Preface

Every now and again I receive a lengthy manuscript from a kind of theoretician known to psychiatrists as the "triangle people" - kooks who have independently discovered that everything in the universe comes in threes (solid, liquid, gas; protons, neutrons, electrons; the Father, the Son, the Holy Ghost; Moe, Larry, Curly; and so on). At the risk of sounding like a triangle person, let me explain why I think that the topic of this volume — storage and computation in the language faculty — though having just two sides rather than three, is the key to understanding every interesting issue in the study of language.

I will begin with the fundamental scientific problem in linguistics: explaining the vast expressive power of language. What is the trick behind our ability to fill each others' heads with so many different ideas? I submit there is not one trick but two, and they have been emphasized by different thinkers throughout the history of linguistics.

There is the arbitrary sound-meaning pairing underlying the word, emphasized by Locke, de Saussure, and Austin; it allows us to communicate tens of thousands of ideas by exploiting the capacious faculty of human memory — that is, storage. And there is the infinite use of finite media, emphasized by the Cartesian linguists, Humboldt, and Chomsky, which allows us to combine words into an unlimited number of sentences in which the meaning of the sentence can be inferred from the meanings of the words and the way they are arranged — that is, computation.

I am fond of the storage–computation distinction for a more parochial reason, too: it has made sense of my own area of research, the psychology of irregular inflection (as in *take–took* and *foot–feet*). Linguists and language learners have long wondered why irregularity should exist at all. Why should languages have a feature whose only purpose is to torment foreign language learners, complicate linguistic theories, and set traps for children to produce errors like *taked* and *foots*? The answer, I argue, is that the human language faculty consists of a component for storage and a component for computation, and they can often do the same kind of work in the language system as a whole. *Feet* is stored; *hands* may

be computed by concatenating the stem *hand* with the suffix *-s* (though sometimes the product of a computation may be stored as well).

The idea that the language faculty consists of these two systems means that they can compete for the franchise to express a linguistic distinction. Many examples of historical change can be understood as a series of battles in that struggle. Anything computed by one person may be simply stored by another person, and at various points in history, one listener may have memorized a product of computation (such as the umlauted *o* in the Old English plural of *foot*) as an unanalyzed chunk, which was then free to mutate and drift as it was passed from memory to memory down the centuries. Conversely, the storage/computation fight can explain how a word can pass in the other direction, from irregular to regular. Storage depends on human memory, a fallible system that is apt to forget low-frequency words. When a generation of Middle English speakers could not remember that *chid* is the past of *chide*, they defaulted to the computation system, converting the word to *chided* for their generation and all subsequent ones.

We tend to call a phenomenon "irregular" when it belongs to inflectional morphology, but the same thing can happen in every other component of language. Idioms, collocations, and verb-particle combinations can be seen as chunks of syntax that are stored in memory rather than computed by compositional syntax and semantics. Productive "Level 2" derivation and not-so-productive "Level 1" derivation in theories of level-ordered morphology and lexical phonology may represent the same distinction: Level 2 is computed, Level 1 stored. We see the distinction in phonology in the difference between *obscene–obscenity* and *obese–obesity*, and everyone knows about it in orthography, where, as the poem says, "A *moth* is not a *moth* in *mother*, Nor *both* in *bother*, *broth* in *brother*."

The major theoretical approaches to language can be distinguished by how they partition storage and computation. Classic Bloomfieldian structural linguistics, and classic Chomskyan and Hallean generative linguistics, try to squeeze every drop of redundancy out of linguistic constructions and retain only the dessicated residue in the storage component. The full forms are reconstituted by interacting rules, constraints, and other symbolic combinatorial operations. The theory most diametrically opposed to this tradition, neural networks or connectionism, does the opposite: it pretends that the computational component does not exist, and tries to account for linguistic productivity by beefing up the storage component so that it no longer merely stores and retrieves items verbatim but can generalize by analogy to similar items.

In my view the most promising approach to language, deriving from Mark Aronoff, Joan Bresnan, Edwin Williams, Rochelle Lieber, Ray Jackendoff, and others, makes full use of both storage and computation. Some linguistic regularities really are computed on-line by symbolic operations, others are patterns of redundancies stored in the lexicon. The storage mechanism need not be like a computer memory with a list of addresses; it may have some of the properties of connectionist pattern associators, superimposing similar items and thereby strengthening their overlapping features. And the computational component need not be a set of rules; it may be a general mechanism of unifying structures that contain variables. This strikes me as an optimal partition, allowing one to avoid overly baroque formulations of the computational system typical of some grammatical theories, without pretending that our magnificent capacity to combine words does not exist. This synthetic theory is worked out in complementary ways in my own *Words and Rules* (1999), and in Jackendoff's *The Architecture of the Language Faculty* (1997).

Child language acquisition, too, can be understood in terms of an interplay between storage and computation. In many areas of development, children appear to memorize a large set of forms (words, chunks of phrases, verbs and their argument structures) and only later analyze them into pieces that may be productively recombined. The textbook example is the so-called "U-shaped" development of the past tense in English, in which children may produce unmarked verbs and correct irregular past tense forms like *broke* for months before producing their first error like *breaked.* In my books *Language Learnability and Language Development* (1994) and *Learnability and Cognition: The Acquisition of Argument Structure* (1989), I tried to account for many phenomena of language development in these terms, and in our 1992 monograph *Overregularization in Language Acquisition*, Gary Marcus, Michael Ullman, and I were able to confirm statistically that it is a real phenomenon.

Finally, the differences between storage and computation may shed light on the evolution of language — both its adaptive engineering design and its phylogenetic history. Imagine a language with no computation, only storage. The use and acquisition of language would be a snap, because we could just store and retrieve chunks of sound. But we would be restricted to blurting out single words, or at best clichés for familiar situations, and would have no way of sharing new combinations of ideas about novel thoughts or events. Now imagine a language with a minimum of storage, such as the "perfect languages" of 17th-century John Wilkins and other enlightenment thinkers. Every vowel or consonant in a word would pick out a semantic feature of its referent, exhaustively specifying every conceivable object in the universe. The problem here

is that we would have to engage in full-blown computation in figuring out what every lexical item meant, in effect playing a game of Twenty Questions as we worked our way through the vowels and consonants of each word. I submit that language is like many other biological systems that combine two mechanisms with different costs and benefits, like the fast-twitch and slow-twitch fibers making up our muscles. We have lexical storage for common and simple entities, which we can quickly look up whole, and we have computational operations for novel combinations of entities, which we can assemble and parse on-line.

And even the neural substrate and evolutionary origin of these mechanisms may not be utterly obscure. On the basis of a series of experiments that presented several kinds of neurological patients with regular and irregular forms to inflect, Ullman and I have suggested that the storage component of language is concentrated in the declarative memory system of the brain, comprising the temporal and parietal lobes and structures that connect with it, primarily the hippocampus. The computational component may be concentrated in the procedural memory system, comprising the basal ganglia and the frontal lobe areas that form circuits with it. Both systems are phylogenetically very old, suggesting that the storage/computation distinction may be deeply rooted indeed, having appeared literally hundreds of millions of years ago.

These are some of the reasons I think the distinction between storage and computation runs through the study of language and is indispensable to understanding its many manifestations. The volume you are holding contains the ideas and data of many ingenious scholars, including other triangle people like me, who feel that the distinction is worth exploring.

STEVEN PINKER

http://www.springer.com/978-1-4020-0526-8

Storage and Computation in the Language Faculty
Nooteboom, S.G.; Weerman, F.; Wijnen, F.N.K. (Eds.)
2002, 356 p. 3 illus., Hardcover
ISBN: 978-1-4020-0526-8