Effects of Compression on Language Evolution

Tracy K. Teal
Charles E. Taylor
Department of Organismic Biology, Ecology, and Evolution, Box 951606
University of California at Los Angeles
Los Angeles, CA 90095, USA

Abstract For many adaptive complex systems information about the environment is not simply recorded in a look-up table, but is rather encoded in a *theory*, *schema*, or *model*, which compresses information. The grammar of a language can be viewed as such a schema or theory. In a prior study [Teal et al., 1999] we proposed several conjectures about the learning and evolution of language that should follow from these observations: (C1) compression aids in generalization; (C2) compression occurs more easily in a "smooth," as opposed to a "rugged," problem space; and (C3) constraints from compression make it likely that natural languages evolve towards smooth string spaces. This previous work found general, if not complete support for these three conjectures. Here we build on that study to clarify the relationship between Minimum Description Length (MDL) and error in our model and examine evolution of certain languages in more detail. Our results suggest a fourth conjecture: that all else being equal, (C4) more complex languages change more rapidly during evolution.

Keywords
compression, language, artificial life, evolution

1 Introduction

A key feature of nearly all successful complex adaptive systems is the ability to distill information about the environment into schemas and then use these models to make predictions or adapt to new situations [5, 6]. Instead of simply recording information in a look-up table, compact representations are created by identifying and using regularities that exist in the data [18]. This compression not only allows large amounts of information to be stored more efficiently, but can also enable the system to generalize.

Language learning, as a mapping from a set of observed data to a grammar, or model, of the language, is a paradigm for which compression is extremely important. Sentences cannot simply be recorded in a look-up table. Recording sentences in a look-up table is not only an inefficient use of space, but only sentences previously heard could be spoken and no novel sentences or words could be generated. Human languages are extremely complex with semantics, syntax, and an inherent linguistic structure all potentially important factors for learning [3, 16]. Because these factors make human language difficult to study analytically, formal languages can be used instead. These languages are well understood and generally contain fewer confounding variables [9, 10, 12, 13, 15].

We have therefore chosen to study the effects of compression on language acquisition in the context of formal languages. To minimize the number of assumptions made about the way language is learned, we have developed a model of language acquisition that learns syntactical languages from positive input alone. It has been shown that

regular languages can be learned from this type of input [4, 20]. Our system is simple enough to be understood, but contains two theoretical properties: Input can be of varying complexity and there is no genetic transmission of the data. Both of these are important for our investigations. By using a simple system, with few model-specific variables, we hope that some of our results can be applied generally to language and cultural evolution and other types of systems that employ compression.

In previous work [20] we studied some of these general features of compression as related to adaptation. In this work we proposed and investigated three conjectures. We initially found the first conjecture, (C1) that compression aids in generalization, to be true. From this basis we proposed that (C2) compression occurs more easily on a "smooth" as opposed to "rugged" problem space and (C3) that constraints from compression make it likely that natural languages come to have smooth problem spaces. In this article we first provide some background on the data compression and introduce the *Minimum Description Length* algorithm which we use as the compression algorithm in our model. Next, we discuss formal languages and finite state automata, the grammar representation used. Then, we discuss methods used for language acquisition. Last, we review some previous results, expanding on them to discuss problems in conjectures 2 and 3, and extend them to investigate a fourth conjecture, that (C4) more complex languages change more rapidly during evolution.

2 A Model of Compression

2.1 Data Compression

We are interested in how compression affects the ability of a system to generalize and adapt. Compression requires a *model* of regularity, such as a linear relationship in a set of data points. It is this model that permits generalization [18]. Consider a set of points arranged more or less along a line in the xy plane. These might be stored as a look-up table of x, y pairs; or they might be represented as all points on a line, and stored simply as the equation for a line in the x, y plane. From the compressed representation, the equation, it is easy to interpolate or to extrapolate the data in ways for which a look-up table would not suffice. Of course it may also be the case that the *true* relationship among the variables is not linear—for example, there might be gaps in their arrangement, or they might plateau just after the highest data point, so that interpolation or extrapolation is not warranted. Also, the relationship might be just *approximately* linear, and not exactly so. There is clearly a trade-off between generalization and error that will be specific to each situation. It is also important to recognize that some compression schemes might provide very efficient coding of data, but would involve lengthy and complicated computation to encode and decode it.

The *minimum description length (MDL)* algorithm is a widely used algorithm which recognizes both factors. First introduced by Rissanen and Ristad [17], it states that the best model is that which permits the shortest encoding of the observed data together with the model itself. The performance of a model is therefore measured by both the length of the description of the theory and the length of the data when encoded with the help of the theory. When this combined code length is minimized a balance has been struck between the correctness of the model and its complexity [8]. The MDL measure has been used extensively in the fields of statistics and machine learning, and its theoretical properties have been well investigated [14, 17]. It has also been used to successfully model language acquisition and biological computation [1, 8, 14, 17].

2.2 Formal Language and FSAs

The data we are encoding are strings that can be analyzed in the context of formal languages. A formal language is a set of strings of symbols from some finite alphabet,

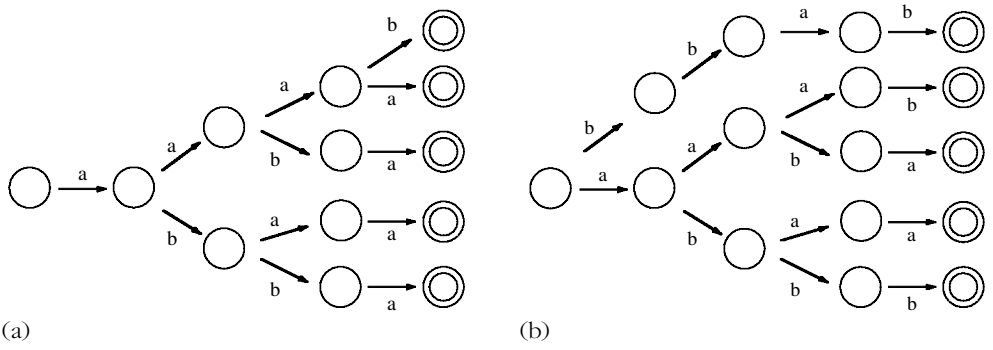


Figure 1. Finite state diagrams for uncompressed grammars: (a) Language 1 (b) Language 2. Double circles are the end or stop states.

where a string is a finite sequence of symbols juxtaposed and an alphabet is a finite set of symbols. For example, a, b and c are symbols in the alphabet (a,b,c) and abac is a string [10]. A language consists of a set of these strings. The language can be a random set of strings such as abcc, bacbc, bbaca or it can contain some regularity such as the language a[ab][ab] which is the set of all strings drawn from the alphabet (a,b) of length three that start with an a—that is, aaa, aab, aba, abb.

These languages can be represented as directed graphs, termed "deterministic finite state automata" (FSA) having unique vertices for each symbol connecting the states [10]. This representation allows states to be combined so that a more compressed model of the language can be found—a feature important for our model of language acquisition. The graphs in Figures 1a and 1b for Language 1 (aaab, aaaa, aaba, abaa, abba) and Language 2 (bbab, aaab, aaba, abaa, abbb) are deterministic finite state automata. Each string or sentence in the language is a path along the nodes or states represented by circles. The description of this FSA is then the *grammar* for the language. The diagram is termed the *finite state diagram* for the language.

Another way to describe the language is with a quintuple, $(Q, \Sigma, \delta, q_0, F)$, where Q is a set of states, Σ is an input alphabet, $\delta \subseteq Q \times \Sigma \times Q$ is a transition function, $q_0 \in Q$ is the initial state, and $F \subseteq Q$ is the set of final states [10]. We will use this representation for determining the MDL, below. For the examples in Figure 1 it can be seen that Q are states represented by circles, $\Sigma = (a, b)$, δ is the set of all the

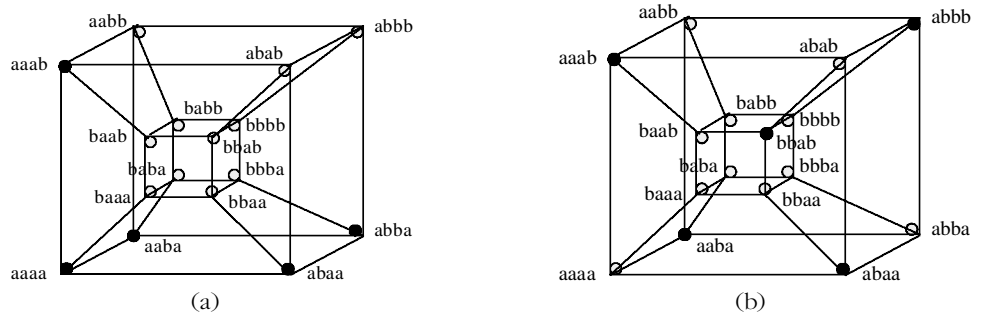


Figure 2. Hypercube representations of the string space landscape for (a) Language 1 and (b) Language 2. Nodes connected by lines are one symbol-substitution away from each other. Black dots are included in the language. Gray dots are not included.

transitions, or circle, arrow, circle combinations, q_0 = the starting circle, and F = states represented by double circles. The automaton accepts a string s if, and only if, the string labels a path from the initial state to a final state. Since an FSA is the defining grammar for a language, any string that can be accepted by the FSA is said to be a part of that language.

2.3 Language Acquisition

Language learners can be regarded as systems that identify rule systems that describe the (potentially infinite) language of the community after being presented with a finite set of examples. This can be successful only in certain circumstances, depending on whether one assumes that success is perfect identification in the limit (the "Gold" paradigm) [7], or that success is feasible convergence to arbitrarily good approximate identification [11, 21].

2.3.1 The Source of Input

In our model the language being learned consists of sets of strings of varying smoothness, with string length 6 drawn from the alphabet (a,b) or (a,b,c). The smoothness or ruggedness of the set, and therefore the language, is determined by the set's *string-edit-distance*, d. To generate the language which will be used as the target language, a single string of symbols of length 6 is randomly selected. The next string in the set is made by changing one or more symbols of that string, depending on the string-edit-distance. For example, if a language of $d = 2$ is being generated and string abaaba is drawn at random, one position in the string would be chosen randomly and the symbol at that position would be changed. Then another position in the string would be randomly selected and its symbol changed. This process is continued until the specified set size has been generated. Languages generated with $d = 1$ are considered smooth, while languages of $d = 4$ are more rugged.

2.3.2 The Model of Language Learning

Our model of language learning will mimic that of a child, who when listening to an adult speaking a language she/he does not yet understand, tries out different candidate grammars that might accept that language. The approach followed here supposes that the child tentatively accepts the grammar with the shortest Minimum Description Length from among those deterministic grammars that accept all the input sentences.

To calculate the MDL we take the sum of the *data-encoding-length* (Δ), the cost of the coding of the data, and the *grammar-encoding-length* (Γ), the cost of the rule set.

Since each string can be specified by a path through a deterministic automaton, we find Δ by counting the number of choices that have to be made as each sentence's path is traced out. This is calculated in bits by the formula:

$$\Delta = \sum_{i=1}^{m} \sum_{j=1}^{|s_i|} \log_2 z_{i,j},$$

where m is the number of sentences in the sequence of strings encoded, $|s_i|$ is the length of the ith string s_i, and $z_{i,j}$ is the number of ways to leave the state reached on the jth symbol of sentence s_i. (A more succinct encoding is obviously possible when the probabilities of the transitions are not uniform. This approximation suffices for purposes of this preliminary investigation.)

In this model, Γ is the number of bits required to encode the FSA. To specify the automaton itself, we must specify all the triples $(q_1, a, q_2) \in \delta$ and we must also specify

Algorithm 1. Best grammar derivable in one step (*current-grammar*).

```
1: best-encoding ← ∞
2: best-grammar ← ∅
3: for all possible pairings (s1, s2) of states from current-grammar do
4:    if s1 and s2 can be combined into a single state without making a
      nondeterministic FSM then
5:       grammar ← CombineStatesInGrammar(current-grammar, s1, s2)
6:       Calculate the encoding length for this grammar via the equations given.
7:       if encoding(grammar) < best-encoding then
8:          best-encoding ← encoding(grammar)
9:          best-grammar ← grammar
10:      end if
11:   end if
12: end for
```

the final states, so we calculate the grammar-encoding-lengths as

$$\Gamma = |\delta|[2(\log_2|Q|) + \log_2|\Sigma|] + |F|[\log_2|Q|],$$

where $|\delta|$ is the number of triples in δ, $|Q|$ is the number of states, $|\Sigma|$ is the size of the alphabet and $|F|$ is the number of final states in F.

Using this algorithm to determine the appropriateness of its grammar, the agent learns the language presented in the following way:

An agent is exposed to all the legal sentences in the language—say of Language 1 or Language 2. It constructs a deterministic automaton with an associated grammar which accepts all of the sentences from the initial input and nothing else. This is a prefix tree FSA where each sentence is explicitly drawn out and no states are combined, as seen for Language 1 and Language 2 in Figure 1a and Figure 1b. The agent then compresses the original grammar by attempting to combine states and transitions in such a way that the outcome is still a deterministic automaton and then combines them if and only if the MDL of the new automaton is smaller than that of the starting one. Given a machine $A = (Q, \Sigma, \delta, q_0, F)$, the result of merging states $q_i, q_j \in Q$ is the machine A' which has these two states replaced by a new state q_{ij} as follows:

$$A' = ((Q - q_i, q_j) \cup \{q_{ij}\}, \Sigma, \delta', q_0', F')$$

where: $q_0' = q_{ij}$ if q_0 is either q_i or q_j, $F' = (F - q_i, q_j) \cup \{q_{ij}\}$ if either q_i or $q_j \in F$, and δ' is the result of replacing all instances of both q_i and q_j by q_{ij} in all the triples (q_n, a, q_m) that define δ. This is the best grammar found at one step according to *Algorithm 1*, outlined above.

A hill-climbing search is then performed using *Algorithm 2*, until the FSA with the smallest MDL is found.

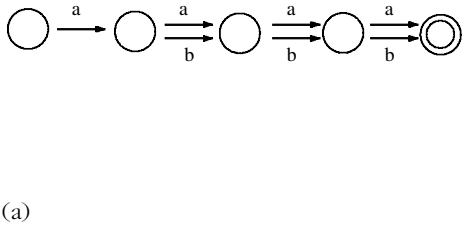
Take, for example, Languages 1 and 2 above. The prefix tree FSA for Language 1 shown in Figure 1a has $\Gamma = 119.3$, $\Delta = 12.0$, and $MDL = 131.3$. The program runs until it cannot find an FSA with a smaller MDL. This gives the grammar in Figure 3a with $\Gamma = 41.8$, $\Delta = 15.0$, and $MDL = 56.8$. The cost trade-offs between the data-encoding-length and the grammar-encoding-length have been optimized. A language with string edit distance 2, Language 2, compresses to the grammar in Figure 3c with $\Gamma = 75.6$, $\Delta = 17.0$, and the $MDL = 92.6$ from $\Gamma = 155.0$, $\Delta = 13.0$, and $MDL = 168.0$.

Algorithm 2. Hill-climbing search (*current-grammar*).

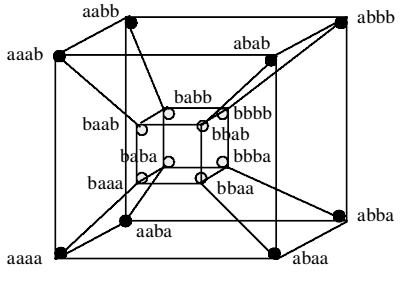
```
1: current-length ← encoding length of current-grammar
2: loop
3:    grammar ← best grammar derivable in one step from current-grammar
      (Algorithm 1)
4:    grammar-length ← encoding length of grammar
5:       if grammar-length < current-length then
6:       current-grammar ← grammar
7:       current-length ← grammar-length
8:    else
9:       return current-grammar
10:   end if
11: end loop
```

Both of these grammars accept more sentences than the original input, generalizing the language from the original subset.

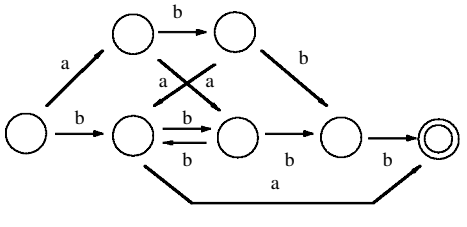
This procedure can be used to model language evolution. We allow an agent who learned a language in the manner described above to teach a new generation in the same way, and continue this process for several generations with each generation presenting its own language to the next. That is, the first generation learns the language which we create, just as described above. After this first-generation learner has de-



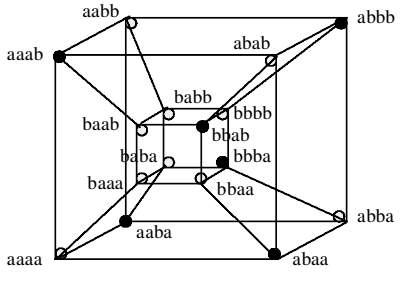
(a)



(b)



(c)



(d)

Figure 3. Transition diagrams of the compressed grammar for Language 1 (a) and Language 2 (c). Hypercube representations of the sentences included in the compressed grammar for Language 1 (b) and Language 2 (d). More sentences are accepted by the compressed grammar than by the original grammar.

veloped a grammar, it has a set of sentences of string length 6 that its grammar can accept or speak. From this first-generation grammar, a set of these sentences is chosen randomly and used as the input for the learner of the next generation. This agent then uses the same model for learning, and so on for as many generations as specified. Languages are thus acquired and change through cultural evolution with vertical transmission [2].

In summary, the steps for an agent to learn a language are:

1. An input set of generated sentences is presented to the agent.
2. The agent constructs an automaton with an associated grammar which accepts all of the sentences from the initial input and nothing else.
3. The agent then compresses the original grammar by attempting to combine states and transitions in such a way that the outcome is still a deterministic automaton and then combines them if and only if the MDL of the new automaton is smaller than that of the starting one. This enables the best grammar to be found at one step.
4. A hill-climbing search is then performed until the FSA with the smallest MDL is found and the language is learned.
5. A subset of all the sentences of length 6 that the agent can produce are selected at random and used as the input for a new agent. Steps 2 through 5 are repeated for the desired number of generations.

This is the procedure used for all of our experiments.

3 Discussion

In previous work, we investigated how compression affects generalization and how easy or difficult it is for languages to be generalized and proposed conjectures C1, C2 and C3 described above [20]. Here we will give a brief summary of our previous study and address outstanding issues regarding these conjectures. We also extend our results and, based on our observations, propose another conjecture: (C4) more complex languages change more rapidly during evolution.

3.1 Conjecture 1: Compression Aids in Generalization

From a series of observations like "Crow A is black" and "Crow B is black" we compress a look-up table of crows and their colors to the generalization that "All crows are black." The generalization is clearly smaller, more "compressed" than a list of many instances. The precise characterization of the circumstances in which such generalization is appropriate, the problem of induction, is a philosophical problem dating back to Aristotle.

We addressed this question in previous experiments and found our first conjecture, that compression aids in generalization, to be true. As seen in the hypercube diagrams in Figure 2 and Figure 3, when a compressed representation of the input for Language 1 is found, the input is generalized to include the outer edge of the cube. This was found to be true for all of the languages we examined [19].

3.2 Conjecture 2: Compression Occurs More Easily on a "Smooth" as Opposed to a "Rugged" Problem Space

Since there is more regularity to be exploited in smoother languages, compression can occur more easily on "smooth" than "rugged" string spaces. In previous work we

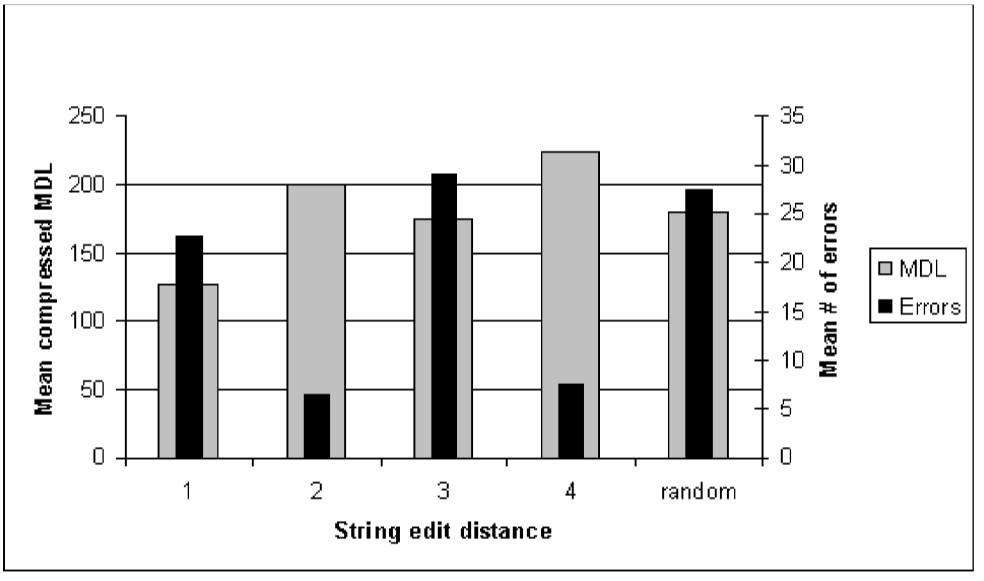


Figure 4. Mean compressed MDL measure versus mean number of errors after compression for string edit distances 1, 2, 3, 4, and random.

looked at the grammars generated by languages with string edit distances 1, 2, 3, 4 as well as random [20]. We found that the smoothness of the language used as input affects the ability of the learner to generalize as well as the types of sentences the learner can produce once the grammar is formed. For example, after compression Language 1 becomes the set (aaaa, aaab, aaba, abaa, aabb, abab, abba, abbb) and Language 2 becomes (aaab, aaba, abaa, bbab, bbba, abbb). As seen in Figure 3a, the representation of the grammar, or its finite state automata, becomes very compressed. Language 1 easily generalizes to the outer cube of the hypercube in this representation. However, for Language 2 there is still little structure in the final language, as can be seen in Figure 3c. This FSA is much less compressed because there is not enough regularity available to be used to develop a more compressed representation.

In our experiments we found that compression generally did occur more easily on smoother problem spaces, but the transition from smooth to rugged languages was not so straightforward as we expected, since string-edit-distance should be a correlate to compressibility. Figure 4 shows that the languages we studied with even string-edit-distances, d, had higher MDL and lower error than expected, where *error* is defined as the number of sentences that are in the final language which were not in the original input set. Error in this context is not considered "bad," but is simply used as a term for the defined measure. We have conducted further experiments and found that this inconsistency seems to be due to trade-offs between error and compression. While this trade-off is incorporated into the definition of the MDL algorithm, there is also the effect that if *data-encoding-length* is optimized, there are few errors and the grammar is not very compressed, but if the *grammar-encoding-length* is minimized, the representation is smaller and more errors occur. The trade-offs we see between error and compression in the final language should be due to and thus correlated with the trade-offs made between the data-encoding-length and the grammar-encoding-length in the generation of the grammar. These trade-offs should be equilibrated so that overall MDL is mini-

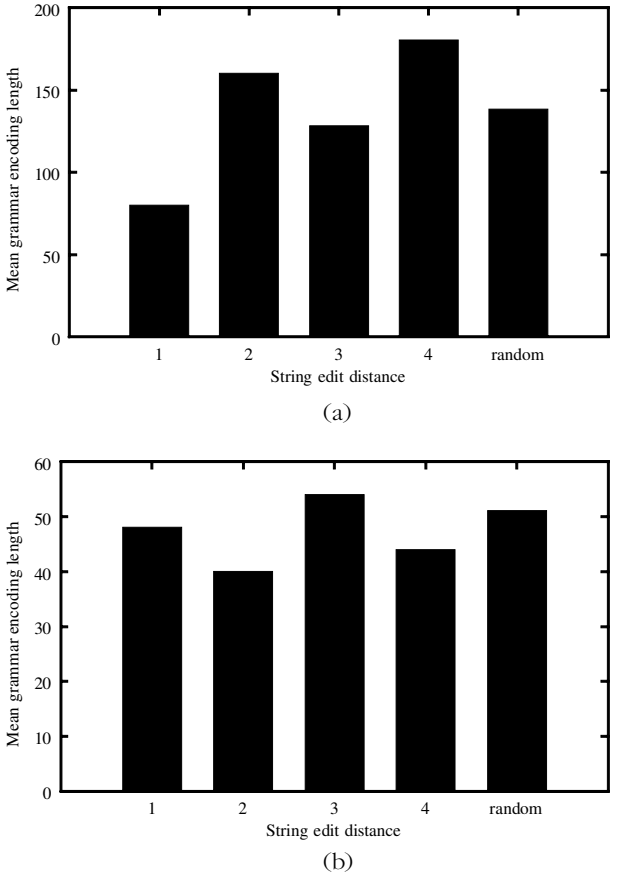


Figure 5. (a) Mean compressed grammar-encoding-length for string edit distances 1, 2, 3, 4, and random. (b) Mean compressed data-encoding-length for string edit distances 1, 2, 3, 4, and random.

mized, but if grammar-encoding-length or data-encoding-length is originally minimized over the other, languages can get stuck at a local optimum.

To investigate this trade-off, we first needed to ensure that the measurements of error and compression are correlated with grammar-encoding-length and data-encoding-length. We performed 20 runs of one generation for string edit distances 1, 2, 3, 4, and random as described above. Compression was measured by the MDL, and the grammar-encoding-length (Γ) and data-encoding-length (Δ) were measured separately as well.

We found that this trade-off between error and compression in the final grammar is highly correlated with the data-encoding-lengths and grammar-encoding-lengths of the grammars. The data-encoding-length is positively correlated with error ($r^2 = 0.82$), and the grammar-encoding-length is positively correlated with compression ($r^2 = 0.98$) as expected, and as seen in Figure 5 there is a trade-off between the two that matches the trade-offs seen in Figure 4.

With the high data-encoding-length and low grammar-encoding-length, it seems likely that languages with even string-edit-distances are optimizing data-encoding-length, but at the expense of grammar-encoding-length. Since the MDL should balance Δ and Γ, it is possible that the search is getting stuck at a local optimum before the two factors are equilibrated. We found that fewer steps were taken to find the final grammar for

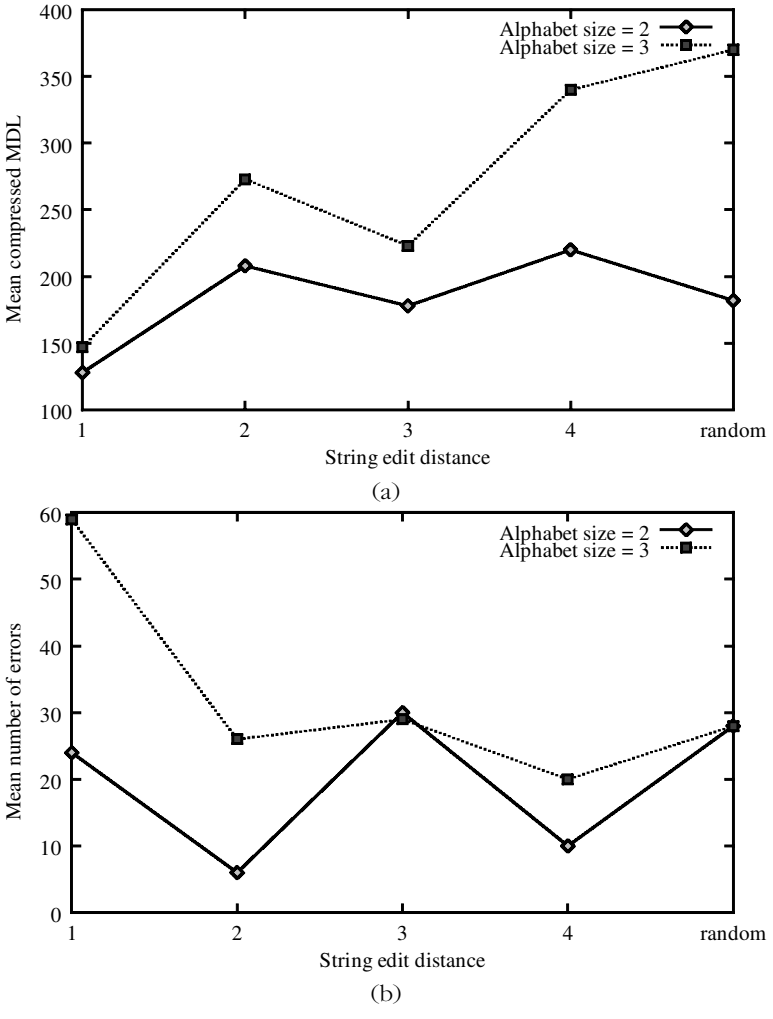


Figure 6. Patterns for the (a) MDL measure and (b) number of errors for alphabet sizes 2 and 3. Patterns for alphabet size 3 are smoother.

even d. The mean numbers of steps to halt were 45.2 steps for $d = 1$, 40.3 steps for $d = 2$, 49.8 for $d = 3$, 43.8 for $d = 4$, and 53.0 for random d. This provides some evidence that a local optimum is being found, although we could not determine this conclusively. We attempted to overcome the possibility of a local optimum by using a depth-first search for a set of examples. This was, however, computationally infeasible.

We believe that constraints causing an interplay between sets of even string-edit-distance and MDL are due to duplicate sentences in the input. They occur with more frequency in the sets of even string-edit-distance because there is a 17% probability that the same position will be selected twice, thereby changing the string back to the original in a system with an alphabet size of 2. Using an alphabet size of 3 can correct for this somewhat, because while there is the same probability that a position will be selected twice, there is a lower probability that it will be changed back to the original string. When we performed 10 runs of $d = 1, 2, 3, 4$, and random for the alphabet (a,b,c), shown in Figure 6, we saw that the trade-offs are less extreme and the transition from smooth to rugged languages is more gradual with an alphabet size

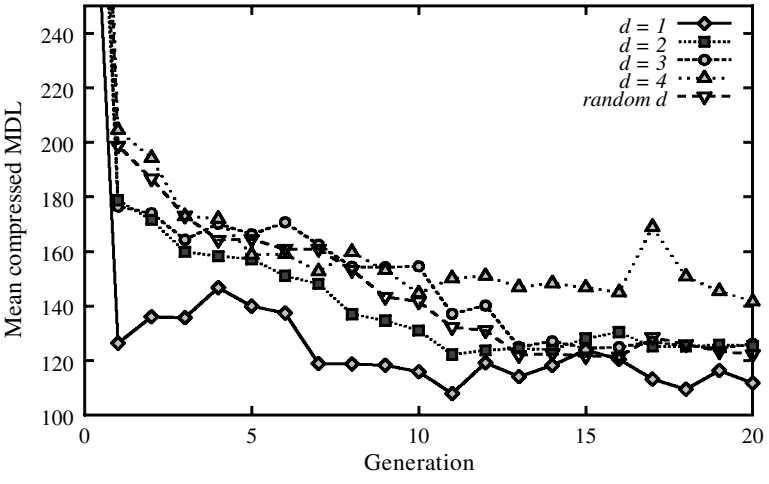


Figure 7. Mean compressed Minimum Description Length (*MDL*) for several string-edit-distances, *d*, at each generation.

of three. The duplicates are providing another source of regularity in the data that is not being accounted for by the string-edit-distance, which leads to these unexpected results. This shows that the irregularities seen in our previous work probably resulted from constraints in the model, and that Conjecture 2 still holds.

3.3 Conjecture 3: Constraints from Compression Make it Likely that Natural Languages Come to Have Smooth String Spaces

One benefit from formalizing the language acquisition system is that we can study the effects of varying initial conditions, something that would be difficult to do with human language learners [15]. In previous studies, we found that regardless of input size or alphabet size, the language being learned is the most important factor for generating the final language and grammar [19]. We expect the grammar to become simpler or more compressed, but this simplicity is not always reflected in the language that the learner finally speaks, or the sentences accepted by the learner's grammar. In previous work we saw indications that constraints due to compression have more effect on the smoothness of the grammars than that of the final languages, but we still thought Conjecture 3 to be true. Here we investigate the question further, looking at the data for more generations and measuring the MDL, number of accepted sentences, and smoothness of the final grammars and languages.

Evolution of these languages was studied over a period of 20 generations. We began by generating a target language as the input set. For each replicate a target language was generated from an input set of 10 examples. The set of examples had $d = 1$, 2, 3, 4, or random; there were 10 replicates of each.

As expected, we found that MDL decreases over time, with the greatest decrease occurring in the first generation when the learner compresses the language from the prefix-tree FSA. This can be seen in Figure 7.

Some disparity between the grammar compression and the smoothness of the language was evident from the mean number of accepted sentences per generation for each value of d. As language evolved, the number of accepted sentences changed. While the dynamical behavior appeared to differ, all the languages did eventually converge to have more or less the same number of accepted sentences, as seen in Figure 8. So while the sizes of grammars remained ordered, the sizes of languages converged.

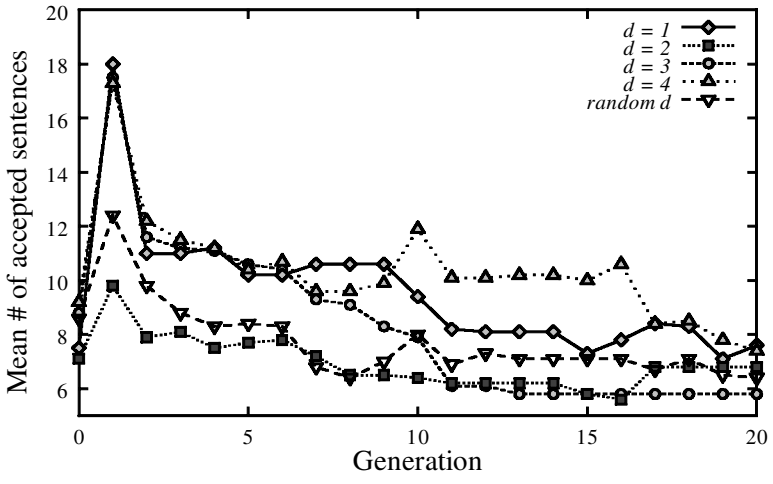


Figure 8. Mean number of sentences accepted by the grammar at each generation, for several string-edit-distances *d*.

One can measure smoothness of languages, rather than of grammars, by measuring the average string-edit-distance (d) of the final languages. We found that they did not decrease significantly from their original string-edit-distance. Measuring a subset of each string edit distance, the average d of the final languages for languages of string-edit-distance 1, 2, 3, and 4 originally were respectively 1.2, 1.3, 2.2, and 3.4. Except for $d = 1$, which could only go up, the ending distances were all just a bit less than when they started, and still in the same rank order.

To summarize, while compression acted to minimize the MDL of grammars, this was not necessarily the same as acting on the size or smoothness of the language. These results indicate that selection acted directly on the grammars, and only indirectly on the languages. So the languages did not become smoother, as we had originally expected. In the light of this new evidence Conjecture 3 was not supported, which refutes our claim in previous work [20].

3.4 Conjecture 4: More Complex Languages will Change More Rapidly during Evolution

Evolution of the most rugged of the languages was examined in the prior section; for $d = 4$ in Figures 7 and 8 the dynamics did seem different at the intermediate generations, suggesting that more complex languages might evolve differently than simpler ones.

From Conjecture 2, which states that rugged languages compress less well than smooth ones, it would follow that across generations the rugged languages continue to resemble look-up tables, whereas smoother languages get compressed into more general rules. General rules can be learned quickly, while look-up tables require that each example be separately encoded. That is to say, general rules are used by many examples, and so will be acquired from a presentation of even a subset of the language. Special instances, characteristic of complex languages, might be generalized a bit, but the grammar will be more susceptible to chance omissions in presentation from teacher to learner and thus drift more. The result would be that more complex languages will change more rapidly.

To investigate this question we looked at the trajectories of the languages as well as those of the grammars. We measured the mean number of accepted sentences per generation. After the first few generations, from about generation 5, the number of

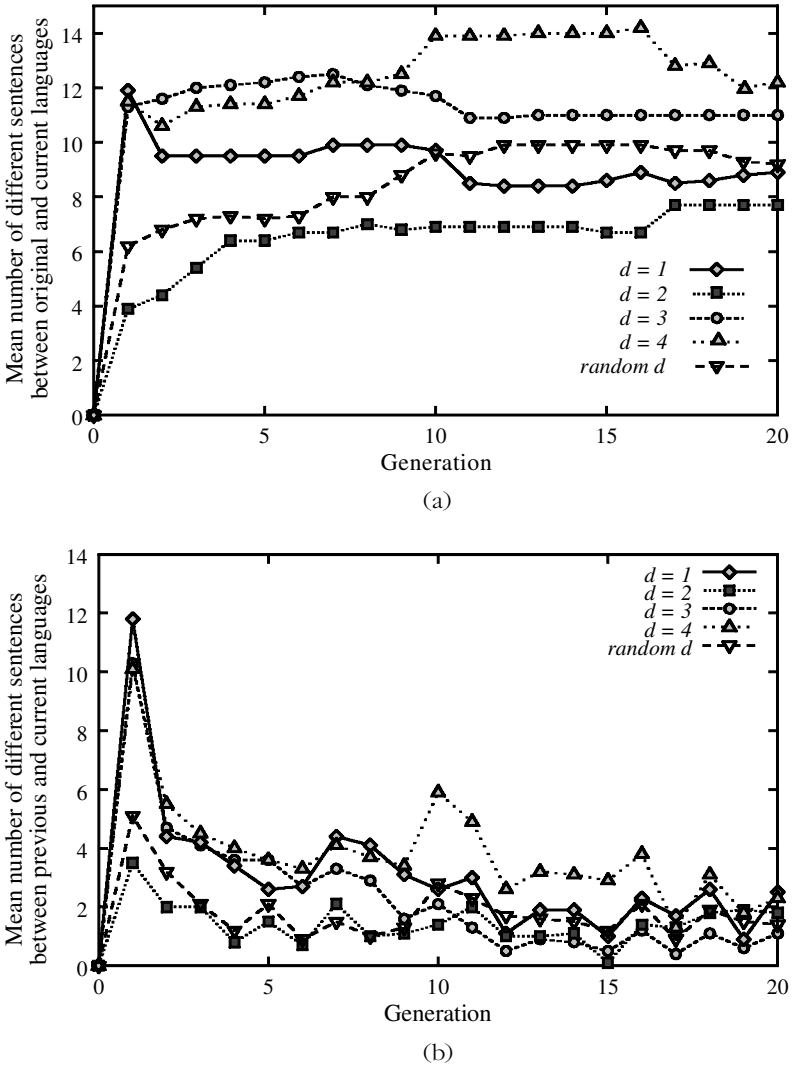


Figure 9. (a) Mean number of different sentences between those given in the original language at generation 0 and those in the current language. (b) Mean number of sentences in the current language that are different from those in the previous language, or errors. Both graphs show values for each generation at several string-edit-distances, *d*.

accepted sentences was more or less constant in the smooth languages, but fluctuated in the more rugged languages. There were typically several changes in the language occurring from one generation to the next. This can be observed in two ways: Figure 9a shows the mean difference between the original input and the current language for each generation; and Figure 9b shows the number of errors between the current language and that of the prior generation. It is evident that the more rugged languages, $d = 3$ and 4, consistently show more differences from the original language, and the mean difference from generation to generation tended to be greater for $d = 4$ than for the others, especially from generations 10 to 18.

These results demonstrate that in simple, regular languages there is less change over time. The best grammar to describe that language has been found, so similar examples are presented to it each generation, and the same language is found in every generation.

In more complex language systems, however, there is some fluctuation in the languages that are produced by the agents in each generation. There is not enough structure to be able to find a completely accurate model, so at each generation, the agent's perception of the language is slightly different. The model of the language is dependent upon the input received, and there are errors transmitted from one generation to the next. Due to this mode of learning and transmission, languages of different complexity follow different trajectories over time, with more complex languages being able to change the most quickly. While this study shows general trends, this subject clearly needs further investigation.

4 Conclusions

We have shown that changes in language can occur as a result of transmission error alone. Different types of languages evolve differently; in our study simpler languages changed considerably less through time. More complex or rugged languages, which presumably more closely resemble human languages, do converge towards grammars with an optimal MDL, but these grammars still allow for significant changes in the languages over time.

Some outstanding issues remain, however, with regard to human languages, such as the role of semantics in language evolution and the effectiveness of this model when using input from human languages. Would use of more natural languages or a more realistic process of language acquisition affect the results? Would a grammar representation different from FSAs help us avoid the limits of local optima? How would using a population of learners affect the resulting languages learned? We are interested to see if these results regarding compression and adaptation can be applied to other situations.

The language trajectories could also be important in collective learning of a language by agents or robots. Different types of environments might provide different information that would influence the language the robots learn. Agents learning in different environments might have difficulty communicating once their language has evolved for several generations and has followed a distinct communication trajectory. This would have important implications not only for language learning, but for any type of collective information sharing.

It seems likely that these results can be applied to other adaptive complex systems that make use of lossy compression. Trade-offs between error and compression are likely to be critical for such systems; which compression algorithms are employed and how they are used will affect the ability of such systems both to adapt to new environments and to evolve.

5 Acknowledgments

We thank Murray Gell-Mann and Stuart Kauffman for prompting our study of these issues. Ed Stabler, Dan Albro, and others in the UCLA Cognitive Science LIS Study Group were helpful at all stages in the study. We thank the reviewers for helpful comments. This work was supported by National Science Foundation grant 9720410.

References

1. Ballard, D. (1997). *An introduction to natural computation*. Cambridge, MA: MIT Press.
2. Cavalli-Sforza, L. L. & Feldman, M. W. (1981). *Cultural transmission and evolution: A quantitative approach*. Princeton, NJ: Princeton University Press.
3. Chomsky, N. (1995). *The minimalist program*. Cambridge, MA: MIT Press.

4. de Marcken, C. G. (1996). Unsupervised language acquisition. PhD Thesis. Department of Electrical Engineering and Computer Science. MIT. Cambridge, MA.

5. Gell-Mann, M. (1999, January 9). *The Santa Fe Institute.* Talk at Santa Fe Institute.

6. Gell-Mann, M. (1999, January 9). *The regular and the random.* Santa Fe Institute Sixth Annual Stanislaw Ulam Lectures.

7. Gold, E. M. (1978). Complexity of automaton identification from given data. *Information and Control, 37,* 302–20.

8. Grünwald, P. (1996). A minimum description length approach to grammar inference. In G. Scheler, S. Wermter, & E. Riloff (Eds.) *Symbolic, Connectionist and Statistical Approaches to Learning for Natural Language Processing,* LNCS #1040 (pp. 203–16). Berlin: Springer-Verlag.

9. Hashimoto, T. (1997). Usage-based structuralization of relationships between words. In P. Husbands & I. Harvey (Eds.), *Fourth European Conference on Artificial Life* (pp. 483–492). Cambridge, MA: MIT Press.

10. Hopcroft, J. E. & Ullman, J. D. (1979). *Introduction to automata theory, languages and computation.* Reading, MA: Addison Wesley.

11. Kearns, M. J. & Vazirani, U. V. (1994). *An introduction to computational learning theory.* Cambridge, MA: MIT Press.

12. Kirby, S. (in press). Syntax without natural selection: How compositionality emerges from vocabulary in a population of learners. In J. R. Hurford, M. Studdert-Kennedy, & C. Knight (Eds.), *Approaches to the Evolution of Language.* Cambridge, U.K.: Cambridge University Press.

13. Kirby, S. & Hurford, J. (1997). Learning, culture and evolution in the origin of linguistic constraints. In P. Husbands & I. Harvey (Eds), *Fourth European Conference on Artificial Life* (pp. 493–502). Cambridge, MA: MIT Press.

14. Li, M. & Vitányi, P. (2000). Minimum description length induction, bayesianism and kolomogorov complexity. In *IEEE Transactions in Information Theory,* IT-46:2 (pp. 446–64). Cambridge, MA: MIT Press.

15. Niyogi, P. & Berwick, R. C. (1995, July). The logical problem of language change. Technical Report A. I. Memo No. 1516, MIT Artificial Intelligence Laboratory.

16. Pinker, S. (1994). *The language instinct.* London, England: Penguin Books.

17. Rissanen, J. & Ristad, E. (1994). Language acquisition in the {MDL} framework. In E. Ristad (Ed.), *Language Computations.* Philadelphia, PA: American Mathematical Society.

18. Sayood, K. (1996). *Introduction to data compression.* San Francisco, CA: Morgan Kauffman.

19. Teal, T. (1999). The effects of compression on language acquisition and compression. Master's Thesis. Department of Organismic Biology Ecology and Evolution. University of California, Los Angeles.

20. Teal, T., Albro, D., Stabler, E., & Taylor, C. E. (1999). Compression and adaptation. In *Fifth European Conference on Artificial Life* (pp. 709–19). Heidelberg, Germany. Berlin: Springer-Verlag.

21. Vapnik, V. N. (1998). *Statistical learning theory.* New York: Wiley.