Supporting Information

Bannard et al. 10.1073/pnas.0905638106

SI Text

Recording, Transcription, and Annotation of Data. All recordings were made in the children's own homes. The mothers of both children spent most of their week alone with the children and were typically the only adult speakers present during recording. They were employed as research assistants for the duration of the study, making 1-h tapes of themselves and their children in typical interactions in their home for 5 days per week over the 6 weeks. Illness prevented recording on 2 days for Annie at 2 years, so there are 30 h of recordings for Brian and 28 h for Annie. At 3 years there are 30 h of recordings for both children. All recordings were transcribed in the CHAT format (1).

After transcription we made the following decisions in using the data. We discarded all utterances containing speech that was marked as unintelligible or where the transcriber indicated uncertainty. We also removed all imitated and interrupted utterances. Where there were repetitions or retraces, the initial or aborted part of the utterance was excluded. Phonological fragments and pauses were not treated as part of utterances and all special form information was stripped. Finally, the CHAT conventions distinguish between different kinds of "s" affixes/clitics, marking them as e.g., 'has or 'is to indicate the abbreviated form. Because such information is not recoverable directly from the speech signal, but requires interpretation from the context by the transcriber, and we cannot be sure that the child is aware of the relationship of the realized form to the assumed-to-be underlying form, we remove this distinction, marking them all as simply 's.

Extraction of Candidate Grammars. An example set of alignments and extracted elements is shown in Fig. 3. At the top of Fig. 3 is the target phrase *Mummy have this one*. At the bottom of Fig. 3 are a sample of other utterances with which this utterance aligns. So, for example, the target is aligned with *I have that one*. Going through the utterance we can see that the word *Mummy* has been substituted for the word *I*, and the word *have* has been matched. Next, the word *this* mismatches with the word *that*, and then the word *one* aligns. It is important to note that the production of tokens for alignment or substitution is greedy, so that in such cases we subsume all available lexical material under as long a unit as possible, as for the subphrase *a red* in the utterance *I have a red one*. We can then use this alignment (and indeed all of the others shown at the bottom of Fig. 3) to extract the three circled signs shown in the middle of Fig. 3: the schema *X have X one*, and the two concrete elements *Mummy* and *this*. The Xs correspond to possible slots. At the point of extraction these slots are general but during the inferential process we will discover the constraints on what can be inserted into them. One thing that should be noted here is that while schemas can at this point only be extracted from an analysis of a whole utterances, the analysis can be licensed by alignments with part utterances. So in this case *Mummy have this one* has been aligned with *Mummy have that one story*, with the material after the end of the aligned word *one* being ignored.

Once we complete this initial process we generate multiple possible flat analyses for each utterance in the corpus (this is only one of the alignment sets that we will find for *Mummy have this one*). We then take each of the concrete elements extracted and repeat the whole process for these. We treat them as targets, identify those subsequences in the corpus that share lexical material with them (the candidate matches), and then align our target with these to produce an analysis of the subphrase into schemas and concrete signs in exactly the same way as for the whole utterances. And again we then take each of the concrete subphrases of these analyses and produce an analysis of these. We perform this process recursively until it halts. Each time we recurse over a new subphrase we generate a whole new candidate analysis, meaning that for the utterance *walk the dog*, we would include both the analysis (*walk* (*the dog*)), yielding the elements *walk X* and *the dog* and the analysis (*walk* (*the* (*dog*))), yielding the elements *walk X*, *the X* and *dog* as separate possible analyses for future consideration. The only constraint on what available schemas can be identified by using this process is that for an alignment to be generated, the basic pattern must occur twice (once in the target and at least once in the match). For consistency, then, we further apply the rule that for any whole utterance to be considered as a candidate concrete sign it must have occurred at least twice.

Model Details. ϕ, π, and σ are component distributions in our mixture model, over which we will integrate during inference. This being a Bayesian model the probability of a set of model parameters is the product of two elements, the probability of the data given that model and the prior probability of the model. Both ϕ and π are probability distributions over a set of discrete values (signs and categories, respectively). The natural choice of prior for such a multinomial distribution is the dirichlet distribution. The dirichlet is conjugate to the multinomial, meaning that samples from a multinomial distribution using a dirichlet prior will themselves have a dirichlet distribution. The dirichlet fits well the kind of skewed distributions seen for words and categories in natural languages (2). Using such a distribution requires us to define concentration hyperparameters for categories (α_c) and signs (α_s), which essentially control the sparcity of the distributions, with values of 1 making signs/categories equiprobable and values < 1 preferring models that explain the data with fewer signs/categories. For all models we assign $\alpha_c = 0.1$ and $\alpha_s = 0.1$. These small values build in a strong preference for sparse models.

π is the distribution over categories. One crucial additional aspect of our model is the number of categories to be allowed. We follow recent work on latent variable modeling (3) in leaving the number of categories unspecified (potentially infinite) and to be discovered as part of the inference procedure by using a stick-breaking procedure to assign exponentially decaying weights (β) to the prior probabilities of the categories. This works as follows. Our weight space can be considered as a stick β with a unit length. The weight for each category used in our model is decided by snapping the stick. The length of one part of the stick is the weight of the current category, while the other part of the stick is the amount of the unit-length stick left to generate the weights for all of the subsequent classes assigned. A new break is made each time a new category is assigned. There is always some remaining stick so weights can be assigned to an infinite number of categories, but with weights that decay exponentially. This means that the prior probability of an analysis will reduce exponentially with each new category that it introduces. The break applied to the stick is drawn from a β distribution $\beta\,(1,\alpha_p)$. We assign a value of $\alpha_p = 1$, producing an equivalent of the uniform distribution, meaning that any breakpoint is equally likely. The use of this stick-breaking prior means that our models can use a theoretically infinite number of categories but that there is a cost associated with each one added.

Probabilities in our model are calculated as follows. The probability of a tree t is calculated as in Eq. **1**. Each x represents a sign, and each z represents a category node, with the subscripts representing the order in which they are generated.

$$P(t) = P(x_k|z_k) \prod_{c_i \in C_k} P(z_{c_i}|z_k)P(t'). \quad [1]$$

The probability of a node $z = i$ producing a child node $z_{ci} = k$ is calculated as in Eq. **2**, where n_{ik} is the count for the joint occurrence of i and k and $n_{i\cdot}$ is the count for occurrence of i with all descendants. Conditioning on previous descendants is added straightforwardly as shown in Eq. **3**. Finally, the probability of an observation or sign x = a given $z = i$ is calculated as in Eq. **4**, with N being the total number of signs included in the model.

$$p(z_{c_i} = k|z = i) = \frac{n_{ik} + \alpha_c \beta_j}{n_{i.} + \alpha_c} \quad [2]$$

$$p(z_{c_i} = k|z_{c_{i-1}} = j, z = i) = \frac{n_{ijk} + \alpha_c \beta_j}{n_{ij.} + \alpha_c} \quad [3]$$

$$p(x = a|z = i) = \frac{n_{ia} + \alpha_s}{n_{i.} + N\alpha_s}. \quad [4]$$

Our MCMC Procedure. We use a MCMC technique known as Gibbs sampling to approximate the posterior distribution $p(t,\theta|D)$, where t is the set of trees, θ is the vector of rule probabilities, and D is our data, from which we then sample our grammars. The space of possible grammars (all possible ways of combining signs and labeling nodes/slots with categories to account for the children's speech) represents a large space of possible states. MCMC techniques involve taking a chain of samples from this possible state space in which each sample is conditional on only the previous sample (this is what is meant by a Markov chain). The chain is initialized by taking a completely random sample from the state space (a random set of sample parses). There are then many thousands of iterations in which repeated samples are drawn until the chain converges on a stationary distribution. By sampling from a chain that has converged we can approximate the posterior distribution.

There are many MCMC algorithms of which Gibbs sampling is one of the simplest. Events in the initial random sample are counted to produce an initial set of probabilities for the distributions that make up our model. At the next iteration a sample analysis of the data is drawn conditional upon these counts and the prior. At each subsequent iteration a new set of counts are taken and the process is repeated. Upon reaching convergence the initial burn-in iterations are discarded and samples are drawn from the chain at intervals (as adjacent samples are not independent).

We apply this algorithm to our data and model as follows. We begin by randomly assigning analyses to our utterances and extracting counts from this set to give us initial model parameters (our distributions ϕ, π, and σ). Each random analysis is built by starting at the utterance node and randomly deciding at each decision point whether to choose a concrete filler or a further schematic sign and if the latter then randomly picking a sign. Categories are assigned to nodes by first selecting a randomly chosen number of categories between 1 and 50, and then randomly assigning these to signs with uniform probability. We then sample our stick β and produce a new set of analyses by random sampling of analyses conditioned on our starting distributions. We generate trees by sampling at each step of the function defined in Eq. **1**. Having drawn a new sample we then extract counts and update the model. We repeatedly perform these three steps (sampling β, sampling analyses, and updating counts) for many thousand iterations.

For each child at 2 years we ran 10 separate chains with different randomly selected starting analyses. At 3 years we ran five chains for each child (because of the greater computational requirements imposed by the greater complexity of the utterances). The models for the 2-year-olds converged within 5,000 iterations, and the models for the 3-year-olds converged within 10,000. We ran all chains for 20,000 samples and sampled grammars at 100 iteration intervals over the last 10,000 iterations. This process gave us 1,000 grammars for the 2-year-old data and 500 grammars for the 3-year-old data. We report the mean performance achieved when parsing with all of these grammars.

Induction of Fully Abstract PCFGs. For the induction of the fully abstract PCFGs used in Exp. 1, we use the Bayesian method of ref. 4. This is a Bayesian parameter estimation technique for PCFGs. Given a particular grammar G, it assigns a vector of probabilities θ to the rules. θ is a product of dirichlet distributions, one for each nonterminal (this results from putting a dirichlet prior on the multinomial distribution over the right side of the rules given the left side). By using a concentration parameter < 1, we are able to prefer grammars in which most rules have a probability very close to zero. This allows one to estimate very effective PCFGs over a set of rules that is massively ambiguous (contains many possible rewrite rules for each category). We refer readers to ref. 4 for full details.

Our requirement in inducing a PCFG from our data was that it be able to parse any utterance that our usage-based PCFG could parse, to allow useful comparison. We wanted grammars that would model the data well while still being able to assign a nonzero probability to any tree structure over words that are in its lexicon. The fact that the technique of ref. 4 can estimate good models over hugely ambiguous grammars means we can expect to infer excellent models while also having guarantees concerning coverage. We accomplish this as follows. We estimate a vector of probabilities θ for grammars in which every category can rewrite to every word in the corpus and every category or every binary or ternary combination of every category in every possible order. The categories were (as in our induction of UB-PCFGs) simply arbitrary labels for classes (here of distributionally similar rules or words) to be discovered during inference. The inference procedure, similar to our method for inferring UB-PCFGs, uses Gibbs sampling to approximate the posterior distribution. We assigned a concentration hyperparameter of 0.1, thereby preferring models that assign most of the probability mass to a small subset of the rules and give the majority a vanishingly small probability. We chose the number of categories to include in the model via the standard model selection technique of first inferring models that include two categories for each dataset and then inferring models in which additional categories were added in a stepwise fashion until the addition of new categories stopped improving the fit of the model. The resulting grammars for both Brian and Annie at 2;0 were provided by grammars with seven categories. The best grammars at 3;0 were provided by grammars with 10 categories for Brian and 9 categories for Annie.

As with our concrete grammars, inference of these grammars involves taking multiple samples from the posterior distribution but for ease of comparison with our multiple sampled concrete grammars (to avoid intractable sets of many-to-many comparisons) we use the approach taken by Johnson et al. (4) and average our rule probabilities over the samples to produce a single model.

Parsing. We use the CYK algorithm (5) for parsing with our grammars. This algorithm requires grammars to be in Greibach normal form and thus we binarize the rules. However, this step is purely for processing convenience (any CFG can be represented in this way) and has no bearing on the results we report.

The Perplexity of Our Grammars When Parsing the Main Corpus. We have reported the perplexity of our grammars on the test data, a different sample from that which had been used to acquire the grammars. The ability to predict new datasets is the correct measure of the explanatory value of our grammars. However, it is also interesting to observe the fit of the models to the data from which they were inferred (the "main" sessions). In fitting statistical models to data there is always a tradeoff between fitting the sample at hand well and allowing generalization to other samples. Because they directly predict whole sequences of words in the corpus from which they are inferred, our UB-PCFGs can be expected to have a better fit to that corpus (and thus lower perplexity) than a fully abstract PCFG, and indeed they do, as shown in Fig. S1. In Fig. S1 the empty bars represent the mean perplexity of our UB-PCFGs for each child at each age, and the error bars represent the interval around this mean within which results for 95% of the sampled grammars fall. The filled bars in Fig. S1 represent the abstract PCFGs. The perplexity of the UB-PCFG is smaller than that of the traditional PCFG in all cases.

1. MacWhinney B (2000) *The CHILDES Project: Tools for Analyzing Talk* (Erlbaum, Mahwah, NJ).
2. Zipf GK (1935) *The Psycho-Biology of Language* (Houghton Miffin, New York).
3. Teh Y, Jordan M, M B, Blei D (2006) Hierarchical dirichlet processes. *J Am Stat Assoc* 101:1566–1581.
4. Johnson M, Griffiths T, Goldwater S (2007) Bayesian inference for PCFGs via Markov Chain Monte Carlo. *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, eds Sidner C, Schultz T, Stone M, Zhai C (Association for Computation Linguistics, Stroudsburg, PA), pp 139–146.
5. Kasami T (1965) *An Efficient Representation and Syntax Algorithm for Context-Free Languages* (Air Force Cambridge Research Center, Bedford, MA), Air Force Cambridge Research Lab Scientific Report 65-758.

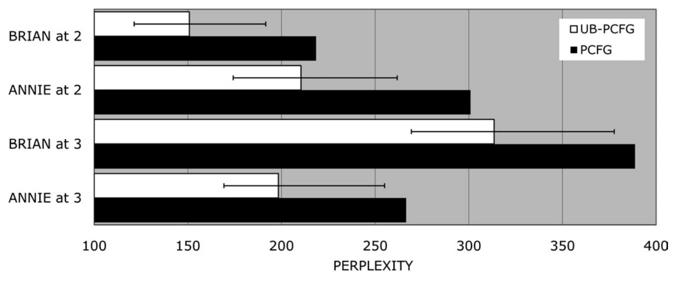

Fig. S1. Perplexity of the different grammars when parsing the main session data.