Do humans use push-down stacks when learning or producing center-embedded structures?

Authors: Stephen Ferrigno[1,2], Samuel J. Cheyette[3], Susan Carey[2]

Affiliations:

[1] Department of Psychology, University of Wisconsin-Madison, Madison, WI

[2] Department of Psychology, Harvard University, Cambridge, MA

[3] Department of Brain and Cognitive Sciences, Massachusetts Institute of Technology, Cambridge, MA

*Correspondence to: sferrigno@wisc.edu

Abstract: Recursive tree-structures are ubiquitous in human mental life, structuring representations within many different cognitive domains, including language, music, mathematics, and logic. However, the representational and computational machinery used to represent and process these structures is unknown. Here, we use an artificial grammar learning task to test if adults can abstract center-embedded and cross-serial grammars and generalize beyond the level of embedding of the training sequences. We find that adults learn both grammars, and that the cross-serial grammar was easier to learn and produce than the matched center-embedded grammar. We test untrained generalizations to longer lengths and use item-by-item error patterns, on-line response times, and a Bayesian mixture model to test two possible representational structures that might underlie both grammars: stacks and queues. Contrary to widely held assumptions, we find no evidence that a stack-like architecture is used to generate center-embedded structures. Instead, we find that both center-embedded and cross-serial sequences are represented using an iterative queue. Items are stored in a first-in-first-out memory structure and then accessed via an iterative search over the list to generate the matched base pairs of a center-embedded or cross-serial structure. We discuss ways this work adds to our current understanding of the memory structures and computational processes used to represent lists and hierarchical sequences.

Keywords: Recursion; Center-embedded structures; Cross-serial structures; Artificial grammar; Sequence learning

1. Introduction

Human beings create recursive tree-structures that underlie language production and comprehension, as well as the representations of music, mathematics, logic, and perhaps many more domains as well. Nonetheless, we know very little about the representational or computational machinery the mind uses to build such structures.

The word "recursion" has been used to mean a variety of things. A *recursive operation* is an operation that includes a self-call (Lobina, 2011). In contrast, a *recursive structure* is property of a sequence or construction which has "a constituent that contains a constituent of the same kind" (Pinker & Jackendoff, 2005, p. 203). For example, in English you can embed the phrase "the dog chased the cat" into the phrase "the cat drank some milk", to create the sequence "the cat the dog chased drank some milk." which has a center-embedded recursive structure. Similarly, a recursive structure can be a property of a non-linguistic sequence. For example, the sequence $A_1A_2B_2B_1$ has a recursive structure; the constituent A_2B_2 is contained inside the constituent A_1__B_1. Importantly, there is no direct relation between recursive operations and recursive structures. Both iterative and recursive operations can create the same structures (including recursive structures) and any recursive process can be rewritten as an iterative process (Liu & Stoller, 1999). Here we focus on *recursive structures* and investigate the mental processes that generate them in the human mind.

We follow common terminology in characterizing recursive structures at various levels of abstraction:

Structure – A property of a particular sequence or string--example: center-embedded.

Grammar – Abstract characterization of the (potentially infinite) set of sequences with a common structure.

Language – The set of sequences that satisfy any given grammar.

Many thinkers have speculated that the capacity to represent center-embedded structures is uniquely human, and perhaps, that this capacity evolved in the context of the evolution of natural language (Berwick & Chomsky, 2016). To address the hypothesis of human uniqueness, researchers must develop non-linguistic tasks that might engage recursive structures and demonstrate that success on said tasks does in fact draw on abstract grammars that characterize those structures, as opposed to being solved in some other way. Specifically, just because an animal can produce or recognize a particular sequence does not mean that the animal has any representation of the underlying recursive structure. To test for the representation of recursive structures, one needs to not only test if participants can represent a given sequence but rather whether subjects can represent the commonalities between the sequences in the language (i.e., has represented the grammar). As reviewed below, many non-linguistic studies have required participants to distinguish novel sequences that comply with a target recursive structure grammar from those that do not, either by measures of responses to sequences that violate the grammar or by measures of success in generating novel sequences that respect the grammar. Some studies include tests of generalization to novel levels of embedding--e.g., training on center-embedded sequences with one level of embedding ($A_1A_2B_2B_1$) and testing generalization to sequences with 2 or more levels of embedding ($A_1A_2A_3B_3B_2B_1$ or $A_1A_2A_3A_4$ $B_4B_3B_2B_1$; Shin & Eberhard, 2015; McCoy et al., 2021).

1.1. Brief history of research on animals' and humans' representations of center-embedded nonlinguistic structures.

Early work investigating whether non-linguistic animals can represent center-embedded, recursive structures found mixed and inconclusive results because the possibility that the animals learned the sequences without representing the underlying recursive structure was not ruled out (see Ferrigno 2022, for review). For example, Fitch and Hauser (2004) habituated tamarin monkeys to novel auditory sequences that respected an A^nB^n center-embedded grammar (e.g., AAABBB, AABB, AAAABBBB). They found that the animals failed to detect violations (e.g., AABB***B*)**. In contrast, using a similar task, starlings could detect the violations (Gentner et al., 2006). However, follow up studies found that even human adults do not represent the center-embedded structure of the strings (Perruchet &

Rey, 2005). Instead, adults represent the sequences via a counting strategy: if the number of As equals the number of Bs, then it is allowed under the grammar the grammar (nA = nB; Corballis, 2007; Fitch & Friederici, 2012). Given this alternative representation of the sequence structure induced by human adults, the successes and failures found in animals are difficult to interpret.

For this reason, recent studies with non-human animals have tested their ability to learn *indexed* A^nB^n grammars, in which each unique A item is paired with a unique B item (Ferrigno et al., 2020; Jiang et al., 2018; Liao & Brecht et al., 2022). A center-embedded indexed A^nB^n grammar would generate sequences such as $A_1A_2A_3B_3B_2B_{1.}$ One recent study tested monkeys' ability to produce sequences with such a center-embedded grammar by requiring them to represent and produce the second half of spatial sequences in reverse order (Jiang et al., 2018; Malassis et al., 2020). Using a version of the Corsi block-tapping task (Corsi, 1972; Milner, 1971), monkeys were presented with six spatial locations on a screen and then saw a series of two or three sequential flashing dots in the spatial locations. Participants were then required to touch the locations indicated in the reverse order, finishing the second half of the sequence generated by the recursive, center-embedded indexed A^nB^n grammar. For example, after seeing locations 1, 2, and 3 light up, monkeys had to touch locations 3, 2, then 1. After extensive training (10,000-25,000 trials), monkeys could produce the second half of such sequences more often than chance.

Although this finding is consistent with the possibility that monkeys can represent center-embedded spatial sequences, it is unknown how the monkeys represented these sequences. One possibility is that this success was due to an associative mechanism with working memory constraints (Rey et al., 2012). Under this proposal, the monkeys could touch the second half of the sequence using the strength of their memory (e.g., they touch the location that was most recently emphasized, followed by the next most recently emphasized location). In fact, Rey et al. provided evidence that monkeys can use this type of strategy, consistent with the hypothesis that under some circumstances center-embedded sequences can be produced as a byproduct of associative learning and working memory constraints.

To rule out this associative strategy and to test recursive sequencing in a non-spatial domain, Ferrigno et al. (2020) used a non-linguistic sequence generation task in which participants were required to produce the entire center-embedded sequence ($A_1A_2B_2B_1$). Participants were monkeys, children (age 3.5-5 yrs.), US adults, and Tsimane' adults who had little formal education. The participants were shown arrays consisting of 4 brackets, such as } [{], randomly arrayed in 2D, and had to learn to touch them in the order: {, then [, then], then }. This sequence of touches satisfies a center-embedded structure: the base pair [ ] is embedded within the base pair { }. Of course, even monkeys can learn to order 4 arbitrary items in a sequencing task (e.g., first dog, then square, then moon, then table; Terrace, 2005). To establish that participants were not using this array specific sequence learning strategy, Ferrigno et al. (2020) tested generalization to novel arrays with different shaped open and closed brackets. They found that all populations of participants represented the abstract grammar, which they could spontaneously generalize to novel arrays that required the same level of embedding (i.e., one). The untrained generalization trials ruled out the possibility that participants succeeded on the basis of associative learning of transitional probabilities among specific pairs of items (representing how often a specific item follows another specific item), or solely the basis of ordinal positions - learning the sequential position in which specific items must be touched (see Ferrigno, 2022).

Thus, associative learning procedures defined over specific items cannot underlie these successes. However, the abstract grammar that was induced could be specific to only 4-item sequences with 1 level of embedding. A finite state grammar, deploying abstract variables, could represent 4-item sequences: $A_1A_2B_2B_1$, where A and B are fixed variables (open and closed), and the indexes 1, 2… represent particular brackets individuated by shape. This type of grammar underlies the Marcus style rule learning studies: habituated to pi pi la, mo mo zu, be be tu (A A B), infants dishabituate to (A B A) or (A B B; Marcus et al., 1999). Hochmann (2022) shows these grammars are instantiated in fixed length working memory models that represent the position of each abstract variable name in its place in the sequence. For that reason, we call such a grammar a "fixed length slot-filler grammar." The center-embedded structures in the Ferrigno et al. (2020) could well be generated by such a grammar. The signature of such a grammar is that it would not generalize to longer sequences that involve multiple levels of embedding. A fixed length slot-filler grammar relevant to the Ferrigno et al. (2020) study would be articulated in terms of one fixed variable (here A, B, corresponding to open and closed, and one open variable, the subscripts 1 and 2, representing distinct shapes, and the working memory representation would be an abstract representation of the *whole 4-item sequence;* namely $A_1A_2B_2B_1$).

The results from Ferrigno et al. (2020) on 4-item sequence generation converge with previous studies of adults learning center-embedded 4-item sequences (Bahlmann et al., 2008; Perruchet & Rey, 2005). In these previous studies, as in the Hauser and Fitch and the Marcus abstract variable studies, the stimuli were consonant-vowel syllables, and the task involved recognizing strings that violate the grammar after a few minutes worth of familiarization strings. After training on the 2-item base pairs (to learn which A goes with which B), humans adults represented the center-embedded sequences generated by at least the abstract fixed length 4-item and 6-item slot-filler grammars. They noticed violations not only when the number of A & B items were unequal, but also when the base-pairs were incorrectly matched ($A_1A_2B_1B_2$; Bahlmann et al., 2008). Similarly, Liao & Brecht et al. (2022) tested crows' ability to represent and generalize 4- and 6-item center-embedded structures to novel arrays of the same length and found marked success. This suggests that both humans and crows can represent the center-embedded sequences generated by at least the abstract fixed length 4-item and 6-item slot-filler grammars.

Follow up studies tested whether human participants could extrapolate learned grammars to greater depths of embedding (Shin & Eberhard, 2015; McCoy, et al., 2021). After training on sequences with one or two embeddings, $A_1A_2B_2B_1$ & $A_1A_2A_3B_3B_2B_1$, participants were then tested on their ability to evaluate strings with three embeddings: $A_1A_2A_3A_4B_4B_3B_2B_1$. These studies have found that some, but not all, participants extrapolate the trained center-embedded grammar to novel lengths. The successful discrimination of the novel sequence lengths rules out that the successful participants were representing the trained sequences with fixed length 4-item or 6-item slot-filler representations because such representations would not generalize to novel sequence lengths.

The present study addresses two issues. First, we seek unequivocal evidence that human adults do solve the $A_1A_2B_2B_1$ sequence production task of Ferrigno et al. (2020), by abstracting an A^nB^n grammar that generates center-embedded sequences of novel, untrained lengths. We train participants on 4-item sequences, test generalization to novel 4-item sequences, and then test spontaneous generalization to novel 6- and 8-item sequences involving 2 or 3 levels of embedding, respectively. Second, because the Ferrigno et al. (2020) task requires sequence production rather than allowable string recognition, it generates data on item-to-item production times. This allows us to explore the actual computational procedures underlying the on-line production sequences that satisfy indexed, center-embedded A^nB^n grammars.

1.2. The Chomsky Hierarchy, Stacks and Queues, Center-Embedded and Cross-Serial Grammars

In *Syntactic Structure* (1957), Chomsky characterized grammars of different levels of complexity, capable of generating all and only sequences with different underlying structural descriptions. He explored what level of complexity the grammar of natural language is likely to be. The first level was *finite automata* (no recursive rules). The second was *context free grammars*, which can easily generate center-embedded languages. If unbounded (i.e., extendable to arbitrarily many levels of embedding), context free grammars require more computational capacity than a finite automaton. They require a push-down stack or equivalent. A push-down stack is a data structure that stores an ordered sequence, A B C, with a computational architecture that can access that sequence from its end only. If you think of this structure as vertically organized, the items are stacked or “pushed” one on top of another, and then can be accessed and removed or “popped” only from the top. Items can only be accessed and read after being popped off of the stack. Clearly such a data structure/procedure can easily underlie the production of sequences with center-embedded structures (e.g., hold ABC in working memory stack, pop off from the top and immediately read each item popped to create ABC|CBA; see Fig. 1A). Simple center-embedded sequences of any length could be generated, so long as the capacity of the stack is unbounded (e.g.,$A_1A_2B_2B_1$ or $A_1A_2A_3B_3B_2B_1$). The third level in Chomsky’s hierarchy was *context sensitive* grammars, which require additional memory structure, relative to context free grammars, such as multiple push-down stacks. An example of a grammar that is context sensitive is a cross-serial grammar which generates such sequences as $A_1A_2A_3B_1B_2B_3$. As shown on Figure 1A, one could realize a cross-serial grammar using two stacks, but not with the single stack of a context free grammar. Thus, in terms of the Chomsky hierarchy, cross-serial grammars are more complex than center-embedded ones because they would require multiple stacks, or a memory structure that is more complex than a stack (e.g., the tape of a Turing machine). Chomsky argued that human language requires at least context sensitive grammars.

Chomsky’s work led to many studies exploring whether the relative complexity of grammars within the Chomsky hierarchy predicted greater difficulty in either discovering the grammar common to a set of sequences or in generating a given sequence once the grammar is known. Many of these studies compared center-embedded languages with cross-serial languages in both natural language and artificial grammar tasks (Bach et al., 1986; De Vries et al., 2011; De Vries et al., 2012; Öttl et al., 2015; Uddén et al., 2012). All of these studies found either no difference in difficulty between these two kinds of languages or marked differences in which the cross-serial languages, that require the more complex context sensitive grammars of the Chomsky Hierarchy, are *easier* both to learn and to execute than the less complex center-embedded languages easily generated by context free grammars.

Fig. 1

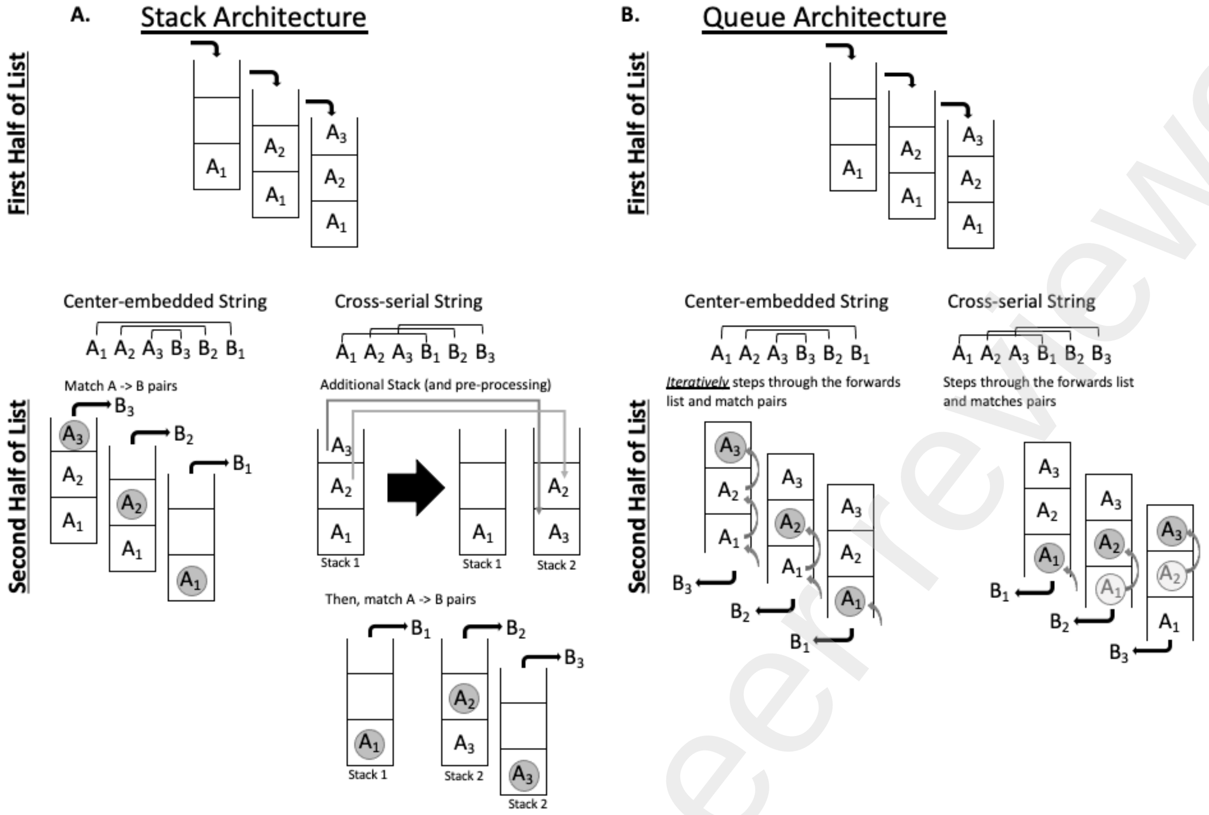
A.
Stack Architecture
First Half of List
A_1
A_2
A_3
Center-embedded String
A_1 A_2 A_3 B_3 B_2 B_1
Match A -> B pairs
B_3
B_2
B_1
Second Half of List
Cross-serial String
A_1 A_2 A_3 B_1 B_2 B_3
Additional Stack (and pre-processing)
Stack 1
Stack 2
Then, match A -> B pairs
B.
Queue Architecture
First Half of List
Center-embedded String
A_1 A_2 A_3 B_3 B_2 B_1
Iteratively steps through the forwards list and match pairs
Cross-serial String
A_1 A_2 A_3 B_1 B_2 B_3
Steps through the forwards list and matches pairs
Second Half of List

Figure 1. Proposed push-down stack and queue architectures for producing center-embedded and cross-serial strings. For both string types and architectures, the first half of the sequence is chosen then stored (pushed onto a stack or queue). (**A)** In the stack model, the second half of cross-serial sequences require more processing (and an additional stack) to produce than center-embedded sequences. **(B)** In the queue model, the second half of center-embedded sequences require more processing (iteratively searching through the queue) to produce than cross-serial sequences using a queue.

Of course, the Chomsky Hierarchy was not meant to be a model of on-line processing or ease of grammar induction. Stack architecture is merely another way of characterizing different levels of structural complexity. Therefore, it is an additional question whether the mind, like some computer programs, creates stacks that can be accessed in a last in, first out manner, and uses them to generate center-embedded recursive sequences. What is missing in this literature is an attempt to test the hypothesis that push-down stacks underlie the processes through which center-embedded sequences and/or cross-serial sequences are generated. Such an investigation requires contrasting stacks with other possible data structures/computational architectures and seeking processing signatures of each. One such architecture is a queue, or a representation of an ordered list which must be accessed from the beginning (see Fig.1B).

In fact, the literature on working memory establishes the viability of queues for representing ordered lists. It is well known that forward recall spans are much longer than backward recall spans (Anders & Lillyquist, 1971; Biretta et al., 2010; Haberlandt et al., 2005; Hurlstone, Hitch, & Baddeley, 2014). That is, repeating a list of digits, nonsense words, or familiar words, in the same order that they were presented (hear “3 8 6 2,” produce “3 8 6 2”) than repeating them in the backward order “2 6 8 3”. The whole sequences (model plus response) in forward recall tasks have cross-serial structure; the whole sequence is $A_1A_2A_3A_4B_1B_2B_3B_4$, (the As are the model; the Bs the response) whereas the whole sequences generated in backward recall tasks have center-embedded structure: $A_1A_2A_3A_4B_4B_3B_2B_1$.

One data structure/computational process that can be deployed in working memory tasks is an iterative loop over a queue (see Figure 1B; Anders & Lillyquist, 1971; Haberlandt et al., 2005; Hurlstone, Hitch, & Baddeley, 2014; Page & Norris 1998; Thomas, Milner & Haberlandt, 2003). A queue represents a sequence of items in order, first, second, third…, and can only be accessed from the beginning. To generate a backwards list, one must step through the queue to find the last item, then begin at the beginning again and step through the queue to find the next to the last item, and so on. This process explains the longer overall reaction times in the backward relative to the forward span task. It also explains the decreasing reaction times as a function of list position in the backwards version of the task shown in Figure 2. In contrast, for the forward span task, there is no iterative search process, instead one can just read the items off in the order of the queue, from front to end leading to relatively flat reaction times (Figure 2; Biretta et al., 2010).

Although the present artificial grammar learning task and the forward/backward span both involve producing lists, the tasks are very different. In the grammar learning task participants need to *induce* the underlying grammar. In contrast, in the memory tasks participants are given the first half of the list in order as a prompt and explicitly told the order to repeat the items back (e.g., “repeat these items in reverse order”). The present studies involve a grammar induction task in which the grammar must be learned from just a few examples, and in which the participant is required to order both the first and second half of the list, where this order can change from trial to trial based on the order the first half of the sequence was selected.

Fig. 2

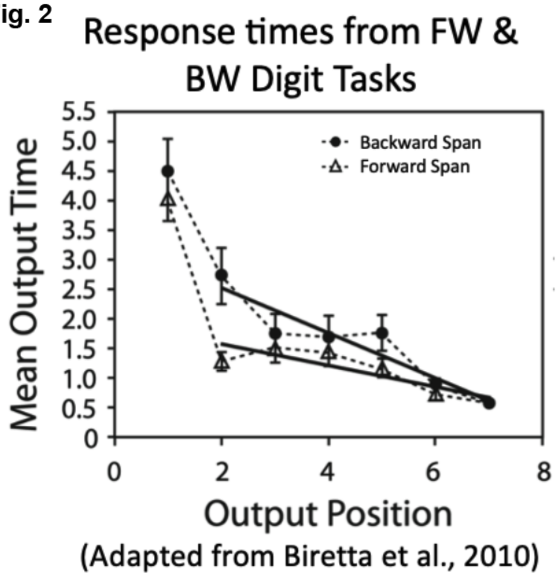

Figure 2. Response times showing the increased processing and a negative slope when using an iterative queue to produce backward orders in the second half of a backward span task (similar to the second half of a center-embedded sequence). Adapted from Biretta er al., 2010.

1.3. The Present Study

Our first goal is to test whether in the Ferrigno et al. (2020) sequence production task centered-embedded grammars are harder to learn, and the sequences in the language are harder to generate on-line, than are cross-serial grammars. That is, we seek to replicate the pattern observed in other artificial grammar learning studies (De Vries et al., 2011; Ottl et al., 2015; Uddén et al., 2009; Uddén et al., 2012). Our second goal is to extend the results of Ferrigno et al. in several ways. First, we test whether participants are actually learning indexed center-embedded, A^nB^n grammars that generalize beyond the level of embedding of the training sequences. Second, we include cross-serial grammars for direct comparison. Third we use stimuli other than open and closed brackets, which mathematically literate participants have had relevant experience but the other participant groups in Ferrigno et al. have not. Finally, we use error and response time data from sequence production to test whether one of two possible computational architectures, push-down stacks or queues, underlie the on-line generation of these center-embedded and cross-serial sequences.

The experiment (which was pre-registered at osf.io/6pe9h/) unfolds in several phases, each of which provides evidence to the participants about the grammar that describes what is common among the sequence of touches in their condition (cross-serial or center-embedded). In Phases 1 and 2, the participant is trained to correctly sequence two specific arrays and receives a first test for what grammar, if any, they have abstracted. Phase 3 then introduces entirely novel 4-item arrays that provides a stronger test for what grammar had been learned. Phase 3 then continues with completely novel arrays of lengths never seen before (i.e., a 6-item array and an 8-item array). The novel test arrays are never error corrected; any sequence order is accepted so it is possible to infer what description of the abstract structure of the sequences the participant has abstracted by that point in the experiment.

2. Methods

2.1. Participants

Participants were 100 adults (*mean age: 31, SD: 8.4, 68 male)*, 50 per condition. All participants were recruited from Amazon Mechanical Turk and paid $3 for participation. Procedures for recruitment and the experiment itself were approved by the Harvard University Research Subjects Review Board.

2.2. Procedure

Participants completed the entire task on their home computers while in full screen mode. The experiment began with 4-item arrays. Participants were instructed that the goal of the task was to click on images in the correct order and that they must learn this order through trial and error. Throughout the whole experiment, all training trials began by clicking the start stimulus, a small robot. Then, four pictures were displayed randomly distributed within a box outlined on their monitor (see Figure 3 for examples of arrays). When a picture was clicked, it gave auditory feedback (a ding) and visual feedback (briefly fading for .2 seconds) to cue that the click was registered. Throughout a trial, all items remained on the screen after being clicked, thus remaining selectable. Participants then moved onto the next choice until all items had been selected in the correct order, or until an error had been made. If an error was made, participants received negative auditory feedback (a buzz), a blank face image, and a 2s time out screen immediately after the incorrect touch to show that the touch was incorrect, before a new trial was initiated. A new array was then presented (with different spatial arrangements of the images, requiring the participant to press the start button again). If a participant completed the trial correctly (i.e., clicked on all items in the correct order), they heard positive auditory feedback (chimes) and saw a happy face to indicate that the trial was correct. After a correct trial, participants immediately moved onto the next trial.

Fig. 3

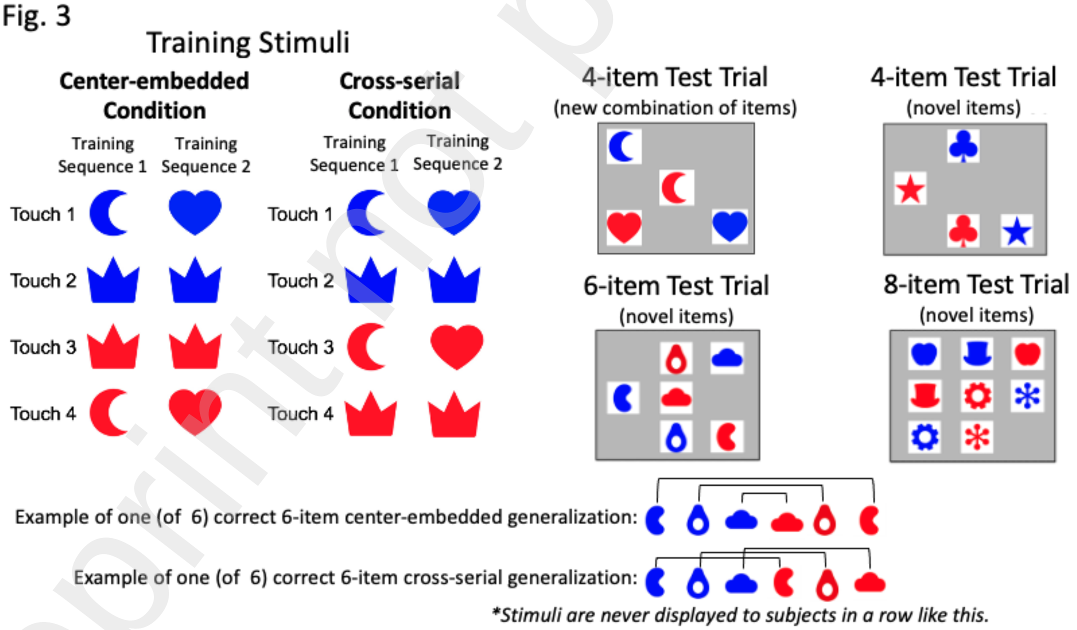

Figure 3. Examples of each trial type. The training stimuli (phase 1) consisted of two center-embedded or cross-serial lists depending on condition. After training, participants received the “4-item Test Trials” (Phase 2) which consisted of a novel combination of items from the training list that were presented without feedback. After phase 2, participants were given five trials of the 4-item Test Trials without reinforcement. Then were given additional training trials on the novel array (with feedback), before moving to the next array size. For each of the arrays, there are multiple correct answers. One correct response for each condition is illustrated for the 6-item array length.

2.3. Stimuli

The stimuli consisted of paired blue and red simple shapes (see Fig. 3). The color of the shapes determined the class within the indexed A^nB^n grammar (blue = A items, red = B items). The shape of the image determined the base pair unit (expressed by subscript values) within the indexed $A^n B^n$ grammar (e.g., in the first training sequence A_1 & B_1 were both moons, one blue and one red). These two dimensions (color & shape) allowed us to create the full indexed center-embedded (and cross-serial) sequences such as the center-embedded structure: $A_{1\ [\text{Blue Moon}]}$, $A_{2\ [\text{Blue Crown}]}$, $B_{2\ [\text{Red Crown}]}$, $B_{1\ [\text{Red Moon}]}$. Here the crown unit is embedded within the moon unit. In the second training sequence, a crown unit is embedded within a heart unit.

2.4. Conditions

Participants were randomly assigned to one of two conditions (center-embedded or cross-serial). In the center-embedded condition, participants were trained to click the images in a center-embedded sequence (e.g., $A_1A_2B_2B_1$, where the correct order of the B items was the reverse of the A item order, see Fig. 3 - Training Stimuli). In the cross-serial condition, participants were trained to click the images in a cross-serial sequence (e.g., $A_1A_2B_1B_2$, where the correct order of the B items was the same as the A items; see Fig. 3 Training Stimuli).

2.5. Phase 1: Initial Training Trials

Participants learned the correct sequence of touches for the first training array, followed by learning the sequence of touches on the second training array. There was one correct sequence of touches for each array. The order of touches of the blue items was arbitrarily decided and had to be discovered by trial and error, but the order of touches of the red items was computable from the structure of the sequences (center-embedded or cross-serial). Participants were required to correctly sequence the first array 4 out of 5 trials in a row before they moved onto the second initial training array. As can be seen on Figure 2, the second array included one base pair that was the same as in the first array (the crowns), and the blue item from this base-pair was the second item touched, i.e., in both sequences the second item touched was the blue crown. The purpose of having two arrays was to introduce participants to the idea that there was a common structure that applied to the correct sequencing of touches on different arrays.

2.6. Phase 2: 4-item test trials (novel combination of trained items, no feedback) combined with further training trials from Phase 1.

After completing both training arrays, participants moved to test trials with a novel array consisting of four items that had not appeared together in a single array before (the hearts and moons of Array 1 and Array 2, respectively; see Figure 2, 4-item Test). There were two correct sequences of touches on the novel test array (e.g., in the center-embedded condition: with center-embedded structure: "Blue Heart -> Blue Moon ->Red Moon ->Red Heart" OR "Blue Moon -> Blue Heart -> Red Heart -> Red Moon"), and there were similarly two correct sequences of touches with cross-serial structures.

One goal of these test trials was to rule out that participants had learned each of the 4-item arrays as two independent arbitrary sequences, not noticing the regularities due to color order or repetition orders of shapes. Research on list learning with monkeys shows that if taught two independent lists (A B C D) and (E B C F) and then tested on arrays that include A D E and F, monkeys produce the items that had come first in each of the previous lists first and those that had come last in each of the previous lists last, which would yield correct orders on the novel combination test arrays half of the time in each of the conditions. And having only learned the partial grammar (*touch blue before red)* would also predict half correct orders on the novel combination test arrays. Thus, comparing the proportion of correct sequences on the novel combination test arrays to 50% will assess whether participants memorized only the specific sequences of shapes for the two training arrays as well as whether they had abstracted a more complete grammar than merely the rule to touch blue shapes before red shapes.

Participants received positive feedback regardless of their accuracy on these test trials; that is, they were not corrected if they failed to produce a sequence that satisfied the center-embedded or cross-serial structures they had been trained on. Thus, these test trials didn't provide new evidence about the correct grammar, as the novel combination arrays were never differentially reinforced. The test trials were intermixed with further reinforced training trials from the original training arrays (Figure 2, Training Stimuli). These additional training trials were used to provide further training on the original arrays, to ensure that the original training was maintained, and to help participants remember the order within each of the trained arrays. Participants received 20 total trials (10 4-item novel combination test trials (no feedback) and 10 additional training trials, 5 using each training array).

2.7. Phase 3: 4-item, 6-item, & 8-item test trials (test arrays with completely novel stimuli (see Figure 2).

Over Phases 1 and 2, participants had received error feedback on only two sequences that satisfied each grammar–the training sequences shown on Figure 3. In Phase 3 participants were tested with arrays constructed from stimuli they had never seen before. The images had the same color structure (blue and red), with color determining the ordering within a unit and a unit consisting of two identical shaped pairs as before (e.g., A_1 B_1; see Figure 2). These shapes were new. The novel 4-item array test trials assessed whether participants had induced a grammar that abstracted beyond the two training arrays. They tested for, at least, a fixed length abstract variable slot-filler grammar, or equivalently, a grammar like *first all the blues, then all the reds; make the first and last item the same shapes*. Grammars of these sorts could create either center-embedded or cross-serial structures for 4-item sequences but would not generalize accurate sequencing to the whole 6- or 8-item sequences.

The 6- and 8-items arrays were designed to test whether participants had learned abstract A^nB^n center-embedded and cross-serial grammars that generalized to additional levels of embeddings. There were many different correct sequences for these 6-item and 8-item sequences trials. Participants had to first decide on an order to touch the blue stimuli (A items) and then had to implement a procedure that produced either the reverse of that order (center-embedded condition) or a repeat of that order (cross-serial condition). Any sequence that was correctly center-embedded or cross-serial (depending on the condition) was treated as correct, regardless of which blue items were first, second, or third.

For each array size in phase 3, participants first received 5 non-differentially reinforced test trials with never before seen shapes. Then, starting on trial 6 for each novel array, participants received additional training trials (with feedback) on the *same* novel array. These feedback trials were used to test if participants could learn the correct sequence structure for that particular array, even if they had not done so already, as well as allowing us to measure the difficulty of implementing each kind of structured sequence. These trials thus provided the first new evidence beyond that received on the training arrays as to what the target grammars were. Participants received a total of 25 4-item trials (5 with no feedback and then 20 training trials with error correction) and 45 6- and 8-item trials (5 with no feedback and then 40 training trials with error correction for each array size). The error correction was the same as in the initial training, except that there was no constraint on the order of the initial blue items. That is, if participants did not touch 4 unique blue items first (i.e., if they included a red item in the first 4 touched, or touched a particular blue item twice), or if they touched a single red item out of order according to their condition, they heard the error signal and had a 2 second time out before the next trial began.

2.8. Exclusion Criteria:

As specified in the preregistration, we excluded outliers for each analysis (individuated by condition, unreinforced generalization test trial vs. reinforced training trial, and array size). No participants were excluded from the entire study. Instead, participants' responses were excluded from specific analyses based on preset criteria. See supplemental materials for the preset exclusion criteria, as well as the number of participants whose data were excluded from each analysis. This allowed us to include as many participants as possible while removing any outlying data points that would drastically increase variability and noise in the data. As specified in the preregistration, we also excluded some specific trials if there were indications of inattention within that trial (trials where the same item was pressed >2 times). As expected, the exclusion of data as described in the SI decreased noise. The SI also reports qualitatively similar results to those reported here when no outliers were removed.

3. Results

3.1. Phase 1: Training Results:

It was equally easy for participants in each condition (center-embedded, cross-serial) to learn to touch in sequence the individual items of two different four-item training arrays. On average participants took 14.1 (SD = 9.2) trials to reach the criterion on the first array, and 6.1 trials (SD = 2.4) to criterion on the second array. An ANOVA examined the effects of array order (first, second) and condition (center-embedded, cross-serial) on trials to criterion. There was a main effect of array order, with faster mastery of the correct sequence of touches on the second array (array order: F[1, 172] = 63.48, *p* < .001]. There was no main effect of condition (F[1, 172] = 1.48, *p* = .23) or interaction between condition and array order [condition*array order: F(1, 172) = .001, *p* = .97]. The improvement from first to second array could be due to having represented the underlying structure in the first array. Alternatively, the four-item sequences in both conditions might have been learned as arbitrary sequences (A B C D) and (E B C F), where B and C were blue and red crowns) without extracting the underlying structure. Associative models of arbitrary sequence learning (e.g., Terrace, 2005) could accommodate the effect of array order, since two specific items were identical, and in the same sequencing position, across the two training arrays. The novel combination test trials adjudicate between these two possibilities.

3.2 Phase 2: 4 item test trials, novel combination of base pairs

The percentage of completely correct sequences in the test trials was compared to that expected from random responding (8.3%), and also to what would be expected if participants had merely learned the generalization “touch blue, and then touch red” (50%) or associative average position merging of the two lists (50%). Participants performed much better then even the latter rules: Center-embedded: 77% correct sequences, *t*(49) = 5.76, *p* <. 001; Cross-serial: 91% correct sequences, *t*(49) = 16.05, *p* <. 001, see Figure 4). Participants did not represent the training arrays as unique arbitrary sequences, nor did they learn only the rule that blue stimuli should be touched before red ones. Instead, they represented the sequences with a grammar that was abstract enough to produce correct sequences from novel arrays that combined items from distinct training arrays. While participants in both conditions successfully sequenced the novel test arrays, performance was significantly better in the cross-serial condition than in the center-embedded condition (t-test, 2-tailed: *t* = 2.61, *p* = .011). The cross-serial structure was easier to learn, or on-line implementation of the cross-serial order sequences was easier (and thus led to fewer errors), or both.

Fig. 4

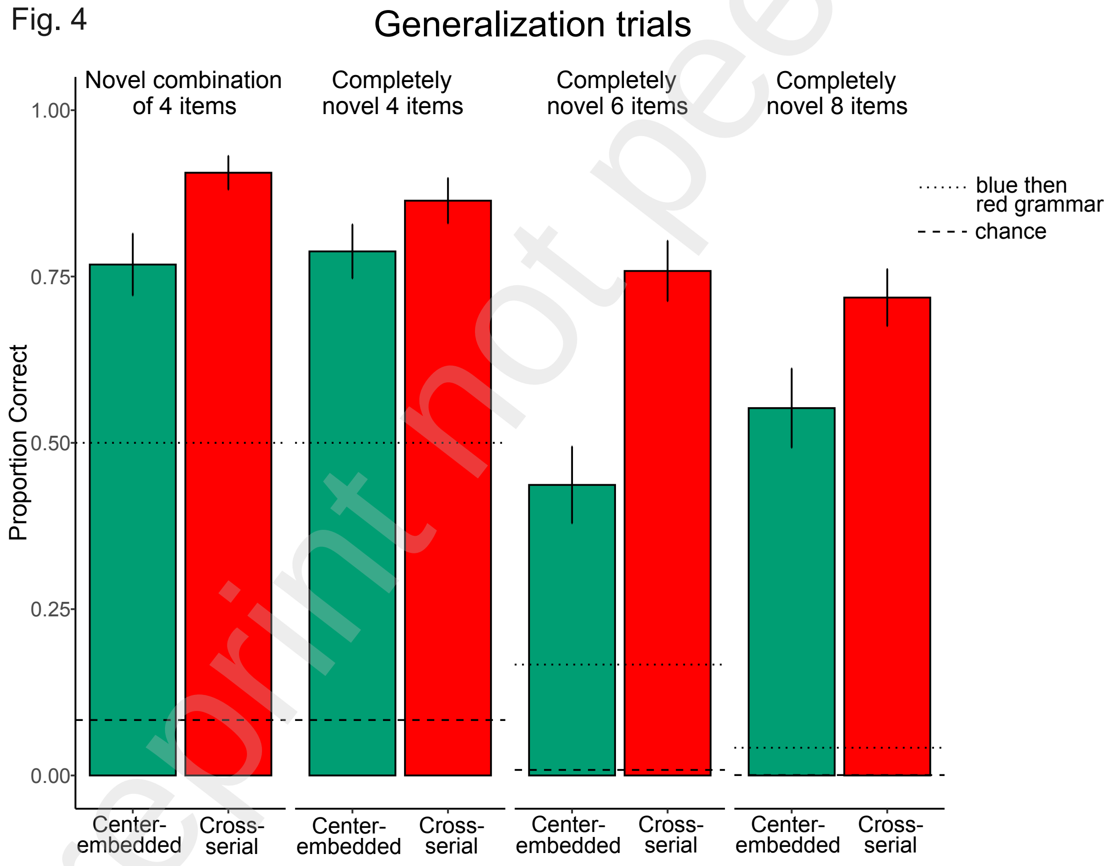

Figure 4. The proportion of fully correct sequences in the 4-item test trials from Phase 2 (novel combination of base pairs from the training arrays) and Phase 3 (completely novel items) in each condition and array length. Error bars represent the standard error of the mean. The dotted line represents predicted accuracy from a blue then red grammar (i.e., disregarding the base pairs) and the dashed line represents chance (random responding).

As in Ferrigno et al., (2020) which used brackets as stimuli (e.g., "{ [ ] }"), participants were able to generalize the 4-item sequences to novel combinations of items. Success here suggests that the excellent performance of the US adults in the Ferrigno et al. (2020) paradigm was not just due to having experience with brackets in mathematical, logical, or linguistics formalisms.

3.3 Phase 3: Test trials. Completely novel 4-item, 6-item, & 8-item arrays

The data on the Phase 2 test trials show that from experience with just two training arrays, arrays that had some common shapes in the same positions at that, was sufficient for adults to abstract a grammar that went beyond *touch blue shapes before red shapes.* Phase 3 test trials tested whether participants could generate correct sequences of touches on arrays that contained red and blue items with shapes not used in training (see Figure 3: star and club for 4-item arrays; bean, avocado, cloud for 6-item arrays; apple, hat, snowflake, gear for 8-item arrays). No associative strategy operating over the two memorized lists that were reinforced during training would generalize to complete sequences using novel shapes.

Novel 4-item array test trials.

These test trials tested whether the very minimal training in Phases 1 and 2 led to grammars that were at least equivalent to a fixed length a slot-filler structure stated in abstract variables. If subjects have abstracted a grammar that generates the relevant 4-item structures, performance on the 4-item novel array test trials (involving blue and red stars and clubs) at least should be above 50% as we found in the 4-item novel combination test trials of Phase 2. As seen on Fig. 4, which displays the proportion of correct sequences in the 5 unreinforced (i.e., no error feedback) test trials, this is what was observed. Participants in both conditions produced correct sequences on the novel 4-item arrays more than would be expected by chance, and more than would be expected if they had only learned that they should touch blue items before red ones, which would generate correct sequences 50% of the time in each condition (blue then red strategy = 50%, Center-embedded: 79%, $t(48) = 7.09$, $p <. 001$; Cross-serial: 86%, $t(45) =$

10.70, *p* <. 001).

Novel 6-item array test trials

Performance on the 5 novel 6-item arrays with no error correction assessed whether the grammar participants in each condition had abstracted by the end of training trials of the new 4-item list from Phase 3 would generalize to an additional level of embedding. This 6-item array was the first array of this length the subjects had encountered in the experiment.

As seen in Figure 4, performance in both conditions was better than would be expected by chance on the 6-item novel test trials, and better than would be expected if participants had learned only that they should touch blue items before red items (17% correct): Center-embedded: 44%, *t*(48) = 4.69, *p* <. 001; Cross-serial: 76%, *t*(47) = 13.05, *p* <. 001). At least some participants, but certainly not all, had learned a grammar from the training on 4-item arrays that generalized to one further level of embedding. This is a conceptual replication of the findings of Shin & Eberhard (2015) and McCoy, et al., (2021).

The great decrease in correct sequencing from the 4-item novel array test trials to the 6-item novel array test trials shows that not all participants generalized the grammars acquired during the 4-item training to new levels of embedding. This drop would be expected if some participants had learned only fixed length abstract grammars from their 4-item training. As shown in the logistic regression reported below, this was true for both conditions, but significantly more so in the center-embedded condition than in the cross-serial condition.

Novel 8-item array test trials

Similarly, the novel 8-item array was the first array of this size participants had ever encountered. Performance on the five unreinforced test trials assessed whether participants had learned grammars that would generalize to an additional level of embedding at least after training with both 4-item (1 level of embedding) and 6-item (2 levels of embedding) sequences. As seen in Figure 4, performance in both conditions was better than would be expected if participants had learned only that they should touch blue items before red items (4% correct): 8-item arrays: Center-embedded: 55%, $t(44) = 8.58$, $p <. 001$; Cross-serial: 72%, $t(48) = 15.80$, $p <. 001$). Furthermore, the drop in performance observed between 4- and 6-item arrays was not seen between 6- and 8-item arrays, in either condition (see regression analysis below). This suggests that the success on 6-item arrays was not due to having learned new fixed length slot filler grammars, but was subserved by A^nB^n grammars that generalized to a new level of embedding.

Preregistered logistic regression

A single preregistered logistic regression examined the effects of condition (center-embedded vs. cross-serial), array size (4-item, 6-item, 8-item), the interactions between these variables, and a random effects term of participant, on the proportion of sequences produced with the full target structure during the test trials that had no feedback. See SI for complete model details. The goal of this analysis was to explore whether cross-serial grammars were easier to learn than center-embedded grammars, as would be expected if participants used queues rather than stacks to implement the indexed A^nB^n grammars that were eventually induced. To assess the difference between conditions, we ran a pairwise comparison between the center-embedded and cross-serial conditions, collapsed across array length. Overall, participants in the cross-serial condition were more likely to give the correct response compared to those in the center-embedded condition ($\beta_{\text{condition}} = -1.13$, $p < .004$). To assess whether there was a cost of increasing array length, we ran pairwise comparisons between each of the three array sizes, We found significantly worse performance on 6-item than 4-item arrays ($\beta_{\text{4-item vs. 6-item}} = 1.76$, $p < .001$) and between 8-item than 4-item arrays ($\beta_{\text{4-item vs. 8-item}} = 1.46$, $p < .001$), and similar performance between the 8-item arrays than 6-item arrays ($\beta_{\text{6-item vs. 8-item}} = -.29$, $p = .18$). This suggests that not all participants spontaneously generalized from 4- to 6-item arrays, and that after training on 6-item arrays, at least some participants had abstracted a grammar that generalized to an additional level of embedding.

Furthermore, we found a significant interaction between condition and array size in the 4- vs. 6-

item array size comparison ($\beta_{\text{condition * 4-item vs. 6-item}}$ = 1.66, $p < .001$), but no significant interaction between condition and array size in the 4- vs. 8-item array size comparison ($\beta_{\text{condition * 4-item vs. 8-item}}$ = .76, $p = .077$). That is, when the array size increased from 4- to 6-items, performance decreased more in the center-embedded condition relative to the cross-serial condition (see Figure 4). Performance fell drastically from the 4-item array (79% correct) to 6-item array (44% correct), in the center-embedded condition. To test if this difference was significant and compare it to changes in the cross-serial condition and the 6- to 8-item change, we ran a series of post-hoc pairwise comparisons. We found that there was a significant decrease in correct sequences between the 4- and 6-item length in the center-embedded condition ($\beta_{\text{Center-embedded: 4-item vs. 6-item}}$ = 2.59, $p < .001$). Although there was also a significant decrease in the cross-serial condition, the effect size was about a third the size ($\beta_{\text{Cross-serial 4-item vs. 6-item}}$ = .93, $p = .005$; center-embedded: a change of -.35 probability of producing a center-embedded sequence; cross-serial: -.11 probability of producing a cross-serial sequence). This decrease was significantly less than in the center-embedded condition (β = 1.66, $p < .001$). That performance fell from the 4-item novel arrays to 6-item novel arrays in both conditions is consistent with the possibility that some participants may have initially represented the structures in the 4-item arrays in terms of a fixed length slot-filler grammar, and the interaction between condition and array size shows that this was more likely in the center-embedded condition.

To assess whether further training on the novel 6-item arrays led to indexed A^nB^n grammars that generalized to an additional level of embedding (i.e., to 8-item sequences), we compared the initial success on the 8-item arrays to the initial success on the 6-item arrays. We found significant *increase* in performance in the center-embedded ($\beta_{\text{Center-embedded 6-item vs. 8-item}}$ = -.79, $p = .01$) and no difference in the cross-serial condition ($\beta_{\text{Cross-serial 6-item vs. 8-item}}$ = .20, $p = .95$). This is consistent with some participants in the center-embedded condition needing additional training on a 6-item list to abstract a length independent center-embedded grammar, whereas those in the cross-serial condition were more likely to have done so by the end of the 4-item training.

As an exploratory analysis, we analyzed the errors produced during the 5 test trials with no error feedback in each condition. In the 6- and 8-item center-embedded trials, we found a large portion of errors where all “A” items are pressed first and “B” items second and a mismatched pairing of the base items (e.g., $A_1A_2A_3B_2B_1B_3$). Although these types of errors were also produced in the cross-serial condition they were much less frequent (6-item array: center-embedded: 38%, cross-serial: 13%, 8-item array: center-embedded: 32%, cross-serial: 16%). Furthermore, in the center-embedded condition, there were also many responses that were fully cross-serial orders (6-item: 16% crossed and 8-item: 12% crossed), but the reverse was not seen in the cross-serial condition (<1% in both 6- and 8-item conditions).

One interpretation of the entire pattern of results is that for *some participants*, especially in the center-embedded condition, further learning of the structure underlying the correct sequence of touches is required to generalize those structures to 6- and 8-item arrays. Suppose for some participants the abstract structure learned in the 4-item training trials in the center-embedded structure was $\text{Blue}_{\text{Shape1}}\text{Blue}_{\text{Shape2}}\text{Red}_{\text{Shape2}}\text{Red}_{\text{Shape1}}$--the slot-filler, abstract variable, representation. This does not generalize to the 6 item or 8 item lists. The learner could then analyze this representation to seek hypotheses relevant to the larger arrays. A first obvious hypothesis is that blues proceed red. That hypothesis is reflected in the high level of sequences that satisfy only this generalization in the center-embedded novel array 6- and 8-item test trials. A second obvious hypothesis is that there is something systematic about the order of the blues that determines the order of the reds, and the cross-serial systematicity is a more salient hypothesis than the center-embedded one. This is reflected in the high level of fully cross-serial sequences among participants on the center-embedded novel array (6- and 8-item) test trials.

The data presented so far are consistent with the hypothesis that the center-embedded grammar is more difficult to learn. Being based on generalization error data alone, they are also consistent with the hypothesis that even after the unbounded indexed center-embedded and cross-serial grammars are learned, the on-line process used to generate center-embedded sequences is harder to execute that for center-embedded sequences. In section 3.4 we turn to the reaction time data to explore the latter hypothesis. In section 4 we will present a computational model that separates errors due to not having learned the grammar from errors due to difficulty producing the sequences.

3.4. Response time analyses (correct sequences in the further training trials: 4-, 6-, and 8-item arrays)

The above analyses reveal that participants made more generalization errors in the center-embedded grammar than the cross-serial grammar. If the above results reflect differences in the difficulty of generating each structure, we should observe reaction time differences that persist once the correct structures are learned. Such a finding would suggest that queues are used to produce both sequence types. However, it is also possible that queues are used to produce cross-serial structures and stacks are used to produce center-embedded structures and that stacks are harder to use than queues.

We can adjudicate between these two possibilities comparing item-by-item response times between successive touches on the second half of the list (the red items) in the 6- and 8 item sequences. A stack memory structure would predict flat response times for the second half of center-embedded sequences and a long delay for the first item in the second half of cross-serial sequences. It would also predict overall longer response times for the cross-serial sequences, because the items would have to be put into a second stack before they could begin to be read off. In contrast a queue-like memory would predict a negative slope for the second half of center-embedded sequences, a relatively flatter slope for cross-serial sequences, and overall longer response times for center-embedded sequences.

We only included correct trials and excluded the first item in the second half of the list (e.g., item 4 in the 6-item list) because these response times could reflect the time to start producing the second half of the list (e.g., the time to switch from pushing items onto a stack to popping them, or the time to begin searching through a queue). Using a preregistered mixed effects linear regression we found a main effect of condition, such that overall, the second half of the center-embedded embedded sequences took longer to produce than cross-serial sequences in both the 6- and 8- item arrays (6-item: $\beta_{condition}$= -87.7, *p* = .03; 8-item: $\beta_{condition}$= -126.3, *p* = .02, see SI for full model description and results). We also found a main effect of touch number, such that in the base condition (center-embedded) response time decreased for successive touches as each trial progressed (6-item: β_{touch}= -68.4, *p* = .03; 8-item: β_{touch}= -185.7, *p* < .001). Lastly, we also found a significant interaction between condition and touch number such that the center-embedded condition had a steeper slope compared to the cross-serial condition for both sequence lengths (see Figure 5; linear regression: six-item array: $\beta_{condition*touch}$ = 50.22, *p* < .001; eight-item array: $\beta_{condition*touch}$ = 84.47, *p* = .001). This negative slope is a signature of an iterative queue in which items are responded to as soon as they can be accessed (Thomas et al., 2013).

These item-by-item response time results (as well as the overall response time results, see SI) suggest that participants use a queue to represent both center-embedded and cross-serial sequences and are inconsistent with the possibility that participants use different memory structures (stacks for center-embedded and queues for cross-serial) to represent these sequences.

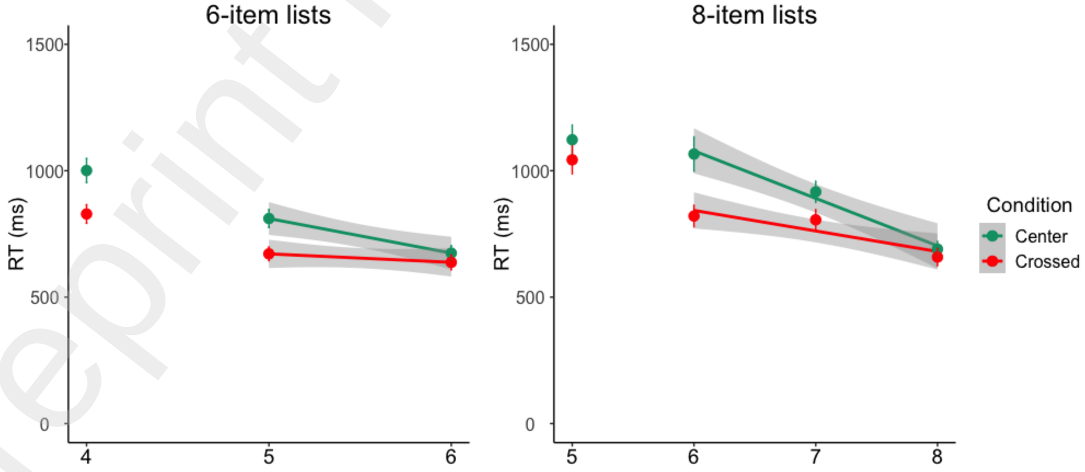

Figure 5: Response Times between touches in the second half of the sequences. That is, mean RTs (points) and model estimates (lines) between 3rd touch and 4th, the 4th and 5th, and the 5th and 6th on the 6-item sequence

list, and mean RTs (points) and model estimates (lines) between the 4th and 5th, 5th and 6th, 6th and 7th, and 7th and 8th touches on the 8-item sequence. Error bars represent the SE of the mean, and the shaded region represents the 95% confidence interval of the model.

4. Computational Models

A central premise of this paper is that it should be possible to infer the structure of memory based on how quickly and accurately people are able to generate sequences from different grammars. Specifically, if memory stores data like a stack, people should be faster and more accurate in generating center-embedded relative to crossed-serial sequences; conversely, if memory stores data like a queue, people should be faster and more accurate in generating crossed-serial relative to center-embedded sequences. In addition, if memory stores data like a queue, the decrease in item-to-item RTs over the second half of the list should be steeper in the center-embedded condition. And indeed, the previous analyses have demonstrated all of these predictions are born out.

Here we develop two memory models corresponding to the two different possible memory structures – a stack and a queue – and fit both models to the experimental data. The data we model here are those from the further training trials on 4-, 6-, and 8-item arrays. After the 5 uncorrected test trials discussed above (Figure 3), participants received 20 additional training trials on the same novel arrays, where an error led to the end of the trial (20 at length 4 and 40 at lengths 6 and 8). Here we analyze data from both correct and incorrect sequences. No analyses from incorrect sequences were included in the results presented so far, so these analyses provide new information that may converge with, or deviate from, the conclusions we have drawn so far.

The models also allow us to investigate the sources of errors – whether they derive from an incorrect grammar or from difficulty in implementing the sequences specified by the grammar. We explore whether the greater error rate in the center-embedded condition derived from greater difficulty learning the correct structure, greater difficulty in correctly implementing that structure, or some combination of both.

In each model, the stack and the queue, we assume that there are two possible programs that people may use to generate sequences in this experiment: a program that generates center-embedded sequences and a program that generates crossed-serial sequences. These programs get compiled into a list of push/pop operations from either a stack or a queue — the exact series of push and pop operations needed to pick each item in a center-embedded versus crossed-serial sequence depends on which memory architecture is used. The queue and stack architectures thus make distinct predictions about how quickly people will choose each item when generating center-embedded and crossed-serial sequences. Additionally, we included a program that generates sequences with blue items first and red items second, with no systematic order of the blue or red items. Because the further training trials stop immediately at the first incorrect touch, we did not attempt to differentiate between various other possible incorrect programs. This program does not rely on the use of either a stack or a queue, and thus to the extent that participants were using this (incorrect) program to generate sequences, would not distinguish them. We also assume that the memory operations introduce the possibility of corruption (noise), which implies that the chance of making a mistake increases with the number of push/pop operations involved — so, e.g., if the memory architecture is a queue, people should make more errors generating a center-embedded sequence than a crossed-serial sequence, since there are a greater number of push/pop operations involved and hence a greater possibility that memory will be corrupted.

As detailed in SI, we assume that participants generate a response as soon as they can identify the icon that should be selected. We model the RTs across all responses until there was an error, if there was an error. (This is unlike the data we report above, where RTs were only included for completely correct sequences and only tested on the second half of the list). We assume that there are multiple potential sources of error in generating sequences: people may be using the wrong program, implementing a program incorrectly due to a memory error, or simply not paying attention. We account for each of these possibilities in the model with independent parameters. It may be that participants do not immediately grasp how to pick items in the right order (e.g., they may think the rule is simply "pick blue items then pick red items," without respect to order of the base pairings). To account for this, we include a latent parameter representing the probability that participants are using the correct program on a given trial. This parameter is allowed to change over the course of training at each sequence length, following a Gompertz function (see SI). In addition to using the wrong program, another significant source of error may be mis-remembering the order of previously chosen items or even forgetting that an item has already been chosen. This is modeled by a different parameter, assumed to be constant for a given participant across the experiment, specifying how likely memory is to be corrupted each time an item is stored. We allow two types of corruption, "swaps" (reordering items in memory) and "deletions" (removing items from memory), which we assume for simplicity are equiprobable. Finally, we allow that on some proportion of trials, participants may not be paying attention or not trying, so may be picking items at random. Additionally, there are parameters in the model to capture reaction times under both a stack and queue memory architecture — these are detailed in the SI.

4.1 Model Comparison

We found maximum likelihood parameter estimates for each participant during the further training trials under both the queue- and stack-based models. The likelihood was the product of the probability of each particular response (which item was pressed) and the reaction time to generate that response, given the set of model parameters. The probability of each response and reaction time was computed by simulating 100,000 possible responses from the model for each sampled set of parameters. We then computed the AIC difference between the queue and stack model, which was just dependent on the difference in likelihoods since they had the same number of free parameters. The details of model fitting can be found in SI.

Figure 6a shows how the queue (left) and stack (right) model fit the aggregate response-time data in the center-embedded (blue, top) and crossed (orange, bottom) condition. The mean model predictions are shown as black lines and the human data are shown as colored points.

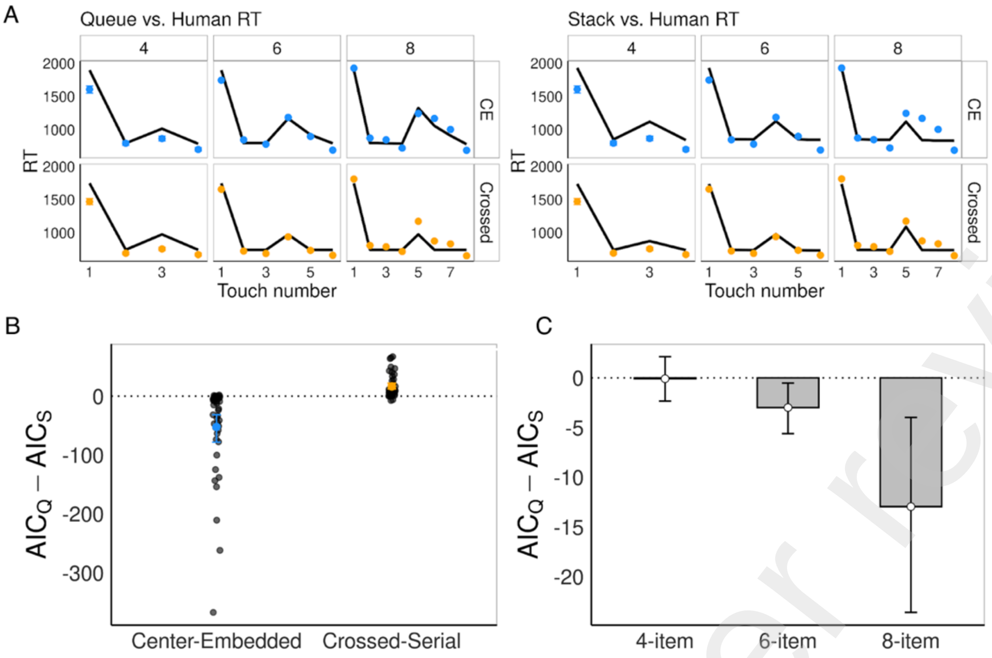

Figure 6: **(A)** Model predictions (black lines) and human data (colored points) for mean reaction times as a function of the touch number within each sequence, in 4-, 6- and 8-item lists (panels, left-to-right). The queue model's predictions are on the left and the stack model's predictions are shown on the right. The center-embedded data is shown in blue on top and the crossed-serial data is shown in orange on the bottom. **(B)** The AIC difference of each subject under the queue and stack models in the center-embedded and crossed-serial conditions; lower values indicate a better fit of the queue model relative to the stack model. **(C)** The mean AIC difference per subject between the queue and stack model in the 4-, 6-, and 8-item lists.

The differences between the two models are relatively subtle overall, but clear in the 8-item center-embedded condition. In that case, the stack model fails to capture the key downward-sloping trend of the reaction times in the second half of the list, whereas the queue model captures this trend closely. Neither model fully captures the slightly downward-sloping reaction times in the first half of each list, nor the downward-slope of the response-times in the crossed-serial condition (though the stack model does slightly better with the first item in the second half of the list). Figure 6b shows the AIC difference between the queue and stack model for each subject in both conditions, where lower values (more negative) indicate a better fit of the queue model over the stack model. The data from each condition point in contradictory directions, with a slightly better fit of the stack model in the crossed-serial condition (ΔAIC = 791) and a significantly better fit of the queue model in the center-embedded condition (ΔAIC = 2250).

The fact that the stack model fits better in the crossed-serial condition is because as the list length increases, people take increasingly long to respond to the first item of the second half of the list in the cross-serial condition — something the stack model predicts but the queue model does not (see SI). However, this by itself is fairly weak evidence for the use of a stack, since this effect is seen also on the first item on the first half of the list as well, in both conditions, and is observed on the second half of the list in both conditions. It is plausible that upon seeing the arrays for the first time, participants scan the blue items to decide which order to touch them in, and this takes more time as the lists increase from 4 to 6 to 8 items. Similarly, before switching to the red items, they again scan all the red items before beginning, and this also takes more time as the lists increase from 4 to 6 to 8 items. While this is independently predicted by the stack model for cross serial sequences alone, and so the model captures this effect, it most probably does not arise from creating a second stack in the crossed-serial condition alone. Importantly, as mentioned above and shown dramatically on Figure 6C, the queue model's overall fit to the data is much better than is the stack's, and increasingly so as array length increases. The queue and stack models predictions are more distinguishable for center-embedded sequences, where the predicted pattern of reaction time differs across the entire second half of the list. Overall, the AIC favors the queue model over the stack model by 8 in the 4-item lists, 272 in the 6-item lists, and 1179 in the 8-item lists (Fig 7C).

4.2 Analyzing the sources of error

Given the evidence that participants are using queues to generate both sequences offered above, we now focus on the value of inferred parameters in the queue model only, to better understand why participants made errors — and specifically why they made them at different rates in the two conditions. We assumed that there were three potential reasons participants might make an error: 1) they were using a program that failed to generate sequences with the correct structure beyond blue before red; 2) they intended to generate the correct structure but mis-remembered which items they had already chosen or the order in which they were chosen; or 3) they were not paying attention or pressing randomly. Each of these potential sources of error would generate different error patterns: 1) would create errors in the second half of the list, particularly toward the beginning of the second half of the list; 2) would generate patterns that are close to the correct structure but have an error, particularly likely toward the end of the sequence; and 3) would generate an equal proportion of errors across the entire sequence. We included parameters for each of these possibilities in the model, allowing us to determine the relative contribution of each source of error (See SI).

We assumed that participants may not have learned the correct program before the beginning of each new sequence length but may learn it over the course of the training trials. We therefore fit participants' probability of using a correct program as a function of trial number under a Gompertz curve, which allows us to fit a learning asymptote, slope, and intercept. Figure 7a shows the inferred average probability of using the correct program over the course of training trials in each sequence length in both conditions. See SI for modeling details. The difference between the two conditions is substantial for 4- and 6-item sequences, especially at the beginning of the set of trials, but small for 8-item sequences and at the end of the 4- and 6-item training trials as well. This indicates that learning to generate center-embedded structures is substantially harder than learning to generate crossed-serial structures. Overall, the inferred rate of errors from incorrect program was twice as great in the center-embedded condition (0.12, CI=[0.10, 0.15]) than in the crossed-serial condition (0.06 CI=[0.05,0.07]).

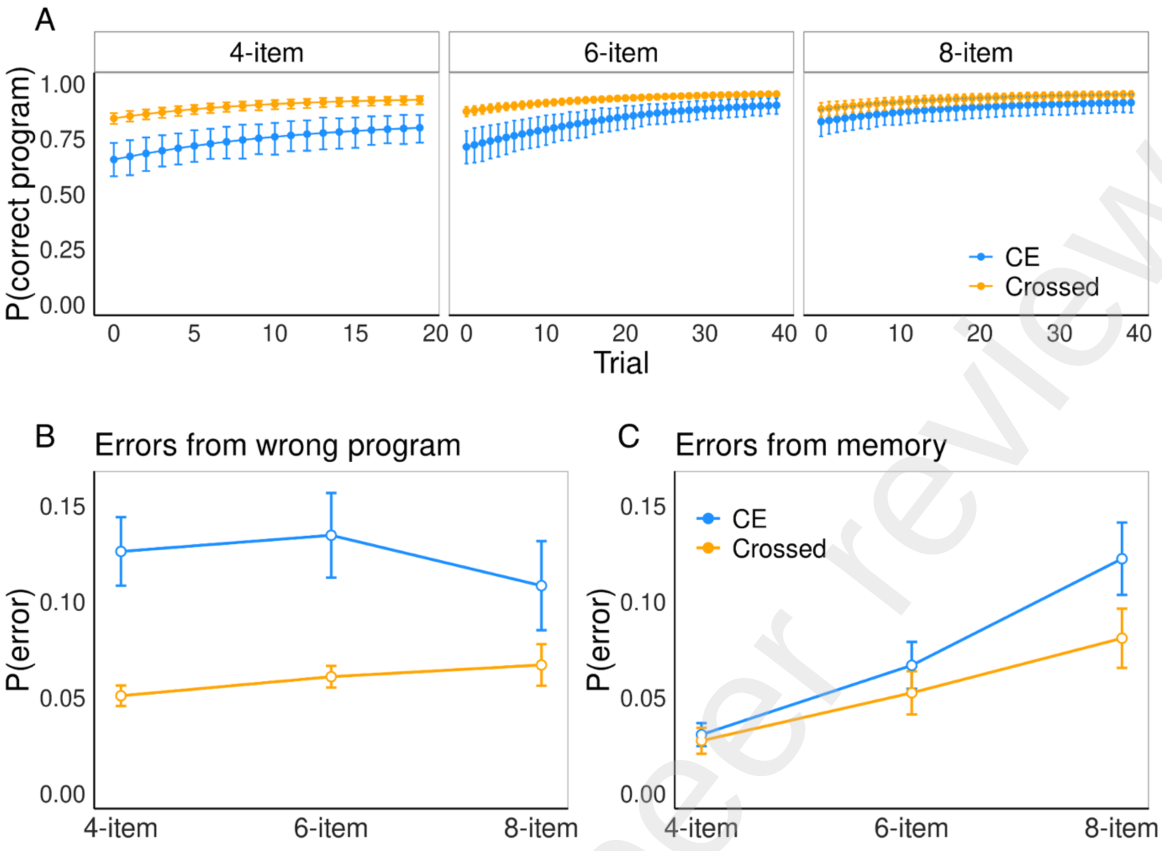

Figure 7: (**A**) The inferred probability that participants are using the correct program over the course of training trials of each sequence length, in both the center-embedded (blue) and crossed-serial (orange) conditions. (**B**) The average probability participants make an error from using the wrong program across 4-item, 6-item, and 8-item lists. (**C**) The probability participants make an error from misremembering which items they already chose or the order in which they chose them.

We can compare the rates at which errors were caused by difficulty learning the correct program relative to difficulty implementing that program. Figure 7b shows the rate at which errors were caused by using the wrong program and Figure 7c shows the rate at which errors were caused by misremembering which items were previously chosen or their order. Errors caused by using the wrong program were more common across all sequence lengths in the center-embedded condition relative to the crossed condition, and were more common earlier in learning (i.e., 4- and 6-item sequences). However, the effect of memory noise only causes substantially more errors in the center-embedded condition relative to the crossed-serial condition in the 8-item sequences, with a rate of 0.12 (CI=[0.10, 0.14]) memory errors in the center-embedded condition and a rate of 0.08 (CI=[0.07, 0.10]) memory errors in the crossed-serial condition. As would be intuitively expected, the rate of errors caused by memory noise grows with the number of items displayed — but it does so more quickly in the center-embedded condition, as predicted by the queue model but not the stack model (see SI).

5. General Discussion

There were four main results from this study. First, sequencing performance was well above chance on 4-, 6-, and 8-item novel test arrays. This establishes that at least some participants had abstracted grammars that generated full center-embedded and cross-serial sequences that generalized to more levels of embedding than in their training. Second, participants in the center-embedded condition required more training to learn the grammar to generalize to new levels of embedding. This difference was seen from the very beginning of training, but especially in the generalization from one level to embedding to two. This result replicates other studies in finding that center-embedded grammars are harder to learn than closely matched cross-serial grammars (Bach et al., 1986; De Vries et al., 2011; Ottl et al., 2015; Uddén et al., 2009; Uddén et al., 2012). Third, participants in the center-embedded condition had more difficulty implementing the procedure that generates such structures once learned than did those in the cross-serial condition, as shown by longer RTs between touches in the second half of the list in the center-embedded condition. Fourth, over the second half of the list, the item-to-item RTs decreased significantly more in the center-embedded condition than in the cross-serial condition.

It is often assumed that humans use push-down stacks to represent center-embedded and cross-serial structures (Fitch & Hauser, 2004; Jiang et al., 2018; Kinsella, 2010; Malassis, Dehaene, & Fagot, 2020; Rodriguez, & Granger, 2016; Joshi, 1989; Joshi, Shanker, Weir, 1991). The finding that cross-serial structures are easier to learn and to produce is inconsistent with the assumption that push-down stacks are the basis of representing both types of recursive structures, because cross-serial structures would require two stacks (and additional processing), relative to center-embedded structures which would require only one (see Figure 1).

However, the relative difficulty in learning/execution is consistent with the possibility that push-down stacks are used in the representation/execution of center-embedded structures, and some other representations of sequences are used for cross-serial structures, *and* that push-down stacks make greater information processing demands than the alternatives. To evaluate this hypothesis, we must specify an alternative to stack architecture. For these structures an alternative is readily available: a queue. Our studies examined item-to-item RTs as well as error rates. The fourth result from this study was unequivocal evidence that queues were used to represent and execute both center-embedded and cross-serial sequences. In accord with center-embedded sequences being generated through a process of iterative forward search through a queue, participants in the center-embedded conditions showed steeper linearly decreasing response times for the second half of the sequence than did those in the cross-serial conditions, reflecting the fact that they had to iteratively step through the queue to find the next item in the sequence. These response time findings, along with the robust evidence that center-embedded grammars are harder to learn than cross-serial ones, point to a queue-based representational system for both center-embedded and cross-serial sequences in this task.

These conclusions were further explored in a comparison of two computational models, one based on stacks as its memory structure and one based on queues as its memory structure, in the fit to the observed data in generation of both cross-serial and center-embedded sequences. Overall, the queue model fit the data better. Accordingly, we used the queue model to separate sources of error into having failed to learn the correct sequence from memory errors due to the difficulty implementing the process which produces them. The modeling results found that both sources of error were greater in the center-embedded condition. Furthermore, separating these sources of error confirmed that errors due to learning the correct center-embedded structure were greater at the beginning of the experiment (i.e., more evident on the 4- and 6-item novel lists than on the 8-item novel list), whereas, not surprisingly, memory errors increased with length of list.

Although we were able to test sequence lengths of up to 8-items, we could not directly test if the grammars induced were truly *unbounded*. Participants could have learned either a bounded grammar (A^nB^n, $0 \leq n \leq 4$) or an unbounded grammar (A^nB^n, , $0 \leq n$). However, to induce the bounded version of this grammar, participants would have needed to acquire a grammar (e.g., "A^nB^n, $0 \leq n \leq 4$") at the 6-item stage in order for it to be used to generalize to the never before seen 8-item length. Inducing this type of bounded grammar for unobserved number of embeddings seems highly unlikely (McCoy, et al., 2021). Although there is no way of testing for unbounded grammar learning in the sequencing task used here, future work could ask subjects to specify the grammar learned, i.e., to describe the procedure they were using to generate sequences, which could provide further evidence of the type of grammar induced.

These data raise several questions for further research and discussion. The first question is why center-embedded grammars are harder to *learn.* The conclusion that queues are used to represent both cross-serial AND center-embedded structures makes sense of why center-embedded structures are harder to *produce on-line*, generating more errors in sequencing and the distinctive item-to-item RT profile of iterative stepping through a queue in the second half of the list. But why is the center-embedded structure harder to learn?

To learn what the structure is, a participant must *notice* that at least one training sequence had a center-embedded structure in order to *hypothesize* that the sequences might always satisfy a center-embedded grammar. Both center-embedded structures and cross-serial structures require representing the order of touches of the blue items, and in these studies, these are represented as a queue. Thus, the "same queue order" hypothesis is immediately available as a hypothesis for the order of the red shapes. To even notice the "reverse queue" order for the red shapes, one must iteratively step through the queue. That is, it is quite possible that the greater processing difficulty of generating center-embedded structures from queue representations makes them harder to learn, as well as harder to generate online even when learned.

Accepting this argument has major implications for the original motivation for the project that led to comparing the ease of learning and producing cross-serial and center-embedded structures: exploring whether formal grammar complexity (in terms of the Chomsky Hierarchy) has implications for the *mental representations* of these grammars. We find no evidence that formal complexity predicts relative processing difficulty. Complexity matters to ease of learning, but it is processing complexity that does so.

The present data raise an important question for further research with non-human animals and with young children on the present sequencing task. Ferrigno et al. (2020) showed that both rhesus macaques and 3- to 5-year-olds could learn the two initial training sequences in the center-embedded condition (the cross-serial condition was not tested) and generalize what they had learned to the novel combination test trials. The monkeys were also tested on completely novel test arrays, and successfully generalized the structure of the trained sequence: $A_1A_2B_2B_1$. Subsequent work with 3- to 4-year-olds added a cross-serial condition and tested generalization to novel test arrays (Ferrigno & Carey, 2020). They found that the center-embedded sequences were harder to learn and harder to process than cross-serial ones, as with adults, but there was successful generalization to novel arrays in both cases. What is not known, either in the case of children or monkeys, is whether the animals and children, who *do* learn an abstract grammar that generates the 4-item sequences, is whether they had learned only a bounded grammar that only generates the correct 4-item sequences (e.g., a slot filler representation of the 4-item sequence; although see Ferrigno, 2019 for suggestive positive evidence from a single monkey on generalization to a novel 6-item sequence). Such representations would not lead to successful generalization to novel 6- and 8-item sequences. Recent work by Liao et al. (2022) found that crows could represent 6-item center-embedded sequences with some training, but they did not test for spontaneous generalization from 4- to 6-items. A high priority is to establish whether cross-serial and center-embedded grammars that generalize to additional levels of embedding can be learned on the basis of the 4-item training by non-human animals and young children. If not, the next question would be whether they *can* be learned with further training on 6-item sequences by these populations, as they were by some adults in the present experiment.

Computational modeling approaches might provide insights into how different grammars are learned. Note, the model we report here is not a model for *how* the grammars are learned; it models when some grammar sufficient to generate the strings has been learned, and the sources of errors in string production as a function both of not having learned a relevant grammar yet and also of difficulty executing the process that generates the correct strings. A recent paper by Yang and Piantadosi (2022) takes on the learning problem, albeit not with the goal of elucidating the learning processes in experiments such as these. Rather its goal is to challenge arguments that grammars of natural language are not learnable in the absence of innate domain specific constraints. Yang and Piantadosi tested 82 different grammars, including regular, context free, and context sensitive grammars, most of them vastly more complicated than those studied here, as well as fragments of natural language grammars. They showed that almost all are learnable from small amounts of positive data alone. The model included logical and set manipulation primitives, and small set of representational/computational primitives related to representations of lists: including list, or string, and a few functions on lists including adding a new character on the end of the list, identifying the first character of the list, identifying the rest of the list, appending or joining lists *X* and *Y.* Interestingly, there is no predefined function that identifies the last character in the string, although it would be easy to build a procedure for doing so from the primitives of the system. Thus, the above functions on lists have the properties of functions on queues, not stacks in this model. However, there is one primitive function which makes the primitive list structures unlike either stacks or queues, namely *insert (X, Y)*, which inserts list *X* into the middle of list

Y. Although this model was not intended to capture the learning process at an algorithmic level, it opens a dialog about the actual learning mechanisms that induce grammars from encountering strings that satisfy the grammar, and, like the present studies, calls for further research on the nature of list representations and the list manipulation functions animals and people have.

An open question is whether the memory structures and processes used to represent the visual center-embedded and cross-serial sequences in the current task are the same (or share at least some structural similarities) to the processes and memory structures used to represent these structures in natural language. The finding that cross-serial sequences are easier than center-embedded sequences has also been seen in psycholinguistic tasks (Bach et al., 1986). Although center-embedded dependencies are more prevalent in languages, there are some languages that prefer cross-serial dependencies. Bach et al. (1986) tested differences in comprehension and ratings of comprehensibility between different dependency types: German (center-embedded) and Dutch (cross-serial) dependencies. They found center-embedded sequences in German were harder to comprehend and rated as less comprehensible than matched cross-serial sequences in Dutch. This suggests that center-embedded sequences are harder to represent in language than cross-serial structures. Therefore, it is unlikely that push-down stacks are the basis for representing these structures.

Nonetheless, there are important differences between center-embedded structures in natural language from those in the artificial grammar tasks we have discussed in this paper. One difference is that in language there is a main clause which not only contains the other pairs, but is modified by them (e.g., in the sentence “The cat the dog chased ran”, the phrase “the dog chased” is describing which cat ran). In contrast, the sequences tested here and in the previous artificial grammar studies reviewed here do not have this aspect of the hierarchical structure found in center-embedded sentences. This difference limits the conclusions that can be made from the present results as to whether center-embedded structures in language should be harder to learn than cross serial ones, or harder to process. However, the fact that center-embedded structures are harder to process in *both* natural and artificial language is consistent with the possibility of at least some shared processing mechanism in the two cases. Whether center-embedded structures are harder to learn in the course of language acquisition is not yet known. Studies of language acquisition across the Dutch-learning and German-learning children could test whether center-embedded linguistic structures are harder to learn as well.

Many previous explanations for the relative difficulty of center-embedded sequences in natural language depend on features specific to natural language (Joshi, 1989). Other explanations break down at the depths of recursions tested here or make predictions that are not seen in the current data (Christiansen & Chater; 1999; Christiansen & MacDonald, 2009; Gibson, 1998; Lakretz et al., 2021). For example, Joshi's (1989) embedded push-down automaton theory, implemented entirely in terms of stacks, provides an explanation for why cross-serial sequences might be easier, but appeals to features specific to language (main and subordinate clauses) that are not shared by the simplest center-embedded and cross-serial grammars studied here. Other explanations which don't commit to specific memory structures, such as usage-based accounts, rely on Simple Recurrent Networks (SRNs) or Long Short-Term memory networks (LSTMs) which breakdown to at the longer and untrained depths of embedding tested here (Christiansen & Chater; 1999; Christiansen & MacDonald, 2009; Lakretz et al., 2021) and require highly specific training to make hierarchical generalizations (Coopmans et al., 2022). Lastly, the Syntactic Prediction Locality Theory, while not a processing theory, does make the prediction that cross-serial sequences should be easier to represent than center-embedded ones of the same length and can scale to any length (Gibson, 1998). However, this model predicts that the maximal memory complexity should be for the final item in a center-embedded structure, yet we found increasing accuracy for the second half of the center-embedded sequences (see SI). Thus, these previous models fail to explain the relative difficulty of center-embedded structures compared to cross-serial structures tested here.

It is likely that aspects of the explanation for the greater difficulty processing center-embedded than cross-serial structures in natural language are different from the explanation of the comparable difference in the present studies. However, that parallel results in the current artificial grammar study and in natural language are also consistent with a common explanation – that queues may be used to represent *some* linear orders within sentences as well, and perhaps computing center-embedded and cross-serial structures sometimes deploy queues. Clearly, there is much future research to be done investigating how center-embedded and cross-serial structures are represented and learned in German and in Dutch. Another approach to these questions might be to build aspects of the structure of natural language into the artificial grammars to explore when, if ever, the pattern of results reported here cease to hold.

Importantly, the goal of this paper was not to study cross-serial and center-embedded structures in natural language. Rather our target was the learning of artificial grammars in the tradition begun by Chomsky in the 1950's (e.g., Chomsky 1957), and continuing to today (e.g., Yang and Piantadosi, 2021). Specifically, our target was an algorithmic description of the actual representations of order, how ordered lists are represented in memory, and how ordered lists are processed while creating strings that satisfy the learned grammar.

5. Conclusion

To our knowledge, the present study is the first to explore whether the mental representations of lists that underlie the induction of indexed A^nB^n grammars involve stacks (representations of string order than can only be accessed from the end) or queues (representations of string order than can only be accessed from the beginning). Drawing on the literature on forward and backward list recall, we found no evidence that mind deploys stacks in these grammar induction paradigms, and positive evidence that participants deployed queues in the learning of both simple context free and context sensitive grammars as well as in the production of sequences that satisfied both context free and context sensitive grammars.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

We thank Nancy Soja, Kelsey Davison, & Sadaf Khan for research support. Abhishek Dedhe & Nour Al-Zaghloul for assistance programming the task. We also thank Brian Leahy, Joshua Rule, Ben Pitt, Alessandra Silva & the Carey Lab helpful for comments and discussion. This research was supported by NIH grant #F32-HD101208 and McDonnell Foundation grant #220020418.

Data availability

The data and analysis code are available for download here: https://github.com/Sferrigno/Creating-recursive-center-embedded-sequences-with-an-iterative-queue.git The Bayesian model code is available for download here: https://github.com/samcheyette/stacks_and_queues

References

Anders, T. R., & Lillyquist, T. D. (1971). Retrieval time in forward and backward recall. *Psychonomic Science*, *22*(4), 205-206.

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

Bahlmann, J., Schubotz, R. I., & Friederici, A. D. (2008). Hierarchical artificial grammar processing engages Broca's area. *Neuroimage*, *42*(2), 525-534.

Berwick, R. C., & Chomsky, N. (2016). *Why Only Us: Language and Evolution*. MIT press.

Bireta, T. J., Fry, S. E., Jalbert, A., Neath, I., Surprenant, A. M., Tehan, G., & Tolan, G. A. (2010). Backward recall and benchmark effects of working memory. *Memory & Cognition*, *38*(3), 279-291.

Chomsky, N. (1957). *Syntactic Structures.* Mouton.

Christiansen, M. H., & Chater, N. (1999). Toward a connectionist model of recursion in human linguistic performance. *Cognitive Science*, *23*(2), 157-205.

Christiansen, M. H., & MacDonald, M. C. (2009). A usage-based approach to recursion in sentence processing. *Language Learning*, *59*, 126-161.

Coopmans, C. W., De Hoop, H., Kaushik, K., Hagoort, P., & Martin, A. E. (2022). Hierarchy in language interpretation: evidence from behavioural experiments and computational modelling. Language, *Cognition and Neuroscience*, 37(4), 420-439.

Corballis, M. C. (2007). Recursion, language, and starlings. *Cognitive Science*, *31*(4), 697-704.

De Vries, M. H., Monaghan, P., Knecht, S., & Zwitserlood, P. (2008). Syntactic structure and artificial grammar learning: The learnability of embedded hierarchical structures. *Cognition*, *107*, 763–774.

De Vries, M. H., Petersson, K. M., Geukes, S., Zwitserlood, P., & Christiansen, M. H. (2012).

Processing multiple non-adjacent dependencies: Evidence from sequence learning. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *367*, 2065–2076.

De Vries, M., Christiansen, M., & Petersson, K. M. (2011). Learning recursion: Multiple nested and crossed dependencies. *Biolinguistics*, *5*, 10–35.

Ferrigno, S. (2018). The evolutionary and developmental origins of human thought. University of Rochester.

Ferrigno, S. (2022). Sequencing, Artificial Grammar, and Recursion in Primates. In Schwartz, B. L. & Beran, M. J. (Ed.), *Primate Cognitive Studies.* Cambridge University Press.

Ferrigno, S. & Carey, S. (2020). The representation of recursive center-embedded and cross-serial sequences in children and adults. Proceedings for the 42nd Annual Conference of the Cognitive Science Society.

Ferrigno, S., Cheyette, S. J., Piantadosi, S. T., Cantlon, J. F. (2020). Recursive sequence generation in monkeys, children, US adults, and native Amazonians. *Science Advances.*

Fitch, W. T., & Friederici, A. D. (2012). Artificial grammar learning meets formal language theory: an overview. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *367*(1598), 1933-1955.

Fitch, W. T., & Hauser, M. D. (2004). Computational constraints on syntactic processing in a nonhuman primate. *Science*, *303*(5656), 377-380.

Gentner, T. Q., Fenn, K. M., Margoliash, D., & Nusbaum, H. C. (2006). Recursive syntactic pattern learning by songbirds. *Nature*, *440*(7088), 1204-1207.

Gibson, E. (1998). Linguistic complexity: Locality of syntactic dependencies. *Cognition*, *68*(1), 1-76.

Hochmann, J. R. (2022). Representations of Abstract Relations in Infancy. *Open Mind*, 1-20

Hochmann, J. R., & Toro, J. M. (2021). Negative mental representations in infancy. *Cognition*, *213*, 104599.

Haberlandt, K., Lawrence, H., Krohn, L., Bower, K., & Thomas, J. G. (2005). Pauses and durations exhibit a serial position effect. *Psychonomic Bulletin & Review*, *12*(1), 152-158.

Hurlstone, M. J., Hitch, G. J., & Baddeley, A. D. (2014). Memory for serial order across domains: An overview of the literature and directions for future research. *Psychological Bulletin*, *140*(2), 339.

Jiang, X., Long, T., Cao, W., Li, J., Dehaene, S., & Wang, L. (2018). Production of supra-regular spatial sequences by macaque monkeys. *Current Biology*, *28*, 1851–1859.

Joshi, A. K. (1990). Processing crossed and nested dependencies: An automation perspective on the psycholinguistic results. *Language and Cognitive Processes*, *5*(1), 1-27.

Joshi A. K., Shanker K.V., Weir D. (1991) The convergence of mildly context-sensitive grammar formalisms. In: Sells P, Shieber S, Wasow T, (ed.) *Foundational Issues in Natural Language Processing.* (pp. 31–81) MIT Press, Cambridge.

Kinsella, A. R. (2010). 10. Was recursion the key step in the evolution of the human language faculty?. *Studies in Generative Grammar 104*, 179.

Lakretz, Y., Desbordes, T., King, J. R., Crabbé, B., Oquab, M., & Dehaene, S. (2021). Can RNNs learn recursive nested subject-verb agreements?. *arXiv preprint arXiv:2101.02258*.

Liu, Y. A., & Stoller, S. D. (1999, November). From recursion to iteration: what are the optimizations?. In *Proceedings of the 2000 ACM SIGPLAN workshop on Partial evaluation and semantics-based program manipulation* (pp. 73-82).

Lobina, D. J. (2011). Recursion and the competence/performance distinction in AGL tasks. *Language and Cognitive Processes*, *26*(10), 1563-1586.

Malassis, R., Dehaene, S., & Fagot, J. (2020). Baboons (*Papio papio*) process a context-free but not a context-sensitive grammar. *Scientific Reports*, *10*, 1–12.

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

McCoy, R. T., Culbertson, J., Smolensky, P., & Legendre, G. (2021). Infinite use of finite means? Evaluating the generalization of center embedding learned from an artificial grammar. In *Proceedings of the Annual Meeting of the Cognitive Science Society* (Vol. 43, No. 43).

Öttl, B., Jäger, G., & Kaup, B. (2015). Does formal complexity reflect cognitive complexity? Investigating aspects of the Chomsky hierarchy in an artificial language learning study. *PloS One*, *10*, e0123059.

Page, M., & Norris, D. (1998). The primacy model: a new model of immediate serial recall. *Psychological Review*, *105*(4), 761.

Perruchet, P., & Rey, A. (2005). Does the mastery of center-embedded linguistic structures distinguish humans from nonhuman primates?. *Psychonomic Bulletin & Review*, *12*(2), 307-313.

Pinker, S., & Jackendoff, R. (2005). The faculty of language: what's special about it?. *Cognition*, *95*(2), 201-236.

Rey, A., Perruchet, P., & Fagot, J. (2012). Centre-embedded structures are a by-product of associative learning and working memory constraints: Evidence from baboons (*Papio papio*). *Cognition*, *123*, 180–184.

Rodriguez, A., & Granger, R. (2016). The grammar of mammalian brain capacity. *Theoretical Computer Science*, *633*, 100-111.

Shima, K., Isoda, M., Mushiake, H., & Tanji, J. (2007). Categorization of behavioural sequences in the prefrontal cortex. *Nature*, *445*, 315–318.

Shin, W. J., & Eberhard, K. M. (2015). Learning a Center-Embeddding Rule in an Artificial Grammar Learning Task. *Proceedings of the Annual Meeting of the Cognitive Science Society, 2015.*

Terrace, H. S. (2005). The simultaneous chain: A new approach to serial learning. *Trends in Cognitive Sciences*, *9*, 202–210.

Thomas, J. G., Milner, H. R., & Haberlandt, K. F. (2003). Forward and backward recall: Different response time patterns, same retrieval order. *Psychological Science*, *14*(2), 169-174.

Udden, J., Ingvar, M., Hagoort, P., & Petersson, K. M. (2012). Implicit acquisition of grammars with crossed and nested non-adjacent dependencies: Investigating the push-down stack model. *Cognitive Science*, *36*, 1078–1101.

Yang, Y., & Piantadosi, S. T. (2022). One model for the learning of language. *Proceedings of the National Academy of Sciences*, *119*(5), e2021865119.

Do humans use push-down stacks when learning or producing center-embedded structures?

Authors: Stephen Ferrigno[1,2], Samuel J. Cheyette[3], Susan Carey[2]

Affiliations:

[1] Department of Psychology, University of Wisconsin-Madison, Madison, WI

[2] Department of Psychology, Harvard University, Cambridge, MA

[3] Department of Brain and Cognitive Sciences, Massachusetts Institute of Technology, Cambridge, MA

*Correspondence to: sferrigno@wisc.edu