# An extensible framework for mechanistic processing models: From representational linguistic theories to quantitative model comparison

**Adrian Brasoveanu (abrsvn@gmail.com)**
Department of Linguistics, 1156 High Street
Santa Cruz, CA 95065, USA

**Jakub Dotlačil (j.dotlacil@gmail.com)**
ILLC, Science Park 107
1098 XG Amsterdam, The Netherlands

## Abstract

We introduce a Python3 reimplementation of ACT-R (Anderson and Lebiere 1998, Anderson 2007) in which we build an end-to-end simulation of syntactic parsing in a typical self-paced reading experiment. The model uses an eager left-corner parsing strategy implemented as a skill in procedural memory (following Lewis and Vasishth 2005), makes use of independently motivated components of the ACT-R framework (procedural memory, content-addressable declarative memory – cf. Wagers et al. 2009), and explicitly models the motor and visual processes involved in self-paced reading. The ACT-R model can be embedded in a Bayesian statistical model to estimate its subsymbolic parameters and perform model comparison.

**Keywords:** ACT-R, Bayesian models, incremental processing, syntax, semantics, self-paced reading

## Introduction: framework & case study

The overarching goal of the research we report on here is to build an extensible framework in which formally and computationally explicit processing models for natural language syntax and semantics can be formulated. Specifically, we want to build cognitively realistic models for incremental parsing of discourse representations structures (DRSs, Kamp 1981; Kamp and Reyle 1993) or similar representations. In the models we built so far, the semantic and syntactic representations are created mostly in parallel, so we will be able to focus only on modeling *syntactic* representations in this paper.

This extensible framework enables us to formulate *mechanistic* models of natural language processing, which is the preferred level of explanation in cognitive science (see Lewandowsky and Farrell 2010 among many others). When building the framework, our strategy was to use an independently motivated, general cognitive architecture, and to embed processing models for natural language in this architecture. Since parsing is easy to embed in hybrid (symbolic and subsymbolic) cognitive architectures, we chose to focus on them. Two hybrid architectures are in common use in psycholinguistics, namely Soar (Hale 2014; Young and Lewis 1999) and ACT-R (Dillon et al. 2013; Engelmann et al. 2013; Kush 2013; Lewis and Vasishth 2005; Nicenboim and Vasishth 2018; Rij 2012; Taatgen and Anderson 2002; Vasishth et al. 2008). ACT-R is the more popular architecture, so it was a natural choice for our framework.

Currently, psycholinguistic ACT-R models are mainly used to model recall of syntactic structures (Dillon et al. 2013; Engelmann et al. 2013; Lewis and Vasishth 2005; Nicenboim and Vasishth 2018; Vasishth et al. 2008). This focus on recall-related modeling does not take advantage of the generality of ACT-R as a cognitive architecture and its "no magic" policy. If we want to make explicit all the various parsing components and actions involved in processing models for specific natural language phenomena, we have to rely on the full implementation of ACT-R in LISP, which is not a very popular programming choice now. Furthermore, since LISP is a relatively isolated programming language, it does not have a thriving ecosystem of statistical estimation / machine learning libraries that could be leveraged in processing models. Being a cognitive architecture, ACT-R comes with many parameters, but because of the relative paucity of the LISP library ecosystem, these parameters are set to their default values or manually changed even when LISP ACT-R is used. Manually changing parameters makes modeling hard to replicate and systematic quantitative model comparison hard to perform.

In this paper, we introduce a new Python3 implementation of ACT-R (**pyactr**, Brasoveanu and Dotlačil in prep.), which makes two main contributions. On one hand, it is easy to combine ACT-R modeling and Bayesian estimation methods: ACT-R models are embedded in Bayesian models, which makes it possible for us to systematically explore parameter values and quantify our uncertainty about them, perform quantitative model comparison, and replicate / build on previous modeling work. On the other hand, the ACT-R component comes with a working, extensible parsing framework for syntax and semantics, which includes visual and motor interfaces and a variety of models for both syntactic and semantic phenomena. The framework has a modular structure: alternative models for peripherals (visual, motor) and other components can in principle be swapped in, and the resulting models can be systematically compared.

We showcase the framework by modeling Experiment 1 in Grodner and Gibson, (2005) (also used in Lewis and Vasishth 2005). This is a self-paced reading experiment (non-cumulative moving-window; Just et al. 1982). Participants read word-by-word sentences in which the subject noun phrase (NP) is modified by a subject or object extracted relative clause (RC). A subject-gap example is provided in (1), and an object-gap in (2).

(1) The reporter who GAP sent the photographer to the editor hoped for a story.

(2) The reporter who the photographer sent GAP to the editor hoped for a story.

There are 9 regions of interest (ROIs) that we will model, underlined in the examples above. These are word 2 (the matrix noun in subject position) through word 10 (the matrix verb). The ACT-R model goes through a series of cognitive (parsing) steps and decides to press the space bar to reveal the next word at certain times during this process.

The remainder of this paper is dedicated to unpacking this ACT-R model (**ACT-R and eager left-corner parsing**) and quantitatively comparing three variations on it that differ in several theoretically-relevant ways (**Modeling results**). We conclude with a summary and future research directions.

## ACT-R and eager left-corner parsing

We unpack the ACT-R model of self-paced reading at three different levels of detail. We start with a broad overview of the ACT-R architectural components we need. We then outline how various parts of an eager left-corner parser are distributed over the ACT-R components. Finally, we discuss how the model functions on a per-word basis; that is, we outline the cognitive steps the model goes through starting immediately after a word is revealed on the virtual screen and ending with the point at which the model decides to press the space bar to reveal the next word.

There are two types of memory in ACT-R: (i) declarative memory (roughly, 'knowing that') – knowledge of facts, which are represented as chunks (attribute-value matrices), e.g., the lexical chunk for the word *car* in (3); and (ii) procedural memory (roughly, 'knowing how') – the set of productions that fire in series to generate cognitive behavior / processes. These productions have the form of rewrite rules in formal grammars (e.g., context free / phrase structure grammars), but in ACT-R, they are conditionalized cognitive actions: the ACT-R mind fires a production, i.e., takes the action encoded in it, if the current cognitive state satisfies the preconditions of that production.

(3)

| | |
|---|---|
| ISA: | word |
| FORM: | car |
| MEANING: | ⟦car⟧ |
| CATEGORY: | noun |
| NUMBER: | sg |

(4) Goal>

| | |
|---|---|
| TASK: | reading |
| FORM: | car |

$\Rightarrow$

Goal>

| | |
|---|---|
| TASK: | retrieving category |

Retrieval>

| | |
|---|---|
| ISA: | word |
| FORM: | car |

An example production is provided in (4):

- if the current cognitive state is such that the goal buffer (which drives cognitive processes in ACT-R) encodes a TASK of 'reading' the FORM 'car',
- then (⇒) we take a cognitive action that takes us to a new cognitive state,
- where the TASK is to retrieve the syntactic category of that form, and in which we simultaneously place a request in the Retrieval buffer to search declarative memory for a chunk of type 'word' that has the FORM 'car'.

Implicit in this example production is that an ACT-R mind is composed of modules, which include declarative and procedural memory, but also visual and motor modules etc. Modules are not directly accessible: they can only be accessed through associated buffers (e.g., the Retrieval buffer is associated with declarative memory). Buffers serve a dual purpose: individually, they provide the input/output interface to specific modules; as a whole, however, buffers represent the current cognitive state of the mind. Crucially, productions fire based on the current cognitive state, i.e., conditioned on the contents of various buffers. The ACT-R architecture constrains cognitive behavior in various ways, two of which are that buffers can hold only one chunk, and only one production can fire at any given time.

Let us now move on to how we can implement an eager left-corner parser in ACT-R (building on Lewis and Vasishth 2005; Resnik 1992; see also Hale 2014 for an introduction). We distribute parser components over ACT-R modules and buffers as follows. Lexical knowledge is encoded in declarative memory, knowledge of grammar and parsing actions are encoded in procedural memory, expectations about upcoming syntactic categories (which guide parsing) are encoded in the goal buffer, information about the current partially-built syntactic parse is encoded in the imaginal buffer (a secondary goal-like buffer), visual information from the environment is transferred via the visual buffer, and, finally, key press commands are issued via the manual buffer. The visual module we implement is EMMA (Salvucci 2001), and the motor module is EPIC (Kieras and Meyer 1996; Meyer and Kieras 1997). Other choices are also possible.

Running this eager left-corner parser on a simple input sentence will shed more light on its inner workings and how they are deployed in real time. Assume we have a simple grammar with three phrase structure rules S → NP VP, NP → Det N, and VP → V. Also, assume that we are reading the sentence *A boy sleeps* in a self-paced reading task.

We start with a screen in which all words are covered with dashes: `- --- ------`. Our goal stack (stored in the goal buffer) consists of just S: our goal is to parse a sentence. After the first space-bar press, the first word is revealed: `A --- ------`, its visual form is transferred via the visual buffer, and its syntactic category Det (determiner) is retrieved from declarative memory. At that point, we take a series of cognitive steps – that is, we fire a series of productions – that take us to a new state. The goal stack in this new state is N NP S: we now have two subgoals of finding an N (noun) and an NP (noun phrase) on the way to finding an S. Also, we build a partial syntactic structure of the form shown in Figure 1 and store it in the imaginal buffer. We see here the left-corner nature of our parser: we trigger all the syntactic rules that have the determiner *a* or a node dominating *a* as their left branch.

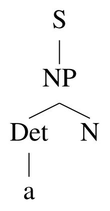

Figure 1: Partial tree after reading the determiner *a*

After another space-bar press, the noun is revealed (`- boy ------`), its form is transferred via the visual buffer and its syntactic category N is retrieved from declarative memory. At this point, we trigger a series of productions that discharge all the N, NP and S goals (this reflects the eager nature of the parser) and replaces them with the single goal of finding a VP (verb phrase). At the same time, a richer partial tree, shown in Figure 2, is stored in the imaginal buffer.[1]

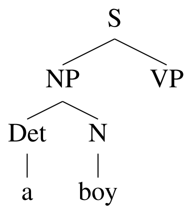

Figure 2: Partial tree after reading the noun *boy*

Finally, the verb is revealed after one more space-bar press: `- --- sleeps`. Its visual form is transferred via the visual buffer and its syntactic category V is retrieved from declarative memory. At that point, the VP goal is satisfied, resulting in an empty goal stack $\varnothing$, and the final structure in Figure 3 is built and encoded in the imaginal buffer.

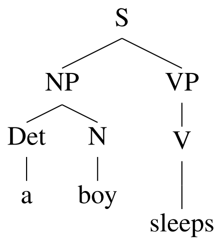

Figure 3: Partial tree after reading the noun *boy*

We have now examined our model at two levels of detail: the general cognitive architecture (ACT-R) and the way the eager left-corner parser is distributed over this architecture. We zoom in one more time to reach the final level of detail at which we want to examine the parser, and describe the series of cognitive steps it takes beginning immediately after seeing a word and ending with the decision to press the space bar to reveal the next word. This sequence of steps is summarized in the flowchart in Figure 4 below.

[1]Strictly speaking, only parts of the tree in Figure 2 are stored in the imaginal buffer at any given time: in the broader spirit of ACT-R, syntactic chunks encode only one level of embedding in the tree, e.g., $[^{NP}$Det N] or $[^{S}$NP VP], but not both.

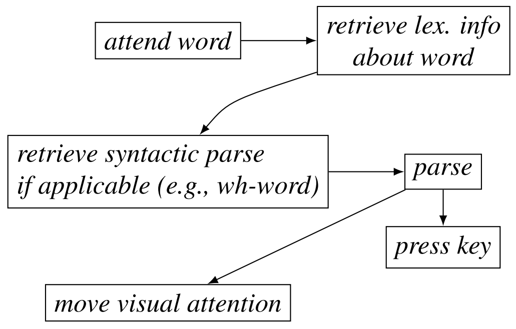
attend word
retrieve lex. info about word
retrieve syntactic parse if applicable (e.g., wh-word)
parse
press key
move visual attention

Figure 4: Flowchart of parsing process per word

As this flowchart indicates, we first attend to the visual form of the word, then retrieve lexical information about the word, and also a syntactic structure if applicable (e.g., we retrieve the wh-word at GAP sites). We then proceed with all the parsing actions we can take given our eager left-corner parser. When these parsing actions are complete – *and only then* – we proceed in parallel to moving visual attention and issuing the key-press motor command.

## Modeling results

We estimate four different parameters associated with the ACT-R parsing model outlined in the previous section. We could in principle estimate more, but we confine ourselves to these four parameters for simplicity.

The first one is the angle parameter $k$ that modulates visual encoding; the time of visual encoding $T_{enc}$ is given by the function $K \cdot D \cdot e^{k \cdot d}$, where $k$ is the angle parameter we estimate, $d$ is visual distance, $D$ specifies relevant visual object properties (in our case, word length), and $K$ is set to its default value of 0.01. We estimate this parameter mostly to show that parameters for peripheral modules can be estimated at the same time as the more commonly estimated parameters associated with declarative and procedural memory.

The second one is the time $r$ it takes to fire a production rule (a condition-action pair). This is necessary because our processing models incorporate linguistic theories in a fairly transparent way, which makes it necessary to fire more rules per word / region of interest (ROI) than it would be possible with the 50 ms default. It might be that a judicious use of production compilation will increase rule-firing time closer to its ACT-R default, but this is a topic for future research. Apart from the need to estimate rule-firing time in such 'theoretically-transparent' linguistic applications, the ACT-R + Bayes framework we introduce here enables us to quantify our uncertainty about rule-firing times in *any* ACT-R model; as an anonymous reviewer points out, a good understanding of the uncertainty associated with the $r$ parameter is relevant to ongoing discussions about the need to possibly add noise to it, and will benefit ACT-R models across the board.

The third and fourth parameters are the latency factor $F$ and the latency exponent $f$ that modulate the latency of retrieval from declarative memory. Retrieval latency $T$ is a function of activation $A$, specifically, $F \cdot e^{-f \cdot A}$.[2] The latency factor $F$ is commonly estimated, but the latency exponent $f$ is usually set to its default value of 1. We estimate both of them here because the latency exponent has proved crucial in estimating latencies in lexical decision tasks like the ones in Murray and Forster, (2004) – see Chapter 7 in Brasoveanu and Dotlačil, (in prep.) for a detailed discussion. Given that lexical retrieval is a necessary component of any cognitively-realistic parsing model, we estimate both parameters.

The model is fit to data by estimating the posterior distributions of these four free parameters $k$, $r$, $F$ and $f$. Standardly, modelers rely on default values or manually changing the values, but this process is subjective and time consuming (e.g., a grid-based search over only 20 parameter values for just these 4 parameters would require manually evaluating $20^4 = 160000$ combinations). In contrast, **pyactr** enables us to easily interface ACT-R models with standard statistical estimation methods implemented in widely-used Python3 libraries. Specifically, we use ACT-R models as the likelihood component of full Bayesian models (implemented in **pymc3**). We are therefore able to take advantage of much more efficient search methods in multi-dimensional parameter spaces, specifically, Markov Chain Monte Carlo (MCMC) methods, when we fit the ACT-R parameters to experimental data.

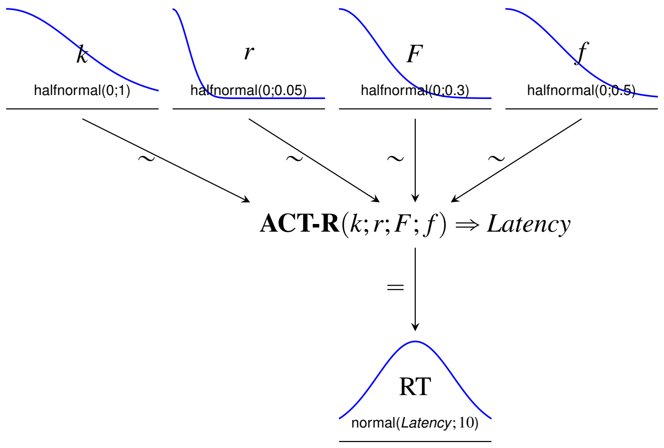

Figure 5: The structure of the Bayesian model

The structure of the Bayesian model is provided in Figure 5: vague / low-information priors for the parameters are listed at the top, the entire ACT-R model provides the likelihood function, which outputs latencies (times between successive key presses) that can be matched against the reading times (RTs) observed in Grodner and Gibson, (2005, Exp. 1). Bayesian methods have many advantages, including the fact that we obtain full posterior distributions for the parameters of interest. We are therefore able to find good parameter values for hybrid (symbolic & subsymbolic) models, and also to quantify our uncertainty about these values. Posterior estimates: $k$ – mean=0.87, sd=0.32, 95% HPD $[0, 1.23]$; $r$ – mean=0.02, sd=0.006, 95% HPD $[0.01, 0.03]$; $F$ – mean=0.01, sd=0.03, 95% HPD $[0, 0.1]$; $f$ – mean=0.23, sd=0.47, 95% HPD $[0, 1.34]$.

The fit of the model is most easily evaluated by examining its posterior predictions for the 9 ROIs, plotted in Figure 6. The diamonds indicate the observed mean RTs for each word, the segments provide the 95% CRIs (credible intervals) for the mean

[2] Base activation $A$ is a function of time periods $t_k$ since previous word usages $k$ from 1 to $n$, where $n$ is determined by the frequency of the word: $A = \log\left(\sum_{k=1}^{n} t_k^{-0.5}\right)$. For reasons of space, we do not discuss spreading activation.

RT predicted by our model, and the dots are the predicted mean RTs. When evaluating the model, recall that the parameters are estimated once for a full run through the experiment – they are not estimated ROI by ROI (falsely assuming independence between ROIs), as it is usually done in the psycholinguistic literature. That is, we model here in one go the full, hybrid (symbolic and subsymbolic) stochastic process of incremental parsing in a self-paced reading task.

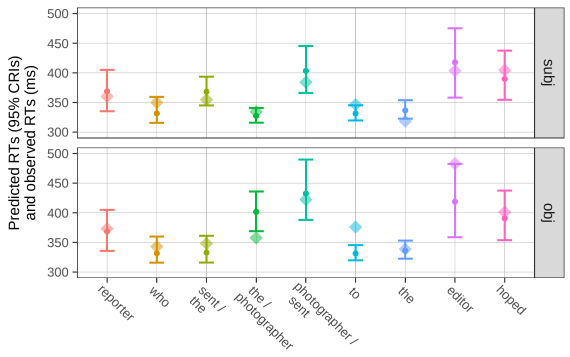

Figure 6: Model 1: postulated subject gaps

Figure 6 shows that wh-gap retrieval is modeled well: this is the 3rd word (*sent*) in the top panel (**subj**-gaps) and the 5th word (also *sent*) in the bottom panel (**obj**-gaps). But the spillover effect on the word after the object gap – the 6th word (*to*) in the bottom panel – is not captured; we return to this.

Finally, we see that the wh-word and the following word (the 2nd and 3rd words in both panels) are modeled well for both subject and object gaps. This is particularly interesting because the model is formulated so that it predictively postulates a subject gap when parsing the wh-word. This postulated gap has to then be reanalyzed for object gaps.

We therefore consider a second model that does not postulate a subject gap as soon as the wh-word is parsed. The posterior predictions of this model, provided in Figure 7, are clearly worse: the 95% CRIs are completely below the observed mean RTs for the wh-word in both conditions, and also for the word immediately following the wh-word in the object-gap condition. This indicates that the model underestimates the parsing work triggered by the wh-word, and it also underestimates the reanalysis work that needs to be done on the word immediately following the wh-word in the object-gap condition.

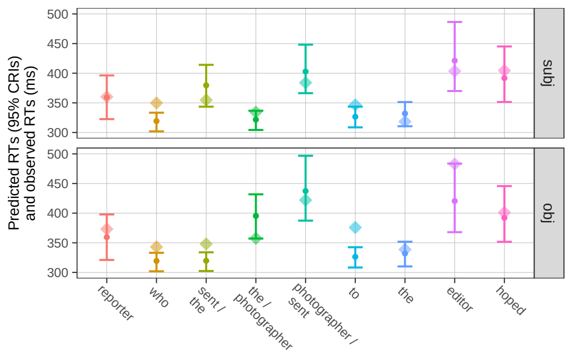

Figure 7: Model 2: no postulated subject gaps

Thus, our first model (postulated subject gaps; Figure 6) is the better one. This model comparison shows that our ACT-R + Bayes framework has empirical bite; not everything goes. Furthermore, the framework can be used both for formalizing (symbolic) processing hypotheses and for quantitative hypothesis testing.

A final, third model we consider aims to capture the spillover effect on the word following the retrieval of object gaps (the 6th word in the bottom panel of Figure 6 above). In this model, parsing actions proceed in parallel to moving visual attention and issuing key press commands, unlike the flowchart in Figure 4 where we see that parsing has to always precede visual and motor actions. This is sufficient to capture the spillover effect for object gaps, and also increases the precision of our model (smaller CRIs), as shown in Figure 8 below.

This increase in precision for Model 3 is clearly visible in its much lower $\text{WAIC}_2$ value:[3] Model 1

[3]Watanabe-Akaike/Widely Available Information Criterion; see Gelman et al., (2013, pp. 173-174) a.o. for discussion of both $\text{WAIC}_1$ and $\text{WAIC}_2$. We compute both

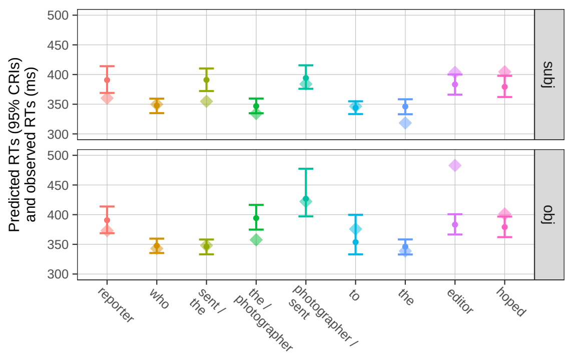

Figure 8: Model 3: 'parallel' reader

(postulated subject gaps) – $WAIC_1$ = 387.8, $WAIC_2$ = 1469; Model 2 (no postulated subject gaps) – $WAIC_1$ = 433.1, $WAIC_2$ = 1613; Model 3 ('parallel' reader) – $WAIC_1$ = 389.8, $WAIC_2$ = 553.4. These WAIC values provide a good summary of the conclusions we drew based on the posterior-prediction plots in Figures 6, 7 and 8: Model 1 is better than Model 2 with respect to both $WAIC_1$ and $WAIC_2$, and Model 3 provides sharper posterior predictions than either Model 1 or Model 2, as its $WAIC_2$ value shows (recall that $WAIC_2$ is variance based).

As a final way to evaluate our processing models, we can compare observed RTs and model predictions for individual words (in individual items) rather than focusing only on mean RTs, as we have done above. Let's focus only on the predictions made by Model 1; a linear regression with observed RTs for individual words as the response variable and predicted word RTs as the sole predictor estimates a slope of 1 (*SE*=0.009, *t*=5.7). That is, a 1 ms increase in predicted RTs corresponds to a 1 ms increase in observed RTs, indicating a very good data fit for our models at individual word level.

Finally, we used Model 1 to predict both eye-tracking (ET) and self-paced reading (SPR) data from Frank et al., (2013) (a variety of syntactic structures, no RCs; we selected 22 sentences that the parser, with its limited set of syntactic rules, parses correctly). For eye-tracking, we simply remove the key-press motor component from the model. Once again, we run a linear regression with observed and predicted word-level RTs as the response and predictor variables, respectively. The model fits both kinds of data fairly well: SPR – 1ms increase in predicted RT corresponds to 0.79ms increase in observed RT (*t*=2.1); ET – 1ms increase in predicted RT corresponds to 0.82ms increase in observed RT (*t*=3.31). The relative decrease in fit (the slope is not 1 anymore) is due to the fact that the model was really tailored to the RC data in Grodner and Gibson, (2005, Exp. 1).

WAIC values based on the estimated posterior `pyactr` RTs for the 18 ROIs (9 subject-gap ROIs + 9 object-gap ROIs).

## Conclusion

We introduced an extensible framework for mechanistic processing models and investigated 3 models incorporating an eager left-corner parser with visual and motor interfaces. The models differed in various theoretically-relevant respects, and the framework was used to quantitatively compare these different models / theoretical hypotheses.

We have only done quantitative comparisons based on posterior-prediction plots and WAIC values, but systematic across-the-board model comparison via Bayes factors is possible in the framework, as well as modeling a variety of other tasks (eye tracking, lexical decision).

## Acknowledgments

We are grateful to Amanda Rysling, Donka Farkas, Abel Rodriguez, Matt Wagers, the UCSC S-lab audience (March 2018) and two ICCM 2018 anonymous reviewers for comments and discussion.

We want to thank Ted Gibson and Dan Grodner for providing the items and full datasets for the two experiments reported in their paper (Grodner and Gibson 2005). Jakub Dotlačil was supported by the NWO VENI 275-80-005 grant. The usual disclaimers apply.

## References

Anderson, John R. (2007). *How can the human mind occur in the physical universe?* Oxford University Press.

Anderson, John R. and Christian Lebiere (1998). *The Atomic Components of Thought*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Brasoveanu, Adrian and Jakub Dotlačil (in prep.). *Formal Linguistics and Cognitive Architecture*. Language, Cognition, and Mind (LCAM) Series. The current version of the *pyactr* library (Python3 ACT-R) is available here: https://github.com/jakdot/pyactr; the (frozen) version used in the book, together with the code for the models in the book, is available here: https://github.com/abrsvn/pyactr-book. Dordrecht: Springer.

Dillon, Brian et al. (2013). "Contrasting intrusion profiles for agreement and anaphora: Experimental and modeling evidence". In: *Journal of Memory and Language* 69.2, pp. 85–103.

Engelmann, Felix et al. (2013). "A Framework for Modeling the Interaction of Syntactic Processing and Eye Movement Control". In: *Topics in Cognitive Science* 5.3, pp. 452–474. DOI: 10.1111/tops.12026.

Frank, Stefan L et al. (2013). "Reading time data for evaluating broad-coverage models of English sentence processing". In: *Behavior Research Methods* 45.4, pp. 1182–1190.

Gelman, A. et al. (2013). *Bayesian Data Analysis, Third Edition*. Chapman & Hall/CRC Texts in Statistical Science. Taylor & Francis. ISBN: 9781439840955.

Grodner, Daniel and Edward Gibson (2005). "Consequences of the Serial Nature of Linguistic Input for Sentenial Complexity". In: *Cognitive Science* 29, pp. 261–291.

Hale, John T. (2014). *Automaton Theories of Human Sentence Comprehension*. Stanford: CSLI Publications.

Just, Marcel A. et al. (1982). "Paradigms and processes in reading comprehension". In: *Journal of Experimental Psychology: General* 111.2, pp. 228–238. DOI: 10.1037/0096-3445.111.2.228.

Kamp, Hans (1981). "A Theory of Truth and Semantic Representation". In: *Formal Methods in the Study of Language*. Ed. by Jeroen Groenendijk et al. Amsterdam: Mathematical Centre Tracts, pp. 277–322.

Kamp, Hans and Uwe Reyle (1993). *From Discourse to Logic. Introduction to Model theoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*. Dordrecht: Kluwer.

Kieras, David E and David E Meyer (1996). "The EPIC architecture: Principles of operation". Unpublished manuscript from ftp://ftp. eecs. umich. edu/people/kieras/EPICarch. ps.

Kush, Dave W (2013). "Respecting relations: Memory access and antecedent retrieval in incremental sentence processing". PhD thesis. University of Maryland, College Park.

Lewandowsky, S. and S. Farrell (2010). *Computational Modeling in Cognition: Principles and Practice*. Thousand Oaks, CA, USA: SAGE Publications.

Lewis, Richard and Shravan Vasishth (2005). "An activation-based model of sentence processing as skilled memory retrieval". In: *Cognitive Science* 29, pp. 1–45.

Meyer, David E and David E Kieras (1997). "A computational theory of executive cognitive processes and multiple-task performance: Part I. Basic mechanisms." In: *Psychological review* 104.1, p. 3.

Murray, Wayne S and Kenneth I Forster (2004). "Serial mechanisms in lexical access: the rank hypothesis." In: *Psychological Review* 111.3, p. 721.

Nicenboim, Bruno and Shravan Vasishth (2018). "Models of retrieval in sentence comprehension: A computational evaluation using Bayesian hierarchical modeling". In: *Journal of Memory and Language* 99, pp. 1 –34. DOI: https://doi.org/10.1016/j.jml.2017.08.004.

Resnik, Philip (1992). "Left-corner parsing and psychological plausibility". In: *Proceedings of the Fourteenth International Conference on Computational Linguistics*. Nantes, France.

Rij, Jacolien van (2012). *Pronoun processing: Computational, behavioral, and psychophysiological studies in children and adults*. Groningen.

Salvucci, Dario D (2001). "An integrated model of eye movements and visual encoding". In: *Cognitive Systems Research* 1.4, pp. 201–220.

Taatgen, Niels A and John R Anderson (2002). "Why do children learn to say "broke"? A model of learning the past tense without feedback". In: *Cognition* 86.2, pp. 123–155.

Vasishth, Shravan et al. (2008). "Processing Polarity: How the Ungrammatical Intrudes on the Grammatical". In: *Cognitive Science* 32, pp. 685–712.

Wagers, Matthew W et al. (2009). "Agreement attraction in comprehension: Representations and processes". In: *Journal of Memory and Language* 61.2, pp. 206–237.

Young, Richard M. and Richard L. Lewis (1999). "The Soar cognitive architecture and human working memory". In: ed. by Akira Miyake and Priti Shah, pp. 224–256.