weights.

As in the case of the SSH, we assign 1 to a match and 0 to a mismatch.

Table 4. Cue matches and mismatches with NP1 and NP2 in L2 sentence processing

	Structure cue (0.2)	Gender cue (0.3)	Topic cue (0.5)
(7a)			
NP1	Mismatch	Match	Match
NP2	Match	Match	Mismatch
Fan effect	No	Yes	No
(7b)			
NP1	Mismatch	Mismatch	Match
NP2	Match	Match	Mismatch
Fan effect	No	No	No

Note: Values in parentheses are cue weights.

[21]The structure cue is set at .2 because, although the IBH argues that the L2 processor relies less on this cue than the L1 processor, it does not claim that the L2 processor does not use it. Thus, structure cue = 2/10 seems reasonable (assuming cues are equally weighted in L1 sentence processing). We now need to divide the remaining .8 between the gender and topic cues. Splitting it into .4 for each is inconsistent with the IBH because, according to the IBH, the L2 processor relies heavily on the topic cue. This leaves three options: <gender cue = .1, topic cue = .7>, <gender cue = .2, topic cue = .6>, and <gender cue = .3, topic cue = .5>. The

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

```
# (7a)
Cue_Mx_7a <- matrix(c(0, 1, 1, 1, 1, 0), ncol = 3)

# (7b)
Cue_Mx_7b <- matrix(c(0, 1, 0, 1, 1, 0), ncol = 3)
```

These values are then used to calculate the associative strength:

```
# (7a)
P_NP1_7a <- log(Cue_Mx_7a[1,] / (Cue_Mx_7a[1,] + Cue_Mx_7a[2,]) + 0.0001)
P_NP2_7a <- log(Cue_Mx_7a[2,] / (Cue_Mx_7a[1,] + Cue_Mx_7a[2,]) + 0.0001)

# (7b)
P_NP1_7b <- log(Cue_Mx_7b[1,] / (Cue_Mx_7b[1,] + Cue_Mx_7b[2,]) + 0.0001)
P_NP2_7b <- log(Cue_Mx_7b[2,] / (Cue_Mx_7b[1,] + Cue_Mx_7b[2,]) + 0.0001)
```

The resulting values are aggregated with the maximum associative strength to complete the calculation of the associative strength:

```
m <- 4

# (7a)
AS_NP1_7a <- m + P_NP1_7a
AS_NP2_7a <- m + P_NP2_7a

# (7b)
AS_NP1_7b <- m + P_NP1_7b
AS_NP2_7b <- m + P_NP2_7b
```

Next, we calculate the weighted spreading activation. As summarised in Table 4, in our model, the topic cue is heavily weighted, while the contribution of the structure cue is reduced, as claimed by the IBH:

```
# Structure cue = 0.2, Gender cue = 0.3, Topic cue = 0.5
G <- 1
CueWeight <- c(0.2, 0.3, 0.5)
NumberOfCue <- length(CueWeight)

W <- matrix(G * CueWeight, ncol = NumberOfCue)

# (7a)
CW_AS_NP1_7a <- rowSums(W * matrix(AS_NP1_7a, nrow = 1))
CW_AS_NP2_7a <- rowSums(W * matrix(AS_NP2_7a, nrow = 1))

# (7b)
CW_AS_NP1_7b <- rowSums(W * matrix(AS_NP1_7b, nrow = 1))
CW_AS_NP2_7b <- rowSums(W * matrix(AS_NP2_7b, nrow = 1))
```

first two options do not seem to be consistent with the IBH in terms of the relative weights between the structure and gender cues, whereas the third option is largely consistent with the IBH. Therefore, the weighting distribution of structure cue = 2/10, gender cue = 3/10, and topic cue = 5/10 is adopted.

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

The mismatch penalty is then calculated to penalise the mismatched combinations:

```
pm <- 0.2

# (7a)
PM_NP1_7a <- rowSums(matrix(pm * (Cue_Mx_7a[1,] - 1), nrow = 1))
PM_NP2_7a <- rowSums(matrix(pm * (Cue_Mx_7a[2,] - 1), nrow = 1))

# (7b)
PM_NP1_7b <- rowSums(matrix(pm * (Cue_Mx_7b[1,] - 1), nrow = 1))
PM_NP2_7b <- rowSums(matrix(pm * (Cue_Mx_7b[2,] - 1), nrow = 1))
```

We now combine the base-level activation, the spreading activation and the mismatch penalty to calculate the total activation without noise.

```
# (7a)
TA_NP1_7a <- BA_NP1 + CW_AS_NP1_7a + PM_NP1_7a
TA_NP2_7a <- BA_NP2 + CW_AS_NP2_7a + PM_NP2_7a

# (7b)
TA_NP1_7b <- BA_NP1 + CW_AS_NP1_7b + PM_NP1_7b
TA_NP2_7b <- BA_NP2 + CW_AS_NP2_7b + PM_NP2_7b
```

For the noise component, we generate random numbers from a normal distribution:

```
# Scaling parameter
x <- 0.2

# Noise function
variables <- c("noise_NP1_7a", "noise_NP2_7a", "noise_NP1_7b", "noise_NP2_
7b")

for (var in variables) {
  assign(var, rnorm(5000, 0, sqrt(pi^2 / 3 * x^2)))
}
```

In this code, we first set the scaling parameter and then create a vector containing four variables. These variables are used in the for loop. In the for loop, the `assign()` function assigns 5,000 random numbers, generated by the `rnorm()` function, to each variable. These random numbers are then combined with the corresponding total activations:

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

```
# (7a)
TA_NP1_7aNoise <- TA_NP1_7a + noise_NP1_7a
TA_NP2_7aNoise <- TA_NP2_7a + noise_NP2_7a

# (7b)
TA_NP1_7bNoise <- TA_NP1_7b + noise_NP1_7b
TA_NP2_7bNoise <- TA_NP2_7b + noise_NP2_7b
```

Let us now examine the outputs of the model. First, we check which NP has higher activation values in (7a) and (7b):

```
# (7a)
Comparison_Greater_7a <- sum(TA_NP1_7aNoise > TA_NP2_7aNoise)
Comparison_Equal_7a <- sum(TA_NP1_7aNoise == TA_NP2_7aNoise)
TrialN <- length(TA_NP1_7aNoise)

cat("Out of", TrialN ,"trials, NP1 has a higher activation than NP2", Comp
arison_Greater_7a, "times.")
## Out of 5000 trials, NP1 has a higher activation than NP2 5000 times.

cat("Out of", TrialN ,"trials, NP1 and NP2 have the same activation", Comp
arison_Equal_7a, "times.")
## Out of 5000 trials, NP1 and NP2 have the same activation 0 times.

# (7b)
Comparison_Greater_7b <- sum(TA_NP1_7bNoise > TA_NP2_7bNoise)
Comparison_Equal_7b <- sum(TA_NP1_7bNoise == TA_NP2_7bNoise)

cat("Out of", TrialN ,"trials, NP1 has a higher activation than NP2", Comp
arison_Greater_7b, "times.")
## Out of 5000 trials, NP1 has a higher activation than NP2 503 times.

cat("Out of", TrialN ,"trials, NP1 and NP2 have the same activation", Comp
arison_Equal_7b, "times.")
## Out of 5000 trials, NP1 and NP2 have the same activation 0 times.
```

In (7a), NP1 always has a higher activation, while in (7b), NP2 almost always has a higher activation ($\frac{(5000-503)}{5000} \approx 90\%$). This variation in retrieval in (7b) is due to the noise function. Next, we check the retrieval status. For (7a), we focus on NP1, as its activation is consistently higher. For (7b), we examine the trials with higher activation.

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

```
# Retrieval threshold
reth <- -1.5

# (7a)
TA_7aNoise <- pmax(TA_NP1_7aNoise, TA_NP2_7aNoise)
Retrieval_Status <- sum(TA_7aNoise >= reth)
cat("Out of", TrialN ,"trials, retrieval is successful", Retrieval_Status,
 "times.")
## Out of 5000 trials, retrieval is successful 5000 times.

# (7b)
TA_7bNoise <- pmax(TA_NP1_7bNoise, TA_NP2_7bNoise)
Retrieval_Status <- sum(TA_7bNoise >= reth)
cat("Out of", TrialN ,"trials, retrieval is successful", Retrieval_Status,
 "times.")
## Out of 5000 trials, retrieval is successful 4996 times.
```

Recall that the `pmax()` function compares the input vectors and returns the maximum value at each position. In (7a), memory retrieval is always successful, whereas in (7b), it is almost always successful (99%). The few retrieval failures in (7b) are due to the noise function. We now calculate the retrieval times of the 5,000 trials in (7a) and (7b) and add 200 ms to these retrieval times to take into account the processes of L2 sentence processing other than memory retrieval:

```
# Scaling parameters
lf <- 0.2
le <- 0.1

# Calculate retrieval times
# (7a)
RT_7a <- lf * exp(-le * TA_7aNoise) * 1000
RTplus_7a <- 200 + RT_7a

# (7b)
RT_7b <- lf * exp(-le * TA_7bNoise) * 1000
RTplus_7b <- 200 + RT_7b
```

The results are shown in Figure 12, alongside the L2 data from Felser et al. (2009).

Although we have formalised the IBH based on its verbal descriptions, the results show different processing patterns between the data and the model outputs. Specifically, while the data show longer reading times in (7a) than in (7b), the model shows longer reading times in (7b) than in (7a).

Discrepancies between data and model output

In the previous subsection, we formalised the IBH, but the model did not produce retrieval times that were consistent with the data used to support the IBH. What accounts for this discrepancy? Looking at the unweighted spreading activations of the

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

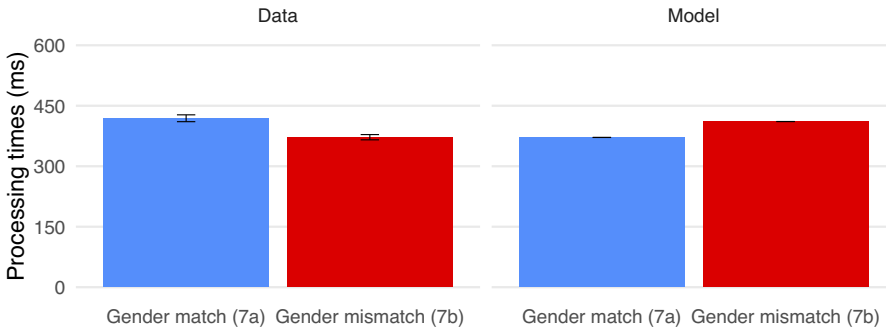


Figure 12. The data from Felser et al. (2009) and the model outputs.

elements that were (almost) always retrieved, specifically NP1 in (7a) and NP2 in (7b), we can see that the former has a lower activation than the latter:

```
# NP1's unweighted activation in (7a)
AS_NP1_7a_D <- matrix(AS_NP1_7a, nrow = 1)
colnames(AS_NP1_7a_D) <- c("Structure cue", "Gender cue", "Topic cue")
AS_NP1_7a_D
##      Structure cue Gender cue Topic cue
## [1,]      -5.21034   3.307053    4.0001

# NP2's unweighted activation in (7b)
AS_NP2_7b_D <- matrix(AS_NP2_7b, nrow = 1)
colnames(AS_NP2_7b_D) <- c("Structure cue", "Gender cue", "Topic cue")
AS_NP2_7b_D
##      Structure cue Gender cue Topic cue
## [1,]        4.0001     4.0001  -5.21034
```

However, with the cue weights assumed by the IBH, this pattern is reversed:

```
# NP1's weighted activation in (7a)
W * AS_NP1_7a_D
##      Structure cue Gender cue Topic cue
## [1,]     -1.042068  0.9921158   2.00005

# NP2's weighted activation in (7b)
W * AS_NP2_7b_D
##      Structure cue Gender cue Topic cue
## [1,]       0.80002    1.20003  -2.60517
```

This is because the heavily weighted topic cue causes the activation of NP1 from this cue to remain high in (7a) and causes the activation of NP2 from this cue to remain low in (7b). Additionally, the underweighted structure cue substantially increases the

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

activation of NP1 in (7a), while in (7b), NP2, which matches this cue, experiences a substantial reduction in activation.[22]

Cunnings's (2017b) interpretation of the Felser et al. (2009) data did not lead the model to produce the observed processing patterns. This incompatibility could have been detected if the IBH had been proposed in conjunction with formal modelling. A discrepancy between data and model output can occur for several reasons: (a) the data do not represent the object of interest (i.e., the data are unreliable), (b) the model is not properly built (e.g., the hypothesis is not properly implemented or is oversimplified), (c) the fit between the data and the model output is not properly evaluated, and (d) the hypothesis implemented by the model is incorrect. The reliability of the data can be assessed by conducting replication experiments and examining whether the results can be replicated. The model fit can be evaluated by using certain methods (some of which are discussed in the general discussion). In our case, (c) does not seem to be an issue, given that Figure 12 clearly shows different patterns between the data and the model outputs. Assuming that Felser et al.'s results are reliable and that we have properly formalised the IBH, if we wish to defend the IBH, we need to make an additional assumption about L2 sentence processing that is compatible with its claims and revise the model of syntactic structure or the model of memory retrieval accordingly. Alternatively, we could formalise the IBH in some other way. Let us consider whether revising the model of memory retrieval works. When revising a model to address a discrepancy with data, it is important to make the revision on reasonable grounds. That is, a revision should not merely be made to enable the model to predict the data better; it should be made because there is a reasonable new hypothesis about the object of interest, and the model implementing that hypothesis may explain the data (recall that the purpose of formal computational modelling is to better understand the object of interest by expressing a hypothesis about it in mathematical form). One aspect to consider about the model of memory retrieval is what occurs after the retrieval of NP1 in (7a). Since NP1 is not the target, for successful sentence interpretation, this memory retrieval must be discarded and another memory retrieval (re-retrieval) must be initiated to search for the target (Fujita, 2024a).[23] As re-retrieval is an additional cognitive process, it increases processing times, which may resolve the incompatibility between the data and the model.

Before formalising this re-retrieval process, we make two assumptions about it. Firstly, when the L2 processor searches for the grammatical antecedent during re-retrieval in (7a), the topic cue is no longer used. This assumption is reasonable because the initial retrieval, in which this cue is heavily weighted, does not lead to the grammatical antecedent and because topic does not constrain the interpretation of the reflexive in (7a). Thus, in our model, re-retrieval only involves the structure and gender cues. Secondly, we assume that these cues are equally weighted during re-retrieval. This assumption is made in order to adopt a neutral stance on cue weights based on the fact that we know little about re-retrieval. Table 5 summarises these assumptions.

[22]Note that with equally weighted spreading activations, NP2 tends to have a higher activation than NP1 in (7a). This is because these NPs match the same number of cues, but NP2 is a more recently encoded element at the point of the reflexive. Using the equally weighted spreading activations brings the model outputs closer to the Felser et al. (2009) data, but there are still some discrepancies.

[23]This implies that the L2 processor immediately recognises that NP1 is not a grammatical antecedent for the reflexive because it does not match the structure cue. This raises the question of why the L2 processor relies on the structure cue despite its initial underweighting.

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

Table 5. A summary of cue matches and mismatches in L2 re-retrieval processes

	Structure cue (0.5)	Gender cue (0.5)
(7a)		
NP1	Mismatch	Match
NP2	Match	Match
Fan effect	No	Yes

Note: Values in parentheses are weights.

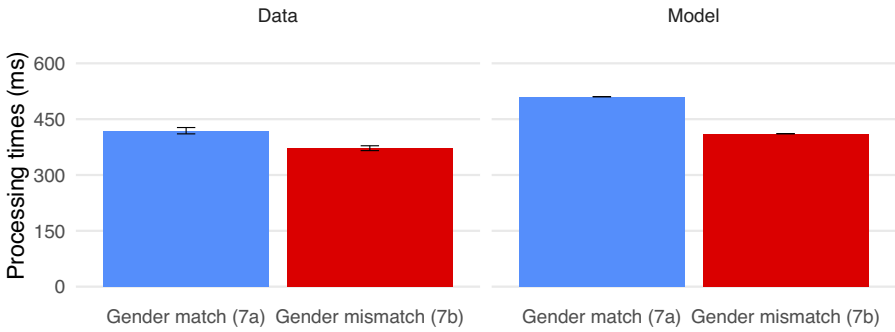


Figure 13. The data from Felser et al. (2009) and the outputs of the revised model.

We now need to formally model the re-retrieval process and combine the processing times of the initial memory retrieval and the re-retrieval. As the calculation process is largely the same as that for the initial retrieval, the code is omitted here for space reasons, but it can be found on the OSF website and in the R notebook on Colab. The results are shown in Figure 13.

Our revised model, which incorporates the re-retrieval process, shows processing patterns compatible with the Felser et al. (2009) data. This suggests that the IBH may require an additional assumption, such as re-retrieval, to account for Felser et al.'s observations.

Discussion

In this paper, I have demonstrated how formal computational modelling can be incorporated into L2 sentence processing research by formalising the SSH and the IBH. The SSH argues that the source of L1/L2 differences lies in the syntactic structures computed during real-time sentence processing, whereas the IBH claims that the L1 and L2 processors differ in their weighting of retrieval cues during memory retrieval. Such verbal descriptions alone can lead to misunderstandings, and I have shown that formal modelling can help to prevent them.

This paper took a step-by-step approach to formal modelling. However, due to its tutorial nature and space limitations, several issues could not be (fully) addressed. These issues are discussed briefly below.

One issue concerns model comparison. Recall that in this paper, we compared two different models of relative clause structures (one computed by the L1 processor and one by the L2 processor) and two different models of memory retrieval (one with and

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

one without a re-retrieval process). These comparisons were based on the same computational architecture (e.g., the two memory retrieval models were both based on the ACT-R architecture). In cognitive psychology, researchers also compare models built on different computational architectures to determine which best captures and explains human cognition. For example, there are cue-based models of memory retrieval that are implemented differently from the activation model (e.g., McElree, 2000; Parker, 2019). Comparing these different computational architectures helps to identify the most suitable one for describing memory retrieval processes (within the framework of cue-based memory retrieval).

Another important issue concerns a measure of the fit between the data and the model output. In this paper, we have compared the model output with the data by examining the presence or absence of effects and their direction. While such qualitative measures are useful, quantitative measures are also used in formal modelling. Quantitative measures often compare each output value from the model with the data to assess the performance of the model. One such measure is the root mean squared deviation (RMSD). The RMSD is the square root of the mean of the squared differences between the data and the model output. For example, for a difference between an observed effect size (y) and a model output (x), the RMSD is defined as $RMSD = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y - x_i)^2}$. A smaller RMSD value indicates that the model output is closer to the data. The RMSD can also be used to determine input values for variables in formal models under development. As discussed earlier in this paper, these values can be determined by fitting the models to data with different values and finding the values that best fit the data. The RMSD can be used as a measure for this data-driven estimation. Note that, as one reviewer suggested, the RMSD can only be used when the data and the model output are on the same scale. If scales are different, scale-independent measures such as the correlation coefficient can be useful. The correlation coefficient (r) is defined as $r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}}$, where x and y are data points and $\bar{x}$ and $\bar{y}$ are their respective means. The output of the correlation coefficient ranges from -1 (perfect negative correlation) to 1 (perfect positive correlation). The correlation coefficient only measures the direction and strength of the covariance between two variables. Therefore, a high correlation alone does not necessarily mean that the data and the model output have similar distributions and should not be interpreted as indicating a good fit between the model and the data. However, when used alongside complementary information (e.g., detailed visual representations of the data), the correlation coefficient can be a useful measure, and motivate further investigation. The online supplementary files on the OSF website and the R notebook on Colab demonstrate how to use the RMSD and correlation coefficient in R.

Another important issue is that, as has been mentioned repeatedly in this paper, real-time sentence processing potentially involves several processes other than memory retrieval, and it is important to model these formally as well. Consider, for example, the parsing system—an algorithm that computes syntactic structures from strings or sounds in real time. Various parsing algorithms have been proposed in the literature, which can be categorised as top-down parsing, bottom-up parsing, or a combination of the two (see, for example, Berwick & Stabler, 2019; Crocker, 1996, 1999; Fujita, 2023; Gibson, 1991; Grune & Jacobs, 2008; Hale, 2014; Johnson-Laird, 1983). Given that

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

some studies have suggested that the L2 processor, unlike the L1 processor, does not engage in predictive processing (e.g., Kaan, Kirkham & Wijnen, 2016), the L1 and L2 processors might use different parsing algorithms, such that the L2 processor relies more on bottom-up parsing than the L1 processor.

I would also like to emphasise the importance of considering individual variation when formally modelling L2 sentence processing. This consideration is crucial because some studies have only observed L1/L2 differences when comparing L1 speakers with a specific group of L2 speakers. For example, previous studies have observed different processing patterns between L1 speakers and L2 speakers with reduced lexical automaticity, L2 proficiency, or working memory capacity (e.g., Dussias & Piñar, 2010; Hopp, 2010, 2014). When considering individual variation, it is important to identify the cause. For example, Hopp (2014) reported that L2 speakers with relatively high lexical automaticity preferred the NP2 modification over the NP1 modification in a particular type of relative clause, whereas those with relatively low lexical automaticity showed no preference. This finding potentially suggests that lexical automaticity plays a role in L1/L2 differences. However, without understanding why different processing patterns emerge between L2 speakers with low and high lexical automaticity, we cannot properly incorporate L1/L2 differences into formal models. For the sake of discussion, let us assume that lexical processing is particularly cognitively demanding for L2 speakers with low lexical automaticity, and as a result, they lack sufficient processing resources for parsing and do not analyse the relative clause as relating to either nominal. Now that we know the cause, we can assume different syntactic structures for L2 speakers with low lexical automaticity and those with high lexical automaticity, and based on these structures, the activation model can be used to model memory retrieval processes for the two L2 groups. There are other ways to formally model individual variation, but it is essential to first establish a theory as to why L1/L2 differences are observed only in comparisons with a particular L2 group.

Finally, at the suggestion of one reviewer, I briefly discuss artificial neural networks (ANNs). ANNs are mathematical objects that mimic the structure of biological neural systems. They consist of multiple layers, each containing nodes (artificial neurons) connected by edges (artificial synapses). ANNs process input data through these layers using a training process that adjusts the weights on the edges to minimise the difference between the model output and the target state (desired output). In cognitive psychology, ANNs are used within the connectionist framework to study human cognition (McClelland, 2009). As discussed in Guest and Martin (2024), ANNs are often used differently today than in the past. In modern times, ANN models are frequently developed without specific hypotheses about human cognition and are trained to perform tasks using vast amounts of data. The model output is then compared with human data, and the degree of correlation between the two is used to assess whether the model represents human cognition. If the correlation is high, the model is often considered to be equivalent to human cognition[24] and is therefore considered worth studying in order to understand human cognition. This modern connectionist approach (Guest & Martin, 2024) differs from the traditional approaches adopted in cognitive psychology, where human cognition is studied (more) directly through experiments involving humans (including introspection), with models serving as tools to facilitate our understanding.

[24]If the model is equivalent to human cognition, it is not a model of human cognition according to the definition given in this paper.

https://doi.org/10.1017/S0272263125101009 Published online by Cambridge University Press

Are modern ANN models useful for studying sentence processing? Can they replace the approaches used in traditional sentence processing research? The answer depends on what people in the field think the goal of sentence processing research is. If the goal is not to predict the behaviour of the processor in a given context (e.g., whether a relative clause is analysed as modifying the first or second nominal in a sentence) but to understand sentence interpretation processes (e.g., why and how forms are paired with specific meanings, and why the processor behaves the way it does), then the role that modern ANN models can play in sentence processing research would be limited, and the traditional approaches would remain essential. This is not only because there is no clear connection between modern ANN models and human cognition—that is, there is no evidence that modern ANN models and human cognition are qualitatively equivalent or similar, even when their outputs correlate highly with human data, especially given the way they are often built and trained—but also because many questions about sentence processing are difficult to answer using modern ANN models alone. For example, it is not clear how modern ANN models can be used to explain why adult L2 learners often do not reach native-like processing abilities, even after a long period of learning, and why certain forms that are rarely encountered in everyday life are processed in a particular way (e.g., the processing of cataphors and parasitic gaps). Also, I am not sure how modern ANN models can be used to understand why certain forms are used more or less frequently or why certain interpretations are favoured or disfavoured during sentence processing (i.e., what factors lead to the use of particular forms or preferred interpretations). To answer such questions, the traditional approaches seem more effective and necessary.

This is not to say that modern ANN models are useless for the study of human sentence processing. These models can be used to obtain information about the likelihood of different forms occurring in a given context. Such probabilistic distributions can then be used to consider possible underlying factors contributing to these probabilities. Nor do I intend to dismiss research that aims to investigate whether theories of sentence processing can be implemented by ANN models in an attempt to understand human cognition, as seen in the classical connectionist framework. Once we understand certain aspects of sentence interpretation processes (e.g., once we understand what factors cause the processor to behave in a certain way in a given context), ANN models can be built based on this understanding to investigate whether they can capture the processor's behaviour (e.g., McRae, Spivey-Knowlton & Tanenhaus, 1998). Such studies can deepen our understanding of the mechanisms of sentence interpretation processes.

Conclusion

In this paper, I provided an overview of formal computational modelling and demonstrated how it can be incorporated into L2 sentence processing research by formalising two verbally formulated hypotheses about L2 sentence processing. I hope that this paper will encourage researchers to use formal modelling and guide them in its application, thereby contributing to advancing our understanding of L2 sentence processing.

Data availability statement. The code used in this paper is available on the OSF website (https://doi.org/10.17605/OSF.IO/9VCX7) and in the R notebook on Colab (https://colab.research.google.com/drive/1zjl1Mc7Sn8d-IfeeB340bErt_fXJ-lJH?usp=sharing).

Funding. This work was supported by the Deutsche Forschungsgemeinschaft (German Research Foundation)—project number 500540359; primary investigator: Hiroki Fujita.

Competing interests. The author has no relevant financial or non-financial interests to disclose.

References

Abney, S. P. (1987). *The English noun phrase in its sentential aspect* [PhD Thesis]. Massachusetts Institute of Technology.

Anderson, J. R. (1974). Retrieval of propositional information from long-term memory. *Cognitive Psychology*, *6*, 451–474. https://doi.org/10.1016/0010-0285(74)90021-8

Anderson, J. R. (2007). *How can the human mind occur in the physical universe?* (pp. ix, 290). Oxford University Press. https://doi.org/10.1093/acprof:oso/9780195324259.001.0001

Bailer-Jones, D. M. (2009). *Scientific models in philosophy of science*. University of Pittsburgh Press. https://doi.org/10.2307/j.ctt5vkdnq

Berwick, R. C., & Stabler, E. P. (Eds.). (2019). *Minimalist parsing*. Oxford University Press. https://doi.org/10.1093/oso/9780198795087.001.0001

Blümel, A., & Holler, A. (2022). DP, NP, or neither? Contours of an unresolved debate. *Glossa: A Journal of General Linguistics*, *7*. https://doi.org/10.16995/glossa.8326

Cheng, Y., Cunnings, I., Miller, D., & Rothman, J. (2022). Double-number marking matters for both l1 and l2 processing of nonlocal agreement similarly: An ERP investigation. *Studies in Second Language Acquisition*, *44*, 1309–1329. https://doi.org/10.1017/S0272263121000772

Chomsky, N. (1956). Three models for the description of language. *IRE Transactions on Information Theory*, *2*, 113–124. https://doi.org/10.1109/TIT.1956.1056813

Chomsky, N. (1981). *Lectures on government and binding: The Pisa lectures*. Foris.

Clahsen, H., & Felser, C. (2006a). Continuity and shallow structures in language processing. *Applied Psycholinguistics*, *27*, 107–126. https://doi.org/10.1017/S0142716406060206

Clahsen, H., & Felser, C. (2006b). Grammatical processing in language learners. *Applied Psycholinguistics*, *27*, 3–42. https://doi.org/10.1017/S0142716406060024

Clahsen, H., & Felser, C. (2018). Some notes on the Shallow Structure Hypothesis. *Studies in Second Language Acquisition*, *40*, 693–706. https://doi.org/10.1017/S0272263117000250

Colombo, M., & Piccinini, G. (2023). *The computational theory of mind*. Cambridge University Press. https://doi.org/10.1017/9781009183734

Crocker, M. W. (1996). *Computational psycholinguistics: An interdisciplinary approach to the study of language* (Vol. *20*). Springer Netherlands.

Crocker, M. W. (1999). Mechanisms for sentence processing. In S. Garrod & M. J. Pickering (Eds.), *Language processing* (pp. 191–232). Psychology Press.

Cunnings, I. (2017a). Interference in native and non-native sentence processing. *Bilingualism: Language and Cognition*, *20*, 712–721. https://doi.org/10.1017/S1366728916001243

Cunnings, I. (2017b). Parsing and working memory in bilingual sentence. *Bilingualism: Language and Cognition*, *20*, 659–678. https://doi.org/10.1017/S1366728916000675

Cunnings, I., & Fujita, H. (2021). Quantifying individual differences in native and nonnative sentence processing. *Applied Psycholinguistics*, *42*, 579–599. https://doi.org/10.1017/S0142716420000648

Cunnings, I., & Fujita, H. (2023). Similarity-based interference and relative clauses in second language processing. *Second Language Research*, *39*, 539–563. https://doi.org/10.1177/02676583211063534

D'Alessandro, R., Franco, I., & Gallego, Á. J. (Eds.). (2017). *The verbal domain*. Oxford University Press. https://doi.org/10.1093/oso/9780198767886.001.0001

Dekydtspotter, L., & Seo, H.-K. (2017). Transitivity in the processing of intransitive clauses: A category-based prediction in low-intermediate learners of English. *Studies in Second Language Acquisition*, *39*, 527–552. https://doi.org/10.1017/S0272263116000188

Dillon, B. (2017). A short discourse on reflexives: A reply to Cunnings (2016). *Bilingualism: Language and Cognition*, *20*, 679–680. https://doi.org/10.1017/S1366728916000973

Dillon, B., Mishler, A., Sloggett, S., & Phillips, C. (2013). Contrasting intrusion profiles for agreement and anaphora: Experimental and modeling evidence. *Journal of Memory and Language*, *69*, 85–103. https://doi.org/10.1016/j.jml.2013.04.003

Dussias, P. E., & Piñar, P. (2010). Effects of reading span and plausibility in the reanalysis of wh-gaps by Chinese-English second language speakers. *Second Language Research*, *26*, 443–472. https://doi.org/10.1177/0267658310373326

Engelmann, F., Jäger, L. A., & Vasishth, S. (2019). The effect of prominence and cue association on retrieval processes: A computational account. *Cognitive Science*, *43*. https://doi.org/10.1111/cogs.12800

Felser, C., Roberts, L., Marinis, T., & Gross, R. (2003). The processing of ambiguous sentences by first and second language learners of English. *Applied Psycholinguistics*, *24*, 453–489. https://doi.org/10.1017/S0142716403000237

Felser, C., Sato, M., & Bertenshaw, N. (2009). The on-line application of binding Principle A in English as a second language. *Bilingualism: Language and Cognition*, *12*, 485–502. https://doi.org/10.1017/S1366728909990228

Fernández Cuenca, S., & Jegerski, J. (2023). A role for verb regularity in the L2 processing of the Spanish subjunctive mood: Evidence from eye-tracking. *Studies in Second Language Acquisition*, *45*, 318–347. https://doi.org/10.1017/S027226312200016X

Fodor, J. A. (1975). *The language of thought*. Harvard University Press. https://books.google.de/books?id=XZwGLBYLbg4C

Fodor, J. D. (1978). Parsing strategies and constraints on transformations. *Linguistic Inquiry*, *9*, 427–473.

Frank, S. L. (2021). Toward computational models of multilingual sentence processing. *Language Learning*, *71*, 193–218. https://doi.org/10.1111/lang.12406

Frazier, L. (1979). *On comprehending sentences: Syntactic parsing strategies* [PhD thesis]. University of Connecticut.

Frenck-Mestre, C., & Pynte, J. (1997). Syntactic ambiguity resolution while reading in second and native languages. *The Quarterly Journal of Experimental Psychology A*, *50*, 119–148. https://doi.org/10.1080/027249897392251

Fujita, H. (2021). On the parsing of garden-path sentences. *Language, Cognition and Neuroscience*, *36*, 1234–1245. https://doi.org/10.1080/23273798.2021.1922727

Fujita, H. (2023). Predictive structure building in language comprehension: A large sample study on incremental licensing and parallelism. *Cognitive Processing*, *24*, 301–311. https://doi.org/10.1007/s10339-023-01130-8

Fujita, H. (2024a). Memory retrieval in online sentence parsing: empirical evidence, computational modelling, and simulations. *Computational Brain & Behavior*, *7*, 457–478. https://doi.org/10.1007/s42113-024-00206-8

Fujita, H. (2024b). Online revision process in clause-boundary garden-path sentences. *Memory & Cognition*, *52*, 73–90. https://doi.org/10.3758/s13421-023-01444-0

Fujita, H. (2025). Active dependency formation, maintenance cost and proactive retrieval interference. *Language, Cognition and Neuroscience*. https://doi.org/10.1080/23273798.2025.2522267

Fujita, H., & Cunnings, I. (2020). Reanalysis and lingering misinterpretation of linguistic dependencies in native and non-native sentence comprehension. *Journal of Memory and Language*, *115*, 104154. https://doi.org/10.1016/j.jml.2020.104154

Fujita, H., & Cunnings, I. (2021a). Lingering misinterpretation in native and nonnative sentence processing: Evidence from structural priming. *Applied Psycholinguistics*, *42*, 475–504. https://doi.org/10.1017/S0142716420000351

Fujita, H., & Cunnings, I. (2021b). Reanalysis processes in non-native sentence comprehension. *Bilingualism: Language and Cognition*, *24*, 628–641. https://doi.org/10.1017/S1366728921000195

Fujita, H., & Cunnings, I. (2022). Interference and filler-gap dependency formation in native and non-native language comprehension. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *48*, 702–716. https://doi.org/10.1037/xlm0001134

Fujita, H., & Cunnings, I. (2023). Interference in quantifier float and subject-verb agreement. *Language, Cognition and Neuroscience*, *38*, 1001–1019. https://doi.org/10.1080/23273798.2023.2189738

Fujita, H., & Cunnings, I. (2024). Subject-verb dependency formation and semantic interference in native and non-native language comprehension. *Applied Psycholinguistics*, *45*, 81–109. https://doi.org/10.1017/S0142716423000498

Fujita, H., & Yoshida, M. (2024). Online reflexive resolution and interference. *Language, Cognition and Neuroscience*, *39*, 513–526. https://doi.org/10.1080/23273798.2024.2329269

Gibson, E. (1991). *A computational theory of human linguistic processing: Memory limitations and processing breakdown* [PhD thesis]. Carnegie Mellon University.

González Alonso, J., Cunnings, I., Fujita, H., Miller, D., & Rothman, J. (2021). Gender attraction in sentence comprehension. *Glossa: A Journal of General Linguistics*, *6*, 20. https://doi.org/10.5334/gjgl.1300

Grune, D., & Jacobs, C. J. H. (2008). *Parsing techniques*. Springer.

Guest, O., & Martin, A. E. (2024). *A metatheory of classical and modern connectionism.* https://doi.org/10.31234/osf.io/eaf2z

Haegeman, L. (1994). *Introduction to government and binding Theory* (2nd ed). Blackwell.

Hale, J. T. (2014). *Automaton theories of human sentence comprehension*. CSLI Publications.

Hopp, H. (2010). Ultimate attainment in L2 inflection: Performance similarities between non-native and native speakers. *Lingua*, *120*, 901–931. https://doi.org/10.1016/j.lingua.2009.06.004

Hopp, H. (2014). Working memory effects in the l2 processing of ambiguous relative clauses. *Language Acquisition*, *21*, 250–278. https://doi.org/10.1080/10489223.2014.892943

Hopp, H. (2015). Individual differences in the second language processing of object–subject ambiguities. *Applied Psycholinguistics*, *36*, 129–173. https://doi.org/10.1017/S0142716413000180

Hopp, H. (2022). Second language sentence processing. *Annual Review of Linguistics*, *8*, 235–256. https://doi.org/10.1146/annurev-linguistics-030821-054113

Hornstein, N., Nunes, J., & Grohmann, K. K. (2005). *Understanding minimalism*. Cambridge University Press. https://doi.org/10.1017/CBO9780511840678

Jacob, G., Lago, S., & Patterson, C. (2017). L2 processing and memory retrieval: Some empirical and conceptual challenges. *Bilingualism: Language and Cognition*, *20*, 691–693. https://doi.org/10.1017/S1366728916000948

Jäger, L. A., Mertzen, D., Van Dyke, J. A., & Vasishth, S. (2020). Interference patterns in subject-verb agreement and reflexives revisited: A large-sample study. *Journal of Memory and Language*, *111*, 104063. https://doi.org/10.1016/j.jml.2019.104063

Johnson-Laird, P. N. (1983). *Mental models*. Cambridge University Press.

Juffs, A., & Harrington, M. (1995). Parsing effects in second language sentence processing: Subject and object asymmetries in wh-extraction. *Studies in Second Language Acquisition*, *17*, 483–516. https://doi.org/10.1017/S027226310001442X

Kaan, E., Kirkham, J., & Wijnen, F. (2016). Prediction and integration in native and second-language processing of elliptical structures. *Bilingualism: Language and Cognition*, *19*, 1–18. https://doi.org/10.1017/S1366728914000844

Kim, H., & Grüter, T. (2021). Predictive processing of implicit causality in a second language: A visual-world eye-tracking study. *Studies in Second Language Acquisition*, *43*, 133–154. https://doi.org/10.1017/S0272263120000443

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

Kornai, A., & Pullum, G. K. (1990). The X-bar theory of phrase structure. *Language*, *66*, 24–50. https://doi.org/10.2307/415278

Kush, D. (2013). *Respecting relations: Memory access and antecedent retrieval in incremental sentence processing* [PhD thesis]. University of Maryland.

Lago, S., & Felser, C. (2018). Agreement attraction in native and nonnative speakers of German. *Applied Psycholinguistics*, *39*, 619–647. https://doi.org/10.1017/S0142716417000601

Lewis, R. L., & Vasishth, S. (2005). An activation-based model of sentence processing as skilled memory retrieval. *Cognitive Science*, *29*, 375–419. https://doi.org/10.1207/s15516709cog0000_25

Marinis, T., Roberts, L., Felser, C., & Clahsen, H. (2005). Gaps in second language sentence processing. *Studies in Second Language Acquisition*, *27*, 53–78. https://doi.org/10.1017/S0272263105050035

McClelland, J. L. (2009). The place of modeling in cognitive science. *Topics in Cognitive Science*, *1*, 11–38. https://doi.org/10.1111/j.1756-8765.2008.01003.x

McElree, B. (2000). Sentence comprehension is mediated by content-addressable memory structures. *Journal of Psycholinguistic Research*, *29*, 111–123. https://doi.org/10.1023/A:1005184709695

McRae, K., Spivey-Knowlton, M., & Tanenhaus, M. K. (1998). Modeling the influence of thematic fit (and other constraints) in on-line sentence comprehension. *Journal of Memory and Language*, *38*, 283–312. https://doi.org/10.1006/jmla.1997.2543

Omaki, A., & Schulz, B. (2011). Filler-gap dependencies and island constraints in second-language sentence processing. *Studies in Second Language Acquisition*, *33*, 563–588. https://doi.org/10.1017/S0272263111000313

Parker, D. (2019). Cue combinatorics in memory retrieval for anaphora. *Cognitive Science*, *43*, e12715. https://doi.org/10.1111/cogs.12715

Pavlik Jr., P. I., & Anderson, J. R. (2005). Practice and forgetting effects on vocabulary memory: An activation-based model of the spacing effect. *Cognitive Science*, *29*, 559–586. https://doi.org/10.1207/s15516709cog0000_14

Phillips, C., & Gibson, E. (1997). On the strength of the local attachment preference. *Journal of Psychalinguistic Research*, *26*, 323–346. https://doi.org/10.1023/A:1025028725042

Rah, A., & Adone, D. (2010). Processing of the reduced relative clause versus main verb ambiguity in l2 learners at different proficiency levels. *Studies in Second Language Acquisition*, *32*, 79–109. https://doi.org/10.1017/S027226310999026X

Reinhart, T. (1981). Pragmatics and linguistics: An analysis of sentence topics. *Philosophica*, *27*. https://doi.org/10.21825/philosophica.82606

Roberts, L., & Felser, C. (2011). Plausibility and recovery from garden paths in second language sentence processing. *Applied Psycholinguistics*, *32*, 299–331. https://doi.org/10.1017/S0142716410000421

Salzmann, M. (2019). A new version of the Matching Analysis of relative clauses: Combining deletion under recoverability with vehicle change In M. Krifka & M. Schenner (Eds.), *Reconstruction effects in relative clauses* (pp. 187–224). De Gruyter. https://doi.org/10.1515/9783050095158-006

Stowell, T. (1981). *Origins of phrase structure* [PhD thesis]. Massachusetts Institute of Technology.

van Riemsdijk, H. C. & Williams, E. (1986). *Introduction to the theory of grammar*. MIT Press.

Vasishth, S., & Engelmann, F. (2021). *Sentence comprehension as a cognitive process: A computational approach*. Cambridge University Press. https://doi.org/10.1017/9781316459560

Wagers, M., Lau, E. F., & Phillips, C. (2009). Agreement attraction in comprehension: Representations and processes. *Journal of Memory and Language*, *61*, 206–237. https://doi.org/10.1016/j.jml.2009.04.002

Wickelgren, W. A. (1974). Single-trace fragility theory of memory dynamics. *Memory & Cognition*, *2*, 775–780. https://doi.org/10.3758/BF03198154

Cite this article: Fujita, H. (2025). Formal computational modelling in second language sentence processing research. *Studies in Second Language Acquisition,* 1–42. https://doi.org/10.1017/S0272263125101009