Cognition 123 (2012) 199–217

Contents lists available at SciVerse ScienceDirect

Cognition

journal homepage: www.elsevier.com/locate/COGNIT

Bootstrapping in a language of thought: A formal model of numerical concept learning

Steven T. Piantadosi [a,*], Joshua B. Tenenbaum [b], Noah D. Goodman [c]

[a] Department of Brain and Cognitive Sciences, University of Rochester, United States
[b] Department of Brain and Cognitive Sciences, MIT, United States
[c] Department of Psychology, Stanford University, United States

ARTICLE INFO

Article history:
Available online 26 January 2012

Keywords:
Number word learning
Bootstrapping
Cognitive development
Bayesian model
Language of thought
CP transition

ABSTRACT

In acquiring number words, children exhibit a qualitative leap in which they transition from understanding a few number words, to possessing a rich system of interrelated numerical concepts. We present a computational framework for understanding this inductive leap as the consequence of statistical inference over a sufficiently powerful representational system. We provide an implemented model that is powerful enough to learn number word meanings and other related conceptual systems from naturalistic data. The model shows that bootstrapping can be made computationally and philosophically well-founded as a theory of number learning. Our approach demonstrates how learners may combine core cognitive operations to build sophisticated representations during the course of development, and how this process explains observed developmental patterns in number word learning.

© 2011 Elsevier B.V. All rights reserved.

1. Introduction

> *"We used to think that if we knew one, we knew two, because one and one are two. We are finding that we must learn a great deal more about 'and'."* [Sir Arthur Eddington]

Cognitive development is most remarkable where children appear to acquire genuinely novel concepts. One particularly interesting example of this is the acquisition of number words. Children initially learn the count list "*one*", "*two*", "*three*", up to "*six*" or higher, without knowing the exact numerical meaning of these words (Fuson, 1988). They then progress through several *subset-knower* levels, successively learning the meaning of "*one*", "*two*", "*three*" and sometimes "*four*" (Lee & Sarnecka, 2010a, 2010b; Sarnecka & Lee, 2009; Wynn, 1990, 1992). Two-knowers, for example, can successfully give one or two objects when asked, but when asked for three or more will simply give a handful of objects, even though they can recite much more of the count list.

After spending roughly a year learning the meanings of the first three or four words, children make an extraordinary conceptual leap. Rather than successively learning the remaining number words on the count list—up to infinity—children at about age 3;6 suddenly infer all of their meanings at once. In doing so, they become cardinal-principal (CP) knowers, and their numerical understanding changes fundamentally (Wynn, 1990, 1992). This development is remarkable because CP-knowers discover the abstract relationship between their counting routine and number-word meanings: they know how to count and how their list of counting words relates to numerical meaning. This learning pattern cannot be captured by simple statistical or associationist learning models which only track co-occurrences between number words and sets of objects. Under these models, one would expect that number words would continue to be acquired gradually, not suddenly as a coherent conceptual system. Rapid change seems to require a learning mechanism which comes to

* Corresponding author.
E-mail address: piantado@mit.edu (S.T. Piantadosi).

0010-0277/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.cognition.2011.11.005

some fundamental knowledge that is more than just associations between words and cardinalities.

We present a formal learning model which shows that statistical inference over a sufficiently powerful representational space can explain why children follow this developmental trajectory. The model uses several pieces of machinery, each of which has been independently proposed to explain cognitive phenomena in other domains. The representational system we use is lambda calculus, a formal language for compositional semantics (e.g., Heim & Kratzer, 1998; Steedman, 2000), computation more generally (Church, 1936), and other natural-language learning tasks (Piantadosi, Goodman, Ellis, & Tenenbaum, 2008; Liang, Jordan, & Klein, 2009, 2010; Zettlemoyer & Collins, 2005, 2007). The core inductive part of the model uses Bayesian statistics to formalize what inferences learners should make from data. This involves two key parts: a likelihood function which measures how well hypotheses fit observed data, and a prior which measures the complexity of individual hypotheses. We use simple and previously proposed forms of both. The model uses a likelihood function that uses the *size principle* (Tenenbaum, 1999) to penalize hypotheses which make overly broad predictions. Frank, Goodman, and Tenenbaum (2007) proposed that this type of likelihood function is important in cross-situational word learning and Piantadosi et al. (2008) showed that it could solve the *subset problem* in learning compositional semantics. The prior is from the *rational rules* model of Goodman, Tenenbaum, Feldman, and Griffiths (2008), which first linked probabilistic inference with formal, compositional, representations. The prior assumes that learners prefer simplicity and re-use in compositional hypotheses and has been shown to be important in accounting for human rule-based concept learning.

Our formal modeling is inspired by the bootstrapping theory of Carey (2009), who proposes that children observe a relationship between numerical quantity and the counting routine in early number words, and use this relationship to inductively define the meanings of later number words. The present work offers several contributions beyond Carey's formulation. Bootstrapping has been criticized for being too vague (Gallistel, 2007), and we show that it can be made mathematically precise and implemented by straightforward means.[1] Second, bootstrapping has been criticized for being incoherent or logically circular, fundamentally unable to solve the critical problem of inferring a discrete infinity of novel numerical concepts (Rips, Asmuth, & Bloomfield, 2006, 2008; Rips, Bloomfield, & Asmuth, 2008). We show that this critique is unfounded: given the assumptions of the model, the correct numerical system can be learned while still considering conceptual systems much like those suggested as possible alternatives by Rips, Asmuth and Bloomfield. The model is capable of learning conceptual systems like those they discuss, as well as others that are likely important for natural language. We also show that the model robustly gives rise to several qualitative phenomena in the literature which have been taken to support bootstrapping: the model progresses through three or four distinct subset knower-levels (Lee & Sarnecka, 2010a, 2010b; Sarnecka & Lee, 2009; Wynn, 1990, 1992), does not assign specific numerical meaning to higher number words at each subset-knower level (Condry & Spelke, 2008), and suddenly infers the meaning of the remaining words on the count list after learning "*three*" or "*four*" (Lee & Sarnecka, 2010a, 2010b; Sarnecka & Lee, 2009; Wynn, 1990, 1992).

This modeling work demonstrates how children might combine statistical learning and rich representations to create a novel conceptual system. Because we provide a fully implemented model which takes naturalistic data and induces representations of numerosity, this work requires making a number of assumptions about facts which are under-determined by the current experimental data. This means that the model provides *at minimum* an existence proof for how children might come to numerical representations. However, one advantage of this approach is that it provides a computational platform for testing multiple theories within this same framework—varying the parameters, representational system, and probabilistic model. We argue that all assumptions made are computationally and developmentally plausible, meaning that the particular version of the model presented here provides a justifiable working hypothesis for how numerical acquisition might progress.

2. The bootstrapping debate

Carey (2009) argues that the development of number meanings can be explained by (*Quinian*) *bootstrapping*. Bootstrapping contrasts with both associationist accounts and theories that posit an innate *successor function* that can map a representation of a number N onto a representation of its successor $N + 1$ (Gallistel & Gelman, 1992; Gelman & Gallistel, 1978; Leslie, Gelman, & Gallistel, 2008). In Carey's formulation, early number-word meanings are represented using mental models of small sets. For instance two-knowers might have a mental model of "one" as $\{X\}$ and "two" as $\{X, X\}$. These representations rely on children's ability for *enriched parallel individuation*, a representational capacity that Le Corre and Carey (2007) argue can individuate objects, manipulate sets, and compare sets using one-to-one correspondence. Subset-knowers can, for instance, check if "two" applies to a set S by seeing if S can be put in one-to-one correspondence with their mental model of two, $\{X, X\}$.

In bootstrapping, the transition to CP-knower occurs when children notice the simple relationship between their first few mental models and their memorized count list of number words: by moving one element on the count list, one more element is added to the set represented in the mental model. Children then use this abstract rule to *bootstrap* the meanings of other number words on their count list, recursively defining each number in terms of its predecessor. Importantly, when children have learned the first few number-word meanings they are able to recite many more elements of the count list. Carey argues that this linguistic system provides a *placeholder structure* which provides the framework for the critical inductive inference. Subset knowers have only stored a few set-based representations; CP-knowers have discovered

[1] Running code is available from the first author.

the generative rule that relates mental representations to position in the counting sequence.

Bootstrapping explains why children's understanding of number seems to change so drastically in the CP-transition and what exactly children acquire that's "new": they discover the simple recursive relationship between their memorized list of words and the infinite system of numerical concepts. However, the theory has been criticized for its lack of formalization (Gallistel, 2007) and the fact that it does not explain how the abstraction involved in number-word meanings is learned (Gelman & Butterworth, 2005). Perhaps the most philosophically interesting critique is put forth by Rips et al. (2006), who argue that the bootstrapping hypothesis actually *presupposes* the equivalent of a successor function, and therefore cannot explain where the numerical system comes from (see also Margolis & Laurence, 2008; Rips, Asmuth, & Bloomfield, 2008; Rips et al., 2008). Rips, Asmuth, & Bloomfield argue that in transitioning to CP-knowers, children critically infer,

> If k is a number word that refers to the property of collections containing n objects, then the next number word in the counting sequence, next (k), refers to the property of collections containing one more than n objects.

Rips, Asmuth & Bloomfield note that to even consider this as a possible inference, children must know how to construct a representation of the property of collections containing $n+1$ objects, for any n. They imagine that totally naive learners might entertain, say, a Mod-10 system in which numerosities start over at ten, with "*eleven*" meaning one and "*twelve*" meaning two. This system would be consistent with the earliest-learned number meanings and thus bootstrapping number meanings to a Mod-10 system would seem to be a logically consistent inference. Since children avoid making this and infinitely many other possible inferences, they must already bring to the learning problem a conceptual system isomorphic to natural numbers.

The formal model we present shows how children could arrive at the correct inference and learn a recursively bootstrapped system of numerical meanings. Importantly, the model can entertain other types of numerical systems like such Mod-*N* systems, and, as we demonstrate, will learn them when they are supported by the data. These systems are not ruled out by any hard constraints and therefore the model demonstrates one way bootstrapping need not assume specific knowledge of natural numbers.

3. Rebooting the bootstrap: a computational model

The computational model we present focuses on only one slice of what is undoubtedly a complex learning problem. Number learning is likely influenced by social, pragmatic, syntactic, and pedagogical cues. Here, we simplify the problem by assuming that the learner hears words in contexts containing sets of objects and attempts to learn structured representations of meaning. The most basic assumption of this work is that meanings are formalized using a "language of thought" (LOT) (Fodor, 1975), which, roughly, defines a set of primitive cognitive operations and composition laws. These meanings can be interpreted analogously to short computer programs which "compute" numerical quantities. The task of the learner is to determine which compositions of primitives are likely to be correct, given the observed data. Our proposed language of thought is a serious proposal in the sense that it contains primitives which are likely available to children by the age they start learning number, if not earlier. However, like all cognitive theories, the particular language we use is a simplification of the computational abilities of even infant cognition.

We begin by discussing the representational system and then describe basic assumptions of the modeling framework. We then present the primitives in the representational system and the probabilistic model.

3.1. A formalism for the LOT: lambda calculus

We formalize representations of numerical meaning using *lambda calculus*, a formalism which allows complex functions to be defined as compositions of simpler primitive functions. Lambda calculus is computationally and mathematically convenient to work with, yet is rich enough to express a wide range of conceptual systems.[2] Lambda calculus is also a standard formalism in semantics (Heim & Kratzer, 1998; Steedman, 2000), meaning that, unlike models that lack structured representations, our representational system can interface easily with existing theories of linguistic compositionality. Additionally, lambda calculus representations have been used in previous computational models of learning words with abstract or functional properties (Piantadosi et al., 2008; Zettlemoyer & Collins, 2005, 2007).

The main work done by lambda calculus is in specifying how to compose primitive functions. An example lambda expression is:

$$\lambda x \cdot (\mathit{not}\ (\mathit{singleton?}\ x)). \tag{1}$$

Each lambda calculus expression represents a function and has two parts. To the left of a period, there is a "λx". This denotes that the argument to the function is the variable named x. On the right hand side of the period, the lambda expression specifies how the expression evaluates its arguments. Expression (1) returns the value of *not* applied to (*singleton*? x). In turn, (*singleton*? x) is the function *singleton?* applied to the argument x.[3] Since this lambda expression represents a function, it can be applied to arguments—in this case, sets—to yield return values. For instance, (1) applied to {Bob, Joan} would yield TRUE, but applied to {Carolyn} would yield FALSE since only the former is not a singleton set.

[2] In fact, untyped lambda calculus could represent any *computable* function from sets to number words. While we use a typed version of lambda calculus, our numerical meanings still have the potential to "loop infinitely", requiring us to cut off their evaluation after a fixed amount of time.

[3] As in the programming language *scheme*, function names often include "?" when they return a truth value. In addition, we use *prefix notation* on functions, meaning that the function f applied to x is written as (f x).

3.2. Basic assumptions

We next must decide on the appropriate interface conditions for a system of numerical meaning—what types of questions can be asked of it and what types of answers can it provide. There are several possible ways of setting up a numerical representation: (i) The system of numerical meaning might map each number word to a predicate on sets. One would ask such a system for the meaning of "*three*", and be given a function which is true of sets containing exactly three elements. This system would fundamentally represent a function which could answer "Are there *n*?" for each possible *n*. (ii) The system of numerical meaning might work in the *opposite* direction, mapping any given set to a corresponding number word. In this setup, the numerical system would take a set, perhaps {duck_A, duck_B, duck_C}, and return a number *word* corresponding to the size of the set—in this case, "*three*". Such a system can be thought of as answering the question "How many are there?" (iii) It is also possible that the underlying representation for numerical meaning is one which relies on *constructing* a set. For instance, the "meaning" of "*three*" might be a function which takes three elements from the local context and binds them together into a new set. Such a function could be cached out in terms of motor primitives rather than conceptual primitives, and could be viewed as responding to the command "Give me *n*".

It is known that children are capable of all of these numerical tasks (Wynn, 1992). This is not surprising because each type of numerical system can potentially be used to answer other questions. For instance, to answer "How many are there?" with a type-(i) system, one could test whether are *n* in the set, for $n = 1, 2, 3, \ldots$. Similarly, to answer "Are there *n*" with a type-(ii) system, one could compute which number word represents the size of the set, and compare it to *n*.

Unfortunately, the available empirical data does not provide clear evidence for any of these types of numerical representations over the others. We will assume a type-(ii) system because we think that this is the most natural formulation, given children's counting behavior. Counting appears to be a procedure which takes a set and returns the number word corresponding to its cardinality, not a procedure which takes a number and returns a truth value.

Importantly, assuming a type-(ii) system (or any other type) only determines the form of the inputs and outputs[4]—the inputs are sets and the outputs are number words. Assuming a type-(ii) system does not mean that we have assumed the *correct* input and output pairings. Other conceptual systems can map sets to words, but do it in the "wrong" way: a Mod-10 system would take a set containing *n* elements and return the *n* mod 10th number word.

3.3. Primitive operations in the LOT

In order to define a space of possible lambda expressions, we must specify a set of primitive functional elements which can be composed together to create lambda expressions. These primitives are the basic cognitive components that learners must figure out how to compose in order to arrive at the correct system of numerical meanings. The specific primitives we choose represent only one particular set of choices, but this modeling framework allows others to be explored to see how well they explain learning patterns. The primitives we include can be viewed as partial implementation of the *core knowledge* hypothesis (Spelke, 2003)—they form a core set of computations that learners bring to later development. Unlike core knowledge, however, the primitives we assume are not *necessarily* innate—they must only be available to children by the time they start learning number. These primitives—especially the set-based and logical operations—are likely useful much more broadly in cognition and indeed have been argued to be necessary in other domains. Similar language-like representations using overlapping sets of logical primitives have previously been proposed in learning kinship relations and taxonomy (Katz, Goodman, Kersting, Kemp, & Tenenbaum, 2008), a theory of causality (Goodman, Ullman, & Tenenbaum, 2009), magnetism (Ullman, Goodman, & Tenenbaum, 2010), boolean concepts (Goodman et al., 2008), and functions on sets much like those needed for natural language semantics. We therefore do not take this choice of primitives as specific to number learning, although these primitives may be the only ones which are most relevant. The primitive operations we assume are listed in Table 1.

First, we include a number of primitives for testing small set size cardinalities, *singleton?*, *doubleton?*, *tripleton?*. These respectively test whether a set contains exactly 1, 2, and 3 elements. We include these because the ability of humans to subitize and compare small set cardinalities (Wynn, 1992) suggests that these cognitive operations are especially "easy", especially by the time children start learning number words. In addition, we include a number of functions which manipulate sets. This is motivated in part by children's ability to manipulate sets, and in part by the primitives hypothesized in formalizations of natural language semantics (e.g., Heim & Kratzer, 1998; Steedman, 2000). Semantics often expresses word meanings—especially quantifiers and other function words—as compositions of set-theoretic operations. Such functions are likely used in adult representations and are so simple that it is difficult to see from what basis they could be learned, or why—if they are learned—they should not be learned relatively early. We therefore assume that they are available for learners by the time they start acquiring number word meanings. The functions *select* and *set-difference* play an especially important role in the model: the recursive procedure the model learns for counting the number of objects in a set first selects an element from the set of objects-to-be-counted, removes it via *set-difference*, and recurses. We additionally include logical operations. The function *if* is directly analogous to a conditional expression in a programming language, allowing a function to return one of two values depending on the truth value of a third; our implementation also allows the third argument to be "undefined" (*undef*) in order to return an undefined value if the condition is not met. This function is necessary for most interesting systems of numerical meaning, and is such a basic computation that it is reasonable to assume children have it as an early conceptual resource.

[4] This is analogous to the type signature of a function in computer programming.

Table 1
Primitive operations allowed in the LOT. All valid compositions of these primitives are potential hypotheses for the model.

Functions mapping sets to truth values	
(*singleton? X*)	Returns true iff the set *X* has exactly one element
(*doubleton? X*)	Returns true iff the set *X* has exactly two elements
(*tripleton? X*)	Returns true iff the set *X* has exactly three elements
Functions on sets	
(*set-difference X Y*)	Returns the set that results from removing *Y* from *X*
(*union X Y*)	Returns the union of sets *X* and *Y*
(*intersection X Y*)	Returns the intersect of sets *X* and *Y*
(*select X*)	Returns a set containing a single element from *X*
Logical functions	
(*and P Q*)	Returns TRUE if *P and Q* are both true
(*or P Q*)	Returns TRUE if either *P or Q* is true
(*not P*)	Returns TRUE iff *P* is false
(*if P X Y*)	Returns *X* iff *P* is true, *Y* otherwise
Functions on the counting routine	
(*next W*)	Returns the word after *W* in the counting routine
(*prev W*)	Returns the word before *W* in the counting routine
(*equal-word? W V*)	Returns TRUE if *W* and *V* are the same word
Recursion	
(*L S*)	Returns the result of evaluating the entire current lambda expression on set *S*

The sequence of number words "*one*", "*two*", "*three*", etc. is known to children before they start to learn the words' numerical meanings (Fuson, 1988). In this formal model, this means that the sequential structure of the count list of number words should be available to the learner via some primitive operations. We therefore assume three primitive operations for words in the counting routine: *next*, *prev*, and *equal-word?*. These operate on the domain of *words*, not on the domain of sets or numerical representations. They simply provide functions for moving forwards and backwards on the count list, and checking if two words are equal.[5]

Finally, we allow for recursion via the primitive function *L* permitting the learner to potentially construct a recursive system of word meanings. Recursion has been argued to be a key human ability (Hauser, Chomsky, & Fitch, 2002) and is a core component of many computational systems (e.g., Church, 1936). *L* is the name of the function the learner is trying to infer and this can be used in the definition of *L* itself. That is, *L* is a special primitive in that it maps a set to the word for that set according to the *current* hypothesis (i.e. the hypothesis where *L* is being used). Thus, by including *L* also as a primitive, we allow the learner to potentially use their currently hypothesized meaning for *L* in the definition of *L* itself. One simple example of a recursive definition is,

$$\begin{aligned}&\lambda\, S \cdot (\mathit{if}\,(\mathit{singleton?}\ S)\\ &\quad \text{“}\mathit{one}\text{”}\\ &\quad (\mathit{next}\ (L\ (\mathit{select}\ S)))).\end{aligned}$$

This returns "*one*" for sets of size one. If given a set *S* of size greater than one, it evaluates (*next* (*L* (*select S*))). Here, (*select S*) always is a set of size one since *select* selects a single element. *L* is therefore evaluated the singleton set returned by (*select S*). Because *L* returns the value of the lambda expression it is used in, it returns "*one*" on singleton sets in this example. This means that (*next* (*L* (*select S*))) evaluates to (*next* "*one*"), or "*two*". Thus, this recursive function returns the same value as, for instance, λS. (*if* (*singleton? S*) "*one*" "*two*").

Note that *L* is crucially *not* a successor function. It does not map a number to its successor: it simply evaluates the current hypothesis on some set. Naive use of *L* can give rise to lambda expressions which do not halt, looping infinitely. However, *L* can also be used to construct hypotheses which implement useful computations, including the correct successor function and many other functions. In this sense, *L* is much more basic than a successor function.[6]

It is worthwhile discussing what types of primitives are *not* included in this LOT. Most notably, we do not include a Mod-*N* operation as a primitive. A Mod-*N* primitive might, for instance, take a set and a number word, and return true if the set's cardinality mod *N* is equal to the number word.[7] The reason for not including Mod-*N* is that there is no independent reason for thinking that computing Mod-*N* is a basic ability of young children, unlike logical and set operations. As may be clear, the fact that Mod-*N* is not included as a primitive will be key for explaining why children make the correct CP inference rather than the generalization suggested by Rips, Asmuth, and Bloomfield.[8] Importantly,

[5] It is not clear that children are capable of easily moving *backwards* on the counting list (Baroody, 1984; Fuson, 1984). This may mean that it is better not to include "prev" as a cognitive operation; however, for our purposes, "prev" is relatively unimportant and not used in most of the interesting hypotheses considered by the model. We therefore leave it in and note that it does not affect the performance of the model substantially.

[6] Interestingly, the computational power to use recursion comes for *free* if lambda calculus is the representational system: recursion can be constructed via the *Y*-combinator out of nothing more than the composition laws of lambda calculus. Writing *L* this way, however, is considerably more complex than treating it as a primitive.

[7] That is, if *S* is the set and $|S| = k \cdot N + w$ for some integer *k*, then this function would return true when applied to *S* and the word for *w*.

[8] This means that if very young children could be shown to compute Mod-*N* easily, it would need to be included as a cognitive primitive, and would substantially change the predictions of the model. Thus, the model with its current set of primitives could be argued against by showing that computing Mod-*N* is as easy for children as manipulating small sets.

though, we also do not include a successor function, meaning a function which maps the representation of N to the representation of $N + 1$. While neither a successor function or a Mod-N function is assumed, both can be constructed in this representational system.

3.4. Hypothesis space for the model

The hypothesis space for the learning model consists of *all* ways these primitives can be combined to form lambda expressions—*lexicons*—which map sets to number words. This therefore provides a space of exact numerical meanings. In a certain sense, the learning model is therefore quite restricted in the set of possible meanings it will consider. It will not ever, for instance, map a set to a different concept or a word not on the count list. This restriction is computationally convenient and developmentally plausible. Wynn (1992) provided evidence that children know number words refer to some kind of numerosity before they know their exact meanings. For example, even children who did not know the exact meaning of "*four*" pointed to a display with several objects over a display with few when asked "Can you show me four balloons?" They did not show this patten for nonsense word such as "Can you show me blicket balloons?" Similarly, children map number words to some type of cardinality, even if they do not know which cardinalities (Lipton & Spelke, 2006; Sarnecka & Gelman, 2004). Bloom and Wynn (1997) suggest that perhaps this can be accounted for by a learning mechanism that uses syntactic cues to determine that number words are a class with a certain semantics.

However, within the domain of functions which map sets to words, this hypothesis space is relatively unrestricted. Some example hypotheses are shown in Fig. 1. The hypothesis space contains functions with partial numerical knowledge—for instance, hypotheses that have the correct meaning for "*one*" and "*two*", but not "*three*" or above. For instance, the 2-knower hypothesis takes an argument S, and first checks if (*singleton? S*) is true—if S has one element. If it does, the function returns "*one*". If not, this hypothesis returns the value of (*if* (*doubleton? S*) "*two*" *undef*). This expression is another *if*-statement, one which returns "*two*" if S has two elements, and *undef* otherwise. Thus, this hypothesis represent a 2-knower who has the correct meanings for "*one*" and "*two*", but not for any higher numbers. Intuitively, one could build much more complex and interesting hypotheses in this format—for instance, ones that check more complex properties of S and return other word values.

Fig. 1 also shows an example of a CP-knower lexicon. This function makes use of the counting routine and recursion. First, this function checks if S contains a single element, returning "*one*" if it does. If not, this function calls *set-difference* on S and (*select S*). This has the effect of choosing an element from S and removing it, yielding a

One-knower

```
λ S . (if (singleton? S)
          "one"
          undef)
```

Two-knower

```
λ S . (if (singleton? S)
          "one"
          (if (doubleton? S)
              "two"
              undef))
```

Three-knower

```
λ S . (if (singleton? S)
          "one"
          (if (doubleton? S)
              "two"
              (if (tripleton? S)
                  "three"
                  undef))
```

CP-knower

```
λ S . (if (singleton? S)
          "one"
          (next (L (set-difference S
                                   (select S)))))
```

Singular-Plural

```
λ S . (if (singleton? S)
          "one"
          "two")
```

Mod-5

```
λ S . (if (or (singleton? S)
              (equal-word? (L (set-difference S)
                              (select S))
                           "five"))
          "one"
          (next (L (set-difference S
                                   (select S)))))
```

2-not-1-knower

```
λ S . (if (doubleton? S)
          "two"
          undef)
```

2N-knower

```
λ S . (if (singleton? S)
          "one"
          (next (next (L (set-difference S (select S))))))
```

Fig. 1. Example hypotheses in the LOT. These include subset-knower, CP-knower, and Mod-N hypotheses. The actual hypothesis space for this model is infinite, including all expressions which can be constructed in the LOT.

new set with one fewer element. The function calls L on this set with one fewer element, and returns the *next* number after the value returned by L. Thus, the CP-knower lexicon represents a function which recurses down through the set S until it contains a singleton element, and up the counting routine, arriving at the correct word. This is a version of bootstrapping in which children would discover that they move "one more" element on the counting list for every additional element in the set-to-be-counted.

Importantly, this framework can learn a number of other types of conceptual systems. For example, the Mod-5 system is similar to the CP-knower, except that it returns "*one*" if S is a singleton, or the word before for the set S minus an element is "*four*". Intuitively, this lexicon works similarly to the CP-knower lexicon for set sizes 1 through 4. However, on a set of size 5, the lexicon will find that (L (*set-difference* S) (*select* S)) is equal to "*four*", meaning that the first *if* statement returns "*one*": sets of size 5 map to "*one*". Because of the recursive form of this lexicon, sets of size 6 will map to "*two*", etc.

Fig. 1 also contains a few of the other hypotheses expressible in this LOT. For instance, there is a singular/plural hypothesis, which maps sets of size 1 to "*one*" and everything else to "*two*". There is also a $2N$ lexicon which maps a set of size N to the $2 \cdot N$th number word, and one which has the correct meaning for "*two*" but not "*one*".

It is important to emphasize that Fig. 1 does not contain the complete list of hypotheses for this learning model. The complete hypothesis space is infinite and corresponds to all possible ways to compose the above primitive operations. These examples are meant only to illustrate the types of hypotheses which could be expressed in this LOT, and the fact that many are not like natural number.

3.5. The probabilistic model

So far, we have defined a space of functions from sets to number words. This space was general enough to include many different types of potential representational systems. However, we have not yet specified how a learner is to choose between the available hypotheses, given some set of observed data. For this, we use a probabilistic model built on the intuition that the learner should attempt to trade-off two desiderata. On the one hand, the learner should prefer "simple" hypotheses. Roughly, this means that the lexicon should have a short description in the language of thought. On the other hand, the learner should find a lexicon which can explain the patterns of usage seen in the world. Bayes' rule provides a principled and optimal way to balance between these desiderata.

We suppose that the learner hears a sequence of number words $W = \{w_1, w_2, \ldots\}$. Each number word is paired with an object type $T = \{t_1, t_2, \ldots\}$ and a context set of objects $C = \{c_1, c_2, \ldots\}$. For instance, a learner might hear the expression "two cats" (w_i = "two" and t_i = "cats") in a context containing a number of objects,

$$c_i = \{\text{cat}_A,\ \text{horse}_A,\ \text{dog}_A,\ \text{dog}_B,\ \text{cat}_B,\ \text{dog}_C\}. \tag{2}$$

If L is an expression in the LOT—for instance, one in Fig. 1—then by Bayes rule we have

$$P(L|W,T,C) \propto P(W|T,C,L)P(L) = \left[\prod_i P(w_i|t_i,c_i,L)\right]P(L) \tag{3}$$

under the assumption that the w_i are independent given the observed data L. This equation says that the probability of any lexicon L given the observed data W, T, C is proportional to the prior $P(L)$ times the likelihood $P(W \mid T,C,L)$. The prior gives the learner's *a priori* expectations that a particular hypothesis L is correct. The likelihood gives the probability of the observed number words W occurring given that the hypothesis L is correct, providing a measure of how well L explains or predicts the observed number words. We discuss each of these terms in turn.

The prior $P(L)$ has two key assumptions. First, hypotheses which are more complex are assumed to be less likely a priori. We use the *rational rules* prior (Goodman et al., 2008), which was originally proposed as a model of rule-based concept learning. This prior favors lexicons which re-use primitive components, and penalizes complex, long expressions. To use this prior, we construct a probabilistic context free grammar using all expansions consistent with the argument and return types of the primitive functions in Table 1. The probability of a lambda expression is determined by the probability it was generated from this grammar, integrating over rule production probabilities.[9]

The second key assumption for the prior is that recursive lexicons are less likely a priori. We introduce this extra penalty for recursion because it seems natural that recursion is an additionally complex operation. Unlike the other primitive operations, recursion requires a potentially unbounded memory space—a *stack*—for keeping track of which call to L is currently being evaluated. Every call to L also costs more time and computational resources than other primitives since using L requires evaluating a whole lambda expression—potentially even with its own calls to L. We therefore introduce a free parameter, γ, which penalizes lexicons which use of recursion:

$$P(L) \propto \begin{cases} \gamma \cdot P_{RR}(L) & \text{if } L \text{ uses recursion} \\ (1-\gamma) \cdot P_{RR}(L) & \text{otherwise} \end{cases} \tag{4}$$

where $P_{RR}(L)$ is the prior of L according to the rational rules model.

We use a simple form of the likelihood, $P(w_i \mid t_i,c_i,L)$, that is most easily formulated as a generative model. We first evaluate L on the set of all objects in c_i of type t_i. For instance suppose that c_i is the set in (2) and t_i = "cat", we would first take only objects of type "cat", $\{\text{cat}_A, \text{cat}_B\}$[10] We then evaluate L on this set, resulting in either a number word or *undef*. If the result is *undef*, we generate a number word uniformly at random, although we note that this is a simplification, as children appear to choose words from a non-uniform baseline distribution when they do not know

[9] Similar results are found using simply the PCFG production probability as a prior.

[10] Note that since we condition on observing the types t_i, the unmentioned types of objects in a context are irrelevant to the likelihood.

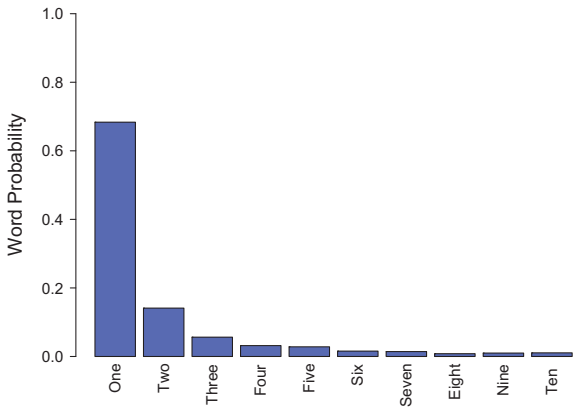

Fig. 2. Number word frequencies from CHILDES (MacWhinney, 2000) used to simulate learning data for the model.

the correct word (see Lee & Sarnecka, 2010b, 2010a; Sarnecka & Lee, 2009). If the result is not *undef*, with high probability α, we produce the computed number word; with low probability $1-\alpha$ we produce the another word, choosing uniformly at random from the count list. Thus,

$$P(w_i|t_i,c_i,L) = \begin{cases} \frac{1}{N} & \text{if } L \text{ yields } \textit{undef} \\ \alpha + (1-\alpha)\frac{1}{N} & \text{if } L \text{ yields } w_i \\ (1-\alpha)\frac{1}{N} & \text{if } L \text{ does not yield } w_i \end{cases} \tag{5}$$

where N is the length of the count routine.[11] This likelihood reflects the fact that speakers will typically use the correct number word for a set. But occasionally, the listener will misinterpret what is being referred to and will hear an incorrectly paired number word and set. This likelihood therefore penalizes lexicons which generate words for each set which do not closely follow the observed usage. It also penalizes hypotheses which make incorrect predictions over those which return *undef*, meaning that it is better for a learner to remain uncommitted than to make a strong incorrect predictions.[12] The likelihood uses a free parameter, α, which controls the degree to which the learner is penalized for data which are not predicted by their hypothesis.

To create data for the learning model, we simulated noisy pairing of words and sets of objects, where the word frequencies approximate the naturalistic word probabilities in child-directed speech from CHILDES (MacWhinney, 2000). We used all English transcripts with children aged between 20 and 40 months to compute these probabilities. This distribution is shown in Fig. 2. Note that all occurrences of number words were used to compute these probabilities, regardless of their annotated syntactic type. This was because examination of the data revealed many instances in which it is not clear if labeled pronoun usages actually have numerical content—e.g., "give me one" and "do you want one?" We therefore simply used the raw counts of number words. This provides a distribution of number words much like that observed cross-linguistically by Dehaene and Mehler (1992), but likely overestimates the probability of "*one*". Noisy data that fits the generative assumptions of the model was created for the learner by pairing each set size with the correct word with probability α, and with a uniformly chosen word with probability $1-\alpha$.

3.6. Inference & methods

The previous section established a formal probabilistic model which assigns any potential hypothesized numerical system L a probability, conditioning on some observed data consisting of sets and word-types. This probabilistic model defines the probability of a lambda expression, but does not say how one might find high-probability hypotheses or compute predicted behavioral patterns. To solve these problems, we use a general inference algorithm similar to the tree-substitution Markov-chain Monte-Carlo (MCMC) sampling used in the rational rules model (Goodman et al., 2008).

This algorithm essentially performs a stochastic search through the space of hypotheses L. For each hypothesized lexicon L, a change is proposed to L by resampling one piece of a lambda expression in L according to a PCFG. The change is accepted with a certain probability such that in the limit, this process can be shown to generate samples from the posterior distribution $P(L \mid W,T,C)$. This process builds up hypotheses by making changes to small pieces of the hypothesis: the entire hypothesis space need not be explicitly enumerated and tested. Although the hypothesis space is in principle infinite, the "good" hypotheses can be found by this technique since they will be high-probability, and this sampling procedure finds regions of high probability.

This process is not necessarily intended as an algorithmic theory for how children actually discover the correct lexicon (though see Ullman et al., 2010). Children's actual discovery of the correct lexicon probably relies on numerous other cues and cognitive processes and likely does not progress through such a simple random search. Our model is intended as a computational level model (Marr, 1982), which aims to explain children's behavior in terms of how an idealized statistical learner would behave. Our evaluation of the model will rely on seeing if our idealized model's degree of belief in each lexicon is predictive of the correct behavioral pattern as data accumulates during development.

To ensure that we found the highest probability lexicons for each amount of data, we ran this process for one million MCMC steps, for varying γ and amounts of data from 1 to 1000 pairs of sets, words, and types. This number of MCMC steps was much more than was strictly necessary to find the high probability lexicons and children could search a much smaller effective space. Running MCMC for longer than necessary ensures that no unexpectedly good lexicons were missed during the search, allowing us to fully evaluate predictions of the idealized model. In the MCMC run we analytically computed the expected log likelihood of a data point for each lexicon, rather than using simulated data sets. This allowed each lexicon to be efficiently evaluated on multiple amounts of data.

Ideally, we would be able to compute the exact posterior probability of $P(L \mid W,T,C)$ for any lexicon L. However, Eq. (3)

[11] The second line is "$\alpha + (1-\alpha)\frac{1}{N}$" instead of just "$\alpha$" since the correct word w_i can be generated either by producing the correct word with probability α or by generating uniformly with probability $1-\alpha$.

[12] It would be interesting to computationally study the relationship of number learning to acquisition of other quantifiers, since they would likely be other alternatives that could be considered in the likelihood.

only specifies something *proportional* to this probability. This is sufficient for the MCMC algorithm, and thus would be enough for any child engaging in a stochastic search through the space of hypotheses. However, to compute the model's predicted distribution of responses, we used a form of selective model averaging (Hoeting, Madigan, Raftery, & Volinsky, 1999; Madigan & Raftery, 1994), looking at all hypotheses which had a posterior probability in the top 1000 for any amount of data during the MCMC runs. This resulted in approximately 11,000 hypotheses. Solely for the purpose of computing $P(L \mid W,T,C)$, these hypotheses were treated as a fixed, finite hypothesis space. This finite hypothesis space was also used to compute model predictions for various γ and α. Because most hypotheses outside of the top 1000 are extremely low probability, this provides a close approximation to the true distribution $P(L \mid W,T,C)$.

4. Results

We first show results for learning natural numbers from naturalistic data. After that, we apply the same model to other data sets.

4.1. Learning natural number

The precise learning pattern for the model depends somewhat on the parameter values α and γ. We first look at typical parameter values that give the empirically demonstrated learning pattern, and then examine how robust the model is to changing these parameters.

Fig. 3 shows learning curves for the behavioral pattern exhibited by the model for $\alpha = 0.75$ and $\log \gamma = -25$. This plot shows the marginal probability of each type of behavior, meaning that each line represents the sum of the posterior probability all hypotheses that show a given type of behavior. For instance, the 2-knower line shows the sum of the posterior probability of all LOT expressions which map sets of size 1 to "*one*", sets of size 2 to "*two*", and everything else to *undef*. Intuitively, this marginal probability corresponds to the proportion of children who should look like subset- or CP-knowers at each point in time. This figure shows that the model exhibits the correct developmental pattern. The first gray line on the left represents several different hypotheses which are high probability in the prior—such as all sets map to the same word, or undefined—and are quickly dispreferred. The model then successively learns the meaning of "*one*", then "*two*", "*three*", finally transitioning to a CP-knower which correctly represents meaning of all number words. That is, with very little data the "best" hypothesis is one which looks like a 1-knower, and as more and more data is accumulated, the model transitions through subset-knowers. Eventually, the model accumulates enough evidence to justify the CP-knower lexicon that recursively defines all number words on the count list. At that point, the model

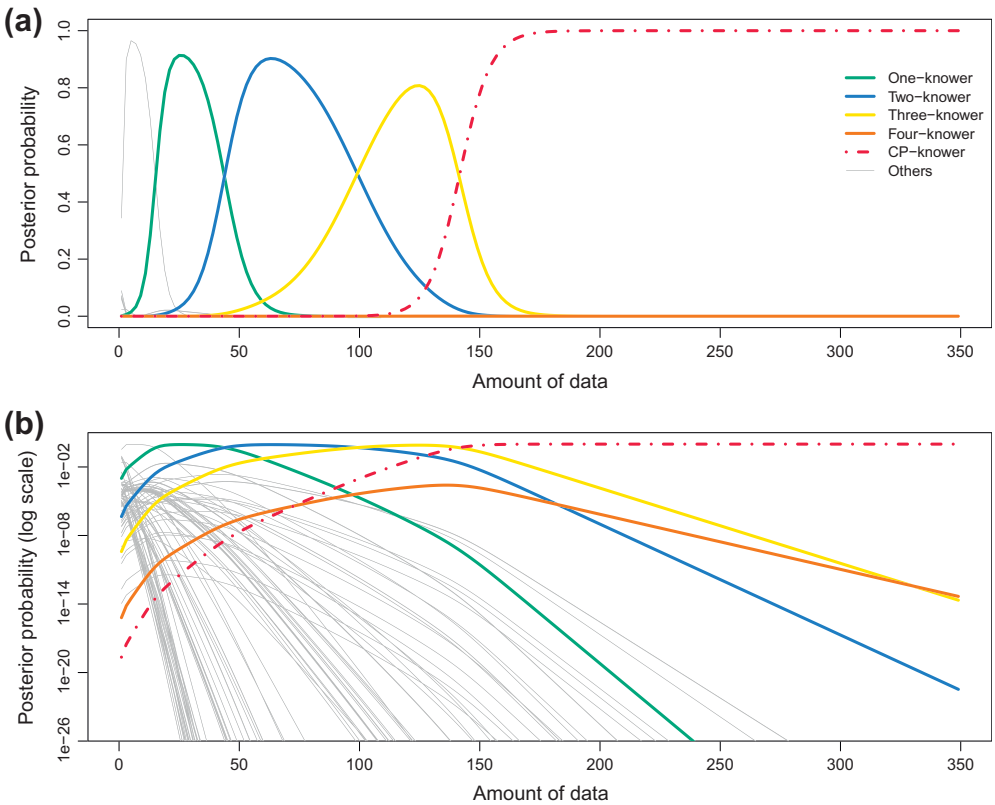

Fig. 3. (a) shows marginal posteriors probability of exhibiting each type of behavior, as a function of amount of data and (b) shows the same plot on a log *y*-axis demonstrating the large number of other numerical systems which are considered, but found to be unlikely given the data.

Table 2
Several hand-selected example hypotheses at 200 data points.

Rank	Log posterior	Behavioral pattern	LOT expression
1	−0.93	(1 2 3 4 5 6 7 8 9 10)	λ *S* · (*if* (*singleton? S*) "*one*" (*next* (*C* (*set-difference S* (*select S*)))))
2	−2.54	(1 2 3 4 5 6 7 8 9 10)	λ *S* · (*if* (*singleton? S*) "*one*" (*next* (*C* (*set-difference S* (*select* (*select S*))))))
3	−3.23	(1 2 3 4 5 6 7 8 9 10)	λ *S* · (*if* (*not* (*singleton? S*)) (*next* (*C* (*set-difference S* (*select S*)))) "*one*)
1423	−11.92	(1 2 3 U U U U U U U)	λ *S* · (*if* (*tripleton? S*) *three* (*if* (*singleton? S*) "*one*" (*if* (*doubleton? S*) "*two*" U)))
1604	−14.12	(1 2 3 U U U U U U U)	λ *S* · (*if* (*tripleton? S*) "*three*" (*if* (*doubleton?* (*union S S*)) "*one*" (*if* (*doubleton? S*) "*two*" *U*)))
4763	−20.04	(1 2 U U U U U U U U)	λ *S* · (*if* (*doubleton?* (*union S S*)) *one* (*if* (*doubleton? S*) "*two*" *U*))
7739	−39.42	(1 U 3 U U U U U U U)	λ *S* · (*if* (*tripleton? S*) *three* (*if* (*singleton? S*) "*one*" *U*))
7756	−44.68	(1 U U U U U U U U U)	λ *S* · (*if* (*singleton? S*) "*one*" *U*)
7765	−49.29	(1 2 4 4 4 4 4 4 4 4)	λ *S* · (*if* (*doubleton? S*) "*two*" (*if* (*singleton? S*) "*one*" *four*"))
9410	−61.12	(1 2 1 1 1 1 1 1 1 1)	λ *S* · (*if* (*not* (*not* (*doubleton? S*))) "*two*" "*one*")
9411	−61.12	(1 2 2 2 2 2 2 2 2 2)	λ *S* · (*if* (*not* (*not* (*singleton? S*))) "*one*" "*two*")
9636	−71.00	(1 2 7 1 1 1 1 1 1 1)	λ *S* · (*prev* (*prev* (*if* (*doubleton? S*) "*four*" (*if* (*tripleton? S*) "*nine*" "*three*"))))
9686	−100.76	(1 3 3 3 3 3 3 3 3 3)	λ *S* · (*if* (*singleton? S*) "*one*" "*three*")
9695	−103.65	(1 1 3 1 1 1 1 1 1 1)	λ *S* · (*if* (*tripleton? S*) (*prev* "*four*") "*one*)
9765	−126.01	(1 1 1 1 1 1 1 1 1 1)	λ *S* · (*next* (*prev* "*one*))
11126	−396.78	(2 U U U U U U U U U)	λ *S* · (*if* (*singleton? S*) "*two*" *U*)
11210	−444.68	(2 U 2 2 2 2 2 2 2 2)	λ *S* · (*if* (*not* (*doubleton? S*)) "*two*' *U*)

exhibits a conceptual re-organization, changing to a hypothesis in which all number word meanings are defined recursively as in the CP-knower lexicon in Fig. 1.

The reason for the model's developmental pattern is the fact that Bayes' theorem implements a simple trade-off between complexity and fit to data: with little data, hypotheses are preferred which are simple, even if they do not explain all of the data. The numerical systems which are learned earlier are simple, or higher prior probability in the LOT. In addition, the data the model receives follows word frequency distributions in CHILDES, in which the earlier number words are more frequent. This means that it is "better" for the model to explain the more frequent number words. Number word frequency falls off with the number word's magnitude, meaning that, for instance, just knowing "*one*" is a better approximation than just knowing "*two*": children become 1-knowers before they become 2-knowers. As the amount of data increases, increasingly complex hypotheses become justified. The CP-knower lexicon is most "complex," but also optimally explains the data since it best predicts when each number word is most likely to be uttered.

The model prefers hypotheses which leave later number words as *undef* because it is better to predict *undef* than the wrong answer: in the model, each word has likelihood $1/N$ when the model predicts *undef*, but $(1-\alpha)/N$ when the model predicts incorrectly. This means that a hypothesis like λ *S*. (*if* (*singleton? S*) "*one*" *undef*) is preferred in the likelihood to λ *S*. "*one*". If the model did not employ this preference for non-commitment (*undef*) over incorrect guesses, the 1-knower stage of the model would predict children say "*one*" to sets of all sizes. Thus, this assumption of the likelihood drives learners to avoid incorrectly guessing meanings for higher number words, preferring to not assign them any specific numerical meaning.

Fig. 3b shows the *same* curves with a log *y*-axis, making clear that many other types of hypotheses are considered by the model and found to have low probability. Each gray line represents a different kind of knower-level behavior—for instance, there is a line for correctly learning "*two*" and not "*one*", a line for thinking "*two*" is true of sets of size 1, and dozens of other behavioral patterns representing thousands of other LOT expressions. These are all given very low probability, showing the data that children plausibly receive is sufficient to rule out many other kinds of behavior. This is desirable behavior for the model because it shows that the model needs not have strong *a priori* knowledge of how the numerical system works. Many different kinds of functions could be built, considered by children, and ruled-out based on the observed data.

Table 2 shows a number of example hypotheses chosen by hand. This table lists each hypothesis' behavioral pattern and log probability after 200 data points. The behavioral patterns show what sets of each size are mapped to[13]: for instance, "(1 2 U U U U U U U U)" means that sets of size 1 are mapped to "*one*" ("1"), sets of size 2 are mapped to "*two*" ("2"), and all other sets are mapped to *undef* ("U"). Thus, this behavior is consistent with a 2-knower. As this table makes clear, the MCMC algorithm used here searches a wide variety of LOT expressions. Most of these hypotheses have near-zero probability, indicating that the data are sufficient to rule out many bizarre and non-attested developmental patterns.

Fig. 4 shows the behavioral pattern for different values of γ and α. Fig. 4a and b demonstrate that the learning rate depends on α: when α is small, the model takes more data points to arrive at the correct grammar. Intuitively this is sensible because α controls the degree to which the model is penalized for an incorrect mapping from sets to words, meaning that when α is small, the model takes more data to justify a jump to the more complex CP-knower hypothesis.

Fig. 4c–f demonstrate the behavior of the model as γ is changed. Fig. 4c and d show that a range of log γ that roughly shows the developmental pattern is from approximately −20 to −65, a range of 45 in log space or over nineteen orders of magnitude in probability space. Even though we do not know the value of γ, a large range of values show the observed developmental pattern.

Fig. 4e and f shows what happens as log γ is made even more extreme. The plot for $\gamma = \frac{1}{2}(\log\gamma = -0.69)$

[13] For conciseness, we use "U" for "undef", the numeral 1 for "one", 2 for "two," etc. in this table.

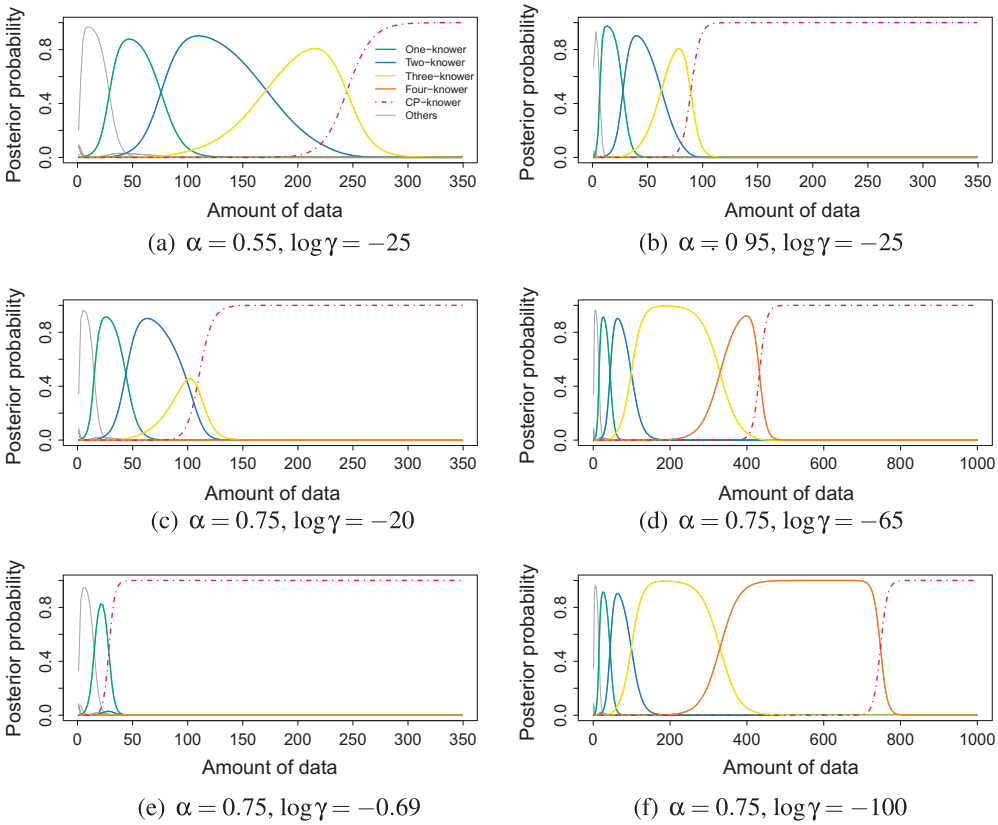

Fig. 4. Behavioral patterns for different values of α and γ. Note that the x-axis scale is different for d and f.

corresponds to *no* additional penalty on recursive hypotheses. This shows a CP-transition too early—roughly after becoming a 1-knower. The reason for this may be clear from Fig. 1: the CP-knower hypothesis is not much more complex than the 2-knower in terms of overall length, but does explain much more of the observed data. If log $\gamma = -100$, the model is strongly biased against recursive expressions and goes through a prolonged 4-knower stage. This curve also shows a long three-knower stage, which might be mitigated by including *quadrupleton?* as a primitive. In general, however, it will take increasing amounts of data to justify moving to the next knower-level because of the power-law distribution of number word occurrences—higher number words are much less frequent. The duration of the last knower-level before the CP-transition, though, depends largely on γ.

These results show that the dependence of the model on the parameters is fairly intuitive, and the general pattern of knower-level stages preceding CP-transition is a robust property of this model. Because γ is a free parameter, the model is not capable of predicting or explaining the precise location of the CP-transition. However, these results show that the behavior of the model is not extremely sensitive to the parameters: there is no parameter setting, for instance, that will make the model learn low-ranked hypotheses in Table 2. This means that the model can explain why children learn the correct numerical system instead of any other possible expression which can be expressed in the LOT.

Next, we show that the model is capable of learning other systems of knowledge when given the appropriate data.

4.2. Learning singular/plural

An example singular/plural system is shown in Fig. 1. Such a system differs from subset-knower systems in that all number words greater than "*one*" are mapped to "*two*". It also differs from the CP-knower system in that it uses no recursion. To test learning for singular/plural cognitive representations, the model was provided with the same data as in the natural number case, but sets with one element were labeled with "*one*" and sets with two or more elements were labeled with "*two*". Here, "*one*" and "*two*" are just convenient names for our purposes—one could equivalently consider the labels to be singular and plural morphology.

As Fig. 5 shows, this conceptual system is easily learnable within this framework. Early on in learning this distinction, even simpler hypotheses than singular/plural are considered: λ *S.* "*one*" and λ *S.* "*two*. These hypotheses are almost trivial, but correspond to learners who initially do not distinguish between singular and plural markings—a simple, but developmentally attested pattern (Barner, Thalwitz, Wood, Yang, & Carey, 2007). Here, λ *S.* "*one*" is higher probability than λ *S.* "*two*" because sets of size 1 are more frequent in the input data. Eventually, the model learns the correct hypothesis, corresponding to the singular/plural hypothesis shown in Fig. 1. This distinction

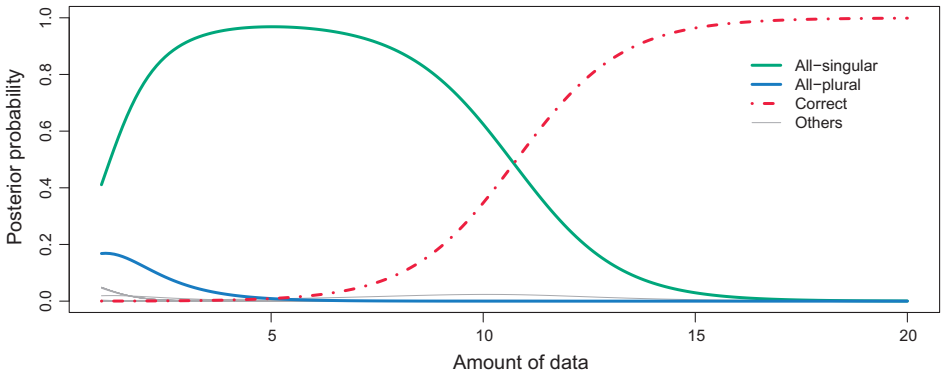

Fig. 5. Learning results for a singular/plural system.

is learned very quickly by the model compared to the number hypotheses, matching the fact that children learn the singular/plural distinction relatively young, by about 24 months (Kouider, Halberda, Wood, & Carey, 2006). These results show one way that natural number is not merely "built in": when given the different kind of data—the kind that children presumably receive in learning singular/plural morphology—the model infers a singular/plural system of knowledge.

4.3. Learning Mod-N systems

Mod-*N* systems are interesting in part because they correspond to an inductive leap consistent with the correct meanings for early number words. Additionally, children do learn conceptual systems with Mod-*N*-like structures. Many measures of time—for instance, days of the week, months of the year, hours of the day—are modular. In numerical systems, children eventually learn the distinction between even and odd numbers, as well as concepts like "multiples of ten". Rips et al. (2008) even report anecdotal evidence from Hartnett (1991) of a child arriving at a Mod-1,000,100 system for natural number meanings. They quote,

> D.S.: The numbers only go to million and ninety-nine.
> Experimenter: What happens after million and ninety-nine?
> D.S.: You go back to zero.
> E: I start all over again? So, the numbers do have an end?
> Or do the numbers go on and on?
> D.S.: Well, everybody says numbers go on and on because you start over again with million and ninety-nine.
> E:. ..you start all over again.
> D.S.: Yeah, you go zero, one, two, three, four—all the way up to million and ninety-nine, and then you start all over again.
> E: How about if I tell you that there is a number after that?
> A million one hundred.
> D.S.: Well, I wish there was a million and one hundred, but there isnt.

When the model is given natural number data Mod-*N* systems are given low probability because of their complexity and inability to explain the data. A Mod-*N* system makes the wrong predictions about what number words should be used for sets larger than size N, although such data is rare for large *N*. As Fig. 1 shows, modular systems are also considerably more complex than a CP-knower lexicon, meaning that they will be dispreferred even for large N, where presumably children have not received enough data. This means that Mod-*N* systems are doubly dispreferred when the learner observes natural number data.

To test whether the model could learn a Mod-*N* system when the data support it, data was generated by using the same distribution of set sizes as for learning natural number, but sets were labeled according to a Mod-5 system. This means that the data presented to the learner was identical for "*one*" through "*five*", but sets of size 6 were paired with "*one*", sets of size 7 were paired with "*two*", etc. As Fig. 6 shows, the model is capable of learning from this data, and arriving at the correct Mod-5 system.[14] Interestingly, the Mod-learner shows similar developmental patterns to the natural number learners, progressing through the correct sequence of subset-knower stages before making the Mod-5-CP-transition. This results from the fact that the data for the Mod system is very similar to the natural number data for the lower and more frequent set sizes. In addition, since both models use the same representational language, they have the same inductive biases, and thus both prefer the "simpler" subset-knower lexicons initially. A main difference is that hypotheses other than the subset-knowers do well early on with Mod-5 data. For instance, a hypothesis which maps all sets to "*one*" has higher probability than in learning number because "*one*" is used for more than one set size.

The ability of the model to learn Mod-5 systems demonstrates that the natural numbers are no more "built-in" to this model than modular-systems: given the right data, the model will learn either. As far as we know, there is no other computational model capable of arriving at these types of distinct, rule-based generalizations.

[14] Because of the complexity of the Mod-5 knower, special proposals to the MCMC algorithm were used which preferentially propose certain types of recursive definitions. This did not change the form of the resulting probabilistic model.

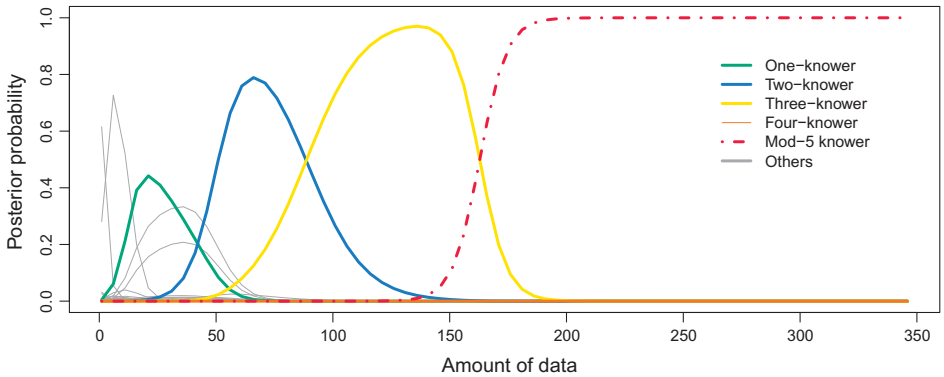

Fig. 6. Learning results for a Mod-5 system.

5. Discussion

We have presented a computational model which is capable of learning a recursive numerical system by doing statistical inference over a structured language of thought. We have shown that this model is capable of learning number concepts, in a way similar to children, as well as other types of conceptual systems children eventually acquire. Our model has aimed to clarify the conceptual resources that may be necessary for number acquisition and what inductive pressures may lead to the CP-transition. This work was motivated in part by an argument that Carey's formulation of bootstrapping actually presupposes natural numbers, since children would have to know the structure of natural numbers in order to avoid other logically plausible generalizations of the first few number word meanings. In particular, there are logically possible modular number systems which cannot be ruled out given only a few number word meanings (Rips et al., 2006; Rips et al., 2008; Rips et al., 2008). Our model directly addresses one type of modular system along these lines: in our version of a Mod-*N* knower, sets of size *k* are mapped to the *k* mod *N*th number word. We have shown that these circular systems of meaning are simply less likely hypotheses for learners. The model therefore demonstrates how learners might avoid some logically possible generalizations from data, and furthermore demonstrates that dramatic inductive leaps like the CP-transition should be expected for ideal learners. This provides a "proof of concept" for bootstrapping.

In implementing a fully working version of this model we have had to make several design choices about the representational system and statistical model. These choices of course affect the final "adult state" of the model. It is useful to distinguish between the choices made in our specific implementation of the model, and our general approach to understanding numerical development. It might turn out, for instance, that children's number word meanings are lower-bounded (with "one" meaning "one or more") or that the representational system we assume is either too weak or too powerful. While such discoveries may be inconsistent with our specific implementation, one could modify the model's representational basis to accommodate such facts.

We have discussed empirical data that provide support for our modeling approach, but it is also important to ask what kind of empirical findings would or would weigh against our model. Evidence against our most fundamental claim—numerical knowledge comes from statistical inference over a structured representational system—could be found by discovering developmental patterns that are inconsistent with the predictions of a statistical model operating on a plausible representational system—for instance, representational changes that cannot be explained as a response to evidence. Or, one might show that children's representational primitives give an inductive bias unlike that required for the model to work[15].

It is tempting to suppose that because the model discovers a recursive form of numerical meaning, it must have knowledge of "the successor principle". However, the model produces explicit representations only of functions which map sets to truth values, not of, for instance, counting principles. Knowing that "one more element" is "one more on the count list" is only *implicit* in the computations performed by the model. Explicit knowledge of successorship and counting might require "looking at" the representations learned by the model and noticing that, for instance, every additional element makes the model return one higher word on the count list. Knowledge of such abstract properties of numbers does not come for free, even to learners who have discovered a function that can correctly map sets to number words.

This work leaves open the question of whether our approach can learn the *full* concept of natural number. This is a subtle issue because it is unclear what it means to have this concept (though see Leslie et al., 2008). Does it require knowledge that there are an infinity of number concepts, or number words? What facts about numbers must be explicitly represented, and which can be implicitly represented? The model we present learns natural number concepts in the sense that it relates an infinite number of set-theoretic concepts to a potentially infinite list of number words, using only finite evidence. However, the present work does not directly address what may be an equally interesting inductive problem relevant to a full natural

[15] For instance, as above, if *Mod* was a primitive.

number concept: how children learn that *next* always yields a new number word. If it was the case that *next* at some point yielded a previous number word—perhaps (*next* "*fifty*") is equal to "*one*"—then learners would again face a problem of a modular number system. Rips, Asmuth, and Bloomfield's arguments about the need to rule out alternative modular number-system hypotheses could apply to both the Mod-*N* systems we considered earlier or the challenge described here, in which *next* is defined cyclically. The latter framing is not directly addressed by our work here. But, it is likely that similar methods to those that we use to solve the inductive problem of mapping words to functions could also be applied to learn that *next* always maps to a new word. It would be surprising if *next* mapped to a new word for 50 examples, but not for the 51st. Thus, the most concise generalization from such finite evidence is likely that *next* never maps to a previous linguistic symbol.[16]

In addition, the model developed here suggests a number of hypotheses for how numerical acquisition may progress:

5.1. The CP transition may result from bootstrapping in a general representation language

This work was motivated in large part by the critique of bootstrapping put forth by Rips et al. (2006), Rips et al. (2008), and Rips et al. (2008). They argued that bootstrapping presupposed a system isomorphic to natural numbers; indeed, it is difficult to imagine a computational system which could not be construed as "building in" natural numbers. Even the most basic syntactic formalisms—for instance, finite-state grammars—can create a discrete infinity of expressions which are isomorphic to natural numbers. This is true in our LOT: for instance the LOT can generate expressions like (*next x*), (*next* (*next x*)), (*next* (*next* (*next x*))), etc.

However, dismissing the model as "building in" natural number would miss several important points. First, there is a difference between representations which could be interpreted *externally* as isomorphic to natural numbers, and those which play the role *internally* as natural number representations. An outside observer could interpret LOT expressions as isomorphic to natural numbers, even though they do not play the role of natural numbers in the computational system. We have been precise that the space of number meanings we consider are those which *map sets to words*: the only things with numerical content in our formalization are functions which take a set as input and return a number word. Among the objects which have numerical content, we did not assume a successor function: none of the conceptual primitives take a function mapping sets of size N to the Nth word, and give you back a function mapping sets of size $N+1$ to the $N+1$st word. Instead, the correct successor function is embodied as a recursive function on sets, and is only one of the potentially learnable hypotheses for the model. Our model therefore demonstrates that a bootstrapping theory is not inherently incoherent or circular, and can be both formalized and implemented.

However, our version of bootstrapping is somewhat different from Carey's original formulation. The model bootstraps in the sense that it recursively defines the meaning for each number word in terms of the previous number word. This is representational change much like Carey's theory since the CP-knower uses primitives not used by subset knowers, and in the CP-transition, the computations that support early number word meanings are fundamentally revised. However, unlike Carey's proposal, this bootstrapping is *not* driven by an analogy to the first several number word meanings. Instead, the bootstrapping occurs because at a certain point learners receive more evidence than can be explained by subset-knowers. According to the model, children who receive evidence only about the first three number words would never make the CP-transition because a simpler 3-knower hypothesis could better explain all of their observed data. A distinct alternative is Carey's theory that children make the CP-transition via analogy, looking at their early number word meanings and noticing a correspondence between set size and the count list. Such a theory might predict a CP-transition even when the learner's data only contains the lowest number words, although it is unclear what force would drive conceptual change if all data could be explained by a simpler system.[17]

5.2. The developmental progression of number acquisition may result from statistical inference

We have shown that the developmental trajectory of the model tracks children's empirically observed progression through levels of numerical knowledge. In the model, this behavior results from both the prior and the likelihood, which were chosen to embody reasonable inferential principles: learners should prefer simple hypotheses which can explain observed patterns of word usage.

The fact that these two ingredients combine to show a developmental progression anything like children is surprising and we believe potentially quite informative. One could imagine that the model, operating in this somewhat unconstrained hypothesis space, would show a developmental pattern nothing like children—perhaps learning other bizarre hypotheses and not the subset- and CP-knower hypotheses. The fact that the model does look like children provides some evidence that statistical inference may be the driving force in number-word learning. This theory has the advantage that it explains why acquisition proceeds though the observed regular stages—why don't children exhibit more chaotic patterns of acquisition? Why should children show these developmental stages at all? Children should show these patterns because, under

[16] In some programming languages, such as *scheme*, there is even a single primitive function *gensym*, for creating new symbols in this manner.

[17] These two alternatives could be experimentally distinguished by manipulating the type of evidence 3-knowers receive. While it might not be ethical to deprive 3-knowers of data about larger number words, analogy theories may predict no effect of *additional* data about larger cardinalities, while our implementation of bootstrapping as statistical inference does.

the assumptions of the model, it is the way an ideal learner should trade off complexity and fit to data.

Additionally, the model addresses the question of why the CP-transition happens so suddenly. Why isn't it the case that children successively acquire meanings for later number words? Why is the acquisition all-or-none? The answer provided by the model is that the representational system may be discrete: at some point it becomes better to restructure the entire conceptual system and jump to a different LOT expression. Our model also predicts recent findings that the amount of evidence children receive about numbers correlates with their knowledge of cardinal meanings, even controlling for other factors such as socioeconomic status (Levine, Suriyakham, Rowe, Huttenlocher, & Gunderson, 2010): unlike maturational or strongly nativist theories, idealized statistical learners are highly sensitive to amount of evidence.

5.3. The timing of the CP-transition may depend on the primitives in the LOT and the price of recursion

Carey (2009) suggests that the limitations of children's small-set representational system may drive the timing of the CP-transition. In what appears to be an otherwise puzzling coincidence, children become CP-knowers roughly at the capacity limit of their small-set representational system—around 3 or 4. At this point, children may need to use another strategy like counting to understand exact numerosities, and thus decide to transition to a CP-knower system.

In the model, the timing of the CP-transition depends both on the primitives in the LOT, and the value of γ, the free parameter controlling the additional cost of recursion. The fact that the timing of the model's CP-transition depends on a free parameter means that the model does not strongly predict when the CP-transition will occur. However, the point at which it does occur depends on which primitives are allowed in the LOT. While a full exploration of how the choice of primitives impacts learning is beyond the scope of the current paper, our own informal experimentation shows an intuitive relationship between the primitives and the timing of the CP-transition. For instance, removing *doubleton?* and *tripleton?* will make the CP-transition occur earlier since it makes 2-knowers and 3-knowers more complex. Including *quadrupleton?* and *quintupleton?* would push the CP-transition beyond "*four*" or "*five*" for appropriate settings of γ. Fully and quantitatively exploring how the choice of primitives impacts the learning is an interesting and important direction for future work. In general, the model can be viewed as realizing a theory very much like Carey's proposal: the CP-transition occurs when the learner runs out of small primitive set operations which allow them to simply define early number word meanings, though, in our model the primitives do not solely determine the timing of the CP transition since the transition also depends on the cost of recursion.

Alternatively, it may turn out that algorithmic considerations play a role in the timing of the CP transition. The simple stochastic search algorithm we used can discover the correct numerical system if given enough time, and the difficulty of this search problem may be one factor which causes number-word learning to take such a long time. Algorithmic accounts are not incompatible with our approach: any approximate probabilistic inference algorithm may be applied to the model we present, producing quantitative predicted learning curves.

5.4. The count list may play a crucial role in the CP-transition

The CP-knower lexicon that the model eventually infers crucially relies on the ability to "move" on the counting list using *next*. This function operates on a list of words that children learn long before they learn the words' meanings. For the CP-knower, this function allows learners to return the *next* number word after the word they compute for a set with one fewer element. Without this function, no CP-knower could be constructed since the LOT would not be able to relate cardinalities to successive words in the count list. The necessity of *next* makes the interesting prediction that children who learned to recite the counting words in a different order—perhaps *next* returns the next word in alphabetical order—would not make the CP-transition, assuming the words have their standard English numerical meaning. Such a reliance on the count list matches Carey (2009)'s suggestion that it provides a key component of bootstrapping, a *placeholder structure*, which guides the critical induction. Indeed, cultures without number words lack the capacity to encode representations of exact cardinalities of even moderately sized numbers (Frank, Everett, Fedorenko, & Gibson, 2008; Gordon, 2004; Pica, Lemer, Izard, & Dehaene, 2004),[18] although this does not prevent them from exactly matching large cardinalities (Frank et al., 2008).

In addition, reasoning about LOT expressions may allow learners to understand more explicitly what happens in moving forward and backward on the count list—e.g., that adding one element to the set corresponds to one additional word on the count list. This potentially explains why subset-knowers do not know that adding or subtracting an element from a set corresponds to moving forward and backward on the count list (Sarnecka & Carey, 2008): subset knowers' representations do not yet use *next*.

5.5. Counting may be a strategy for correctly and efficiently evaluating LOT expressions

As we have presented it, the model makes no reference to the act of counting (pointing to one object after another while producing successive number words). However, counting has been argued to be the key to understanding children's numerical development. Gallistel and Gelman (1992) and Gelman and Gallistel (1978) argue children suddenly infer the meanings of number words greater than "three" or "four" when they recognize the simple correspondence between their preverbal counting system and their budding verbal counting system. They posit an innate preverbal counting system governed by a set of principles,

[18] Though see Gelman and Butterworth (2005) and see also Hartnett and Gelman (1998).

along with an innate successor function which can map any number N onto its successor $N + 1$. The successor function allows the learner to form concepts of all natural numbers, given a meaning for "one." Under this theory, the ability to count is a central, core aspect of learning number, and children's main task in learning is not creating numerical concepts, but discovering the relationship between verbal counting and their innate numerical concepts.

In our model there is a role for counting: counting may be used to keep track of which number word is currently in line to be returned by a recursive expression. For instance, perhaps each time *next* is computed on a number word, children say the number word aloud. This helps them keep track of which number word they are currently on. This may simplify evaluation of sequences like (*next* (*next* (*next*...))), which may be difficult to keep track of without another strategy. Similarly, the act of pointing to an object is also interpretable under the model. In the recursive CP-knower hypothesis, the model must repeatedly compute (*set-difference S* (*select S*)). It may be that each time an element of a set is *select*ed, it is pointed to in order to individuate it from other objects. This interpretation of counting differs from Gelman & Gallistel's view in that counting is only a technique for helping to evaluate a conceptual representation. It may be an essential technique, yet distinct from the crucial representational systems of numerical meaning.

This interpretation of counting explains why children generally do not count when they are subset-knowers, but do count when they are CP-knowers (Wynn, 1992).[19] Because the subset-knower lexicons make use of LOT operations like *singleton?* and *doubleton?*, they do not need to use *next*, and therefore do not need to keep track of a location in the recursion along the list of counting words. When the transition is made to a CP-knower lexicon, all number words are computed using *next*, meaning that children need to use counting as a strategy.

5.6. Innovation during development may result from compositionality in a LOT

One of the basic mysteries of development is how children could get something fundamentally new—in what sense can children progress beyond systems they are innately given? Indeed, this was one of the prime motivations in studying number since children's knowledge appears very different before and after the CP-transition. Our answer to this puzzle is that novelty results from *compositionality*. Learning may create representations from pieces that the learner has always possessed, but the cognitive pieces may interact in wholly novel ways. For instance, the CP-knower lexicon uses primitives to perform a computation which is not realized at earlier subset-knower levels, even though the primitives were available.

The apparent novelty seen in other areas across development may arise in a similar way, from combining cognitive primitives in never-before-seen ways. This means that the underlying representational system which supports cognition can remain unchanged throughout development, though the specific representations learners construct may change.[20]

Such compositionality is extremely powerful: as in lambda calculus or programming languages, one need only assume a very small set of primitives in order to allow Turing-complete computational abilities.[21] This provides an interesting framework for thinking about Fodor's radical concept nativism (Fodor, 1975, 1998). Fodor argues that the only sense in which new concepts can be learned is by composing existing concepts. He says that because most concepts are not compositional (e.g., *carburetor*, *banana*, *bureaucrat*), they could not have been learned, and therefore must be innate (see also Laurence & Margolis, 2002). We have demonstrated that learning within a computationally powerful LOT, lambda calculus, may be a viable developmental theory. If development works this way, then concepts that apparently lack a compositional form–like *bureaucrat*–could be learned by only requiring a set of primitives of sufficient computational power. The reason for this is that if a computational theory of mind is correct, all such concepts can be expressed in lambda calculus or other Turing-complete formalisms, with the appropriate primitives. Such concepts could be learned *in principle* by the type of model we present, by searching through lambda expressions.

5.7. A sufficiently powerful representation language may bind together cognitive systems

The LOT we use is one of the most important assumptions of the model. We chose to include primitive LOT operations—like *and*, *not*, and *union*—which are simple and computationally basic. These simple primitives provide one plausible set of conceptual resources 2-year olds bring to the task of learning number-word meanings.

We also included some operations—for instance *singleton?*, *doubleton?* and *tripleton?*—which seem less computationally and mathematically elementary. Indeed, they are technically redundant in that, for instance, *doubleton?* can be written using *singleton?* as λ *S*. (*singleton?* (*set-difference S* (*select S*))). However, we include all three operations because there is evidence that even very young children are capable of identifying small set sizes. This indicates that these three operations are all cognitively "basic", part of the conceptual core that children are born with or acquire very early in childhood. The inclusion of all three of these operations represents an important assumption of the model, since excluding some would change the inductive bias of the model and lead to a different predicted developmental pattern.

Thus, the model presented here formalizes and tests assumptions about core cognitive domains by the primitives it includes in the LOT. Importantly, the core

[19] Though see Bullock and Gelman (1977), Gelman and Meck (1992), Gelman (1993) for contrary evidence.

[20] This hypothesis provides an alternative view on the distinction between continuity and discontinuity in development. The model's learning is continuous in that the representation system remains unchanged; it is discontinuous in that the learner substantially changes the specific representations that are constructed.

[21] The subset of lambda calculus we use is not Turing-complete, though it may not halt. It would be simple to modify our learning setup to include untyped lambda expressions, making it Turing-complete.

set operations must interface via the LOT with operations in other domains. The ability to compute *singleton?* is only useful if it can be combined with other primitives manipulating logic and the counting routine. This requires a representational system which is powerful enough to operate meaningfully across domains. The LOT we use contains several of these cross-domain operations: for example, *L* maps a set to a word, transforming something in the domain of sets to one in the domain of number words. The primitive *equal-word?* transforms something in the domain of words to the domain of truth values, which can then be manipulated by logical operations. The model therefore demonstrates that core cognitive operations can and must be integrated with general computational abilities to support rich learning. As future work elaborates our understanding of children's cognitive capacities, this class of model can provide a means for exploring how different abilities may interact and support learning.

5.8. Further puzzles

There are several phenomena in number learning which further work is required to understand and model.

First, since our model is not intended to be an algorithmic theory, it leaves open the question of why learning number takes so long. This is puzzling because there are many instances in which children learn novel words relatively quickly (Carey & Bartlett, 1978; Heibeck & Markman, 1987). It is possible that it takes so long because children's space of possible numerical meanings is large, and the data is consistent with many different or, the difficulty with learning number words may even point to algorithmic theories under which children stochastically search or sample the space of hypotheses (Ullman et al., 2010), as we did with the model. Such an algorithm typically considers many hypotheses before arriving at the correct one.

A second puzzle is to fully formalize the role of counting and its relationship to the abstract principles of counting (Gallistel & Gelman, 1992; Gelman & Gallistel, 1978). We suggested that counting may play a role in helping to evaluate LOT expressions, but this would require a level of meta-reasoning that is not captured by the current model. Even subset knowers appear to understand that in specific circumstances, counting can be used to determine cardinality, although it is not clear how able they are to generally use counting (Sarnecka & Carey, 2008). It may be that the knowledge of when and how to count is initially learned as a pragmatic—not numerical—principle, and children have to discover the relationship between counting behavior and their system of numerical representation. Future work will be required to address counting and understand how it might interact with the types of numerical representations presented here. Additionally, it will be important to understand the role of social and pedagogical cues in number learning. These are most naturally captured in the likelihood function of the model, perhaps by increasing the importance of certain socially salient data points.

The model we have presented assumed that number word meanings are *exact*: "two" is true of sets containing exactly two elements, and not more. In natural language use, though, numerals can often receive an interpretation of *at least*: if there are six professors named "Mark", then it is also true that there are three named "Mark". There is some evidence that children's meanings of numbers are exact (Barner, Chow, & Yang, 2009; Huang, Snedeker, & Spelke, 2004; Papafragou & Musolino, 2003), in which case the exactness assumptions of the model would be justified. This leaves open the question of how children learn the scalar implicatures common in numerical meanings, and how such pragmatic phenomena relate to pragmatics in other determiners and language more generally (see Barner & Bachrach, 2010). One way to capture this would be to modify the likelihood to include pragmatics, or perhaps even to learn the correct form of the likelihood—learn *how* number words are used. On the other hand, it may turn out that number word representations are lower-bounded. If this were true, one could construct a model very similar to the one presented here, but modify the primitives: *singleton?* would change to *at-least-singleton?* and would be true of sets of cardinality one or greater, *doubleton?* would change to *at-least-doubleton?*, etc. Such a system would require pragmatic factors to be included in the likelihood to achieve exactness.

A final question is how children make the mapping between approximate numerosity (Dehaene, 1999; Feigenson, Dehaene, & Spelke, 2004) and their counting routine. One way the representations of continuous quantity could interface with the representations assumed here is that continuous quantities may have their own operations in the LOT. It could be, for instance, that children also learn a compositional function much like the CP-knower hypothesis which maps approximate set representations to approximate number words, and this process may interact with the LOT representations of exact number.

6. Conclusion

In number acquisition, children make one of their most elegant inductive leaps, suddenly inferring a system of numerical competence which is, in principle, infinitely productive. On the surface, generalization involving such a simple conceptual system is puzzling because it is difficult to see how such a leap might happen, and in what way children's later knowledge could go beyond their initial knowledge.

The model we have provided suggests one possible resolution to this puzzle. Children's initial knowledge may be characterized by a set of core cognitive operations, and the competence to build a vast set of potential numerical systems by composing elements of their representational system. The hypothesis space for the learner might be, in principle, infinite, including many types of Mod-*N* systems, singular/plural systems, even/odd systems, and other systems even more complex. We have shown that not only is it theoretically possible for learners to determine the correct numerical system from this infinity of possibilities, but that the developmental pattern such learning predicts closely resembles observed data. The model we have proposed makes sense of bootstrapping, providing a way of formalizing the roles of key computational systems to explain how children might induce natural number concepts.

Acknowledgments

We'd like to thank Lance Rips, Sue Carey, Ted Gibson, Justin Halberda, Liz Spelke, Irene Heim, Dave Barner, Barbara Sarnecka, Rebecca Saxe, Celeste Kidd, Leon Bergen, Eyal Dechter, Avril Kenney, an anonymous reviewer, and members of CoCoSci and TedLab for helpful discussions and feedback. This work was supported by an NSF Graduate Research Fellowship and a Social, Behavioral & Economic Sciences Doctoral Dissertation Research Improvement Grant in Linguistics (to S.T.P.), and AFOSR Grant FA9550-07-1-0075. S.P. and N.G. conducted this research while at MIT's department of Brain and Cognitive Sciences.

References

Barner, D., & Bachrach, A. (2010). Inference and exact numerical representation in early language development. *Cognitive psychology, 60*(1), 40–62.

Barner, D., Chow, K., & Yang, S. (2009). Finding one's meaning: A test of the relation between quantifiers and integers in language development. *Cognitive psychology, 58*(2), 195–219.

Barner, D., Thalwitz, D., Wood, J., Yang, S., & Carey, S. (2007). On the relation between the acquisition of singular–plural morpho-syntax and the conceptual distinction between one and more than one. *Developmental Science, 10*(3), 365–373.

Baroody, A. (1984). Children's difficulties in subtraction: Some causes and questions. *Journal for Research in Mathematics Education, 15*(3), 203–213.

Bloom, P., & Wynn, K. (1997). Linguistic cues in the acquisition of number words. *Journal of Child Language, 24*(03), 511–533.

Bullock, M., & Gelman, R. (1977). Numerical reasoning in young children: The ordering principle. *Child Development, 48*(2), 427–434.

Carey, S. (2009). *The origin of concepts*. Oxford: Oxford University Press.

Carey, S., & Bartlett, E. (1978). Acquiring a single new word. *Papers and Reports on Child Language Development, 15*, 17–29.

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

Condry, K. F., & Spelke, E. S. (2008). The development of language and abstract concepts: The case of natural number. *Journal of Experimental Psychology: General, 137*, 22–38.

Dehaene, S. (1999). *The number sense: How the mind creates mathematics*. Oxford: Oxford University Press.

Dehaene, S., & Mehler, J. (1992). Cross-linguistic regularities in the frequency of number words∗ 1. *Cognition, 43*(1), 1–29.

Feigenson, L., Dehaene, S., & Spelke, E. (2004). Core systems of number. *Trends in cognitive sciences, 8*(7), 307–314.

Fodor, J. (1975). *The language of thought*. Cambridge, MA: Harvard University Press.

Fodor, J. (1998). *Concepts: Where cognitive science went wrong*. Oxford: Oxford University Press.

Frank, M., Goodman, N.D., & Tenenbaum, J.B. (2007). A Bayesian framework for cross-situational word-learning. *Advances in Neural Information Processing Systems, 20*.

Frank, M., Everett, D., Fedorenko, E., & Gibson, E. (2008). Number as a cognitive technology: Evidence from Pirahã language and cognition. *Cognition, 108*(3), 819–824.

Fuson, K. (1984). More complexities in subtraction. *Journal for Research in Mathematics Education, 15*(3), 214–225.

Fuson, K. (1988). *Children's counting and concepts of number*. New York: Springer-Verlag.

Gallistel, C. (2007). Commentary on Le Corre & Carey. *Cognition, 105*, 439–445.

Gallistel, C., & Gelman, R. (1992). Preverbal and verbal counting and computation. *Cognition, 44*, 43–74.

Gelman, R. (1993). A rational-constructivist account of early learning about numbers and objects. *The Psychology of Learning and Motivation. Advances in Research Theory, 30*, 61–96.

Gelman, R., & Meck, B. (1992). Early principles aid initial but not later conceptions of number.

Gelman, R., & Butterworth, B. (2005). Number and language: How are they related? *Trends in Cognitive Sciences, 9*(1), 6–10.

Gelman, R., & Gallistel, C. (1978). *The child's understanding of number*. Cambridge, MA: Harvard University Press.

Goodman, N., Ullman, T., & Tenenbaum, J. (2009). Learning a theory of causality. In *Proceedings of the 31st annual conference of the cognitive science society* (pp. 2188–2193).

Goodman, N., Tenenbaum, J., Feldman, J., & Griffiths, T. (2008). A rational analysis of rule-based concept learning. *Cognitive Science, 32*(1), 108–154.

Gordon, P. (2004). Numerical cognition without words: Evidence from Amazonia. *Science, 306*(5695), 496.

Hartnett, P. (1991). The development of mathematical insight: From one, two, three to infinity. Ph.D. thesis, University of Pennsylvania.

Hartnett, P., & Gelman, R. (1998). Early understandings of numbers: Paths or barriers to the construction of new understandings? *Learning and instruction, 8*(4), 341–374.

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

Heibeck, T., & Markman, E. (1987). Word learning in children: An examination of fast mapping. *Child Development, 58*(4), 1021–1034.

Heim, I., & Kratzer, A. (1998). *Semantics in generative grammar*. Malden, MA: Wiley-Blackwell.

Hoeting, J., Madigan, D., Raftery, A., & Volinsky, C. (1999). Bayesian model averaging. *Statistical Science, 14*, 382–401.

Huang, Y., Snedeker, J., & Spelke, E. (2004). What exactly do numbers mean. In *Proceedings of the 26th Annual Conference of the Cognitive Science Society* (p. 1570).

Katz, Y., Goodman, N., Kersting, K., Kemp, C., & Tenenbaum, J. (2008). Modeling semantic cognition as logical dimensionality reduction. In *Proceedings of Thirtieth Annual Meeting of the Cognitive Science Society*.

Kouider, S., Halberda, J., Wood, J., & Carey, S. (2006). Acquisition of english number marking: The singular plural distinction. *Language Learning and Development, 2*(1), 1–25.

Laurence, S., & Margolis, E. (2002). Radical concept nativism. *Cognition, 86*, 25–55.

Le Corre, M., & Carey, S. (2007). One, two, three, four, nothing more: An investigation of the conceptual sources of the verbal counting principles. *Cognition, 105*, 395–438.

Lee, M., & Sarnecka, B. (2010a). A model of knower-level behavior in number concept development. *Cognitive Science, 34*(1), 51–67.

Lee, M., & Sarnecka, B. (2010b). Number-knower levels in young children: Insights from Bayesian modeling. *Cognition, 120*(3), 391–402.

Leslie, A. M., Gelman, R., & Gallistel, C. (2008). The generative basis of natural number concepts. *Trends in Cognitive Sciences, 12*, 213–218.

Levine, S., Suriyakham, L., Rowe, M., Huttenlocher, J., & Gunderson, E. (2010). What counts in the development of young children's number knowledge? *Developmental Psychology, 46*(5), 1309–1319.

Liang, P., Jordan, M.I., & Klein, D. (2009). Learning semantic correspondences with less supervision. In *Proceedings of the joint conference of the 47th annual meeting of the ACL and the 4th International joint conference on natural language processing of the AFNLP. Association for Computational Linguistics* (Vol. 1, pp. 91–99).

Liang, P., Jordan, M.I., & Klein, D. (2010). Learning programs: A hierarchical bayesian approach. In *Proceedings of the 27th international conference on machine learning*.

Lipton, J., & Spelke, E. (2006). Preschool children master the logic of number word meanings. *Cognition, 98*(3), B57–B66.

MacWhinney, B. (2000). *The CHILDES project: Tools for analyzing talk*. Hillsdale, New Jersey: Lawrence Erlbaum.

Madigan, D., & Raftery, A. (1994). Model selection and accounting for model uncertainty in graphical models using Occams window. *Journal of the American Statistical Association, 89*, 1535–1546.

Margolis, E., & Laurence, S. (2008). How to learn the natural numbers: Inductive inference and the acquisition of number concepts. *Cognition, 106*(2), 924–939.

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

Papafragou, A., & Musolino, J. (2003). Scalar implicatures: Experiments at the semantics-pragmatics interface. *Cognition, 86*(3), 253–282.

Piantadosi, S., Goodman, N., Ellis, B., & Tenenbaum, J. (2008). A Bayesian model of the acquisition of compositional semantics. In *Proceedings of the Thirtieth Annual Conference of the Cognitive Science Society*.

Pica, P., Lemer, C., Izard, V., & Dehaene, S. (2004). Exact and approximate arithmetic in an Amazonian indigene group. *Science, 306*(5695), 499.

Rips, L., Asmuth, J., & Bloomfield, A. (2006). Giving the boot to the bootstrap: How not to learn the natural numbers. *Cognition, 101*, 51–60.

Rips, L., Asmuth, J., & Bloomfield, A. (2008). Do children learn the integers by induction? *Cognition, 106*, 940–951.

Rips, L., Bloomfield, A., & Asmuth, J. (2008). From numerical concepts to concepts of number. *Behavioral and Brain Sciences, 31*, 623–642.

Sarnecka, B., & Carey, S. (2008). How counting represents number: What children must learn and when they learn it. *Cognition, 108*(3), 662–674.

Sarnecka, B., & Gelman, S. (2004). Six does not just mean a lot: Preschoolers see number words as specific. *Cognition, 92*(3), 329–352.

Sarnecka, B., & Lee, M. (2009). Levels of number knowledge during early childhood. *Journal of Experimental Child Psychology, 103*, 325–337.

Spelke, E. (2003). What makes us smart? Core knowledge and natural language. In D. Gentner & S. Goldin- Meadow (Eds.), *Language in Mind*. Cambridge, MA: MIT Press.

Steedman, M. (2000). *The syntactic process* (Vol. 131). Cambridge, MA: MIT Press.

Tenenbaum, J. (1999). A bayesian framework for concept learning. Ph.D. thesis, Massachusetts Institute of Technology.

Ullman, T., Goodman, N., & Tenenbaum, J. (2010). Theory Acquisition as Stochastic Search. In *Proceedings of thirty second annual meeting of the cognitive science society*.

Wynn, K. (1990). Children's understanding of counting. *Cognition, 36*, 155–193.

Wynn, K. (1992). Children's acquisition of the number words and the counting system. *Cognitive Psychology, 24*, 220–251.

Zettlemoyer, L.S., & Collins, M. (2005). Learning to map sentences to logical form: structured classification with probabilistic categorial grammars. In *UAI* (pp. 658–666).

Zettlemoyer, L.S., & Collins, M. (2007). Online Learning of Relaxed CCG Grammars for Parsing to Logical Form. In *EMNLP-CoNLL* (pp. 678–687).