1

Learning, Bottlenecks and the Evolution of Recursive Syntax

Simon Kirby
Language Evolution and Computation
Research Unit, Department of Linguistics,
University of Edinburgh

1.1 Introduction

Human language is a unique natural communication system for two reasons.[1] Firstly, the mapping from meanings to signals in language has structural properties that are not found in any other animal's communication systems. In particular, syntax gives us the ability to produce an infinite range of expressions through the dual tools of compositionality and recursion. Compositionality is defined here as the property whereby an expression's meaning is a function of the meanings of parts of that expression and the way they are put together. Recursion is a property of languages with finite lexica and rule-sets in which some constituent of an expression can contain a constituent of the same category. Together with recursion, compositionality is the reason that this infinite set of expressions can be used to express different meanings.

Secondly, at least some of the *content* of this mapping is learned by children through observation of others' use of language. This seems *not* to be true of most, maybe all, of animal communication (see review in Oliphant, this volume). In this chapter I formally investigate the interaction of these two unique properties of human language: the way it is learned and its syntactic structure.

[1] The research for this chapter was carried out at the Language Evolution and Computation Research Unit at the Department of Linguistics, University of Edinburgh funded by ESRC grant R000237551. I would like to thank Jim Hurford, Mike Oliphant, Ted Briscoe and Robert Worden for useful discussions relating to the material presented here (although they do not necessarily agree with the content). The author's email and home page are: `simon@ling.ed.ac.uk`, `www.ling.ed.ac.uk/~ simon`.

1.1.1 Evolution without natural selection

Evolutionary linguistics is currently a growing field of research tackling the origins of human language (Bickerton 1990; Pinker & Bloom 1990; Newmeyer 1991; Hurford *et al.* 1998). Of particular interest to many researchers is the origins of syntactic structure. Perhaps the dominant approach to the evolution of this structure is expounded by Pinker & Bloom (1990); they suggest that the best way to view human language is as a biological adaptation that evolved in response to the need to communicate "propositional structures over a serial interface" (p. 707). In their (and many linguists) view, syntax is to a significant extent specified by an innate (and therefore genetically determined) language acquisition device (LAD) which constrains the language learner with prior knowledge about the nature of language.

> Evolutionary theory offers clear criteria for when a trait should be attributed to natural selection: complex design for some function, and the absence of alternative processes capable of explaining such complexity. Human language meets these criteria.
>
> *Pinker & Bloom (1990:707)*

In this chapter I agree that the structure of the human learning mechanism(s) will bring particular prior biases to bear on the acquisition task. Indeed there are good theoretical reasons why this *must* be the case for any learner that can generalise (e.g. Mitchell 1997). However, because language is unique (an autapomorphy in biological terms) we should search very carefully for "alternative processes" before turning to natural selection as an explanation. In fact, recent work of which this chapter is a continuation, (Batali 1998; Kirby 1998a; Kirby 1998b; Hurford 1998; Batali, this volume; Hurford, this volume) has suggested that some of the complex structure of language may be the result of a quite different process from biological evolution. This work shows that learning influences the dynamic process of language transmission, historically, from one generation to the next. In many ways this approach is mirrored in the recent work of linguists from quite different research perspectives (e.g. Niyogi & Berwick 1995; Niyogi & Berwick 1997; Christiansen & Devlin 1997; Briscoe 1998). This chapter aims to demonstrate that, for any reasonable learning bias, basic structural properties of language such as recursion and compositionality will inevitably emerge over time through the complex dynamical process of social transmission — in other words, without being built in to a highly constraining innate language

acquisition device.

1.1.2 A computational approach

If we are to understand the ways in which a learned, socially transmitted, system such as language can evolve we need some sophisticated models of learners embedded in a dynamic context. Verbal theorising about the likely behaviour of complex dynamical systems is often not good enough. As Niyogi & Berwick (1997) point out, our intuitions about the evolution of even simple dynamical systems are often wrong. Recently, many researchers have responded to this problem by taking a computational perspective (e.g. Hurford 1989; Hurford 1991; MacLennan 1991; Batali 1994; Oliphant 1996; Cangelosi & Parisi 1996; Steels 1996; Kirby & Hurford 1997b; Briscoe 1997; Briscoe 1998). This methodology provides a third way between verbal theorising on the one hand and on the other, analytical mathematical approaches — which are often difficult to formulate for these types of system. This chapter follows on from this line of research, developing a working computational simulation of individuals capable of learning to communicate by observing each other's behaviour, and tracking the development of the artificial languages that emerge in the population.

The rest of this chapter is divided into three main sections. Firstly, the computational model is described in some detail, with particular attention being paid to the process of learning (although some details are left to an appendix). The next section deals with two representative experiments with the model dealing with the emergence of compositionality given simple semantics, and with the emergence of recursive subordinate clauses given a more complex semantic space. Finally, an explanation of the behaviour of the simulation is given in theoretical terms along with a discussion of the impact of these results on linguistic theory.

1.2 A working model of linguistic transmission

Language exists in two different domains (Chomsky 1986; Hurford 1987; Kirby 1999):

I-language This is (internal) language as represented in the brains of the population. It is the language user's knowledge of language.

E-language This is the (external) language that exists as utterances in

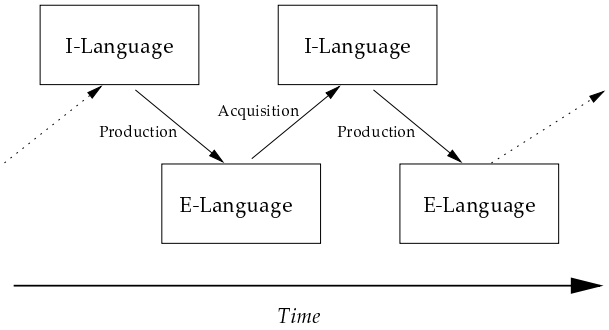


figure 1.1. The transmission of language over time.

the arena of use (Hurford 1987).

These two domains of language influence each other in profound ways via the process of linguistic transmission diagrammed in figure 1.1. E-language is a product of the I-language of speakers. However, the I-language of language learners is a product of the E-language that they have access to. A model of the constraints on these two domains and the transformations that map between them should be sufficient to determine the dynamical properties of linguistic transmission.

A computational simulation of linguistic transmission works within the framework shown in figure 1.2, an elaboration of the model in figure 1.1. The simulation implements these processes:

1. An individual in the simulation is given a set of meanings that must be expressed. These meanings can be thought of as being provided by the external "world", but in the simulation will simply be chosen randomly from some predefined set.
2. The individual then attempts to express each meaning either using their own internalised knowledge of language or by some random process of invention.
3. A new learner takes this set of utterances and uses it as input to learning.
4. Finally, the learner becomes a new speaker the old speaker is discarded and a new individual is added to become a new learner and the cycle repeats.

The utterances that the individuals produce and learn from in these simulations are pairs of strings of letters (which can be thought of as

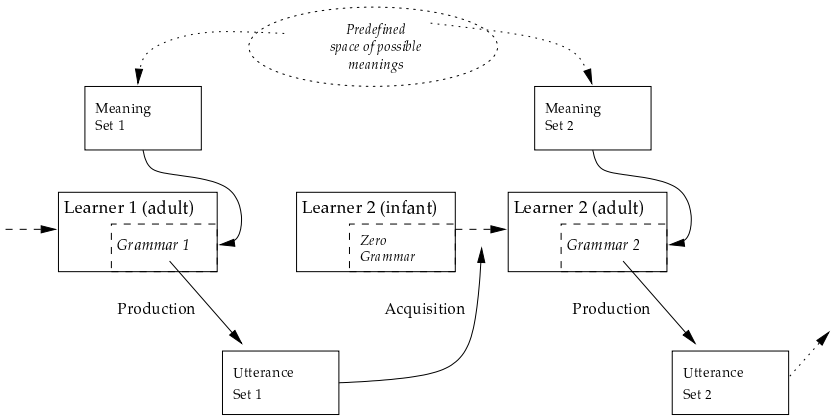


figure 1.2. A computational implementation of linguistic transmission.

basic unanalysable phonemic segments) and meaning representations. In these simulations the world is made up of a set of predefined atomic concepts. These might include:

john tiger
eats fears
knows

These concepts can be combined into simple predicate-argument propositions, which may have hierarchical structure. For example:

fears(john,tiger)
knows(john,eats(tiger,john))

So, an example utterance by an individual in the simulation that happened to know something like English might be the pair:

$$< \texttt{tigereatsjohn}, \text{eats(tiger,john)} >$$

Obviously the biggest component of the simulation will be the part that takes sets of pairs such as these and is able to learn from them in some useful way — in other words, the part of the simulation that takes instances of E-language and maps them into I-language. This is the subject of the next section.

1.2.1 Learning

For simulations such as the one presented in this chapter, which model many generations of speakers and acquirers, the design of the learning algorithm is crucial. Two important criteria for a learning algorithm for us are: efficiency, because the simulations will need to model thousands of learners over time; and ease of analysis, since we are interested in how the language evolves over time and it is therefore important to be able to easily inspect the internal states of the learners.

The algorithm presented here[2] has been designed specifically with simulation tasks in mind — it is extremely simple and efficient, and it enables the internal states of learners to be easily analysed. Although no claims are made here for its efficacy as a *practical* grammar induction tool, it does model in a simple way the dual processes of rote learning of examples and induction of generalisations that must be at the core of any model of language acquisition.

Grammatical representation

For these simulations, the hypothesis space that the learning algorithm searches consists of context-free grammars enriched with the kind of simple semantics described above. In fact, these grammars are a restricted form of definite-clause grammar in which non-terminals may have a single argument attached to them which conveys semantic information. It is important to realise that although the internal knowledge of the individuals is a type of context-free grammar, this does not mean that compositionality or recursion is built-in. Consider a learner that can produce the string `tigereatsjohn` meaning eats(tiger,john). Here are two (of many, many possible) grammars that this learner could have internalised:

$$S/\mathsf{eats}(\mathsf{tiger},\mathsf{john}) \rightarrow \texttt{tigereatsjohn}$$

$$\left\| \begin{array}{l} S/p(x,y) \rightarrow N/x \;\; V/p \;\; N/y \\ V/\mathsf{eats} \rightarrow \texttt{eats} \\ N/\mathsf{tiger} \rightarrow \texttt{tiger} \\ N/\mathsf{john} \rightarrow \texttt{john} \end{array} \right.$$

The S symbol in these grammars is the start symbol, whereas the N and V are arbitrarily named non-terminals. The lower case letters are shorthand for preterminals that expand to atomic phonemic segments. The

[2] In some respects the algorithm is a simplification and development of the one described in Kirby (1998a) and Kirby (1998b).

material following the slashes attached to non-terminals is the semantic representation for that non-terminal.[3]

The grammar on the left is the simplest grammar that could produce the utterance. It states that "`tigereatsjohn`" is a valid sentence meaning eats(tiger,john). Notice that this is entirely non-compositional — in no way is the meaning of the whole a function of meanings of its parts. In fact the string is not broken down or analysed at all, instead it is simply treated as a holistic chunk.

The grammar on the right, however, is compositional. The sub-parts of the string each are assigned meanings. So, for example, `tiger` corresponds to the meaning tiger. The whole string is composed by piecing these substrings together and combining their meanings using the variables x, p, and y.

It should be clear from this example, that although the grammatical formalism (obviously) allows us to represent languages that are structured syntactically, it *does not constrain languages to be of this form.* In other words, the space of languages that the learners have access to includes many that would not be considered possible human languages because, for example, they are non-compositional. The choice of formalism therefore does not build-in the result we are looking for.

Rule subsumption

In the first stage of learning, the grammar contains no rules. Data is presented to the the inducer as a pairing of a string of terminals and a semantic structure. A single pair can be incorporated into the grammar rather trivially. Say the pair

$$< \texttt{tigereatssausages}, \mathsf{eats(tiger,sausages)} >$$

is to be incorporated. The simplest rule that covers this "fact" about the language to be induced is:

$$S/\mathsf{eats(tiger,sausages)} \rightarrow \texttt{tigereatssausages}$$

[3] Formally, the semantic structures attached to non-terminals can take one of three forms: a fully specified form (i.e. a semantic structure with no variables), a partially specified form (i.e. a semantic structure with some variables, although the variables may only occur at the top level), or a variable. The left hand side semantics can take any of these forms in the grammar, but right hand side non-terminals can only take semantic variables in this formalism.

A trivial learning algorithm could involve gathering one of these language facts for every string-meaning pair presented and storing them as one big grammar. This would give us a grammar which can generate exactly and only the sentences in the input. We could add one simple refinement to this technique by deleting duplicate rules in the grammar. In fact these two basic operations — incorporation, and duplicate deletion — are at the core of the final induction algorithm used by the simulation.

The problem with using just these two operations is that the inducer has no power to generalise. As such, this is a rather poor model of learning. A basic strategy for extracting generalisations from rules that are overly specific (similar in some ways to the more general method used in some inductive logic programming — see, e.g. discussion and citations in Mitchell (1997)) is to take pairs of rules and look for the least-general generalisation that can be made that subsumes them within some prespecified constraints. For example, imagine a grammar with these two rules:

$$S/\mathsf{eats}(\mathsf{tiger},\mathsf{sausages}) \rightarrow \texttt{tigereatssausages}$$
$$S/\mathsf{eats}(\mathsf{john},\mathsf{sausages}) \rightarrow \texttt{johneatssausages}$$

What is the *least general* rule that would subsume both of these? Firstly, we need to replace tiger and john in the semantics with a variable. So the left hand side becomes: $S/\mathsf{eats}(x, \mathsf{sausages})$. But this means we need a nonterminal with an x attached to it in the right hand side of the rule. If we replace `tiger` and `john` on the right hand sides with a single new category (let's call it N), the we have our new rule:

$$S/\mathsf{eats}(x,\mathsf{sausages}) \rightarrow N/x\ \texttt{eatssausages}$$

We can now delete the original two rules because we have one that subsumes them both. However, there is a problem here. We have introduced a new nonterminal, N, but there is no rule saying what an N is. At every stage of induction, our generalisation should ensure that the new grammar can still parse the sentences that it could parse previously. In other words, the set of sentences $L(g)$ that a grammar g could generate before generalisation will always be a subset (though not necessarily a proper subset) of the set of sentences $L(g')$ that could be generated after generalisation. So, we must add two new N rules:

$$N/\mathsf{tiger} \rightarrow \texttt{tiger}$$
$$N/\mathsf{john} \rightarrow \texttt{john}$$

This is the most commonly applied subsumption method in the induction algorithm, but there are others. For example, if there are two rules such as:

$$N/\texttt{mary} \rightarrow \texttt{mary}$$
$$M/\texttt{mary} \rightarrow \texttt{mary}$$

then a rule that subsumes these two will simply choose one of the non terminal category symbols N or M. Let us say that it chooses to replace M with N,[4] then to keep the induction preservative we must rewrite all occurrences of M throughout the grammar with N.

The induction algorithm thus proceeds by taking an utterance, incorporating the simplest possible rule that generates that utterance directly, and then searches through all pairs of rules in the grammar for possible subsumptions like the ones described above until no further generalisations can be found, and finally deletes any duplicate rules that are left over. More details about the algorithms for rule-subsumption and the constraints on its application can be found in the appendix to this chapter.

1.2.2 Invention

The particular meanings of the sentences that the speakers produce is controlled by the experimenter. The space of possible meanings can be thought of as the population's "world model", in other words, what they want to talk about. One way to think of it is that the world compels the speaker to try to produce a string for a certain meaning. This means that there is no guarantee that the speaker will have a way of generating a string for the meaning it is compelled to produce. This will be especially true of the early stages of any simulation run, since the population is initialised with no grammatical knowledge at all.

If there was no way for speakers to produce strings *in the absence* of a grammar that can generate them, then a language could never get off the ground. It is important, therefore, that our model of an individual be enriched to allow for *invention*. The invention process is essentially a mechanism for introducing random new words for chunks of meaning, but it should not build in new syntactic structure. In other words, we

[4] In general which it chooses will not matter *except* in the case of the start category, S. The start category is never changed.

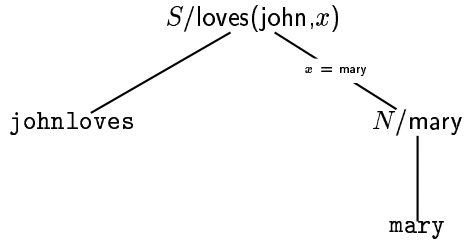
assume that the individuals are able to invent strings of sounds but are not able to spontaneously invent hierarchical structure that they have not observed.

The invention algorithm used here, given a meaning that the speaker does not have a way of producing, tries to find the closest meaning that the speaker *does* have a way of producing. With this new meaning, a string and a parse tree for that string can be generated. The parse tree will show the parts of the string that correspond to the "wrong" parts of the meaning — in other words, the parts of the near meaning that are different to the meaning that should have been produced. These parts of the string are excised, and replaced with a random sequence of symbols.[5] Finally, the speaker's induction algorithm "hears" its own invented string/meaning pair (this ensures that the speaker's output is consistent).

An example should make this clearer. Imagine that a speaker has the following grammar:

$$S/\mathsf{loves(john},x) \rightarrow \texttt{johnloves}\ N/x$$
$$N/\mathsf{mary} \rightarrow \texttt{mary}$$
$$N/\mathsf{jane} \rightarrow \texttt{jane}$$

This speaker is then asked to produce a string for the meaning loves(john, anna). The nearest meanings to this that the speaker can produce strings for are loves(john, mary) or loves(john, jane). We'll pick the first, which produces the tree structure:



The wrong part of this tree is the material dominated by the node that introduces the meaning mary. We therefore delete the string mary and replace this with a random sequence of characters. So, the invented string for the meaning loves(john,anna) might be johnlovesspog. So, in this case, the compositionality of the grammar is reflected in the invented

[5] For the simulation runs presented here, the sequence varies from 1 to 3 letters randomly chosen from the alphabet.

string.

A second example demonstrates an important property of the algorithm which avoids the introduction of novel structure. We'll use the same grammar, but instead try and invent a string for the meaning loves(fred,mary). The nearest meaning to this one using the grammar is loves(john,mary), which generates the same tree as above. This time the wrong bit of meaning is john, which dominates the whole string. An invented string for loves(fred,mary), therefore, might be a totally non-compositional string like `bling`.

1.2.3 Summary of simulation cycle

In general the simulation can contain a population of any number of individuals, but to keep things simple in the experiments described here, there are only ever two individuals at any one time: an adult **speaker** and a new **learner**. At the start of any simulation run, neither the speaker nor the learner has any grammar at all — in other words, they have no knowledge of language. This means that any language that is observed in the simulation is purely emergent from the interactions of the individuals in the simulation.

Each generation in the simulation goes through the following steps:

1. The speaker tries to produce a set of utterances that will form input to the learner. This involves repeating the following sequence of actions some number of times (set by the experimenter):
   (a) The speaker is given a meaning chosen at random from a predefined set.
   (b) If the speaker is able to generate a string for that meaning using its grammar, it does so, otherwise it invents a string.[6] If the speaker has invented a string, the *speaker* uses that string-meaning pair as input to induction. This means that, if an individual invents a new way of saying something, they will learn from that and use that invention again if the need arises.
   (c) The learner is given the string, and tries to parse it with any

[6] Generation is always deterministic. If the grammar allows more than one way of producing a certain string, only one way is ever used. However, which one is used will be random. This is implemented by randomly ordering the grammatical rules once after the learner becomes a speaker, and using this order to inform the choice of rules employed in generation.

grammar it might have. If it is unable to parse the string, then it takes the string-meaning pair and uses it as input to induction.

2. The speaker's grammar is logged and then it is deleted from the simulation.
3. The learner becomes the new speaker, and a new learner with a blank grammar is added to the simulation.

The two main parameters that the experimenter can vary in this model are: the number of utterances that the speaker will be called upon to produce in its lifetime, and the structure and size of the meaning space. In the discussion section of this chapter we will see that these two parameters bear upon each other in an interesting way.

1.3 Example experiments

This section describes in detail two experiments which demonstrate that interesting linguistic structure emerges in initially non-linguistic populations over time in the cycle of acquisition and use. Each experiment has been run many times with differing initial random-number "seeds". In analysing the results we are able to directly examine individual grammars as well as plotting numerical values such as the proportion of meanings that are produced without invention, and size of grammar.

1.3.1 Degree-0 compositionality

In the first simulation run we experiment with the properties of languages that emerge when the individuals only communicate about simple meanings. The meaning space is made up of simple degree-0 two-place predicates (i.e. predicates with no embedding) in a world with five possible "objects" and five possible "actions". Example meanings are:

likes(gavin,mary)
loves(mary,john)
hates(heather,pete)
and so on...

In these simulations, the arguments of the predicates must be distinct — in other words, reflexives like loves(john,john) are not allowed.

This means there 100 distinct meanings that the individuals may wish to express (5 predicates x 5 possible first arguments x 4 possible second

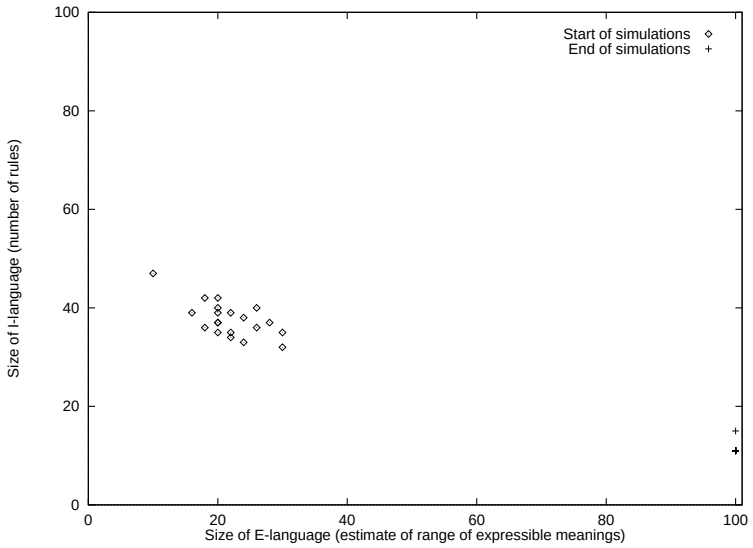


figure 1.3. A scatter plot of the start and end of the simulations in a space of I- vs. E-language size.

arguments). Each speaker produces 50 utterances in a lifetime, each of which is chosen at random from this space of two-place predicates. This means that, even if the meanings were carefully chosen, rather than being picked at random, learners can never be exposed to the entire range of possible meanings.

The results of the simulation can be plotted on a graph of grammar size against grammar expressivity. The former is calculated simply by counting the number of rules in each speaker's grammar, and the latter can be estimated by counting the proportion of utterances that the speaker produced without resorting to invention. These two values can be thought of as the I-language size and E-language size, respectively. Figure 1.3 is a scatter plot on this space, of the state of the languages at the start of the simulation runs (i.e. at the end of the first speaker's "life") and again at the end of the simulation runs. In fact, almost all the simulation runs ended up at the same point (with an expressivity of 100, and 11 grammar rules). The one exception is a language that had not converged on a stable system of 11 rules by the end of the run (we return to this situation later). Figure 1.4 shows the movement through this space of the languages of five typical simulations.

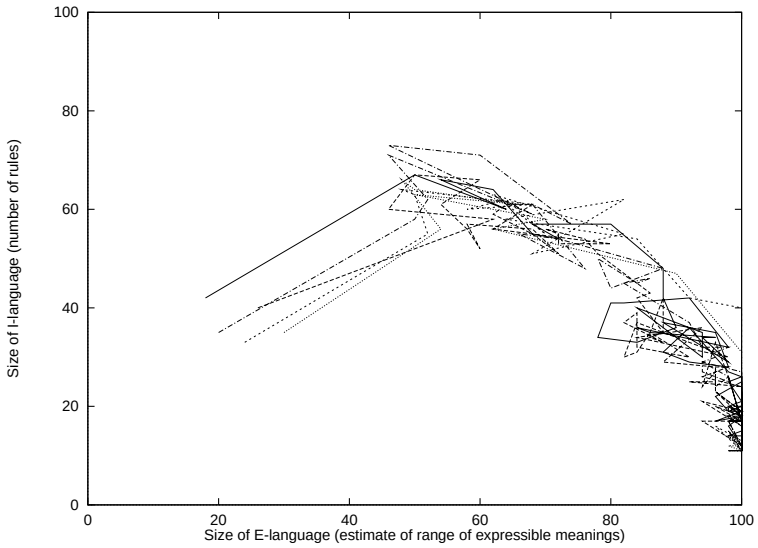


figure 1.4. The movement of some of the simulations through the I/E-language space. In the early stages, all the languages have low expressivity (E-language size) and big grammars. However, the languages universally move towards higher expressivity and smaller grammars.

These graphs are best understood by working through a particular example language as it changed over time. Here is a typical first generation grammar (nonterminals except for the start nonterminal S are arbitrarily chosen capital letters):

Generation 1

S/detests(john,gavin) $\rightarrow$ `nqb`
S/hates(heather,mary) $\rightarrow$ `b`
S/loves(mary,pete) $\rightarrow$ `k`
S/admires(john,mary) $\rightarrow$ `u`
S/detests(pete,john) $\rightarrow$ `ayj`
S/likes(heather,gavin) $\rightarrow$ `g`
S/loves(john,mary) $\rightarrow$ `o`
S/loves(pete,john) $\rightarrow$ `vcs`
S/likes(john,pete) $\rightarrow$ `os`
S/loves(heather,gavin) $\rightarrow$ `e`
S/likes(mary,gavin) $\rightarrow$ `ke`
S/admires(john,gavin) $\rightarrow$ `hy`
S/admires(pete,heather) $\rightarrow$ `dx`
S/admires(gavin,pete) $\rightarrow$ `x`
S/likes(heather,mary) $\rightarrow$ `d`
S/detests(heather,john) $\rightarrow$ `m`
S/detests(john,pete) $\rightarrow$ `fu`
S/detests(mary,gavin) $\rightarrow$ `qqq`
S/hates(gavin,john) $\rightarrow$ `jrx`
S/likes(gavin,john) $\rightarrow$ `w`
S/admires(gavin,mary) $\rightarrow$ `h`
S/hates(heather,gavin) $\rightarrow$ `nln`
S/hates(pete,mary) $\rightarrow$ `r`
S/likes(gavin,pete) $\rightarrow$ `qi`
S/admires(gavin,john) $\rightarrow$ `j`
S/detests(john,mary) $\rightarrow$ `f`
S/detests(heather,pete) $\rightarrow$ `wkm`
S/detests(pete,mary) $\rightarrow$ `sm`
S/loves(heather,john) $\rightarrow$ `i`
S/hates(john,heather) $\rightarrow$ `xf`
S/loves(mary,gavin) $\rightarrow$ `bni`
S/admires(gavin,heather) $\rightarrow$ `yn`
S/hates(heather,pete) $\rightarrow$ `yya`

$S/$admires(x,john) $\rightarrow$ `f` A/x
$A/$mary $\rightarrow$ `lg`
$A/$pete $\rightarrow$ `nv`

This is the grammar of the very first speaker at the end of life. The reason the speaker has any grammar at all is due to the fact that every utterance that it invents it also uses for its own induction. The grammar is essentially an idiosyncratic vocabulary for a random subset of the meaning space. So, for example, the speaker's sentence corresponding to the English "Gavin hates John" is `jrx`, whereas the sentence for "Gavin likes John" is the completely unrelated `w`. This, then is a non-compositional, non-syntactically structured communication system. Notice, however, that a chance similarity of two sentences — `flg` meaning admires(mary,john) and `fnv` meaning admires(pete,john) — has lead to the creation of an A category for mary and pete.

Further on in this same simulation, we have grammars such as this one:

Generation 14
$S/$hates(pete,john) $\rightarrow$ `a`
S/p(john,x) $\rightarrow A/x$ B/p
$S/$likes(gavin,pete) $\rightarrow$ `lw`
$S/$hates(heather,john) $\rightarrow$ `z`
$S/p(x,$ mary) $\rightarrow$ `l` B/p A/x
S/p(pete,gavin) $\rightarrow$ `dx` E/p
$S/$admires(heather,mary) $\rightarrow$ `hhi`
$S/$likes(mary,pete) $\rightarrow$ `h`
$S/p(x,$ heather) $\rightarrow F/p$ A/x
$S/$hates(gavin,mary) $\rightarrow$ `rw`
$S/$detests(gavin,john) $\rightarrow$ `vow`
$S/$hates(heather,gavin) $\rightarrow$ `s`
$S/$detests(x,y) $\rightarrow D/x$ A/y
$S/$hates(mary,x) $\rightarrow D/x$ `rs`
$S/$hates(heather,pete) $\rightarrow$ `kw`
$S/$likes(heather,gavin) $\rightarrow$ `ufy`
$S/$loves(x,y) $\rightarrow A/y$ A/x
$S/$likes(x,y) $\rightarrow$ `l` C/y A/x
$S/$admires(x,y) $\rightarrow A/y$ C/x
$S/p(x,y) \rightarrow C/x$ B/p `n` A/y
$A/$gavin $\rightarrow$ `b`
$A/$mary $\rightarrow$ `ni`
$A/$john $\rightarrow$ `y`
$A/$heather $\rightarrow$ `x`
$A/$pete $\rightarrow$ `h`
$B/$loves $\rightarrow$ `y`
$B/$hates $\rightarrow$ `n`
$B/$likes $\rightarrow$ `z`
$B/$detests $\rightarrow$ `m`
$C/$pete $\rightarrow$ `t`
$C/$gavin $\rightarrow$ `yo`
$C/$heather $\rightarrow$ `gpi`
$C/$john $\rightarrow$ `d`
$D/$heather $\rightarrow$ `kr`
$D/$gavin $\rightarrow$ `q`
$E/$hates $\rightarrow$ `c`
$E/$detests $\rightarrow$ `rp`
$F/$detests $\rightarrow$ `r`
$F/$hates $\rightarrow$ `mofw`
$F/$admires $\rightarrow$ `u,d`

Here, we have some productive generalisations. For example, there are several words of category A, which can be used in different contexts. The A category in fact covers all the objects in the semantic space, although objects are not exclusively expressed in this way. For example, in different contexts, heather can be expressed as `gpi`, `kr` or `x`.

Turning time forward even further, we get some highly regular grammars:

Generation 112
$S/p(x,y) \rightarrow$
$C/y \;\; B/p$ `n` $\;\; A/x$
$S/p(x,y) \rightarrow$
$A/y \;\; C/x \;\; B/p$ `n`
$A/$gavin $\rightarrow$ `b`
$A/$mary $\rightarrow$ `ni`
$A/$pete $\rightarrow$ `re`
$A/$john $\rightarrow$ `y`
$B/$loves $\rightarrow$ `xfh`
$B/$hates $\rightarrow$ `n`
$B/$admires $\rightarrow$ `srw`
$B/$likes $\rightarrow$ `z`
$B/$detests $\rightarrow$ `m`
$C/$heather $\rightarrow$ `fkn`
$C/$pete $\rightarrow$ `t`
$C/$mary $\rightarrow$ `ns`
$C/$gavin $\rightarrow$ `yo`
$C/$john $\rightarrow$ `d`

Now the category B can clearly be thought of as a *verb*. There are two nominal categories C and A, giving us two types of expression for most of the objects in the semantic space as shown in the table below:

Meaning	type 1 (category C)	type 2 (category A)
mary	`ns`	`ni`
pete	`t`	`re`
gavin	`yo`	`b`
john	`d`	`y`
heather	`fkn`	—

There are now only two sentence rules. The first sentence rule gives us an OVS language, with the object nouns of type 1, and the subject nouns of type 2. In the other sentence rule, the word order is OSV. Interestingly, the two types of noun have switched roles, so the object nouns are of type 2, and the subject nouns are type 1.

This form of the language is fairly stable, losing the type 2 form of gavin, but otherwise remaining the same for thousands of generations. In fact, this is similar to the state of the "unusual" language in figure 1.3, which has a larger grammar at the end of its simulation run than those of the rest of the simulations. Eventually, however, the language goes through a rapid and complex series of changes to end up with the following form, which only has one type of noun:

Generation 7944
$S/p(x,y) \rightarrow$ `v` $\;A/y\;$ `g` $\;A/x\;\; B/p$ `n`
$A/$gavin $\rightarrow$ `gw`
$A/$john $\rightarrow$ `gbb`
$A/$pete $\rightarrow$ `k`
$A/$heather $\rightarrow$ `gyt`
$A/$mary $\rightarrow$ `pd`
$B/$hates $\rightarrow$ `n`
$B/$loves $\rightarrow$ `c`
$B/$detests $\rightarrow$ `m`
$B/$admires $\rightarrow$ `srw`
$B/$likes $\rightarrow$ `z`

This result is fairly typical of the simulation run started with different random-number seeds. The language in the population evolves from an idiosyncratic vocabulary for complex meanings to a completely compositional syntax with nominal and verbal categories. The main variation between runs is how quickly the coverage of the basic categories becomes

complete. Sometimes an idiosyncratic sentence rule for a particular action, or particular object survives for a long time, and very occasionally optionality in word order emerges and appears to be stable for a long time.

1.3.2 Infinite language, finite means

The simulation in the previous section used a finite meaning space. The next step is to expand the meaning space so that there is an infinite range of meanings that may be expressed. To do this we include predicates which may take other predicates as arguments. The simulation is run again with five "embedding predicates" (such as know, say etc.) Each speaker tries to produce 50 degree-0 meanings as before, but also then tries to produce 50 degree-1 meanings and finally, 50 degree-2 meanings.[7]

Because the potential expressivity of a language with this kind of semantics is infinite, we cannot visualise the behaviour of the simulation in the same way as we did for degree-0 meanings. Instead, figure 1.5 shows the proportion of degree-0, degree-1 and degree-2 meanings expressed without invention against time averaged over ten simulation runs. Also plotted on these graphs is a line showing the average size of the grammars in these runs.

Once again, this graph can best be understood by looking at the evolution of language in a particular simulation run. The first generation grammars for a simulation starting with these parameters are very large (over 100 rules), because there are three times as many utterances to be produced. Here is a small subset of a typical first generation grammar for this new set of conditions:

Generation 1
S/praises(pete,heather) → k
S/hits(john,mary) → u
S/admires(heather,pete) → y
S/hates(gavin,mary) → qv
S/says(mary,admires(gavin,mary)) → n
S/says(mary,praises(pete,gavin)) → te
S/decides(heather,hits(gavin,john)) → h
S/says(john,hits(mary,pete)) → q
S/knows(gavin,loves(pete,heather)) → r
S/believes(john,praises(heather,mary)) → ei
S/says(mary,loves(heather,gavin)) → l

[7] Notice, this presentation scheme simulates a "starting small" learning paradigm (Elman 1993; Kirby & Hurford 1997a).

S/thinks(gavin,loves(gavin,mary)) → a
S/decides(heather,hates(heather,pete)) → vi
S/decides(john,admires(heather,john)) → jj
S/says(heather,hits(gavin,john)) → lzf
S/decides(heather,hits(john,mary)) → apv
A/praises(heather,pete) → p
S/knows(gavin,p) → g A/p
A/admires(mary,gavin) → ws
S/says(mary,thinks(mary,praises(john,gavin))) → bx
S/thinks(pete,thinks(john,admires(pete,heather))) → gv
S/believes(pete,thinks(john,hates(john,heather))) → bc
S/believes(gavin,thinks(gavin,hates(heather,pete))) → im
S/believes(pete,decides(gavin,hates(pete,heather))) → lsq
S/decides(heather,believes(heather,admires(mary,pete))) → hjg
B/admires(mary,heather) → p
S/knows(pete,p) → m B/p d
B/knows(john,loves(john,mary)) → m
.
.
.

Firstly, notice that the vocabulary obviously includes more complex meanings such as says(mary, thinks(mary, praises(john,gavin))) (in English "Mary says she thinks that John praises Gavin"). As with the last simulation runs, the inducer has already done some work. So, the similarity between the sentences mpd meaning knows(pete, admires(mary,heather)) and mmd meaning knows(pete, knows(john, loves(john,mary))) has lead to the creation of a category B for admires(mary,heather) and knows(john, loves(john,mary)).

The grammars in the simulation rapidly increase in size and complexity, peaking in the mid 200's in terms of number of rules, and they also are quickly able to express the full range of degree-0 meanings using regular sentence rules rather like those that emerged in the simulation runs in the previous section. However, after some time the grammars typically reduce dramatically in size:

Generation 115
$S/p(x,q)$ → S/q C/p gp B/x d
$S/p(x,y)$ → stlw A/p B/y B/x
A/loves → r
A/admires → i
A/hates → wja
A/detests → w
A/likes → btl
B/pete → f
B/heather → v
B/gavin → eks
B/mary → k
B/john → a
C/says → fdbtl
C/decides → b
C/believes → o
C/knows → z
C/thinks → t

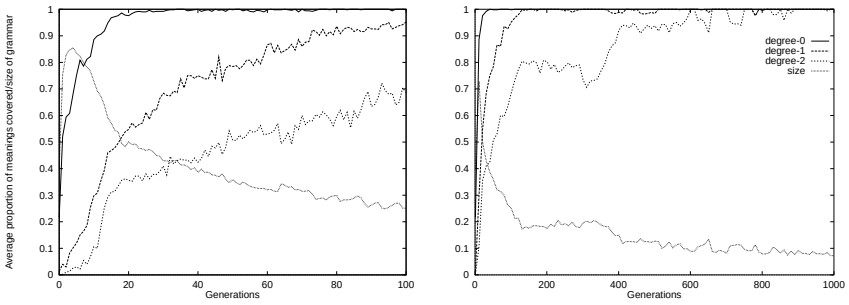


figure 1.5. Proportions of meaning space covered and size of grammars averaged of ten simulation runs, plotted on two different time-scales. The grammar size is scaled down by a factor of 300 in order that it can be plotted on the same scale. Coverage of the different meaning types increases and the size of the grammar decreases over time.

There are two sentence rules in this grammar, and three other categories. The second sentence rule is similar to the ones we saw in the previous section, allowing the language to express the full range of degree-0 sentences. The category A is a verbal category, and B is the nominal category. This language has VOS order in main clauses.

The other sentence rule is the one that allows the language to express meanings greater than degree-0. It introduces a category C for verbs that have a subordinating function (such as `fdbtl` meaning says), and crucially has a category S on its right hand side. This means that the language is recursive, allowing it to build up an infinite range of meanings. The tree in figure 1.6 shows how this particular language copes with complex meanings. It displays the parse for the sentence `stlwrkazgpfd` which, translated into English, means "Pete knows that John loves Mary".

Again, the language in the simulation has evolved simply by being learned and used repeatedly by individuals in the population. An initially random, idiosyncratic non-compositional and relatively inexpressive communication system, has become a compact, compositional language with nominal and verbal categories, word order encoding meaning distinctions and recursive subordinate clauses allowing the speakers to express an infinite range of meanings. The question is why?

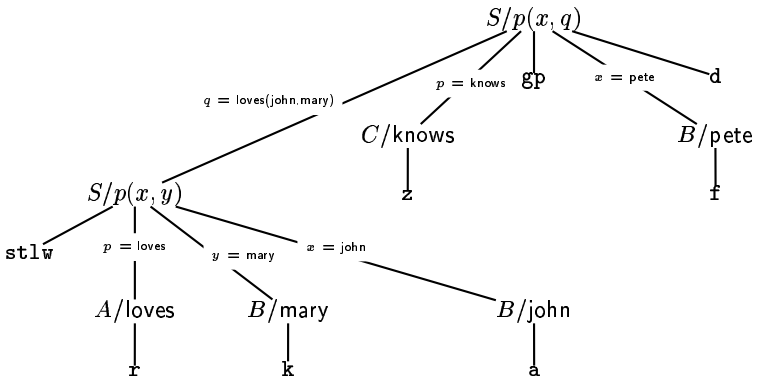


figure 1.6. "`stlwrkazqpfd`" meaning *Pete knows that John loves Mary.*

1.4 Bottlenecks and universal grammar

The individuals in the simulation simply observe each other's behaviour and learn from it, occasionally inventing, at random, new behaviours of their own. From this apparent randomness, organisation emerges. Given that so little is built into the simulation, why is compositional, recursive syntax inevitable? To answer this question we need to look back at how languages persist over time in a population (figure 1.1).

We can divide up I-language into units — *replicators* — that may or may not persist through time. The persistence of an I-language in this view is related to the success of the replicators that make up that language. In other words, the languages which are more easily transmitted from generation to generation will persist.

Within a population, certain replicators actually compete for survival. That is, the success of one must be measured relative to the success of others in the population at that time. These competing replicators are those rules which potentially express the same meaning. If there are two ways of saying *John loves Mary*, then on a particular exposure to this meaning, the learner can obviously only hear one of them. Therefore, on one exposure, only one of the rules (or, more properly, set of rules) that can be used to express *John loves Mary* has a chance of being induced by the learner.

At face value, it would seem that the two competing rules (or rule-

sets) will have an equal chance of being the one chosen for producing the meaning, so the replicative success of all rules in a language should be equal. This would be true *if each rule only ever expressed one meaning.* However, if one rule can be used to express more meanings than another, then, all other things being equal, that rule will have a greater chance of being expressed in the E-language input to the learner. In this case, the more general rule is the better replicator.

For a more concrete example, consider a situation where, in the population of I-languages, there are two competing rules. One is a rule that expresses *John loves Mary* as an unanalysed string of symbols — essentially as one word. The other rule expresses *John loves Mary* as a string of symbols, but can also be used to express any meaning where someone *loves Mary.* So, the latter rule can also be used to express *Gavin loves Mary* and so on. Further imagine that both rules have an equal chance of being used to express *John loves Mary.* The more general rule is a better replicator, because for any randomly chosen set of meanings, we can expect it to be used more often than the idiosyncratic rule. Its chances of survival to the next generation are far more secure than the idiosyncratic rule.

Of course, the more general rule will not be learned as easily as the idiosyncratic rule. In the simulations described above, an idiosyncratic pairing of one meaning to one form takes only one exposure to learn, but the most general rule takes several. However, the idiosyncratic rule only covers one meaning, whereas the most general rule covers an infinite number. It is clear, therefore, that the probability of a acquiring a particular rule given any sample of meanings increases with the generality of that rule. The success of I-languages which contain general rules seems secure.

The picture that emerges, then, is of the language of the population acting as an adaptive system in its own right. Initially, the rules are minimally general, each pairing one string with one meaning. At some point, a chance invention will lead a learner to "go beyond the data" in making a generalisation that the previous generation had not made. This generalisation will then compete with the idiosyncratic rule(s) for the same meaning(s). Given that generalisations are better replicators, the idiosyncratic rules will be pushed out over time. The competition will then be replayed amongst generalisations, always with the more general rules surviving. (Notice that this picture of a move from holistic protolanguage to an emergent syntactic system is similar to the one

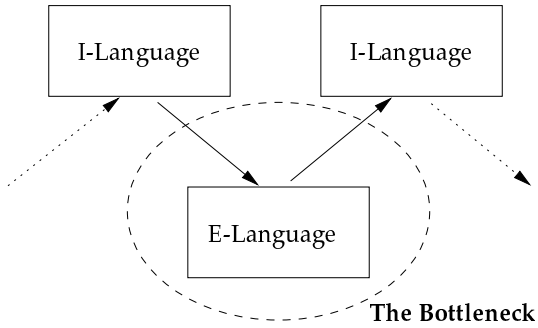


figure 1.7. The E-language domain acts as a bottleneck on the transmission of I-language.

proposed by Wray (1998).)

The inevitable end state of this process is a language with a syntax that supports compositionally derived semantics and recursion in a highly regular fashion. The grammar for such a language appears to be the shortest (in terms of numbers of rules) that can express the entire meaning space. The shorter the grammar, the higher the generality of each of the rules — the shortest grammar that can still do the job of expressing meanings is therefore the one made up of optimal replicators.

We can think of the transformations between I- and E-language as a bottleneck on the transmission of language over time (see figure 1.7). Since the number of meanings that the learners are exposed to is always lower than the total number of meanings, a totally idiosyncratic language *cannot* survive. In order to see this, we can visualise the contrast between idiosyncratic and syntactic languages in terms of types of mappings between structured spaces. Figure 1.8 is a schematic representation of a possible mapping between two spaces. This mapping does not preserve structure from one space to the other. In other words, there is a random relation between a point in the space and its corresponding point in the other space.

Now, imagine that this mapping must be learned. In the diagram, some of the pairings are shown in bold — if these where the only ones a learner was exposed to, would that learner be able to reconstruct the whole mapping? Not easily: for a finite space, the only way a random mapping could be reliably learnt from a subset of pairings would be if the learner had a very informative and domain specific prior bias to learn

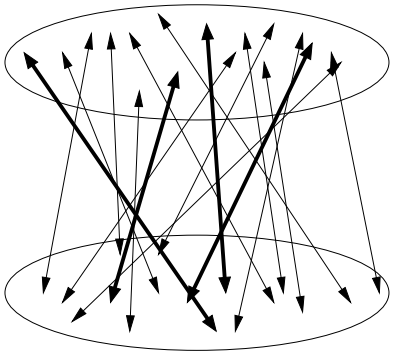


figure 1.8. A non-structure preserving mapping between two spaces with spatial structure. The bold lines indicate an imaginary subsample of the mapping that might be evidence for a learner. This mapping could only be learnt by a learner with a very specific prior bias.

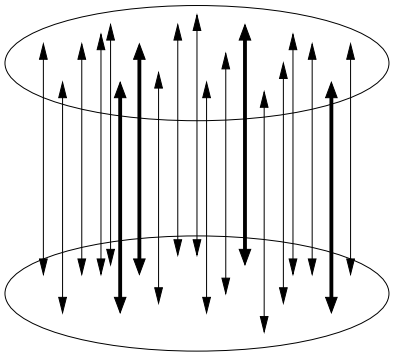


figure 1.9. A mapping in which structure is preserved. The bold lines indicate an imaginary subsample of the mapping that might be evidence for a learner. This mapping is more likely to be successfully learnt by a learner with a more general prior bias.

that particular mapping. Even this is not possible where the spaces are potentially unbounded.

Figure 1.9 on the other hand, shows a mapping in which structure in one space is preserved in the other. Given the sample in bold, it seems that a learner has a higher chance of reconstructing the mapping. A learner that is biased to construct concise models, for example, would learn this mapping more easily than that in the first figure. Importantly,

this bias is more likely to be domain general than one that explicitly codes for a particular idiosyncratic mapping. Furthermore a model can be constructed that would map the spaces even if they were potentially infinite in extent.

In the second set of simulations, as in real language, what is being learnt is a mapping between a meaning space and a signal space both of which are potentially infinite in extent. This means that it is in principle impossible for a learner to acquire a language that looks like the mapping of the first type, that is an idiosyncratic pairing of meanings and strings. This is why the initial, random languages in the simulations are unstable over time. This is not a feature of syntactically structured languages, however. Structure in the mapping improves the survivability of that mapping from one generation to the next.

What we are left with is a very general story about the (cultural) evolution of mappings. Structure-preserving mappings are more successful survivors through the learning bottleneck. This fact, coupled with random invention of pairings in languages that have incomplete coverage of the meaning space, and the unboundedness of the meaning and signal spaces, leads inevitably to the emergence of syntax.

At the start of this chapter the approach taken here was contrasted with the dominant approach in evolutionary linguistics, where the structure of language is taken to match closely with the structure of the language faculty which in turn is shaped by natural selection. We can now more precisely unpack the differences between these two perspectives on the origins of syntax. In particular, the relationship between the model of the acquirer and constraints on cross-linguistic variation are quite different.

Traditionally, the Chomskyan language acquisition device (LAD) directly constrains what makes a possible human language by limiting directly what can or cannot be acquired. This limit is said to closely map the observed constraints on variation (Hoekstra & Kooij 1988). Part of the generative research program involves accounting for variation between languages explicitly within the model of the language acquirer. In fact, Universal Grammar (UG) and the LAD are often treated as synonymous within this tradition. It is not generally considered that the dynamics of language acquisition and use impose further constraints within the boundaries imposed by the structure of the LAD (although see Niyogi & Berwick (1995) & Clark (1996) for interesting exceptions).

Figure 1.10 contrasts this view with that proposed in this paper. The

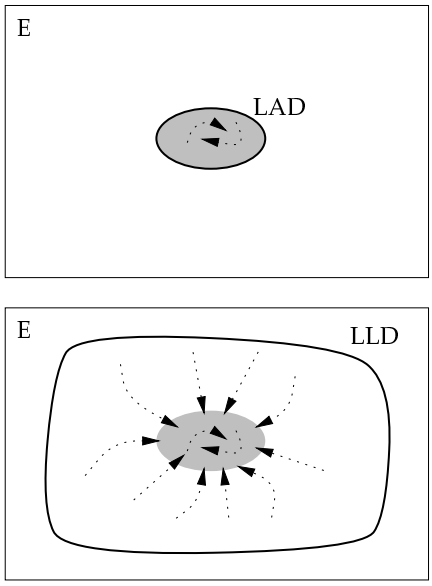


figure 1.10. Two Venn diagrams showing the different approaches to explaining observed constraints on cross-linguistic variation. E is the set of all logically possible languages, the gray area signifies the set of occurring human languages. In the top diagram, the Chomskyan language acquisition device constrains the learner directly and nothing else is required to explain the limits on variation. In the bottom diagram, the language *learning* device is less constraining, and the particular characteristics of human languages are the end result of a historical evolution of languages in populations (represented by arrows).

language learning device clearly does impose constraints directly in a similar fashion — there are certain types of language that the learner simply cannot acquire — however these constraints are far less severe than those imposed by the Chomskyan model of the LAD. As can be seen in the initial stages of the simulation, very un-language like systems can be acquired by this learner. The constraints on variation are not built into the learner, but are instead emergent properties of the social dynamics of learned communication systems and the structure of the semantic space that the individuals wish to express.

The theory presented here gives us a neat explanation of why human languages use syntactic structure to compositionally derive semantics, use recursion to express infinite distinctions in a digital way, have words

with major syntactic categories such as noun and verb, and use syntactic rules of realisation (such as ordering rules) to encode meaning distinctions. However, it does not seem to allow us to understand more specific universals. For example, why particular constituent orders are far more frequent than others across the languages of the world (Hawkins 1983; Dryer 1992).

Perhaps the best explanation for these types of universal should look at the effect of parsing and generation on the transmission of replicators (see Kirby 1999 and Kirby 1997 for details). On the other hand, at least some of these word order constraints may eventually be explained in terms of linguistic adaptation without appealing to processing (see, Christiansen 1994 and Christiansen & Devlin 1997 for some suggestions along these lines). X-bar theory — a sub part of UG which constrains the structure of syntactic trees cross categorially (Jackendoff 1977) — has been implicated in various word order universals. Daniel Nettle (personal communication) has suggested that X-bar is just the sort of over-arching generalisation that the theory put forward in this chapter predicts. It can be characterised as a pair of phrase structure rules:

$$XP \rightarrow Spec\ X' \quad \text{or} \quad XP \rightarrow X'\ Spec$$
$$X' \rightarrow X\ YP \quad \text{or} \quad X' \rightarrow YP\ X$$

These rules are like standard context free rules except that X and Y are variables that can range over the lexical categories in the language.

This use of variables in phrase structure rules is not possible with the formalism adopted here, so this result is not possible in the simulation. Nevertheless, if the language learning device were able to make a generalisation such as that expressed by X-bar, we would expect it to thrive as a replicator. More generally, we should expect languages to behave in such a way that their word orders can be expressed in the most compact way, since this will reflect the behaviour of the optimal, most general, replicator. Dryer (1992) shows with a large-scale cross-linguistic survey, that this is indeed the case; languages tend to order their non-branching nodes on the same side of their branching nodes across the phrasal categories of the language.

1.5 Conclusion

Compositionality and recursion are arguably the most basic features of the syntax of language. These structural properties, along with the way

it is transmitted, are what makes human language a unique natural communication system. This chapter has presented an explanation of the origins of the properties which does not require them to be built-in as hard constraints on learning. This lifts the burden of explanation away from the biological evolution of the human genome and instead relies on very general properties of the dynamics of mappings that must replicate over time through learning.

For a language to survive from generation to generation it must be learned by individuals observing the behaviour of other individuals. The sample of observations will be finite, yet the range of meanings that individuals may wish to communicate about is likely to be very large or infinite. This learning bottleneck leads inevitably to the emergence of a language in which structure is preserved in the mapping between semantics and strings in utterances.

The working model of linguistic transmission presented in this chapter has provided a demonstration of this process of emergence; compositional, recursive grammars arise given a particular model of learning and a particular model of semantics. Treating language as an adaptive system in its own right, in which properties of information transmission impact on its emergent structure, opens up new avenues of explanation in linguistics. Before seeking a biological or functional explanation for a particular feature of human language, or appealing to direct coding in an innate acquisition device, we should be aware of what we might be getting "for free" through the kinds of processes described here.

Bibliography

Batali, J. (1994). Innate biases and critical periods: Combining evolution and learning in the acquisition of syntax. In R. Brooks and P. Maes (Eds.), *Artificial Life IV*, pp. 160–171. MIT Press.

Batali, J. (1998). Computational simulations of the emergence of grammar. In J. Hurford, C. Knight, and M. Studdert-Kennedy (Eds.), *Approaches to the Evolution of Language: Social and Cognitive Bases*, Cambridge, pp. 405–426. Cambridge University Press.

Bickerton, D. (1990). *Language and Species*. University of Chicago Press.

Briscoe, E. J. (1997). Co-evolution of language and of the language acquisition device. In *35th Association for Computational Linguistics*, pp. 418–427. Morgan Kaufmann.

Briscoe, E. J. (1998). Language as a complex adaptive system: co-evolution of language and of the language acquisition device. In P. Coppen, H. van Halteren, and L. Teunissen (Eds.), *8th Meeting of Comp. Linguistics in the Netherlands*, Amsterdam, pp. 3–40. Rodopi.

Cangelosi, A. and D. Parisi (1996). The emergence of a language in an evolving population of neural networks. Technical Report NSAL–96004, National

Research Council, Rome.
Chomsky, N. (1986). *Knowledge of Language*. Praeger.
Christiansen, M. (1994). *Infinite Languages, Finite Minds: Connectionism, Learning and Linguistic Structure*. Ph. D. thesis, University of Edinburgh.
Christiansen, M. and J. Devlin (1997). Recursive inconsistencies are hard to learn: A connectionist perspective on universal word order correlations. In *Proceedings of the 19th Annual Cognitive Science Society Conference*, pp. 113–118. Mahwah, NJ: Lawrence Erlbaum Associates.
Clark, R. (1996). Internal and external factors affecting language change: A computational model. Master's thesis, University of Edinburgh.
Dryer, M. (1992). The Greenbergian word order correlations. *Language 68*, 81–138.
Elman, J. L. (1993). Learning and development in neural networks: the importance of starting small. *Cognition 48*, 71–99.
Hawkins, J. A. (1983). *Word Order Universals*. Academic Press.
Hoekstra, T. and J. G. Kooij (1988). The innateness hypothesis. In J. A. Hawkins (Ed.), *Explaining Language Universals*. Blackwell.
Hurford, J. (1987). *Language and Number: the Emergence of a Cognitive System*. Cambridge, MA: Basil Blackwell.
Hurford, J. (1989). Biological evolution of the Saussurean sign as a component of the language acquisition device. *Lingua 77*, 187–222.
Hurford, J. (1991). The evolution of the critical period for language acquisition. *Cognition 40*, 159–201.
Hurford, J. (1998). Social transmission favours linguistic generalisation. In C. Knight, J. Hurford, and M. Studdert-Kennedy (Eds.), *The Emergence of Language*. To appear.
Hurford, J., C. Knight, and M. Studdert-Kennedy (Eds.) (1998). *Approaches to the Evolution of Language: Social and Cognitive Bases*, Cambridge. Cambridge University Press.
Jackendoff, R. (1977). *X-Syntax: A Study of Phrase Structure*. MIT Press.
Kirby, S. (1997). Competing motivations and emergence: explaining implicational hierarchies. *Language Typology 1*, 5–32.
Kirby, S. (1998a). Language evolution without natural selection: From vocabulary to syntax in a population of learners. Technical Report EOPL–98–1, Department of Linguistics, University of Edinburgh.
Kirby, S. (1998b). Syntax without natural selection: How compositionality emerges from vocabulary in a population of learners. In C. Knight, J. Hurford, and M. Studdert-Kennedy (Eds.), *The Emergence of Language*. To appear.
Kirby, S. (1999). *Function, Selection and Innateness: the Emergence of Language Universals*. Oxford: Oxford University Press.
Kirby, S. and J. Hurford (1997a). The evolution of incremental learning: Language, development and critical periods. Occasional Paper EOPL-97-2, Department of Linguistics, University of Edinburgh, Edinburgh.
Kirby, S. and J. Hurford (1997b). Learning, culture and evolution in the origin of linguistic constraints. In *Fourth European Conference on Artificial Life*, pp. 493–502. MIT Press.
MacLennan, B. (1991). Synthetic ethology: an approach to the study of communication. In C. Langton, C. Taylor, J. Farmer, and S. Ramussen (Eds.), *Artificial Life II*, pp. 631–657. Addison-Wesley.
Mitchell, T. M. (1997). *Machine Learning*. New York: Mc Graw Hill.

Newmeyer, F. J. (1991). Functional explanation in linguistics and the origins of language. *Language and Communication 11*, 3–28.

Niyogi, P. and R. Berwick (1995). The logical problem of language change. Technical Report AI Memo 1516 / CBCL Paper 115, MIT AI Laboratory and Center for Biological and Computational Learning, Department of Brain and Cognitive Sciences.

Niyogi, P. and R. Berwick (1997). Populations of learners: the case of Portugese. Unpublished manuscript, MIT.

Oliphant, M. (1996). The dilemma of saussurean communication. *BioSystems 37*, 31–38.

Pinker, S. and P. Bloom (1990). Natural language and natural selection. *Behavioral and Brain Sciences 13*, 707–784.

Steels, L. (1996). Emergent adaptive lexicons. In P. Maes (Ed.), *Proceedings of the Simulation of Adaptive Behavior Conference*. Cambridge, Ma: The MIT Press.

Wray, A. (1998). Protolanguage as a holistic system for social interaction. *Language and Communication 18*, 47–67.

A.1 Details of rule subsumption

This appendix gives a more thorough treatment of the rule subsumption approach introduced in section 1.2.1. The algorithm uses two methods of subsumption:

Merge *If* the two rules would be the same if two category symbols were merged, *then* merge those categories. In other words, pick one of the categories and rewrite the other one to be the same as it throughout the grammar.

Chunk *If* the two rules would be the same if either one or both of them chunked a sequence of terminals, *then* chunk those terminals. Chunking involves creating a new rule made up of a substring of nonterminals on the right hand side of the old rule, and adjusting the old rule to refer to the new one.

Whilst rule subsumption through merging is straightforward, chunking is rather more difficult to implement. It is best to describe the procedure step-by-step:

1. Take a pair of rules, r_1 and r_2 from the grammar with the same left hand side category symbol, C.
2. Can chunking can be applied to both rules?
   (a) Do the left hand side semantics of the two rules differ in only one position? If so, call the differences m_1 and m_2. If there is no difference, or if there is more than one difference then stop.
   (b) Are there two strings of terminals that, if removed, would make the right hand sides of the two rules the same? If so, call this string difference λ_1 and λ_2. If there isn't one string difference, then go to step 3.
   (c) Create a new category N.
   (d) Create two new rules:

$$N/m_1 \rightarrow \lambda_1$$
$$N/m_2 \rightarrow \lambda_2$$

   (e) Replace the old rules r_1 and r_2 with one rule. This rule is identical to r_1 (or r_2) except that λ_1 (or λ_2) is replaced with C/x on the right hand side, and m_1 (or m_2) is replaced with the variable x on the left hand side.
   (f) Stop.
3. Can chunking can be applied to just one of the rules?

(a) Can the left hand side semantics of the two rules can be unified? If not, stop.
(b) Is there a string of terminals λ in one of the rules which corresponds to a nonterminal label N/m in the other rule? In other words, is this the only difference in the two rules' right hand sides? If not, stop.
(c) Delete the rule containing the substring λ.
(d) Create a new rule:

$$N/m \rightarrow \lambda$$

(e) Stop.

We can work through this chunking procedure using the example in section 1.2.1.

1. start with the two rules, r_1 and r_2:

$$S/\mathsf{eats}(\mathsf{tiger},\mathsf{sausages}) \rightarrow \texttt{tigereatssausages}$$
$$S/\mathsf{eats}(\mathsf{john},\mathsf{sausages}) \rightarrow \texttt{johneatssausages}$$

These have the same left hand side category symbol, S.
2. check to see if chunking can be applied to both of these rules.
   (a) the left hand side semantics differ in one position, so $m_1 = \mathsf{tiger}$ and $m_2 = \mathsf{john}$.
   (b) the shortest pair of strings of terminals that could be removed from both rules to make them the same is `tiger` and `john`, so $\lambda_1 = \texttt{tiger}$ and $\lambda_2 = \texttt{john}$.
   (c) we make up a new category name — for convenience, we'll call it N.
   (d) the two new rules are therefore:

$$N/\mathsf{tiger} \rightarrow \texttt{tiger}$$
$$N/\mathsf{john} \rightarrow \texttt{john}$$

   (e) the two old rules are replaced with a single more general one:

$$S/\mathsf{eats}(x,\mathsf{sausages}) \rightarrow N/x\ \texttt{eatssausages}$$

   (f) stop.

With merging and chunking, the inducer can successfully discover new rules that subsume pairs of rules that it has learnt through simple incorporation. However, in practice it is useful to add an other procedure to the induction algorithm which also makes rules more general.

Wherever possible, the inducer tries to simplify its rules by utilising other rules that are already in the grammar. So, for example, if we had the following pair of rules:

$$S/\mathsf{loves}(\mathsf{john},\mathsf{mary}) \rightarrow \texttt{johnlovesmary}$$
$$N/\mathsf{mary} \rightarrow \texttt{mary}$$

the inducer would simplify the first one to:

$$S/\mathsf{loves}(\mathsf{john},x) \rightarrow \texttt{johnloves}\ N/x$$

index