Mathematical Linguistics and Cognitive Complexity

Aniello De Santo
Dept. of Linguistics
University of Utah
aniello.desanto@utah.edu

Jonathan Rawski
Dept. of Linguistics & Language Development
San Jose State University
jon.rawski@sjsu.edu

Abstract. The complexity of linguistic patterns has been object of extensive debate in research programs focused on probing the inherent structure of human language abilities. But in what sense is a linguistic phenomenon more complex than another, and what can complexity tell us about the connection between linguistic typology and human cognition? This chapter approaches these questions by presenting a broad and informal introduction to the vast literature on formal language theory, computational learning theory, and artificial grammar learning. In doing so, it hopes to provide readers with an understanding of the relevance of mathematically grounded approaches to cognitive investigations into linguistic complexity, and thus further fruitful collaborations between cognitive scientists and mathematically inclined linguist and psychologist.

Keywords. Formal Language Theory, Complexity, Simplicty, Learnability, Artificial Language Learning

1 Introduction

Questions about the complexity of linguistic patterns have always played a central role in research programs focused on probing the inherent structure of human language abilities. In fact, even a quick glance at recent research in (psycho)linguistics will reveal an abundance of references to concepts like *complexity*, *simplicity*, *economy*, often associated with *computational* processes within the language system. But in what sense is a linguistic phenomenon more complex than another? Most times, these appeals to complexity characterizations lack precise, unambiguous definitions, and are more often than not expressed in the form of general verbal theories. This reliance on intuitive notions over proper formalization is not just a problem in linguistics and psycholinguistics, but it seems to be widespread in the psychological sciences at large. In contrast, this chapter aims to highlight the plethora of insights that come from grounding theories of linguistic cognition on the firm foundation provided by rigorous mathematical formalization.

Mathematical characterizations allow researchers to spell-out the assumptions (e.g., about available resources, mechanisms, and representations) and implications of their theories, *before* testing them in experimental settings or computer simulations (Kaplan, 2011; Martin and Baggio, 2019; Guest and Martin, 2021). A focus on transparency and lack of ambiguity is fundamental in the study of complex and often opaque cognitive abilities, that can only be observed indirectly through behavior. Importantly thought, there are risks in computational models themselves. Kaplan (1995) for instance, warns against the “compelling temptations or seductions” one risks to fall into when applying computational methodologies to cognitive questions without explicitly laying out an underlying theoretical stance. That is, it is possible to adopt computational models and focus on their ability to exploit observations about the data to make new (sometimes correct) predictions about expected behavior.

However, if these models are themselves opaque and ill-understood — i.e. employed without an understanding of what *causal* relations connect the way these models generalize from observed data to new predictions — there is little they actually contribute to the cognitive study of language (Newell, 1973; Kaplan, 1995; Martin and Baggio, 2019). What seems to be desirable is to leverage well-understood mathematical approaches, which can make casual predictions about what is expected of one’s theory, by specifying the relation between the assumptions of the theory and the observed data in an *explanatory* fashion (Boone and Piccinini, 2016; Levy and Bechtel, 2013; Guest and Martin, 2021).

The present chapter explores the contributions of the mathematical *theory of computation* in developing questions of this type. First, the chapter discusses the role computability theory played in the radical transformation underwent by the cognitive sciences, psychology, philosophy, and linguistics at the beginning of the twentieth century (Miller, 2003; Núñez et al., 2019). In line with this, the complexity of linguistic patterns is then approached from the perspective of *Formal Language Theory*. The abstractness and rigor of this framework allows for explicit hypotheses on the relation between linguistic phenomena (typological and experimental) and the cognitive mechanisms underlying human linguistic abilities. Formal Language Theory has a rich history with respect to its interactions with formal linguistics, and it has been leveraged in a wide variety of experimental studies probing human linguistic abilities — within and across linguistic domains, as well as for comparative studies of animal communication systems (Reber, 1967; Rogers and Pullum, 2011; Hauser et al., 2002; Fitch and Hauser, 2004; Rogers and Hauser, 2009; Fitch et al., 2012; Udden et al., 2009; Levelt, 2020; Uddén et al., 2020; Wilson et al., 2020, a.o.). Importantly, the reader will not find an in-depth review of the variety of results sparkled by this line of research across multiple fields (psycholinguistics, evolutionary linguistics, neurolinguistics, animal cognition, among many). Instead, the goal of the chapter is to clarify the way Formal Language Theory identifies necessary components of the patterns under analysis, and how such characterizations of pattern complexity are then reflected in predictions about the hypothesis space of learning mechanisms. A detailed discussion of the relation between formal languages, grammars, and the ties between typology and learnability is thus followed by a formal characterization of the *learning* problem. With the formal apparatus in place, the chapter concludes with an overview of the ways these characterizations can be used to test precise experimental predictions in an Artificial Grammar Learning framework, while avoiding common fallacies of such experiments.

Through a formal but accessible overview of these topics, the chapter guides readers to an understanding of foundational and state-of-the-art concepts in formal language theory, how they influenced (and were influenced) by linguistic insights, and how they can be used to inform our understanding of linguistic cognition — thus providing the reader with the tools to successfully explore and evaluate existing experimental results on language complexity and learnability.

2 Mathematical Theories of Language and Cognition

The cognitive sciences owe much to the painstaking work of mathematicians in the beginning of the twentieth century — which set the stage for a variety of commonplace concepts and terminology as understood in cognition today: *computation*, *process*, *algorithm*, and so on. It is often tempting to conceive of these as metaphors. However, as Pylyshyn notes in his discussion of the nature of cognition and of cognitive theories, “the kinds of theories cognitive scientists entertain are intimately related to the set of tacit assumptions they make about the very foundations of the field of cognitive science. In cognitive science the gap between metatheory and practice is extremely narrow” (Pylyshyn, 1984).

With this in mind, what is the contribution of mathematical formalization to the study of cognition, and language in particular? Niyogi (2006) distinguishes between mathematical and computational descriptions, and points out that

> Mathematical models with their equations and proofs and computational models with their programs and simulations provide different and important windows of insight into the phenomena at hand. In the first, one constructs idealized and simplified models but one can now reason precisely about the behavior of such models and therefore be sure of one’s conclusions. In the second, one constructs more realistic models but because of the complexity, one will need to resort to heuristic arguments and simulations. In summary, for mathematical models the assumptions are more questionable but the conclusions are more

> reliable — for computational models, the assumptions are more believable but the conclusions more suspect.
>
> (Niyogi, 2006, pg. 43)

In this sense, it is important to disentangle the notion of computation in cognition from computational models of cognition. The development of a theory of computation came from an attempt to resolve a period of intense upheaval in the scientific and mathematical world: Cauchy (1821)'s methods for solidifying the calculus, Whitehead and Russell (1912)'s axiomatic method in *Principia Mathematica*, the Hilbert program (Hilbert, 1928) to prove the consistency and foundational security of mathematics, Carnap's proposals for logically reconstructing science in an experience (Carnap, 1928), are some examples of the first work along these lines. Hilbert, in particular, was interested in salvaging classical mathematics from the paradoxes of set theory.

> I should like to eliminate once and for all the questions regarding the foundations of mathematics, in the form in which they are now posed, by turning every mathematical proposition into a formula that can be concretely exhibited and strictly derived, thus recasting mathematical definitions and inferences in such a way that they are unshakable and yet provide an adequate picture of the whole science.
>
> (Hilbert, 1928, pg. 464)

Alan Turing's attempt to untangle this problem simultaneously produced what was effectively the first mathematical model of cognition, in the form of a theory of computation. His paper *On computable numbers, with an application to the Entscheidungsproblem* introduced several concepts and results that are fundamental to the modern study of computation (Turing, 1937, 1938). In his words, "We may compare a man in the process of computing a real number to a machine which is only capable of a finite number of conditions" (Turing, 1937). Turing conceived of a mathematical model of the cognitive phenomenon under study (now known as a Turing machine), and the range of tasks or behaviors it could possibly achieve (computably enumerable functions). Notably, Turing's second result was negative: there are some mathematical propositions which cannot be solved or performed mechanically by a machince possessing such universality.

Via these two results, Turing attempted to rigorously analyze the scope of "mechanical" intelligent processes or behaviors. Via explicit mathematical precision, Turing proved that there exist possible tasks, problems, or behaviors where the result is completely certain, but where no amount of mechanically intelligent processing will give that right result for every possible input. What does such a formal cognitive result provide? One could think of infinitely many other informal or intuitive definitions that could then be plausibly tested with experiments. The formal approach necessarily supersedes them all (van Rooij and Baggio, 2021).

The importance of mathematical negative results is well-put by Turing himself, who clarified that his results "are mainly of a negative character, setting certain bounds to what we can hope to achieve purely by reasoning. These, and some other results of mathematical logic, may be regarded as going some way toward a demonstration, within mathematics itself, of the inadequacy of 'reason' unsupported by common sense" (Turing, 1954). Turing explicitly hypothesized that limits of human reasoning have consequences for building theories of cognition, and concluded that theories of unconscious mental capacities must accompany the overt capacities that can be governed by conscious reasoning.

With Turing's discovery, more and more equivalent models of computation emerged. Consequently, Kleene (1952) transformed these results into the **computability thesis**: there is an objective notion of effective computability independent of a particular formalization. What has come to be known as the *Church-Turing thesis* states that a function on the positive integers is effectively calculable if and only if it is computable. More broadly it is the thesis that anything computable can be computed with a Turing machine or its equivalents (e.g., Church's lambda calculus). As the universality of computation emerged, the rigor and analytic transparency underlying the work became increasingly apparent.

The discovery of computational limits immediately transformed fields like biology and behavioral sciences, and facilitated the creation of the new field of Cognitive Science — which incorporated the study of computation as one of its pillars. Perhaps the most well-known case of the transformation in biology is the representation of the storage structure of DNA, discovered by Rosalind Franklin and later built on by James Watson and Francis Crick. The computability thesis demonstrated that a sequence or string of symbols from a fixed finite alphabet is the basis for all computations, since any program can be viewed as a string. The genetic discovery stated that DNA can be stored as strings over an alphabet of four symbols: A, G, T, and C, representing four nucleotides which are read by a particular type of computer (Searls, 2002). This is just one of the myriad ways that biological processes may be viewed as computation (see Brenner, 2012; Danchin, 2008, for more examples).

A large implication of Turing's and Church's insights, as Kleene showed, was that there must be a distinction between the computation and the device doing the computing. This distinction is obvious to anyone who writes programs, but can also be seen in things like traffic signals, maps of a city, or blueprints to a building. Consider this recent statement regarding the distinction in molecular biology:

> If constraints exist as to what sorts of modules and linkages can generate effective and robust behaviours, then fewer possibilities will need to be considered. The tool-kit of modules and of the linkages between them that operate in cells may thus be limited, reducing the complexity of the problem that has to be solved. [...] We need to know how information is gathered from various sources, from the environment, from other cells and from the short- and long-term memories in the cell; how that information is integrated and processed; and how it is then either used, rejected or stored for later use. The aim is to describe how information flows through the modules and brings about higher-level cellular phenomena, investigations that may well require the development of new methods and languages to describe the processes involved.
>
> (Nurse, 2008, pg. 425)

This statement mirrors almost exactly Marr (1982)'s methodological separation of the different levels at which one must understand a particular cognitive faculty (vision, in his case), now widely adopted in cognitive science. Marr distinguished a *computational level*, a *representations and algorithmic level*, and an *implementation level* (Marr et al., 1991). The computational level specifies as precisely as possible a statement of the problem under study (e.g., the nature of the function from an input to an output). The algorithmic level defines what representational systems are used, and how to manipulate their outputs. Crucially, Marr considered representation to be a key element for understanding computation, but put it at the same level as algorithms. Finally, the implementation level specifies how the information-processing model is meant to be solved biologically by cells, synapses, neuronal circuits, etc.

The importance of this separation between the system and its substrate cannot be understated. Independence between software programmers and hardware designers allows unprecedented progress in each area, with the knowledge that the formal links between them are guaranteed to still exist. In the same way, Marr's adoption of this framework allowed cognitive scientists to isolate the processes of cognition, all the while knowing that there was a biological substrate at a more refined level of description. It also allowed neuroscientists to understand what exactly the cell-to-cell interactions and circuit behaviors they observed and theorized about were supposed to be doing.

A linguistically relevant and generally successful example of the computational independence in Marr's framework comes from the study of systems that learn. Just as a computability theory studies a human in the process of computing a real number, a learning theory studies a human in the process of mapping a finite set of experiences to a real number, which represents some finite description of potentially infinite data that the human may not ever experience. The particular characteristics of such a learning system will be explained more in the sections to follow. Just like with computability, negative results again arise when one looks at the full scope of learnability from a formal and analytical perspective. Just as with computation itself, there are distinctions between what is effectively learnable, and what is effectively unlearnable.

In the cognitive study of language, what is often called the "logical problem of language acquisition" (Hornstein and Lightfoot, 1981) is the observation that "there is no free lunch — no way to generalize beyond the specific training examples, unless the learner commits to some additional assumptions" (Mitchell, 2017, pg. 4). Given a finite set of observations, there is an infinite number of possible generalizations consistent with them. This is true for every problem that can be abstracted as inducing generalizations from data. Thus, in order to arrive at a single, "correct" generalization, learners must be constrained in specific ways. Such constraints must exist a priori of the hypothesis space of the learner, and are thus separate from the observed data. Note that this is a logical inevitability of learning theory. The controversy is in where such priors must reside. A connectionist assumes that the biases correspond to the features defining the topology of a "neural" network. A behaviorist might assume constraints on the way generalizations are made. A nativist would say they reside in *a priori* categories, mechanisms, and constraints.

This perspective is a longstanding pillar of the field of ethology which gained a new strength after assaults by early 20th-century behaviorism, thanks to the newfound appreciation for the limits of reasoning and abilities. Eric Lenneberg, a crucial influence on the biological study of language, championed the idea of structural limits in learning since the early 1950s. In his "*Biological Foundations of Language*", Lenneberg noted that "there is no possible way in which we could think of a device, natural or artificial, that is freed from all structural information" (Lenneberg, 1967, pg. 394). Like Turing, Lenneberg understood the importance of hypothesis *classes* for learning, noting that "within the limits set, however, there are infinitely many variations possible. Thus the outer form of languages may vary with relatively great freedom, whereas the underlying type remains constant" (Lenneberg, 1967, pg. 374). Compare this to a later statement by the language acquisition researcher Lila Gleitman (1990), that "the trouble is that an observer who notices *everything* can learn *nothing*, for there is no end of categories known and constructible to describe a situation". Applying this same concept to the biological substrate of a learning system, the cognitive scientist Gallistel (1999) notes that

> Adaptive specialization of mechanism is so ubiquitous and so obvious in biology, at every level of analysis, and for every kind of function, that no one thinks it necessary to call attention to it as a general principle about biological mechanisms [...] From a biological perspective, the idea of a general learning mechanism is equivalent to assuming that there is a general purpose sensory organ, which solves the problem of sensing.
>
> (Gallistel, 1999, pg. 1179)

Thus, any debate that exists regards the *properties* of the innate constraints required for learning, not the *existence* of such innate constraints themselves. In real-world environments, an almost limitless number of distinct possible situations may occur. Thus, theories of the cognitive capacity of learning focus on the general properties of the significant generalization aspect: any learned behavior or task or function must be *productive* — effective in novel situations, not just those identical to what has already been experienced.

Suppose, adapting an example from Valiant (2013) which is probably familiar to many schoolchildren in the US, that a child is playing a game at a county fair. In front of the child is an unreasonably large container with millions of plastic gaming chips, each one labeled with a different number. The child can reach in and draw 10 chips at random, and they must guess which numbers occur at least once among the remaining chips. Will the child be able to win, or should an adult interfere to stop them from being scammed?

It should be immediately obvious that if the child approaches the game with no assumptions at all, there is no way to win. Why? As Valiant explains, there is a possibility that every chip has a unique number. In this case, any 10 chips the child draws will never enable them to identify numbers on the remaining chips. At the opposite extreme, if the game operator tells the child that all the chips are identical, then any single sample gives the child complete knowledge about all the remaining chips. With this constraint, the child can never lose. There are many other interesting constraints between these two extremes.

This game may seem absurd and trivial, but as Valiant (2013) notes, it is precisely the mathematical perspective that allows us to abstract away from the absurd particulars. Which possible constraints on a child/learner ensure that its algorithm is verifiably correct, effective, and feasible? Or, put another way, “We should somehow specify additional constraints on the generating process. But which constraints are plausible?” (Schmidhuber, 2002). As Valiant (2013) discusses, generalizing from a small number of types to an arbitrarily large set of types shows that the amount of evidence required to guarantee that a target hypothesis is chosen grows exponentially. A learning framework where the ability to generalize depends on exponentially many examples in terms of the number of types is completely infeasible, not to mention unrealistic. It is no surprise that the child “feels it all as one great blooming, buzzing confusion” (James, 1910).

However, humans learn language and many other abilities from incredibly small, distributionally skewed samples, even in cases where the number of possible distinguishable individual words and phrases is gigantic (Yang, 2013). Perhaps one could constrain the algorithm to learn from a set of examples that are *polynomial* in the number of types n. A learner which needs n or n^2 or even n^3 examples will learn more feasibly than a learner which relies on exponential data like 2^n. After seeing n examples, the learner should be able to generalize to a set with greater than n members. As Valiant (2013) notes, varying the conditions on an algorithm’s evaluation can mean the nature of the county fair chip game guarantees a loss or win, every time. Robust evaluation means the child does not even have to waste money trying hypotheses, nor does an experimenter have to make a group of participants play the game. The lessons for learning experiments here are powerful.

The distinction between computational and mathematical approaches outlined at the beginning of this section may seem odd, now that computation has been shown to be an analytically rigorous mathematical theory. Historically however, the term *computational* has drifted to encompass the sense outlined in the beginning, that of constructing a specific program and running specific simulations on a phenomenon of interest to see if the system effectively calculates or learns the behavior in question. The argument presented here and in the rest of the chapter is that the general analytical results are uniquely privileged. Practically, they make it possible to avoid redundant simulations or experiments, by clarifying a-priori when a problem will never work given specific resources. Or to find out that in fact the problem was trivially solvable on the conditions one set forth.

This distinction is particularly relevant nowadays, as the fields of Artificial Intelligence (AI) and Cognitive Modeling come more and more to prominence, while surprisingly straying from an analytic relationship to the broader cognitive sciences. The computer scientist Weizenbaum has a more critical view.

> [AI practitioners] simply mistake the nature of the problems they believe themselves to be “solving". As if they were benighted artisans of the seventeenth century, they present “general theories" that are really only virtually empty heuristic slogans, and then claim to have verified these “theories” by constructing models that do perform some tasks, but in a way that fails to give insight into general principles [...] Furthermore, such principles cannot be discovered merely by expanding the range of a system in a way that enables it to get more

> knowledge of the world. Even the most clever clock builder of the seventeenth century would never have discovered Newton's laws simply by building ever fancier and more intricate clocks!
>
> Weizenbaum (1976, p.196)

In line with this, the work reviewed in this chapter wants to underline how, if one strives for a successful theory of cognitive abilities, interpretable formalization cannot be ignored. Mathematical rigor provides a clarity of purpose and a guard against uncertainty, that is unique in the natural sciences. Thus, it is by necessity complementary to empirical studies of phenomena in cognition (Niyogi and Berwick, 1996; Niyogi, 2006). The advantages of this approach have been clear since Euclid: take a system, decompose it into its parts, idealizing when necessary and to varying degrees, and understand what consequences those choices have. This approach might force researchers to slow down, and sometimes produce unsatisfying or counterintuitive results. But, as demonstrated by its adoption in field after field, this is a framework that provides a level of insight that could not be attained otherwise. The remainder of this chapter attempts to showcase the ways in which this kind of formalization can enlighten the study of language, of learning, and of linguistic cognition at large.

3 Formal Language Theory and Cognitive Theories of Language

The previous section discussed in depth how computational concepts and mathematical tools have been fundamental in theory development in the larger cognitive sciences. In this sense, the study of human linguistic cognition has suffered from the same issues of other psychological sciences. That is, privileging establishing effects of linguistic behavior over a search for deep explanatory principles (Cummins, 2000). This might be partially due to skepticism for high level abstraction, and a belief that mathematical formalization leads to losing track of the *real* empirical facts. Moreover, some of the resistance against formalization seems to come from confusing highly specified *formal* theories, with narrative theories enriched with technical *jargon* and complex mathematical notation. In contrast to these views, the rest of this chapter aims at exemplifying how it is possible to draw useful generalizations about linguistic cognition, by evaluating language patterns in terms of (a specific notion of) computational complexity.

When motivated and understood, precise formal characterizations are fundamental to the study of linguistic objects, as they allow for the development of theories that are *falsifiable* and *explanatory*: they not only aim to account for certain sets of data and to make predictions about which kind of data are to be expected, but are able to offer transparent explanation as to the relation between data and predictions (Martin and Baggio, 2019; van Rooij and Baggio, 2021; van Rooij and Blokpoel, 2020; Guest and Martin, 2021).

Focusing on the computational properties of linguistic processes allows researchers to take a step back from framework-specific analyses and ask precise questions about the nature of the generalizations observed in natural languages. In line with this, this chapter takes a particular stance on the complexity of linguistic patterns grounded in formal language theory (FLT). While FLT is extensively studied in the more formal subareas of computer science, it has a long history in generative linguistics — starting with Noam Chomsky and Marcel Schützenberger's early work (Chomsky and Schützenberger, 1959; Chomsky, 1959), based on the results by Alan Turing and Emil Post discussed in the previous section (Church, 1936b,a, 1937; Turing, 1937, 1938; Post, 1944). Crucially, this approach allows for the unambiguous specification of the set of requirements for a specific cognitive theory, by focusing on the fundamental primitives of the cognitive ability under study characterized in terms of functions from a particular input to a corresponding output. The rest of this section is dedicated to intuitively illustrate these ideas about linguistic complexity and their relevance to the formulation of cognitively sound theory of language.

3.1 The Chomsky Hierarchy

The best known formal language theoretical concept in linguistics is probably that of the so-called *Chomsky-Schützenberger Hierarchy* (Chomsky, 1959). Under this view, *languages* are characterized as unbounded sets of *strings*, constructed from a *finite vocabulary* (or alphabet) Σ, and well-formed according to some underlying property. For example, one could imagine a language L_a that only allows sequences of a:

$$L_a := \{a, aa, aaa, aaa, \dots\}$$

The properties of a language are fully encoded in the (finite) *grammar* that can generate it (Chomsky, 1959; Post, 1944). According to the computability thesis, every grammar can be viewed as a function, so the focus turns to the properties of particular types of functions. Such a function may map strings to binary values such as 0 or 1. One could also consider other functions which take strings as input and return various values, depending on the desired properties of the grammar under study. For example, a grammar can be made stochastic by replacing the binary mapping with a mapping to real values between 0 and 1, or to represent linguistic transformations by changing strings to strings, or strings to trees, and so on (see Table 1).

Function	**Description**	**Linguistic Correlate**
$f : \Sigma^* \to \{0,1\}$	Binary classification	(Well-formedness)
$f : \Sigma^* \to \mathbb{N}$	Maps strings to numbers	(Well-formedness)
$f : \Sigma^* \to [0,1]$	Maps strings to real values	(Gradient classification)
$f : \Sigma^* \to \Delta^*$	Maps strings to strings	(Single-valued transformation)
$f : \Sigma^* \to \mathscr{P}(\Delta^*)$	Maps strings to stringsets	(Multi-valued transformation)

Table 1: Grammars as functions. Adapted from Rawski and Heinz (2019).

The use of such string sequences is not a claim that the literal content of linguistic experience is a string. However, this notion of generation presupposes a mapping between the strings belonging to a language and specific structural descriptions with respect to a grammar.

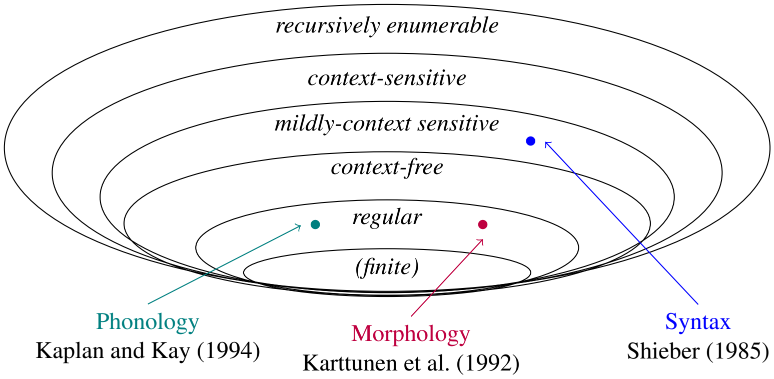


Fig. 1: The Chomsky Hierarchy (including mildly-context sensitive languages), and classical placement of linguistic sub-modules.

> For a class of grammars to have linguistic interest, there must be a procedure that assigns to any pair (σ, G), where σ is a string and G a grammar of this class, a satisfactory structural description of the string σ with respect to the grammar G. In particular, the structural description should indicate that the string σ is a well-formed sentence of the language $L(G)$ generated by G, where this is the case. If it is, the structural description should contain grammatical information that provides the basis for explaining how σ is understood by the speakers who have internalized the grammar; if it is not, the structural description might indicate in what respects σ deviates from wellformedness.
>
> (Chomsky and Schützenberger, 1959, pg. 119)

Thus, this approach highlights fundamental properties of the language patterns in terms of formal requirements on their generative devices, and has been extensively adopted in the attempt to define the class of *possible* natural languages. In this respect, the *Chomsky-Schützenberger Hierarchy* characterizes linguistic dependencies in a hierarchy of nested regions based on the complexity of their grammars (Figure 1).

The Chomsky Hierarchy as originally formulated distinguishes four regions of complexity — recursively enumerable, context-sensitive, context free, and regular — based on the constraints put on the types of grammars that each region allows. *Recursively enumerable languages* (also, computably enumerable) correspond to those languages that can be defined by a *Turing machine* (see Turing (1937) for the original formulation, and Hopcroft et al. (2001) for a technical but accessible introduction). While the debate on the relevance of Turning machines to cognitive science and theory of mind is vast (McCulloch and Pitts, 1990; Piccinini and Bahar, 2013; Pylyshyn, 1984; Putnam, 1967; Sprevak, 2010, a.o.), this particular class seems to be of little interest from the perspective of establishing a cognitive theory of human linguistic abilities, as all languages that can be defined by some formal grammar fit into it. Given this generality, recursively enumerable languages offer the weakest (almost trivial) condition to

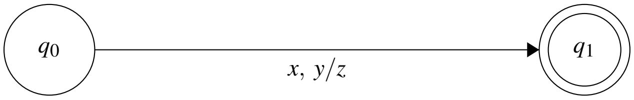


Fig. 2: Graphical representation of a generic pushdown automaton.

the complexity allowed to natural language, and thus lack a precise specification of essential computational requirements.

> [This] condition, on the other hand, has no interest. We learn nothing about a natural language from the fact that its sentences can be effectively displayed,i.e., that they constitute a recursively enumerable set. The reason for this is clear. Along with a specification of the class F of grammars, a theory of language must also indicate how, in general, relevant structural information can be obtained for a particular sentence generated by a particular grammar.
>
> (Chomsky, 1959, pg. 138)

A more interestingly restricted class in connection to the study of natural languages has been, historically, that of *context-free* languages. Context-free languages (CFL) can be recognized by a context-free grammar, which inherently imposes some kind of hierarchical organization on the string. It is possible to decide whether a string belong to a CFL or not via a specific machine called *pushdown automaton* (PDA; *automata* in the plural). A PDA is a machine equipped with a finite number of *states*, and enriched with a memory mechanism knows as a *stack* which keeps track of information about the string. The information in the stack guides the transitions of the machine, together with the current state and information from the input.

Figure 2 represents a PDA with two states: q_0 and q_1. Of these, only q_1 corresponds to an ending state (double circled): the machine succeeds in recognizing a string σ, iff it is in an ending state when the last symbol in σ is received. Arrows between states symbolize possible transitions. These are annotated as $x, y/z$, such that x represents the symbol that triggers the transition (e.g., from q_0 to q_1). Stack usage is instead represented by y/z: in order to transition the top of the stack needs to match y, which is then replaced by z.

A classical example of a CFL language is $a^n b^n$ — which only allows for a contiguous sequence of a's, followed by a contiguous sequence of b's of exactly the same length (that is, the number of a's and b's must be identical and equal to n, where n is an arbitrary integer). A PDA that accepts all and only the strings in this language is in Figure 3, where ε represents the empty string. The idea is that the machine stays in q_0 as long as the input string contains a's, and uses the stack to keep track of how many of these it sees. The first b then acts as a trigger, and the PDA keeps accepting b's only as long as the stack contains a's to match them off.

Consider now the class of *regular* languages. These are languages that can be described by machines consisting only of a a finite number of internal states: finite state automata (FSA). These devices can only keep track on a limited amount of information,

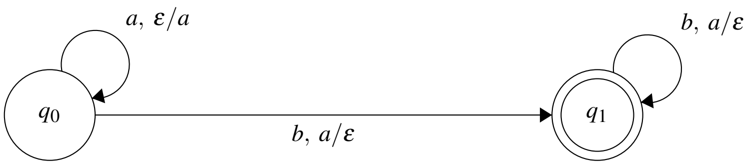


Fig. 3: A pushdown automaton for the language $a^n b^n$.

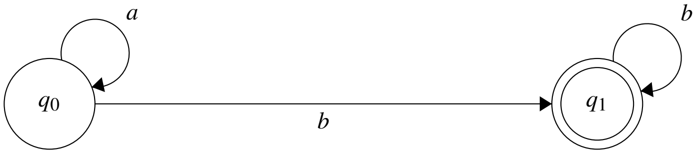


Fig. 4: A finite state automaton for the language $a^m b^n$.

and thus they cannot recognize languages with patterns that rely on information about the full length of a string. For instance, consider the FSA in Figure 4. This automaton recognizes strings belonging to the language $a^m b^n$: that is, a contiguous number of a's followed by a contiguous number of b's. Importantly, in contrast to the CF language $a^n b^n$, here there is no need to keep track of how many a's and b's are in the string. All the device needs is the ability to recognize that once a b is seen in the string, no a is allowed to follow — thus, these devices rely on limited, local memory of recent events in order to make decisions.

The expressivity of the machines associated to each level of the Chomsky Hierarchy has been used extensively to debate the positioning of linguistic dependencies with respect to regions of complexity. For instance, Chomsky (1957) argued that the syntax of English's center embedding constructions cannot be captured by finite state devices, an argument which would then place the upper bound for the expressivity of English syntax beyond regular (in fact, syntactic dependencies in natural language have since then been placed even outside the CF region, see for example Shieber, 1985; Kobele, 2006, a.o.). Similar complexity claims can be made for other domains of linguistics (Karttunen et al., 1992; Kaplan and Kay, 1994, see Figure 1). Note that this reference to linguistics domains should not be interpreted as making any commitment to the modularity of linguistic cognition, as it is merely referring to the study of different types of linguistic data (cf. Fodor, 1983).

While a detailed discussion of the accuracy of these claims is beyond the scope of this chapter, complexity characterizations outlined by the Chomsky hierarchy allow researchers to draw a connection between attested linguistic patterns and the expressivity of the machinery that can be used to evaluate them. In doing so, these classes provide a measure of descriptive adequacy for linguistic theories based on the *typology* of observed linguistic phenomena. In fact, these complexity claims have direct consequences for any theoretical framework that affirms domain specific generalizations — if we aim for theories that are both sufficiently *expressive* and *restrictive* enough to account for existing natural language patterns and nothing else. That is, it is desirable to have theories that account for the distribution of linguistic patterns across natural languages, and that do not predict the existence of unattested patterns.

Consider for instance the case of theories of phonology — e.g., Optimality Theory (OT) — evaluated against the assumption that phonological transformations from underlying to surface forms are *regular* (Kaplan and Kay, 1994). Under this view, the regular region sets up an upper-bound to the computational expressivity of phonology: that is, we do not expect to find natural language phonological patterns that require supra-regular machinery. In this sense, a certain amount of work has been done to translate OT grammars to regular relations, showing that the interaction of very simple OT constraints can yield non-regular patterns (Frank and Satta, 1998). If one subscribes to the idea that a theory that is over-expressive is undesirable, complexity characterizations then highlight the shortcomings of OT, and may suggest ways in which the theory could be restricted in order to reduce over-generation.

From a cognitive perspective however, the reader might still be (reasonably) wondering whether such an approach is offering us any truly valuable insight. Do the formal characterizations outlined so far allow us to make any precise conjecture about cognitive mechanisms? In fact, there are several reasons to believe that the Chomsky hierarchy as presented so far is too opaque to make precise claims about essential attributes of linguistic cognition (Rogers and Pullum, 2011; Jäger and Rogers, 2012; Rogers et al., 2013). One of the main shortcomings of the division in regions as discussed above is the fact that it presupposes very specific types of recognition mechanisms (grammar, automata, etc.):

> Classes of the Chomsky hierarchy provide a measure of the complexity of patterns based on the structure of the mechanisms (grammars, automata) that can distinguish them. But [...] these mechanisms make judgements about strings in terms of specific analyses of their components. When dealing with an unknown mechanism, such as a cognitive mechanism of an experimental subject, we know nothing about the analyses they employ in making their judgements, we know only that they can or cannot make these judgements about strings correctly.
>
> (Jäger and Rogers, 2012, pg. 1961)

What does this imply? Consider again the case of a PDA for context-free languages. By characterizing the $a^n b^n$ language in terms of a PDA, one could be tempted to assume that a memory stack is needed in order to recognize such language, and thus set it up as a cognitive primitive. However, there are multiple alternative mechanism that can recognize that language, which have little in common beyond the general idea that keeping track of information about the length of a string is required to distinguish patterns in the language. What seems necessary as a good basis for a fruitful investigation of the relation between computational complexity and linguistic cognition, is a way to isolate essential properties of a pattern, that must be detected by any device able to recognize such a pattern — while also minimally separating it from properties of patterns of different complexity. These desiderata seem in line with the call to theory development in psychology as described, for example, by van Rooij and Baggio (2021):

> Generally, psychological theories of capacities should be (i) mathematically specified and (ii) independent of details of implementation. The strategy is precisely to try to produce theories of capacities meeting these two requirements, unless evidence is available that this is impossible, e.g., that the capacity cannot be modeled in terms of functions mapping inputs to outputs.
>
> (van Rooij and Baggio, 2021, pg. 7)

Note that the second argument does not imply uninterest in the connection between abstract cognitive processes and their physical implementation. What it remarks instead is the importance of isolating core primitives of the mental abilities under study, that must *necessarily* be relevant to any system implementing them. These issues are addressed by focusing on *descriptive* characterizations of complexity classes (Rogers and Pullum, 2011).

3.2 Beyond the Chomsky Hierarchy: Subregular Languages

Recent formal language theoretical work on the expressivity of linguistic patterns has focused a fine-grained hierarchy of sub-classes which regular languages can be divided into (McNaughton and Papert, 1971; Brzozowski and Knast, 1978; Eilenberg, 1974; Pin, 1986; Thomas, 1997). Crucially for the points made above, languages in this *subregular* hierarchy have been characterized in terms on logics, highlighting those properties of a pattern that do not depend on the structure of specific recognition mechanisms (see Figure 5). These are *descriptive* characterizations that focus on the information necessary to distinguish strings (or, more generally, structures) that present a certain pattern from strings that do not (Rogers and Pullum, 2011). As the hierarchical classification is based on this type of fundamental information, any devices able to recognize members of a class will have to be at least sensitive to such information. Moreover, *subregular* languages have been extensively studied from a variety of alternative perspectives — including those of grammars and automata. Thus, although they do not *presuppose* any particular recognition mechanism, they provide abstract characterizations of *possible* mechanisms that allow generalizations about the cognitive processes involved in the recognition of specific patterns over others.

Overviewing the abundance of work leveraging subregular classes for detailed analyses of linguistic patterns is beyond the scope of this chapter (but see Heinz and Idsardi, 2013; Chandlee, 2014; Heinz, 2015; Jardine, 2015; McMullin, 2016; Graf, 2016; Aksënova et al., 2016; Strother-Garcia, 2019; De Santo and Graf, 2019; Graf and De Santo, 2019; Vu, 2020, a.o. for an overview of results across linguistic domains). Here the interest is in exemplifying how this classification emphasizes core properties of linguistic dependencies. A formal introduction to the properties of all subregular classes is thus burdensome and not necessary, as a purely intuitive understanding of the properties behind the simplest of such classes should be sufficient to outline the main insights of the approach.

Consider *Strictly Local* (SL$_k$) languages, at the base of the hierarchy. These languages comprise patterns that exclusively rely on symbols immediately *adjacent* to each other in a string. An SL$_k$ language can be fully defined by a finite set of allowed

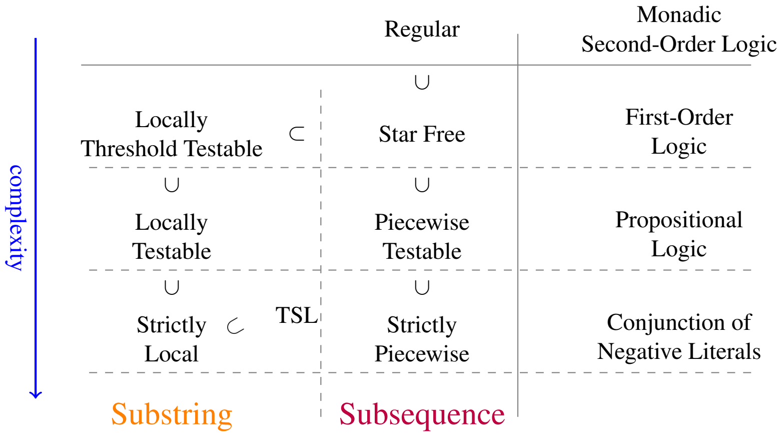


Fig. 5: Classes in the subregular hierarchy, aligned according to their subsumption relations, sensitivity to adjacency vs precedence, and corresponding logical definability, following (Heinz, 2018).

(or disallowed) contiguous *substrings* of symbols of length k. For instance, the string *abcd* contains the following substrings of length 2: *ab*, *bc*, and *cd*. To understand how this can be applied to the study of linguistic patterns, consider now the phenomenon of *intervocalic voicing* in Northern Italian. In this variety, the alveolar fricatives [s] and [z] occur in what is known as complementary distribution: only [z] can occur within two vowels (Bertinetto and Loporcaro, 2005). Thus *casa* (home) can only be pronounced as [kaza] and not [kasa]. This can be captured by forbidding sequences of three symbols, where [s] appears within two vowels:

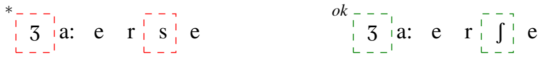
$$^{*}\ \text{k}\ \boxed{\text{a}\ \ \text{s}\ \ \text{a}} \qquad\qquad ^{ok}\ \text{k}\ \boxed{\text{a}\ \ \text{z}\ \ \text{a}}$$

Assuming that the vowel inventory of the language is $\Sigma = \{a,e,i,u,o\}$, the pattern is fully licensed by listing all the trigrams (contiguous substrings of length 3) that are forbidden from occurring within a string of the language:

$$L_{vocalic} = \{asa, ese, isi, oso, usu, ase, asi, aso, asu, esa, \dots\}$$

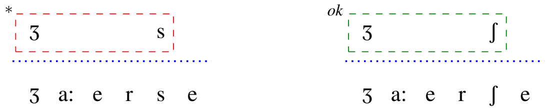
Note that here the vowels are treated as independent symbols in the alphabet, but it would be equally possible to list trigrams that refer to more abstract properties of such symbols (e.g., being a vowel). What is crucial is that whatever cognitive mechanism is in charge of distinguishing [kaza] from [kasa] only needs to keep track on three adjacent symbols at the time. Generally, the core information a system sensitive to SL$_k$ patterns needs to capture is immediate adjacency between k symbols in a string (its k-*factors*). Sensitivity to substrings of finite length distinguishes strictly local languages from other classes in the hierarchy, as for example those that rely on *precedence* instead of adjacency between symbols.

Strictly piecewise (SP$_k$) languages distinguish patterns based on *subsequences* of symbols of length k: that is, k symbols within a string, with arbitrary intervening material between them (Fu et al., 2011). Consider again the string $abcd$. The subsequences of length 2 in it are: ab, bc, and cd, ac, ad, and bd. While SL languages can easily describe a variety of local phenomena in natural languages (like the voicing example described above), human languages present an abundance of non-local patterns (also commonly referred to as long-distance dependencies). For instance, Aari (an Omotic language of south Ethiopia) shows a case of unbounded sibilant harmony: sibilants within a word must agree in *anteriority*, independently of which other symbols occur within them (Hayward, 1990). This pattern is clearly outside of the expressivity of SL languages, as it is based on the relations between non-adjacent symbols. SP languages can easily account for such a phenomenon however, by relying on precedence relations between segments — for instance, by forbidding the occurrence of [ʒ] followed by [s] anywhere else in the string (so banning *ʒs as a subsequence).



As in the case for the local processes captured by SL languages, the ability to keep track of finite-length subsequences is the necessary component of any mechanism able to detect strictly piecewise patterns.

The fine (and growing) granularity of the hierarchy also allows for a variety of distinctions that could be missed otherwise. Consider once again the non-local pattern in Aari. A different way of characterizing it is via the class of *tier-based strictly local* (TSL) languages (Heinz et al., 2011). This class has no access to the precedence operator, as in SP languages. How can it then capture dependencies between non adjacent segments? The intuition is that the class relies on a notion of relativized locality: a TSL dependency is a dependency that is non-local in the input string, but local over a special structure called a *tier*. A tier is defined by considering only a subset of the segments of the input string as relevant, and then evaluating strings based on substrings of length k allowed (or, alternatively, not allowed) on the tier. For Aari, it is possible to enforce long-distance sibilant harmony (in anteriority) by projecting from the input a tier T that only contains sibilants, and ban adjacent *ʒs and *sʒ on T.



The reader might now wonder if the existence of equivalent characterizations of long-distance harmony defeats the whole purpose of this section, as it seems to bring us back to multiple alternatives for the same process. Note however that the difference between TSL and SP is deep. In order to understand how, assume that in Aari the harmony requirement stands unless there is an [r] occurring somewhere in the string between the two sibilants (that is, [r] acts as an harmony blocker). Under a TSL analysis, this can be captured by making [r] a tier segment, thus disrupting immediate adjacency between the sibilants. However, SP languages are unable to capture this, as ʒs is a subsequence in the string independently of which segments intervene between the two symbols.

Thus, while TSL and SP share some of the patterns they can encode, they do so in fundamentally different ways. Understanding the trade off between TSL's relativized locality and SP's use of precedence makes it possible to once again conceive of patterns that unambiguously belong to one class or the other. Characterizing local and non local patterns in terms of SL, SP, or TSL languages then allows for the study of properties of these patterns without *a priory* assumptions about the structure of the cognitive mechanisms that distinguish them.

Importantly, as mentioned a few times before, this is an approach that relies on the exploration of attested linguistic phenomena in order to establish an *upper bound* to the complexity of natural languages. That is, claiming that natural language phonology has the expressive power of TSL languages does not imply that every phonological dependency has to be TSL — there could be an abundance of SL patterns — nor that every conceivable TSL pattern should be attested. It is instead a claim about the general properties of the language system, which allows to make precise predictions about what kind of patterns should *not* be expected (e.g., patterns that require the full power of regular languages). A common objection to this type of approach is that not every human language has been studied in detail, and that it is unreasonable to make generalizations based on patterns that have not been encountered. For example, McCollum et al. argue that "*naive estimates of frequency counts or literature reviews are not a reasonable evidential basis for concluding that a particular type of phenomenon is categorically impossible*" (McCollum et al., 2020, pg. 23). To approach this issue from a different perspective, consider now a somewhat extreme case: *flying pigs*. While the modern taxonomy of animal species is vast, it is hard to claim that we have categorized every species ever existed (in fact, novel species keep being "discovered"; McGregor et al., 2020). However, the cumulative understanding of Earth's biodiversity makes a theory which discards the possibility of a species like flying pigs more *plausible* than a theory which assumes every conceivable species just waiting to be found. Note that the more constrained theory is also the one that would be more informative *should* flying pigs actually exist, as it predicted their non-existence based on a precise sets of biological/evolutionary constraints. The limited nature of the data available to an observer in order to formulate a comprehensive theory is thus a problem common to all empirical investigation.

> The problem that we cannot deduce (or even straightforwardly induce) theories from data is a limitation, or perhaps an attribute, of all empirical science [...] Still, one may abduce hypotheses, including computational-level analyses of psychological capacities. Abduction is reasoning from observations (not limited to experimental data; more below) to possible explanations [...] It consists of two steps: generating candidate hypotheses (abduction proper), and selecting the "best" explanatory one (inference to the best explanation).
>
> (van Rooij and Baggio, 2021, pg. 9)

The ability to formulate hypotheses from available observations that then lead to explanatory predictions is, as discussed above, a core component of any plausible theory of cognitive capacities. In this sense, the goal of the formal approach outlined here is not to rule out phenomena as categorically impossible. A formal analysis of attested patterns in terms of their complexity serves to make precise, explanatory generalizations about *expected* empirical phenomena. Should a phenomenon that exceeds a formerly assumed upper bound arise, it would have no consequence of the characterizations of previously encountered phenomena, and it would directly point to what kind of information was deemed to be beyond the scope of the theory by the previous upper bound.

Finally, while the discussion so far has focused on leveraging typological data about linguistic patterns to formulate analyses in terms of their formal complexity, the claim is that such characterizations inform our idea of language and cognition. It is thus reasonable to wonder whether and how these characterizations are reflected in linguistic behavior (Reber, 1969; Levelt, 2020; Uddén et al., 2020; Fitch and Hauser, 2004; Öttl et al., 2015). A shift from typology to behavioral experiments would also offer observations about complexity patterns that are distinct from typological attestation. In this sense, a possible hypothesis is that more complex patterns should lead to longer processing time — as exemplified, for instance, by self-paced reading experiments (cf. Chesi and Moro, 2014, 2015). Importantly however, there is nothing in the formal arguments per se that support such an hypothesis, as theories of processing have once again to make assumptions about the structure of the underlying mechanism. A different route is to instead investigate how the boundaries assumed by different formal classes influence humans' abilities to *learn* patterns of different complexity. The rest of the chapter is thus focused on exploring the relations between complexity bounds and learning hypotheses, and discuss how the fine-grained insights of the subregular hierarchy can inform experimental explorations of human cognitive biases.

4 Formal Theories of Grammar Learning

In Section 2, the general problem of induction was posed as a facet of the general mathematical study of cognition. Learning theories provide rigorous definitions of what learning means and ask, under those definitions: What can be learned, how so, and why? In this respect, determining which definitions are "correct" or "best" for a given scenario is a core issue.

Learnability results are foundationally important results for all branches of language science, because they are independent (yet characteristic) of particular theories and perspectives. Specifically, learning theories provide non-trivial conditions of explanatory adequacy on any theories of natural language.

> For a class of languages to be the natural languages, the class must be learnable by children on the basis of the kind of linguistic exposure typically afforded the young. Call this the learnability condition on the class of natural languages. Formal learning theory is an attempt to deploy precise versions of the learnability condition in the evaluation of theories of natural language. In the present context, such a theory will specify (a) the kind of linguistic input available to children, (b) the process by which children convert that experience into successive hypotheses about the input language, and (c) the criteria for "internalization of a language" to which children ultimately conform. From (a)-(c)

> it should be possible to deduce (d) the class of languages that can be internalized in the sense of (c) by the learning mechanism specified in (b) operating on linguistic input of the kind characterized in (a). Such a theory is correct only if (d) contains exactly the natural languages.
>
> (Osherson and Weinstein, 1983, pg. 37)

What makes a class of languages learnable? These results have permeated almost every theory of language, and indeed every problem of learning related to structure. This section presents a well-known evaluation framework and discusses its extensions and consequences. Readers interested in an in-depth discussion of these results may refer Heinz (2016) and Niyogi (2006), as well as Heinz and Rawski (orth) for a history of learnability in linguistics.

4.1 Membership problems

Central to a mathematical formulation of language learning, as Niyogi (2006) notes, is that children are not exposed directly to grammar. They get “exposure to the expressions of their language, along with extragrammatical and nonlinguistic cues, yet they reliably acquire a grammar that provides a compact encoding of the ambient language they are exposed to in a particular linguistic community". Mathematically, this basic question of whether some expression in a language is well-formed according to a given grammar is called the membership problem.

The concept of the membership problem allows us to clearly formulate a definition of learning. In particular, is there a learning algorithm A which takes as input a finite subset of data D of the possible acceptable forms of a language S, and returns a grammar which solves the membership problem M for the target language? Formalizing in this way means that each of the objects A, D, S, M can be studied independently or in combination with the others, at various levels of abstraction. It is always crucial that learners acquire a grammar, not a language. This is simply because, mathematically, grammars are of finite size, while the extensions of the grammars may be functionally infinite in size.

Under this inductive view, learners are functions from experience to grammars, which are themselves functions. This characterization of learners is precise but broad, since any inductive learning procedure is a function from experience to grammars. Mirroring the discussion in Section 2, this includes connectionist (Rumelhart and McClelland, 1986), Bayesian (Griffiths et al., 2008), and innatist (Wexler and Culicover, 1980; Berwick, 1985; Niyogi, 2006) approaches, among others.

Learning theory is also concerned with “the circumstances under which these hypotheses stabilize to an accurate representation of the environment from which the evidence is drawn. Stability and accuracy are conceived as the hallmarks of learning” (Osherson et al., 1986). How difficult is it for a learner to arrive at a generalization, and what role does the evidence a learner receives play? There are several general statements one can make (Heinz, 2016). Learners exposed only to *positive evidence* (only elements of the target language) have a harder task than those given both positive and *negative* evidence (both elements in and outside of the language). Learners having access to inaccurate, mislabeled or “noisy” evidence have a harder task than those given accurate evidence. Learners who may query an oracle or teacher about acceptability of a form have access to more information than those who cannot. Requiring that a learner exactly selects the right target grammar is a stricter condition than requiring them to select an approximately correct target grammar (see Heinz, 2016, for further discussion on the variety of learning paradigms and their consequences for theories of cognition).

4.2 Enumeration and Universal Grammar

Does every possible language have a solution to the membership problem? As the reader may have already intuited, another consequence of the computability thesis is that most languages have no solution to the membership problem. Why can a learner not just entertain any possible grammar, moving one by one until a grammar is selected which works?

This problem may be formulated concretely by understanding enumeration. A set S is enumerable, or countable, if its members can be arranged in an ordered list where each member will eventually be encountered. Formally, there is a function f which maps positive integers to members of S such that f is *onto*, i.e. for every element s in the co-domain there is some x in its domain such that $f(x) = s$. For example, consider this list of natural numbers:

$$1, 2, 3, 4, 5, 6, \ldots$$

Every positive natural number in the list will appear in some finite amount of time. Every element can be enumerated — a number can be assigned to it by a function which gives the value n to each positive nth integer. Now consider a different list of natural numbers:

$$1, 3, 5, \ldots, 2, 4, 6, \ldots$$

For this list, it is impossible to assign an index to any even numbers: they will not all be encountered in a finite amount of time. In an acceptable list, each item must appear sooner or later as the nth entry, for some finite n.

It is immediately obvious that the set Σ^* of all strings over an alphabet Σ is enumerable. The usual way to enumerate strings in Σ^* is to order them by length and then alphabetically within strings of the same length, as shown below over the alphabet $\Sigma = \{a, b, c\}$.

$$\begin{gathered} 0 \to \varepsilon \\ 1 \to a \\ 2 \to b \\ 3 \to c \\ 4 \to aa \\ 5 \to ab \\ 6 \to ac \\ \ldots \end{gathered}$$

However, what about languages or stringsets, the subsets of the powerset $\mathscr{P}(\Sigma^*)$? Recall that a powerset is the set of all subsets of a set S, or the set of all languages in the case of Σ^*. An argument from Georg Cantor (1892) demonstrates that the powerset of any countable set is uncountable, meaning there is no way to properly enumerate the members of that set. This has immediate and far-reaching consequences for learnability. Any learning algorithm that solves the membership problem is of finite length. This means it can be written as a finite string, and is an element of Σ^*. Consequently, there are at most countably many languages S which have programs which solve the membership problem of S. But there are uncountably many languages (elements of $\mathscr{P}(\Sigma^*)$), so most languages have no solution to the membership problem. Consequently, any possible learning framework whose target language S is non-enumerable, or uncomputable, will never have a solution. The learning algorithm cannot ultimately return a grammar which solves the membership problem for S.

It is now apparent that learnable languages must come from a well-defined class, and that pure tabula rasa learning is impossible. As mentioned previously, this forms what in the cognitive study of language is often called the “logical problem of language acquisition”. Necessary, *a priori* restrictions on the hypothesis class of candidate grammars forms the requirement of a *Universal Grammar*. Perhaps the constraints given by the Chomsky-Schützenberger Hierarchy in the last section are enough of a restriction to ensure learning will succeed, given a particular formulation of the learning problem. The remainder of the chapter will consider one particularly influential inductive inference paradigm, Gold’s *Identification in the Limit* framework (Gold, 1967). There are of course many other frameworks inspired by Gold, and readers are referred to the overviews mentioned earlier.

4.3 Grammar Identification in the Limit

Gold (1967), inspired directly by mathematical linguists’ application of the theory of computation, introduced the first inductive learnability results for a type of learner over classes of formal languages. In his framework, learning is a continuous process unfolding in time with no end. Evidence comes incrementally, and the learning algorithm incrementally outputs a grammar based on its experience thus far. As time goes on, the output grammar must be identical and must solve the membership problem for the target language in order for learning to succeed. The focus of Gold’s framework squarely puts the problem on generalization: is there a successful learning strategy given generous input, time, and resources, but where the target must be met exactly?

More precisely, according to the Identification in the Limit framework there are no limits on the learner’s computational resources or time, and each input is assumed to be a finite initial portion of an infinitely long data stream drawn from the target language S. Learners map the finite pieces of the data stream to grammars. A particular piece of evidence, or *positive presentation*, of the target language S is a function $f : \mathbb{N} \rightarrow S$ such that f is onto, meaning for every string $s \in S$, there is some number $n \in \mathbb{N}$ such that $f(n) = s$.

Gold’s article also considers evidence of different forms: positive evidence, positive and negative evidence, and arbitrary/primitive recursive evidence. Positive evidence (often called positive data) for a formal language S is such that every element of S can be observed at least once. Positive and negative evidence includes every logically possible string in Σ^* at least once, along with a label indicating whether it belongs to the target language S or not, similar to what is today called *supervised learning*. Primitive recursive evidence is beyond the scope of this chapter.

Consider the case of positive evidence. The learner initially has no input evidence, no cumulative evidence, no grammar, and thus no language. At the first time point, the learner gets evidence $e(1)$ as a positive presentation from the target language. The list of cumulative evidence $\langle e(1)\rangle$, or the finite portion of the data stream seen thus far, updates to include this new evidence. The grammar $G(1)$ updates to incorporate the new evidence according to the constraints of the grammar. Since the grammar is updated, the language (or extension) of the grammar $L(G(1))$ may or may not change. This process continues for each new data point, so evidence $e(n)$ to time n results in cumulative evidence $\langle e(1), e(2), \ldots, e(n)\rangle$ which forms Grammar $G(n)$ whose extension is $L(G(n))$.

What constraints are placed on the learner by this framework? First, success for a learning algorithm means that it converges over time to a correct generalization. As the learner encounters successive data points from this stream, it generates a corresponding stream of hypothesis grammars. In this framework, a learner *converges* to a grammar G if at some finite point, every future hypothesis it generates from new data is exactly G. The learner identifies a language in the limit if G generates the target language for *any* sequence of data from the target language. In particular, at some time point n, the algorithm must output the same program and this program must solve the membership problem for the target set S. The learner identifies a *class* of languages if it identifies in the limit every member language of the class. Readers are referred to Niyogi (2006) for a description of Probably Approximately Correct (PAC) learning in this setting. The second point, which follows naturally, is that the algorithm can make finitely many mistakes when generalizing, even though there is no bound on the number of inputs the learner receives.

When evaluating algorithms against criteria like identification in the Limit, it is important to remember that the algorithm exists independently of the evaluation criteria. Frameworks like Gold's are a way to understand the behavior of the algorithm generally. Gold (1967)'s work derives several important results from these constraints, both positive and negative. A learner exists which identifies the class of finite languages in the limit from arbitrary positive evidence. However, this is not the case for *super*-finite classes of languages, which include all finite languages and *at least one* infinite language. There is no learner which can identify any super-finite class in the limit from arbitrary positive evidence. The major consequence of this result is that none of the traditional classes of the Chomsky hierarchy (Regular, Context-Free, or Context-Sensitive) as discussed before are identifiable in the limit from positive data following this paradigm. However, learners which have access to positive and negative data are able to learn any of the computable languages (see Heinz, 2016, and references therein).

Building on the work of Gold, Heinz (2010) considered a class of hypothesis grammars known as *string extension grammars*, which are finite subsets of some set A. This means they include the Strictly Local, Strictly Piecewise, and other subregular grammar classes. The class of languages they generate are determined by a function f which maps strings to finite subsets of A (chunks of grammars). Since the size of the canonical grammars is finite, a learner which develops a grammar on the basis of the observed forms and the function f identifies this class in the limit from positive data. Heinz called learners inducing these grammars String Extension Learners because each string in the language can be mapped or "extended" to an element of the grammar, which in every case, is conceived as a finite set of elements. Later, Heinz et al. (2012) generalized the finite subsets of the set A to be elements in a lattice, and showed that such "lattice-class" learners can be identified in the limit, and are incremental, globally consistent, locally conservative, and set-driven, and strongly monotonic, making them also learnable in the Probably Approximately Correct sense.

4.4 Learning *k*-Strictly Local and *k*-Strictly Piecewise Languages

The failure of Identification in the Limit frameworks to learn major classes of the Chomsky-Schuützenberger Hierarchy is a major negative result. Does this mean that natural languages are not learnable in this way, or that Gold-style learning is irrelevant for language? No. Recall the previous section's lesson that the class of natural languages are often structured according to specific properties. Just as the case with computable enumeration, further structuring the hypothesis class available to a learner often enables Identification in the Limit to succeed, particularly for classes which more closely fit the types of linguistic patterns. This is the case with the *k*-Strictly Piecewise and *k*-Strictly Local classes discussed in the previous section.

The Strictly Piecewise case is easy to show with an illustrative example. Let $\Sigma = \{a, b\}$ and consider the positive 2-Strictly Piecewise grammar consisting of the set of allowed subsequences $G = \{\lambda, a, b, aa, ab, ba\}$, with λ being the empty string. Then the language of the grammar $L(G) = \Sigma^* \backslash (\Sigma^* b \Sigma^* b \Sigma^*)$. This language allows any string which does not contain two b's even when they are arbitrarily far apart, just like the sibilant harmony example discussed in Section 3.

Intuitively, a successful learning algorithm over this language is straightforward. The learner, *a priori* possessing the concepts of the *k*-value of 2 and the notion of a subsequence, extracts the length-2 subsequences from each piece of evidence $e(t)$ presented to it, and successively updates their grammar $G(t)$ accordingly. Since for each Strictly *k*-Piecewise language there is a finitely sized grammar of *k*-subsequences that will identify the language, there is some point at which new evidence will not add any new subsequences to the grammar's current store for new presentations. Thus, it is guaranteed to Identify in the Limit the target *k*-Strictly Piecewise language from positive data. Additionally, this class is String Extension Learnable as mentioned above.

It is relatively easy to replace the subsequence concept with substring, so that the learner extracts the *k*-length substrings of a string, consistent with a *k*-Strictly Local grammar rather than Strictly Piecewise. In both cases the properties of the grammar constrain the type of the membership function the algorithm infers. The *k*-value is important, because it parameterizes the susbtrings/subsequences by some number k corresponding to the window size that the grammar uses to extract information from a presentation, such that the learner will eventually see all of them. It is for this reason that, while the Strictly Local and Strictly Piecewise classes are not generally identifiable in the limit from positive data, the *k*-Strictly Local and *k*-Strictly Piecewise classes are for any k. Thus, a specific set of precise constraints on the learner allows it to succeed, while other potential restrictions may not.

4.5 Cognitive Lessons from Learning Theory

The preceding section described how an unrestricted learner is unable to learn any language at all, a more specific case of the logical problem of induction. Restricting the hypothesis space available to a learner has measurable effects on the learner's ability. However, the type of restriction matters. As we saw, some restrictions, like the major regions of the Chomsky hierarchy, ensure that a learner in a particular framework will not succeed. In contrast, other restrictions do ensure success, some with more feasibility than others.

The takeaway is that theoretical choices matter, and evidence matters. Is there sufficient evidence to suggest that a learner's hypothesis space necessarily conforms to some class? If so, then a learning algorithm's failure on that class under some framework is evidence that the system is not learning in that way. Conversely, if a learner demonstrates a behavior of learning consistent with some learning framework, then perhaps the learner's hypothesis space is consistent with those that the algorithm is able to learn. Perhaps some facets of a learning framework (say positive and negative data) are present in a learner, but not others. In this case, it is up to the scientist to understand the precise relationship between the properties of the learner and of what is being learned. Understanding such precise relationships is the domain of computational learning theory. There are many many more learning frameworks than what is presented here, but in all cases the precision and analytic guarantees are the focus. Ignorance of the results of computational theories may not preclude language scientists from getting results, but its inclusion guarantees such results will be interpretable. The cognitive scientist studying learning of particular patterns would do well to understand these limits, as the next section demonstrates.

5 Testing Formal Predictions with Artificial Grammar Learning

Artificial Grammar Learning (AGL, also Artificial Language Learning) has been extensively used by linguists and cognitive scientists in general to investigate biases in the acquisition of complex patterns from input observations (Reber, 1967, 1969; Marcus et al., 1999; Gomez and Gerken, 1999; Gomez, 1997; Goodman, 1997; Ferman et al., 2009, a.o.). The core idea of an AGL experiment is to familiarize participants with an artificially constructed mini-language, representative of a specific pattern under study. As discussed above, assuming that the pattern of interest is characterized by a grammar G, the full set of strings presenting such a pattern over a controlled set of symbols (alphabet) can be called $L(G)$. Participants in AGL studies are presented with a finite subset of $L(G)$, after which they are tested on their ability to discriminate between string that present the pattern, and strings that do not (see Figure 6). While the specific details of the experimental paradigm vary significantly among studies and can be used to test different formal learning frameworks – depending, for instance, on whether explicit feedback is provided during an habituation phase, on what kind of observations

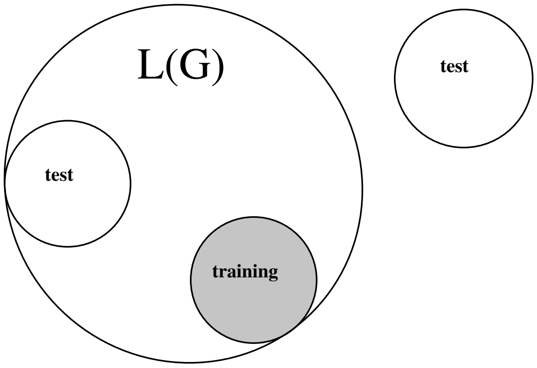


Fig. 6: Basic set up of an Artificial Grammar Learning Experiment.

are provided in input, or how the testing phase is set up — the general design aims at testing how the input influences generalization to novel strings, thus exploring the participants' ability to internalize patterns with specific properties.

A common objections to this approach to this paradigm is that the simplified set of data and the conditions of a laboratory experiment do not truly reflect the complexity of language acquisition in the real world, thus casting doubts on the usefulness of AGL results (Braine et al., 1990, a.o.). While the *ecological validity* of experimental results is an important concern, it is also true that this is the price paid by any attempt to investigate human behavior under controlled conditions (DeKeyser, 1997; Ferman et al., 2009). Once again however, having a theory laid down that is precise in its assumptions and in its predictions comes to help: when the experimental contrasts are well-motivated and situated in a formal learning framework, results of AGL experiments will always have something concrete to tell us about the way participants interacted with the stimuli. In this respect, the goal of this section is not to present an in-depth overview of the abundance of work exploring how the hypotheses of formal language theory can be explored in AGL settings, both at the level of the Chomsky hierarchy, and at the more detailed level of the subregular hierarchy (Koo and Callahan, 2012; Finley, 2012; Lai, 2015; Finley, 2011; Finley and Badecker, 2009; Finley, 2017; McMullin and Hansson, 2019; Avcu and Hestvik, 2020, and references therein). Instead, the aim is to illustrate how subregular characterizations can be used to improve the design of experiments concerned with investigating structural biases in language learning, while avoiding some common fallacies of these experiments (Rogers and Pullum, 2011; Jäger and Rogers, 2012; Rogers et al., 2013; De Santo, 2018; De Santo and Rawski, 2020; Wilson et al., 2020; Levelt, 2020; Uddén et al., 2020).

Learning Non-Local Dependencies: The Fallacy of Generalization. Section 3 already touched on the contrast between local and non-local dependencies between string symbols. This kind of contrast has been focus of a variety of studies in experimental linguistics, since long-distance relations are ubiquitous in human languages. One could, for instance, be interested in investigating the ability of adults to learn such dependencies compared to children at different stages of development. Or compare human learners to other animal species. It is possible to imagine a stimulus set up comparing, for instance, strings with two immediately adjacent *a*'s, to strings where the two *a*'s are separated by some other symbol. Assuming an alphabet $\Sigma = \{a,b,c,d,e\}$, the training samples could look like the following:

$L_{loc} = \{abcd, aabcd, baacd, bcaae, \dots\}$
$L_{dist} = \{abacd, bacad, bcada, bcaea, \dots\}$

Note however that the pattern exemplified in L_{dist} is technically still a *local* pattern: it can be easily captured by SL trigrams. Thus, if participants to such an experiment are then tested on novel strings that just put one single symbol in between two *a*'s, evidence of the fact that they are inferring non-local relations from the input would be fairly weak. The crucial point here, as in all AGL experiments, is to make sure to set up tests that allow the experimenter to evaluate whether the input is leading learner to generalize to the right kind of processes. In the case of long-distance relations, the core property to test is the fact that related symbols can be at an *arbitrary* distance from each other. Of course, there are practical limits to the way this can be tested in a laboratory setting. For instance, testing strings can not be arbitrary long, as length of the string is bound to have some effects on behavioral results that are unrelated to the property under test. However, it is still possible to target the core property being investigated by increasing the distance between relevant symbols by one or two extra element with respect to the training set.

$L_{dist} = \{abcad, abcad, bacda, abcea, \dots\}$

This set up is obviously still imperfect, and one could argue that each of the chosen examples is *still* a local patter by itself. Importantly though, testing over dependencies that are longer than the ones observed targets the participants' ability to generalize from a restricted input to strings containing the property of interest.

Refining the Question: What Kind of Long Distance Relation? The discussion of long-distance processes in Section 3 already showed how it is possible to use subregular characterizations to pinpoint fine-grained properties of the patterns under consideration. In the case of long-distance dependencies, for instance, it is then possible to ask not just whether learners are sensitive to non-local patterns, but what kind of relations characterize such patterns and not others. Recall the difference between TSL and SP languages, as outlined in before in the chapter: TSL languages capture non-locality by recasting adjacency over a subset of a language's whole alphabet, while SP languages rely on the precedence operator. Crucially, TSL and SP languages are *incomparable*: they do not stand in a precise subsumption relation with respect to each other. This means that, while there are some patterns that are both TSL and SP (e.g., the L_{dist} language from above), there are also patterns that are exclusively TSL, and patterns that are exclusively SP. The case of sibilant harmony with blocking already illustrated an exclusively TSL pattern: since SP languages rely of subsequences to discriminate between well-formed and ill-formed strings, they can not take blockers into account. For an exclusively SP pattern, consider the language L that consists of all strings over $\Sigma = \{a,b,c,d,e\}$ that do not contain the subsequences *ac* or *bd*. For instance, $cadeb, cdba, cddddbbbba$ are strings in L, but $abecd, abedc, addddbbbbc$ are not. This language is not TSL since, in order to correctly ban both *ac* and *bd*, *a*, *b*, *c*, and *d* should all be considered as tier members. But then, *a*, and *c* would act as blockers for the *bd* constraint, and vice-versa. Given this clear contrast, it should be possible to set up training sets with artificial languages that are both TSL and SP, only TSL, and only SP, and compare how each class affects the learning trajectories and classification errors of participants. In turn, this could shed light on preferences between adjacency or precedence as operators as the core of long-distance relations. An understanding of the distinctions between patterns through subregular lenses thus allows for a refinement of experimental questions even on sets on dependencies that would seem indistinguishable otherwise.

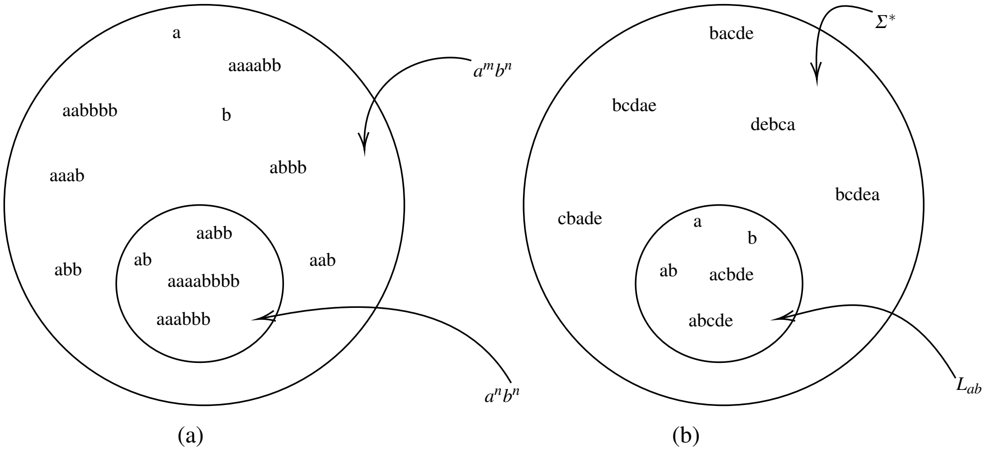


Fig. 7: Examples of set-subset relations for (a) the $a^n b^n$ vs $a^m b^n$ constrast; and (b) a free word order language (Σ^*) vs a language only allowing adjacent *ab* (L_{ab}) given $\Sigma = \{a,b,c,d,e\}$.

Two Cases of the Set/Subset Problem. The examples discussed so far should clarify how important it is to keep the relation between overlapping formal characterizations in mind when picking the training and test sets. In this sense, a common, crucial fallacy in the design of AGL experiments is related to what is commonly known as the *set/subset problem*. Assume that an experiment is set up to explore the ability of humans to learn context-free languages, and that the core of the experiment is the $a^n b^n$ languages. As the reader might recall, this is the language of strings containing an arbitrary number of *a*'s, followed by a matching number of *b*'s. It would seem reasonable to train subjects on a subset of this language, and then test that they are able to recognize well-formed strings in the language. However, consider now the (less-powerful) regular language

$a^m b^n$ — the language of strings containing an arbitrary number of a's, followed by an arbitrary number of b's. The reader might notice that strings in $a^n b^n$ are in fact a subset of the strings in $a^m b^n$ (see Figure 7a) — the context-free requirement being that a's and b's must match. Thus, just observing that a participant recognizes $a^n b^n$ strings as well-formed is not evidence of the fact that they internalized a context-free patterns. Setting up a contrast set that checks whether they correctly exclude $a^m b^n$ strings is also crucial. This is a case in which the set-subset issue can be addressed by being careful about the experimental design.

Let us consider a second case now. Assume an alphabet of symbols $\Sigma = \{a,b,c,d,e\}$, and a language L_{ab}, which only allows b after a (but not vice-versa) — thus, L_{ab} would contain strings like $ab, acb, abcde$. Imagine an experiment focusing on measuring participants' preferences for ordered versus unordered sequences (e.g., Nowak and Baggio, 2017), setting up as a contrast to L_{ab} a language where the order of the symbols is completely unrestrained (a *free order* language). What kind of strings would this language contain? Strings like $ba, bcdea, bacd$ would be part of the language but, since there are no constraints on the possible order of symbols, so would all the strings in L_{ab} (see Figure 7b). In fact, the unordered language is exactly Σ^* — the collection of all possible strings generated from Σ — and thus there is no set of strings based on Σ, that is not a *proper subset* of Σ^*. Therefore, while it is reasonable to train participants on strings from L_{ab} and test them n the contrast between L_{ab} and the larger Σ^*, the opposite (training on Σ^*) is by definition uninformative: Σ^* has no distinctive property that can be isolated in such a setting. In contrast to the previous case then, this example highlights an issue with the set/subset problem that cannot be resolved with a smart experimental design, but it instead ruled out directly following the theoretical formalizations. Note that this does not imply that there are no interesting questions to be asked about how participants would generalize over Σ^* (e.g., would they prefer to impose some kind of constraint over none?), but highlights how such questions would not immediately follow from the one that originally motivated the stimulus design.

Subsumption, and the Importance of Conservative Assumptions. As a specular issue to the one above, it is important to keep in mind that classes in these hierarchies stand in *subsumption* relations with respect to each other. For instance, every regular language is also a context free language, and every subregular pattern is trivially also a regular one. This means that one could be tempted to classify a TSL pattern in terms of its regular characterization. However, as discussed many times in this chapter, what one should strive to keep in mind in the study of opaque cognitive abilities is the distinction between possible and necessary. Focusing on the weakest class that captures a specific pattern fully allows for clarity about which primitives are fundamental in order to distinguish such pattern from others — by formalizing the minimum necessary requirements on transparent properties of the input — consistently with a theory that strives to make the least possible number of assumptions about unobserved mechanisms.

Balancing Measures of Complexity. Formal language theoretical notions are obviously not the only conceivable way to characterize linguistic complexity. A common alternative is to evaluate patterns (and analyses of such patterns) in terms of the succinctness of their descriptions (Rissanen, 1983; Grünwald, 1995; Vitányi and Li, 2000; Hansen and Yu, 2001; Grünwald et al., 2005). For instance, one could examine the way TSL and SP analyze sibilant harmony based on the dimension of the resulting grammars (e.g., measured in terms of banned tokens). While exploring how succinctness plays a role in human cognition is by itself a worthwhile enterprise, it seems important to highlight how it is not a property of the observed patterns themselves, but it once again requires assumptions about the mechanism that we use to encode such patterns — e.g., it is a measure imposed on the grammars, automata, or other means of representation. Crucially then, measures of succinctness are not in contrast with language theoretical characterizations of complexity, and could benefit from examining the functional primitives highlighted by, for instance, the subregular approach.

Relatedly, recall that the characterizations described in this chapter pose as an object of inquiry an upper-bound to the complexity of linguistic dependencies. This is important to remember when thinking about the intersection of multiple measures of complexity. As mentioned before, for instance, claiming that phonological dependencies are at most TSL does not imply that every possible TSL pattern should be attested (see Aksënova et al. (ming)). By characterizing a complexity upper bound based on what kind of operations are allowed though, this perspective then opens the road to a more informed exploration of other cognitively grounded restrictions on language learning (e.g., Heinz and Idsardi, 2013; Aksënova and Deshmukh, 2018).

6 Conclusion

The overarching lesson of this chapter is that mathematical characterizations further the study of language and cognition, by forcing theoretical frameworks to be specific, analytically transparent, and unambiguous. Approaching linguistic complexity through the lens of formal language theory allowed for a discussion of different classes in the Chomsky hierarchy, and the implicit assumptions that accompany them. Importantly, the focus on sub-regular classes served to clarify how alternative characterizations, while formally equivalent, can offer different kinds of insights into cognitive mechanisms. For instance, descriptive characterizations of string languages seem to be more useful in investigating the cognitive reality of complexity distinctions over grammar or automata characterizations, since they allow for the study of necessary properties that are tied to the string patterns themselves.

As discussed, formal language theory has been used extensively to guide the design and interpretation of Artificial Grammar Learning experiments. Importantly, in order to design learnability experiments successfully and avoid critical fallacies that would obfuscate their interpretability, it is crucial to understand the relation between formal characterizations of the linguistic phenomena under study, and the formal properties of a learning system. The detailed mathematical specifications provided by formal languages and learning theory make it possible to evaluate the validity of a cognitive question as framed in an experimental set-ups *a-priori*, before (and beyond what is allowed by) data collection (Planck, 1936; Guest and Martin, 2021). The formal rigor guaranteed by this approach makes it possible for formal theories to reach *explanatory* power, beyond their ability to make correct or incorrect predictions. For instance, AGL experiments have been used in the past not only to address questions about the relevance of computational distinctions to human (and non-human) language learning (Fitch and Hauser, 2004; Fitch et al., 2012; Öttl et al., 2015), but also to make claims about *possible* and *impossible* dependencies in natural languages (Moro et al., 2001, a.o.). What is that makes a language possible or impossible? The claim here is that answering this type of questions becomes possible possible when adopting a theoretical framework precisely formalized in all of its components.

Importantly, this chapter's focus on string languages is not meant to imply a disinterest in different types representations. For instance, tree languages are notoriously of crucial relevance to the study of human syntax, and formal language theoretical approaches have shed light on core properties of these representations (see Hunter, 2021, a.o. for a recent overview). As briefly mentioned before, representational assumptions interact in non trivial ways with complexity characterizations — so that, for instance, a CF string language corresponds to a SL_2 tree language. This is made even more complicated when focusing on human language, as hierarchical representations are assumed to be derived from string-like input and output (i.e. they are hidden in the data). That is to say, it would not be particularly insightful to probe what kind of trees humans learn for a specific set of sentences, by training and testing on trees themselves. Thus, if one is interested in studying how the hypothesis space of specific learners (e.g. humans) is constrained with respect to the computational expressivity of tree languages, it will be necessary to design contrast string languages that isolate a specific property of target tree grammars, while also controlling for possible confounds coming from the process of deriving trees from string (Fowlie, 2017). This remark is not meant to imply that it is impossible to derive insights about syntactic representations from AGL experiments (see, for instance, Culbertson, 2021), but that additional care has to be taken to disentangle complexity and representational biases within the learning process.

In this sense, the hope of this chapter is that providing readers with a detailed understanding of the mechanics of formal language theory and learning theory — and their relevance to the cognitive investigations into linguistic complexity — will further fruitful collaborations between cognitive scientists and mathematically inclined linguist and psychologist.

References

Aksënova, A. and Deshmukh, S. (2018). Formal restrictions on multiple tiers. In *Proceedings of the Society for Computation in Linguistics (SCiL) 2018*, pages 64–73.

Aksënova, A., Graf, T., and Moradi, S. (2016). Morphotactics as tier-based strictly local dependencies. In *Proceedings of SIGMorPhon 2016*.

Aksënova, A., Rawski, J., Graf, T., and Heinz, J. (forthcoming). The computational power of vowel harmony. In van der Hulst, H., editor, *Oxford Handbook of Vowel Harmony*. Oxford University Press. Under review.

Avcu, E. and Hestvik, A. (2020). Unlearnable phonotactics. *Glossa: a journal of general linguistics*, 5(1).

Bertinetto, P. M. and Loporcaro, M. (2005). The sound pattern of standard italian, as compared with the varieties spoken in florence, milan and rome. *Journal of the International Phonetic Association*, 35(2):131–151.

Berwick, R. (1985). *The acquisition of syntactic knowledge*. Cambridge, MA: MIT Press.

Boone, W. and Piccinini, G. (2016). Mechanistic abstraction. *Philosophy of Science*, 83(5):686–697.

Braine, M. D., Brody, R. E., Brooks, P. J., Sudhalter, V., Ross, J. A., Catalano, L., and Fisch, S. M. (1990). Exploring language acquisition in children with a miniature artificial language: Effects of item and pattern frequency, arbitrary subclasses, and correction. *Journal of memory and language*, 29(5):591–610.

Brenner, S. (2012). Life's code script. *Nature*, 482(7386):461–461.

Brzozowski, J. A. and Knast, R. (1978). The dot-depth hierarchy of star-free languages is infinite. *Journal of Computer and System Sciences*, 16(1):37–55.

Cantor, G. (1892). *Über eine elementare Frage der Mannigfaltigkeitslehre*. Druck und Verlag von Georg Reimer.

Carnap, R. (1928). Der logische aufbau der welt: Versuch einer konstitutionstheorie der begriffe. *Welt-Kreis, Berlin*.

Cauchy, A.-L. (1821). *Analyse Algebrique*. Debure Frères, Paris, France.

Chandlee, J. (2014). *Strictly Local Phonological Processes*. PhD thesis, University of Delaware.

Chesi, C. and Moro, A. (2014). Computational complexity in the brain. *Measuring grammatical complexity*, pages 264–280.

Chesi, C. and Moro, A. (2015). The subtle dependency between competence and performance. *Angel J. Gallego & Dennis Ott (Eds.)*, 50:33–45.

Chomsky, N. (1957). Syntactic structures.

Chomsky, N. (1959). On certain formal properties of grammars. *Information and control*, 2(2):137–167.

Chomsky, N. and Schützenberger, M. P. (1959). The algebraic theory of context-free languages. In *Studies in Logic and the Foundations of Mathematics*, volume 26, pages 118–161. Elsevier.

Church, A. (1936a). A note on the entscheidungsproblem. *Journal of Symbolic Logic*, 1(1):40–41.

Church, A. (1936b). An unsolvable problem of elementary number theory. *American Journal of Mathematics*, 58(2):345–363.

Church, A. (1937). *The Journal of Symbolic Logic*, 2(1):42–43.

Culbertson, J. (2021). Artificial language learning. Oxford Handbook of Experimental Syntax (to appear).

Cummins, R. (2000). How does it work? versus What are the laws?: Two conceptions of psychological explanation. *Explanation and cognition*, pages 117–144.

Danchin, A. (2008). Bacteria as computers making computers. *FEMS microbiology reviews*, 33(1):3–26.

De Santo, A. (2018). Commentary: developmental constraints on learning artificial grammars with fixed, flexible, and free word order. *Frontiers in psychology*, 9:276.

De Santo, A. and Graf, T. (2019). Structure sensitive tier projection: Applications and formal properties. In *International Conference on Formal Grammar*, pages 35–50. Springer.

De Santo, A. and Rawski, J. (2020). What can formal language theory do for animal cognition studies? *Royal Society open science*, 7(2):191772.

DeKeyser, R. M. (1997). Beyond explicit rule learning: Automatizing second language morphosyntax. *Studies in second language acquisition*, pages 195–221.

Eilenberg, S. (1974). *Automata, Languages, and Machines*. Academic Press, Inc.

Ferman, S., Olshtain, E., Schechtman, E., and Karni, A. (2009). The acquisition of a linguistic skill by adults: Procedural and declarative memory interact in the learning of an artificial morphological rule. *Journal of Neurolinguistics*, 22(4):384–412.

Finley, S. (2011). The privileged status of locality in consonant harmony. *Journal of Memory and Language*, 65:74–83.

Finley, S. (2012). Testing the limits of long-distance learning: Learning beyond the three-segment window. *Cognitive Science*, 36(4):740–756.

Finley, S. (2017). Locality and harmony: Perspectives from artificial grammar learning. *Language and Linguistics Compass*, 11(1):e12233.

Finley, S. and Badecker, W. (2009). Artificial language learning and feature-based generalization. *Journal of Memory and Language*, 61:423–437.

Fitch, W. and Hauser, M. (2004). Computational constraints on syntactic processing in nonhuman primates. *Science*, 303:377–380.

Fitch, W. T., Friederici, A. D., and Hagoort, P. (2012). Pattern perception and computational complexity: introduction to the special issue. *Phil. Trans. R. Soc. B*, 367:1925–1932.

Fodor, J. A. (1983). *The modularity of mind*. MIT press.

Fowlie, M. (2017). *Slaying the Great Green Dragon: Learning and modelling iterable ordered optional adjuncts*. PhD thesis, UCLA.

Frank, R. and Satta, G. (1998). Optimality theory and the generative complexity of constraint violability. *Comput. Linguist.*

Fu, J., Heinz, J., and Tanner, H. G. (2011). An algebraic characterization of strictly piecewise languages. In *International Conference on Theory and Applications of Models of Computation*, pages 252–263. Springer.

Gallistel, C. R. (1999). The replacement of general-purpose learning models with adaptively specialized learning modules. *The new cognitive neurosciences*, pages 1179–1191.

Gleitman, L. (1990). The structural sources of verb meanings. *Language Acquisition*, 1(1):3–55.

Gold, E. (1967). Language identification in the limit. *Information and Control*, 10:447–474.

Gomez, R. L. (1997). Transfer and complexity in artificial grammar learning. *Cognitive Psychology*, 33(2):154–207.

Gomez, R. L. and Gerken, L. (1999). Artificial grammar learning by 1-year-olds leads to specific and abstract knowledge. *Cognition*, 70(2):109–135.

Goodman, E. B. J. C. (1997). On the inseparability of grammar and the lexicon: Evidence from acquisition, aphasia and real-time processing. *Language and cognitive Processes*, 12(5-6):507–584.

Graf, T. (2016). The power of locality domains in phonology. Ms., Stony Brook University.

Graf, T. and De Santo, A. (2019). Sensing tree automata as a model of syntactic dependencies. In *Proceedings of the 16th Meeting on the Mathematics of Language*, pages 12–26, Toronto, Canada. Association for Computational Linguistics.

Griffiths, T., Kemp, C., and Tenenbaum, J. B. (2008). Bayesian models of cognition. In Sun, R., editor, *The Cambridge handbook of computational cognitive modeling*. Cambridge University Press.

Grünwald, P. (1995). A minimum description length approach to grammar inference. In *International Joint Conference on Artificial Intelligence*, pages 203–216. Springer.

Grünwald, P. D., Myung, I. J., and Pitt, M. A. (2005). *Advances in minimum description length: Theory and applications*. MIT press.

Guest, O. and Martin, A. E. (2021). How computational modeling can force theory building in psychological science. *Perspectives on Psychological Science*, page 1745691620970585. PMID: 33482070.

Hansen, M. H. and Yu, B. (2001). Model selection and the principle of minimum description length. *Journal of the American Statistical Association*, 96(454):746–774.

Hauser, M. D., Chomsky, N., and Fitch, W. T. (2002). The faculty of language: What is it, who has it, and how did it evolve? *Science*, 298:1569–1579.

Hayward, R. J. (1990). Notes on the Aari language. In Hayward, R. J., editor, *Omotic language studies*, pages 425–493. University of London, London, UK.

Heinz, J. (2010). String extension learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 897–906, Uppsala, Sweden. Association for Computational Linguistics.

Heinz, J. (2015). The computational nature of phonological generalizations. Ms., University of Delaware.

Heinz, J. (2016). Computational theories of learning and developmental psycholinguistics. In Lidz, J., Synder, W., and Pater, J., editors, *The Oxford Handbook of Developmental Linguistics*, chapter 27, pages 633–663. Oxford University Press, Oxford, UK.

Heinz, J. (2018). The computational nature of phonological generalizations. In Hyman, L. and Plank, F., editors, *Phonological Typology*, Phonetics and Phonology, chapter 5, pages 126–195. De Gruyter Mouton.

Heinz, J. and Idsardi, W. (2013). What complexity differences reveal about domains in language. *Topics in Cognitive Science*, 5(1):111–131.

Heinz, J., Kasprzik, A., and Kötzing, T. (2012). Learning with lattice-structured hypothesis spaces. *Theoretical Computer Science*, 457:111–127.

Heinz, J., Rawal, C., and Tanner, H. (2011). Tier-based strictly local constraints for phonology. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*, HLT '11, pages 58–64, Stroudsburg, PA, USA. Association for Computational Linguistics.

Heinz, J. and Rawski, J. (forth.). History of phonology: Learnability. In Dresher, E. and van der Hulst, H., editors, *Oxford Handbook of the History of Phonology*, chapter 32. Oxford University Press.

Hilbert, D. (1928). Die grundlagen der mathematik. In *Die Grundlagen der Mathematik*, pages 1–21. Springer.

Hopcroft, J. E., Motwani, R., and Ullman, J. D. (2001). Introduction to automata theory, languages, and computation. *Acm Sigact News*, 32(1):60–65.

Hornstein, N. and Lightfoot, D. (1981). *Explanation in linguistics. The logical problem of language acquisition*. Longman, London.

Hunter, T. (2021). The Chomsky Hierarchy. In *A Companion to Chomsky*, pages 74–95. Wiley Online Library.

Jäger, G. and Rogers, J. (2012). Formal language theory: Refining the chomsky hierarchy. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 367(1598):1956–1970.

James, W. (1910). The principles psychology vol 1.

Jardine, A. (2015). Computationally, tone is different. *Phonology*.

Kaplan, D. M. (2011). Explanation and description in computational neuroscience. *Synthese*, 183(3):339–373.

Kaplan, R. M. (1995). Three seductions of computational psycholinguistics. *Formal Issues in Lexical-Functional Grammar*, 47.

Kaplan, R. M. and Kay, M. (1994). Regular models of phonological rule systems. *Computational linguistics*, 20(3):331–378.

Karttunen, L., Kaplan, R. M., and Zaenen, A. (1992). Two-level morphology with composition. In *COLING 1992 Volume 1: The 14th International Conference on Computational Linguistics*.

Kleene, S. C. (1952). *Introduction to metamathematics*, volume 483. van Nostrand New York.

Kobele, G. M. (2006). *Generating Copies: An investigation into structural identity in language and grammar*. PhD thesis, University of California, Los Angeles.

Koo, H. and Callahan, L. (2012). Tier-adjacency is not a necessary condition for learning phonotactic dependencies. *Language and Cognitive Processes*, 27(10):1425–1432.

Lai, R. (2015). Learnable vs. unlearnable harmony patterns. *Linguistic Inquiry*, 46(3):425–451.

Lenneberg, E. (1967). *Biological foundations of language*. Wiley, Oxford, England.

Levelt, W. J. (2020). On empirical methodology, constraints, and hierarchy in artificial grammar learning. *Topics in cognitive science*, 12(3):942–956.

Levy, A. and Bechtel, W. (2013). Abstraction and the organization of mechanisms. *Philosophy of science*, 80(2):241–261.

Marcus, G. F., Vijayan, S., Rao, S. B., and Vishton, P. M. (1999). Rule learning by seven-month-old infants. *Science*, 283(5398):77–80.

Marr, D. (1982). *Vision: a computational investigation into the human representation and processing of visual information*. W.H. Freeman and Company.

Marr, D., Poggio, T., Hildreth, E. C., and Grimson, W. E. L. (1991). A computational theory of human stereo vision. In *From the Retina to the Neocortex*, pages 263–295. Springer.

Martin, A. and Baggio, G. (2019). Modelling meaning composition from formalism to mechanism. *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences*, 375(1791):20190298–20190298.

McCollum, A. G., Baković, E., Mai, A., and Meinhardt, E. (2020). Unbounded circumambient patterns in segmental phonology. *Phonology*, 37(2):215–255.

McCulloch, W. S. and Pitts, W. (1990). A logical calculus of the ideas immanent in nervous activity. *Bulletin of mathematical biology*, 52(1-2):99–115.

McGregor, D. C., Padovan, A., Georges, A., Krockenberger, A., Yoon, H.-J., and Youngentob, K. N. (2020). Genetic evidence supports three previously described species of greater glider, petauroides volans, p. minor, and p. armillatus. *Scientific reports*, 10(1):1–11.

McMullin, K. and Hansson, G. Ó. (2019). Inductive learning of locality relations in segmental phonology. *Laboratory Phonology: Journal of the Association for Laboratory Phonology*, 10(1).

McMullin, K. J. (2016). *Tier-based locality in long-distance phonotactics?: learnability and typology*. PhD thesis, University of British Columbia.

McNaughton, R. and Papert, S. (1971). *Counter-Free Automata*. MIT Press, Cambridge.

Miller, G. A. (2003). The cognitive revolution: a historical perspective. *Trends in cognitive sciences*, 7(3):141–144.

Mitchell, T. (2017). Key ideas in machine learning. In *Machine Learning: Second Edition*. (forthcoming) `http://www.cs.cmu.edu/~tom/mlbook/keyIdeas.pdf`.

Moro, A., Tettamanti, M., Perani, D., Donati, C., Cappa, S., and Fazio, F. (2001). Syntax and the brain: Disentangling grammar by selective anomalies. *NeuroImage*, 13(1):110–118.

Newell, A. (1973). You can't play 20 questions with nature and win: Projective comments on the papers of this symposium.

Niyogi, P. (2006). *The Computational Nature of Language Learning and Evolution*. Cambridge, MA: MIT Press.

Niyogi, P. and Berwick, R. (1996). A language learning model for finite parameter spaces. *Cognition*, 61:161–193.

Nowak, I. and Baggio, G. (2017). Developmental constraints on learning artificial grammars with fixed, flexible and free word order. *Frontiers in Psychology*, 8:1816.

Núñez, R., Allen, M., Gao, R., Rigoli, C. M., Relaford-Doyle, J., and Semenuks, A. (2019). What happened to cognitive science? *Nature Human Behaviour*, 3(8):782–791.

Nurse, P. (2008). Life, logic and information. *Nature*, 454(7203):424–426.

Osherson, D. and Weinstein, S. (1983). Formal learning theory. In Gazzaniga, M. and Miller, G., editors, *Handbook of Cognitive Neurology*. Plenum, New York.

Osherson, D., Weinstein, S., and Stob, M. (1986). *Systems that Learn*. MIT Press, Cambridge, MA.

Öttl, B., Jäger, G., and Kaup, B. (2015). Does formal complexity reflect cognitive complexity? investigating aspects of the chomsky hierarchy in an artificial language learning study. *PloS one*, 10(4):e0123059.

Piccinini, G. and Bahar, S. (2013). Neural computation and the computational theory of cognition. *Cognitive science*, 37(3):453–488.

Pin, J. E. (1986). *Varieties Of Formal Languages*. Plenum Publishing Co.

Planck, M. (1936). *The philosophy of physics*. W. W. Northon and Company Inc.

Post, E. L. (1944). Recursively enumerable sets of positive integers and their decision problems. *bulletin of the American Mathematical Society*, 50(5):284–316.

Putnam, H. (1967). Psychological predicates. *Art, Mind, and reliogion*, pages 37–48.

Pylyshyn, Z. W. (1984). *Computation and cognition*. Cambridge, MA: MIT press.

Rawski, J. and Heinz, J. (2019). No free lunch in linguistics or machine learning: Response to pater. *Language*, 95(1):e125–e135.

Reber, A. S. (1967). Implicit learning of artificial grammars. *Journal of verbal learning and verbal behavior*, 6(6):855–863.

Reber, A. S. (1969). Transfer of syntactic structure in synthetic languages. *Journal of Experimental Psychology*, 81(1):115.

Rissanen, J. (1983). A universal prior for integers and estimation by minimum description length. *The Annals of statistics*, pages 416–431.

Rogers, J. and Hauser, M. (2009). The use of formal languages in artificial language learning: a proposal for distinguishing the differences between human and nonhuman animal learners. In van der Hulst, H., editor, *Recursion and Human Language*, chapter 12, pages 213–232. De Gruyter Mouton, Berlin, Germany.

Rogers, J., Heinz, J., Fero, M., Hurst, J., Lambert, D., and Wibel, S. (2013). *Cognitive and Sub-regular Complexity*, chapter Formal Grammar, pages 90–108. Springer.

Rogers, J. and Pullum, G. K. (2011). Aural pattern recognition experiments and the subregular hierarchy. *Journal of Logic, Language and Information*, 20(3):329–342.

Rumelhart, D. E. and McClelland, J. L. (1986). On learning the past tenses of English verbs. In McClelland, J. and Rumelhart, D. E., editors, *Parallel Distributed Processing, volume 2*, pages 216–271. Cambridge MA: MIT Press.

Schmidhuber, J. (2002). The speed prior: a new simplicity measure yielding near-optimal computable predictions. In *International conference on computational learning theory*, pages 216–228. Springer.

Searls, D. B. (2002). The language of genes. *Nature*, 420(6912):211–217.

Shieber, S. M. (1985). Evidence against the context-freeness of natural language. *Linguistics and Philosophy*, 8(3):333–343.

Sprevak, M. (2010). Computation, individuation, and the received view on representation. *Studies in History and Philosophy of Science Part A*, 41(3):260–270.

Strother-Garcia, K. (2019). *Using model theory in phonology: a novel characterization of syllable structure and syllabification*. PhD thesis, University of Delaware.

Thomas, W. (1997). Handbook of formal languages, vol. 3. chapter Languages, Automata, and Logic, pages 389–455. Springer-Verlag New York, Inc., New York, NY, USA.

Turing, A. (1938). On computable numbers, with an application to the entscheidungs problem. a correction. *Proceedings of the London Mathematical Society*, 2(1):544–546.

Turing, A. M. (1937). On computable numbers, with an application to the entscheidungs problem. *Proceedings of the London mathematical society*, 2(1):230–265.

Turing, A. M. (1954). Solvable and unsolvable problems. *Science News*, 31:7–23.

Udden, J., Araujo, S., Forkstam, C., Ingvar, M., Hagoort, P., and Petersson, K. M. (2009). A matter of time: Implicit acquisition of recursive sequence structures. In Taatgen, N. A. and van Rijn, H., editors, *Proceedings of the 31st annual conference of the Cognitive Science Society*, pages 2444–2449, Austin, TX. Cognitive Science Society.

Uddén, J., de Jesus Dias Martins, M., Zuidema, W., and Tecumseh Fitch, W. (2020). Hierarchical structure in sequence processing: How to measure it and determine its neural implementation. *Topics in cognitive science*, 12(3):910–924.

Valiant, L. (2013). *Probably Approximately Correct: Nature's Algorithms for Learning and Prospering in a Complex World*. Basic Books (AZ).

van Rooij, I. and Baggio, G. (2021). Theory before the test: How to build high-verisimilitude explanatory theories in psychological science. *Perspect Psychol Sci.*

van Rooij, I. and Blokpoel, M. (2020). Formalizing verbal theories. *Social Psychology*.

Vitányi, P. M. and Li, M. (2000). Minimum description length induction, bayesianism, and kolmogorov complexity. *IEEE Transactions on information theory*, 46(2):446–464.

Vu, M. H. (2020). *A quantifier-based approach to NPI-licensing typology: Empirical and computational investigations*. PhD thesis, University of Delaware.

Weizenbaum, J. (1976). Computer power and human reason: From judgment to calculation.

Wexler, K. and Culicover, P. (1980). *Formal Principles of Language Acquisition*. MIT Press.

Whitehead, A. N. and Russell, B. (1912). *Principia mathematica*, volume 2. University Press.

Wilson, B., Spierings, M., Ravignani, A., Mueller, J. L., Mintz, T. H., Wijnen, F., Van der Kant, A., Smith, K., and Rey, A. (2020). Non-adjacent dependency learning in humans and other animals. *Topics in Cognitive Science*, 12(3):843–858.

Yang, C. (2013). Who's afraid of george kingsley zipf? or: Do children and chimps have language? *Significance*, 10(6):29–34.