C:/TOOLS/WMS/CUP-NEW/5122568/WORKINGFOLDER/KOCK/9781107030077C29.3D 708 [708–733] 26.6.2014 5:37PM

29

Linguistic anthropology in the age of language automata

Paul Kockelman

29.1 Introduction

Computers, as both engineered and imagined, are essentially text-generated and text-generating devices. And computation (in the machine-specific sense) may be understood as the enclosure of interpretation – an attempt to render a highly messy and stereotypically human process relatively formal, quantifiable, and context-independent. To make these arguments, I introduce some of the key concepts and claims of computer science (language, recognition, automaton, transition function, Universal Turing Machine, and so forth), and show their fundamental importance to the concerns of linguistic anthropology.

I argue that no small part of linguistic anthropology constitutes an oppositional culture in relation to computer science: many of its core values and commitments are essentially contrastive (rather than contentful). Such contrasts have hamstrung the ability of linguistic anthropologists to engage productively with the fruits of computer science, such as pervasively networked, digitally mediated, and ubiquitously present environments that more and more constitute the infrastructure for so-called "natural" communication. Here I will show the ways some of the core claims and methods of linguistic anthropologists can be productively applied to, and extended using, such infrastructure – opening up not only a new set of topics, but also a new set of techniques.

Sections 29.2 and 29.4 describe key concepts of computer science in their own terms, developing the relation between different kinds of languages and different kinds of computers. Sections 29.3 and 29.5 show the ways these concepts relate to core concerns in linguistic anthropology, such as interaction versus abstraction and linguistic relativity versus universal grammar. And the conclusion tacks between the concerns of both disciplines, highlighting key areas of future interest.

29.2 The relation between languages and computers

As in linguistic anthropology, the notion of language is crucial to computer science (however differently it is understood). In particular, a computer (or, automaton more generally) may be abstractly understood as a sieving device that accepts certain strings of characters and rejects others (Kockelman 2011). The set of strings that it accepts is called the language that it recognizes. The rest of this section will develop these ideas at length, as grounded in standard works on this subject (Rabin and Scott 1959; Turing 2004 [1936]; Sipser 2007), describing the core operations that computers must be able to perform if they are to sieve strings in these ways.

An *alphabet* may be understood as a set of characters. Examples include: {0, 1}, {0, 1, 2, 3, . . ., 9, #}, {a, b, c, . . ., z}, {the characters of a standard QWERTY typewriter}, {glyphs from an ancient language}, and so forth. Most generally, an alphabet can be any set of types whose tokens are perfectly and reliably readable and writable by the computer in question.[1] A *string* may be understood as a list of characters from such an alphabet (such that a string is said to be "over" the particular alphabet whose characters it incorporates). Examples of strings, over some of the foregoing alphabets, include: "11110111100," "3#29," "hullabaloo," "What did the quick brown fox jump over?" and so forth. More generally, the strings in question may be understood as containing any amount of quantifiable information, as well as encoding any kind of imaginable meaning. And a *language* may be understood as a set of such strings. Examples include: {the set of all *w*, where *w* is a string over the English alphabet that ends in *-ing*}, {the set of all *s*, where *s* is a grammatically acceptable sentence in German}, {the set of all pairs *x#y*, where $y = x^3 + 2$}, and so forth. In this way, with its innards still suitably black-boxed, a computer may be imagined as taking in strings as its input (whatever their length or alphabet), and turning out one of two strings, and thereby instigating one of two actions, as its output ("accept" or "reject," 1 or 0, "True" or "False," "permit" or "prohibit," etc.). See Figure 29.1a.

To be able to perform the task of accepting or rejecting particular strings, and thus, ultimately, of recognizing a particular language, a generalized automaton (or *Turing Machine*, as it will be referred to below) must be able to engage in the following kinds of operations: (1) read and write tokens of particular character types; (2) move along some kind of medium (where such tokens are read and written); and (3) both ascertain and update its own internal state. See Figure 29.1b. At the heart of such a device is a *transition function* that maps a domain of values onto a range of values. And thus, depending on the current state of the device, and the character it is currently reading, the transition function specifies what character to write (if any), what direction to move in (along the medium), and what state to change into. See Figure 29.1c. In essence, that is all such a

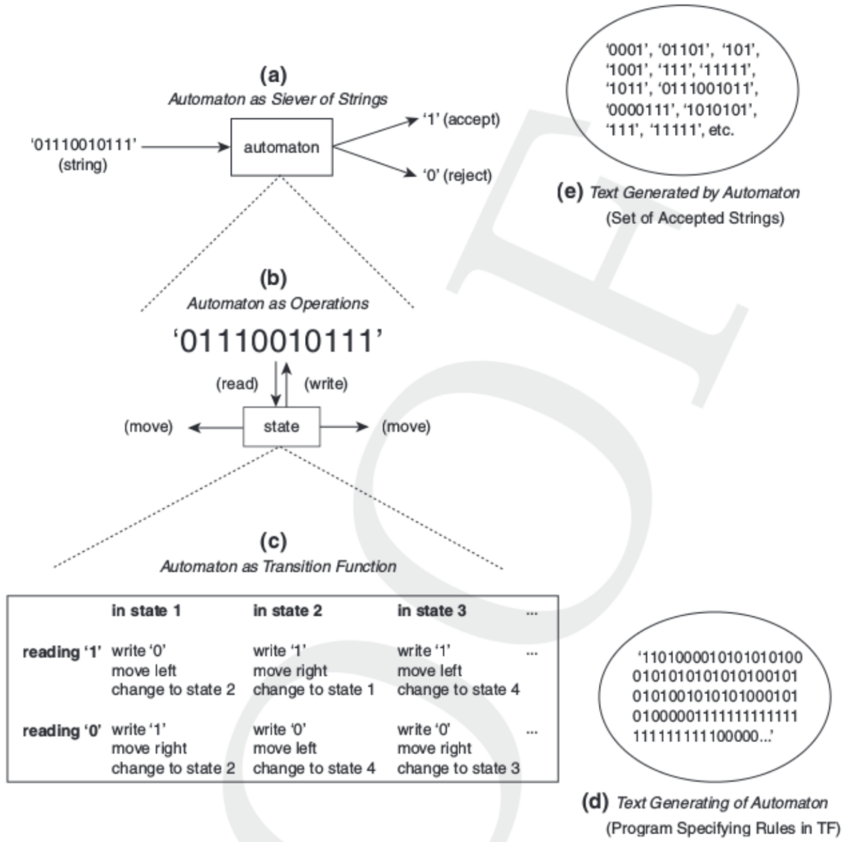


Figure 29.1 Automata as text-generated and text-generating devices

device ever does: having been given some string as its initial input (as written into the medium), and having been put in a particular state at a particular position along the string (usually the beginning), it repeats this mapping procedure (a potentially mind-numbing number of times) until it ends up in one of two particular states as its final output (accept or reject).

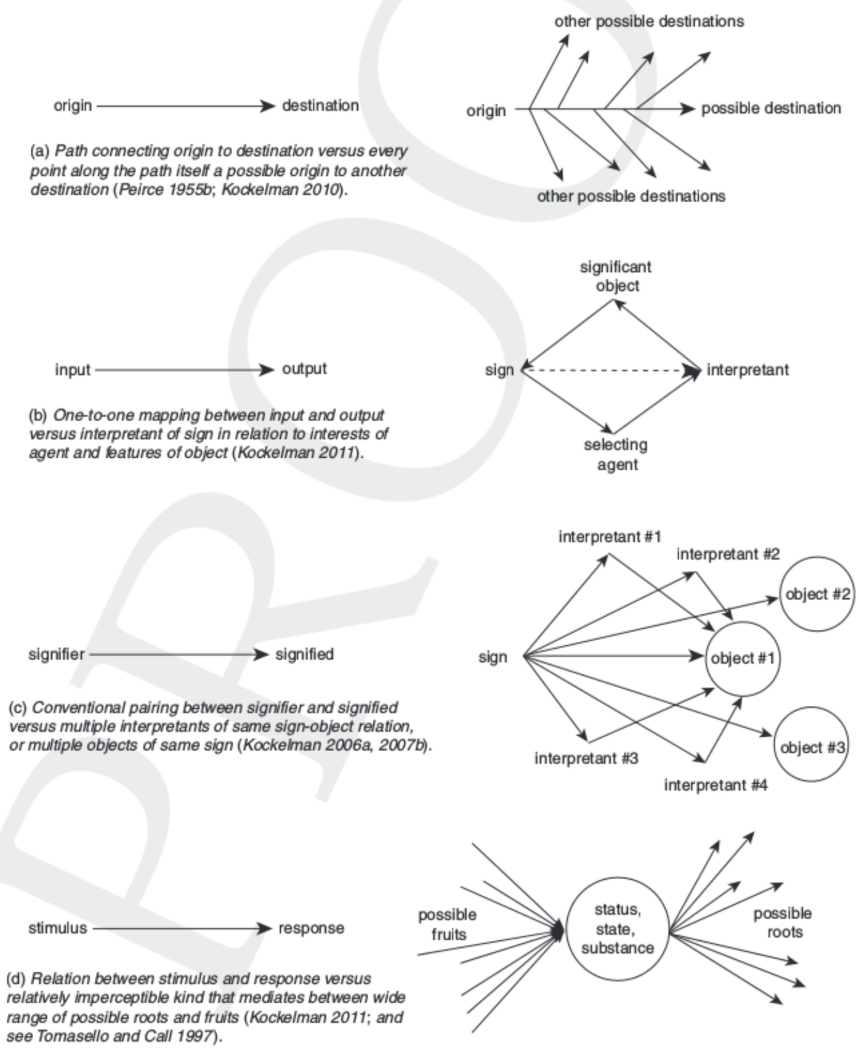
Phrased another way, a transition function consists of a finite set of rules which map current values (character read, current state) onto subsequent values (character written, movement undertaken, next state). To *program* such a device is essentially to specify its transition function (usually by giving the device another, more "primordial" string which encodes the rules in question). See Figure 29.1d. And such a transition function itself determines whether or not the device will accept particular strings; and thus, ultimately, whether or not the device will recognize a particular language. See Figure 29.1e. Crucially, while each rule may be trivial to specify, the list of rules (or program) can be quite complicated to formulate, and the overall behavior of such a device (e.g., the particular patterning of the language it recognizes) impossible to predict without actually observing it (if a pattern is even inferable at all). In all of these ways, then, *as both* *engineered and imagined, computers are essentially text-generated and text-generating devices*. Framed recursively, computers presuppose strings and create languages, where a language is a set of strings, anyone of which might be presupposed by another computer in its creation of a language.

29.3 String-sieving: Abstraction and interaction

Having characterized some of the ways computer scientists understand languages and computers, we may now begin to sketch a linguistic anthropology of strings, and the devices that sieve them. In part, this is done to show how the tools of linguistic anthropology can be applied to the concepts of computer science (as well as to the objects of computer engineering). And, in part, this is done to show and soften the fundamental tension between the culture of linguistic anthropologists and the concepts of computer science – a tension that is otherwise almost laughably overdetermined in its binary simplicity. As will be seen, the title of this chapter is meant to be ironic: for, in fact, linguistic anthropology came of age in the time of language automata, but somehow managed to studiously avoid what it is arguably destined to embrace.

As described in this section, automata are exemplary instances of relatively black-boxed, rule-bound, and deterministic intermediaries. In particular, both the localized mapping of values (e.g., character read and current state to character written, move made, and next state), and the global input–output relation (e.g., string to accept/reject) are radically deterministic,[2] such that there seems to be a maximally rigid (as opposed to flexible) mapping between inputs and outputs.[3] This characteristic puts them at odds with anthropology's strongly humanistic imaginary, which sees human agency as maximally mediated. For example, people are understood as norm-abiding, culture-inhabiting, context-sensitive, interactively emergent, and reflexively conscious agents. And so it is not surprising that linguistic anthropologists have been extremely wary of disciplines (such as cognitive science and formal linguistics) that have invoked computational metaphors in their attempts to understand key features of human behavior.

While Latour (2005: 39) is often cited in relation to this distinction between *intermediaries* (or whatever "transports meaning or force without transformation: defining its inputs is enough to define its outputs") and *mediators* (whose "input is never a good predictor of their output; their specificity has to be taken into account every time"), Michel Serres (2007 [1982]) is arguably the more originary figure. And Serres was himself developing certain ideas of Claude Shannon (1949, 1963[1949]) in regards to noise and enemies (Kockelman 2010), a thinker who – ironically enough – was the key theorist of information (as opposed to meaning, a distinction that will be returned to in the conclusion). Of particular relevance to linguistic anthropologists, an even earlier definition was offered by Peirce, in his use of a path metaphor to distinguish between what he called *secondness* and *thirdness*: "a straight road, considered merely as a connection between two places is second, but so far as it implies passing through intermediate places [themselves possibly connected by other paths to further places] it is third" (1955b: 80; Kockelman 2010: 413). That said, the distinction is really much more general and variable, as may be seen in Figure 29.2, which diagrams a variety of analogous and overlooked relations. And that said, the distinction is itself highly ontology-specific and frame-dependent (points that will be taken up below).



(a) *Path connecting origin to destination versus every point along the path itself a possible origin to another destination* (*Peirce 1955b*; *Kockelman 2010*).

(b) *One-to-one mapping between input and output versus interpretant of sign in relation to interests of agent and features of object* (*Kockelman 2011*).

(c) *Conventional pairing between signifier and signified versus multiple interpretants of same sign-object relation, or multiple objects of same sign* (*Kockelman 2006a, 2007b*).

(d) *Relation between stimulus and response versus relatively imperceptible kind that mediates between wide range of possible roots and fruits* (*Kockelman 2011; and see Tomasello and Call 1997*).

Figure 29.2 Some ways of framing secondness and thirdness

Indeed, an enormous amount of energy has gone into trying to refute any claim that people are in any way automaton-like (and thus in no way like seconds or intermediaries). Forty years of anthropology has spent its time trying to show that each and every social form (such as a practice, structure, sign, identity, or belief) is "emergent," "contested," "fluid," "embodied," "non-deterministic," "dialogic," "constructed," "distributed," "context-bound," "reflexive," "mediated," and so forth. In some sense, computers (or rather a widespread folk-theory of computers), have been the favorite bogeyman of anthropology (and social scientists more generally): where there are rules, give us practices; where there are symbols, give us indices; where there is truth-conditioning, give us poetry and performance; where there is mind, give us body; where there is abstraction give us interaction; where there are ideal languages give us forms of life.

The history of these divisions is institutional as much as intellectual, and deserves a chapter of its own; but some of the key moves are easy enough to sketch. Descartes versus Heidegger in continental philosophy (see, for example, Dreyfus 1991 and Haugeland 1998a). Early Wittgenstein (1961 [1921]) versus late Wittgenstein (1958[1953]) in analytic philosophy (and see, for example, Kripke 1982 and Kockelman 1999). The structuralism of Levi-Strauss (1969[1949]) versus the practice theory of Bourdieu (1977 [1972]) in anthropology.[4] Saussure (1983[1916]) versus Peirce (1955a, b, c) in semiotics (and see, for example, Kockelman 2006b). And formalism (Chomsky 1965) versus functionalism (Jakobson 1990a) in linguistics (and see, for example, Van Valin and LaPolla 1997).[5] Insofar as linguistic anthropology sits downstream, as it were, of all of these currents it has adopted most of their claims, such that its understanding of interpretation (and meaning) is essentially contrastive with stereotypes about automata (and information). As such, it is worth examining one of its key foils with a renewed empathy built on fifty years or so of enmity.

More important than being dismissive of, or trying to circumvent, the rule-like nature of such devices is to understand the ways such rules get coupled to norms: or, rather, the way allegedly human-specific modes of signifying and interpreting meaning (as grounded in mediation, or thirdness) are mediated by and mediating of allegedly machine-specific modes of sending and sieving information (as grounded in intermediaries, or secondness). In particular, much of the current built environment, *qua* communicative infrastructure, consists of precisely such devices. And so natural languages and culture-specific communicative practices more generally, are constantly being mediated by (encoded with and channeled through) such devices. Moreover, the distinction between mediators and intermediaries is itself grounded in mediators: and so there is a culture and history to the ways some community specifies where machine-like things end and human-like things begin; as well as a culture and history of evaluating what is essential to each, and what is good or bad when breached. Phrased another way, there is a lot of firstness in where we draw the line between secondness and thirdness (Kockelman 2013a: 141). More generally, rather than ontologize the world in such binary terms, it is much better to (1) foreground agency as a radically multidimensional, distributed, and graduated process; (2) foreground a variety of practices which not only have the effect of enclosing agency as "agents" but also dichotomizing such agents in terms of distinctions like "intermediary" and "mediator"; and (3) genealogize the recent presumption of this dichotomy among scholars (Kockelman 2007a, 2007b, 2010). And thus a key task for the linguistic anthropology of automatized languages is to trace the politics and pragmatics of such *intermediation*.[6] See Table 29.1.

Indeed, the genealogy of this division may be traced back at least as far as the industrial revolution. For example, the art critic and historian Ruskin endlessly railed against the machine, championing handicraft in the face of widespread industrialization, arguing that the latter insofar as it is mass-produced rather than individually and singularly crafted loses "the traces or symptoms of a living being at work" (quoted in Gombrich 1979: 40; and note the relation to Walter Benjamin's more famous notion of "aura"). Interestingly, Rushkin often aimed his critiques at the "decorative" arts more generally, in their often mechanically produced and repetitive or

Table 29.1 Intermediaries, mediation, and intermediation

Secondness	Thirdness	Via Peirce
Intermediary	*Mediary*	Via Serres and ANT
Ideal Language	*Form of Life*	Via Wittgenstein
Universal Grammar	*Linguistic Relativity*	Via Chomsky and Sapir
Machines Talking	*Humans Talking*	Via Turing
Structure	*Agency*	Via Cultural Anthropology
Computer Science	*Linguistic Anthropology*	Via Disciplinary Boundaries
Computing Machines	*Interpreting Humans*	Via Multiple Encodings
Real Imaginaries	*Symbolic Imaginaries*	Via Ontological Mappings
Artificial Languages	*Natural Languages*	Via Possible Objects
Statistics (Math)	*Semiotics (Meaning)*	Via Possible Methods
Enclosing	*Disclosing*	Via Underlying Imperative
Sieving and Serendipity	*Significance and Selection*	Via Semiotic Framing
Redundancy	*Poetry (qua Metricality)*	Via Shannon and Jakobson

Intermediation as Obviation

(1) Secondness and thirdness are poles of a continuum, not positions in an opposition;
(2) Boundary between secondness and thirdness is itself grounded in thirdness (and secondness);
(3) Each is affecting of, and affected by, the other at various degrees of remove;
(4) Whether some process is understood as one or the other is dependent on degree of resolution and frame of relevance;
(5) Process of making (or seconding and thirding, as it were), and making seem (like secondness and thirdness), as important as the products made (seconds and thirds, per se).

highly patterned nature. This is particularly salient insofar as there is a close linkage between the patterns (*qua* languages) generated by automata and the patterns generated by decoration-producing mechanisms such as looms (which themselves were, in the age of Jacquard, programmable with punch-cards, and thus also generated by text-like patterns). Finally, these issues go back to the origins of rhetoric (and see Bate, this volume, Chapter 21): the admonishment to make one's speech simple (and thus less flowery or "decorated"), and thus less poetic and more referential. Note, then, linguistic anthropology's valorization of mediation over intermediaries is itself grounded in the oldest (or at least most famous and widespread) of language ideologies. Ironically, this is, in a certain sense, the converse of its own explicitly articulated sensibilities as to the importance of poetic regimentation and the multi-functionality of language, as discussed by Kockelman, this volume, Chapter 24.

As another example of intermediation, note that because a sieving device is, in some sense, coupled to its input (the string it is initially given) by way of its transition function (which makes reference to the possible characters on a string), the device and the string are "intimate" (Kockelman 2013a: 109–10; and see Gibson 1986[1979], Simon 1996, and Haugeland 1998b for intimacy as a metaphor). In certain respects, a stringless device is like an organism without its environment; just as a deviceless string is like an environment without an organism. Neither makes much sense except in relation to the other. In this way, such sieving devices are (inverse) iconic-indices of the strings of symbols they sort: each incorporates, creates, and complements the other. Moreover, and closely related to the first point, such devices are shifters in an expanded sense: their input-to-output mapping (string to accept/reject) only counts as a sign–interpretant relation when contextualized – such that the features of an object, and the interests of an agent, can be specified (Kockelman 2011).[7] For both these reasons, such devices are not at all "context free" or "abstract" or "meaningless" or "symbolic," but rather radically grounded, intimate, contextual, iconic-indexical, motivated, embedded, and so forth.[8] In this way, the classic techniques of linguistic anthropology (Silverstein 1976; Jakobson 1990a, 1990b; Lucy 1993; Silverstein and Urban 1996) are perfectly poised to illuminate both the string–sieve relation and the string/sieve–situation relation (not to mention the cultural and disciplinary ontologies that would otherwise figure such relations in simplistic ways). For example, modern approaches to interaction (see, for example, Enfield 2009 and Sidnell 2010) can be brought to bear on what otherwise seems to be canonical cases of abstraction.

As is well known, the actual material instantiation of such devices is "immaterial" in regards to the mathematical specification of the language in question: there is nothing inherently electronic (as opposed to mechanical, quantum mechanical, lively, etc.) about computers – even if their practical instantiation, and widespread adoption, had to await a particular technology. What matters, ultimately, is that the device be able to undertake the kinds of tasks listed above (read and write, move left or right, ascertain and update). In particular, people can do each task; and, ironically, the first "computers" were indeed people (often women), who carried out lengthy (and tedious) calculations according to a finite set of relatively simple rules (Kittler 1989[1986]; Inoue 2011). In this way, several other interrelated tensions are immediately apparent. First, claims to a timeless disembodied abstraction in relation to a history of particular material instantiations. Second, the relation between people and machines (as ontologized by any particular community, or imagined in terms of a particular technology) in relation to the relation between different kinds of people (e.g., genders, classes, ethnicities, and so forth). Third, the kinds of computational tasks asked of sieving devices and their relation to politicized notions like labor, work, and action (not to mention often highly idealized and romanticized notions like creativity, contemplation, and communication). And finally, as will be taken up in subsequent sections, the relation between such artificial languages (both generating and generated) and so-called "natural" languages – in all their forms and functions (news, philosophy, poetry, grammar, conversation, and so forth). While the scholarship relevant to such concerns is enormous, key works include: Benjamin (1968a); MacKay (1969); Kittler (1989 [1986], 1996[1993]); McLuhan (1996[1964]); Hayles (1999); Mirowski (2002); Turing (2004[1950]); Suchman (2007); Benkler (2007); and many of the essays collected in Wardrip-Fruin and Montfort (2003).

29.4 Kinds of languages, kinds of computers

Before taking up other important kinds of tensions, it is worth returning to some key claims of computer science. Particular automata (or particular programs running on a universal Turing Machine, essentially a "computer" in the stereotypic sense) may be characterized in terms of the sets of strings that they accept (and thus the languages that they recognize). And different classes of automata may be characterized in terms of the kinds of languages they can recognize – kinds of languages that can be compared in terms of their relative complexity, and thus classes of automata that can be compared in terms of their relative power.[9] See Figure 29.3.

In particular, three key classes of sieving devices are Deterministic Finite Automata (DFA), Context-Free Grammars (CFG), and Turing Machines (TM). DFAs are the simplest of the three devices. In contrast to TMs (whose innerworkings were detailed in section 29.2), such devices only move in one direction (from the beginning of the string to the end); no characters are ever written; and the medium only ever contains the string in question. Endowed with such capabilities, such devices can recognize the class of

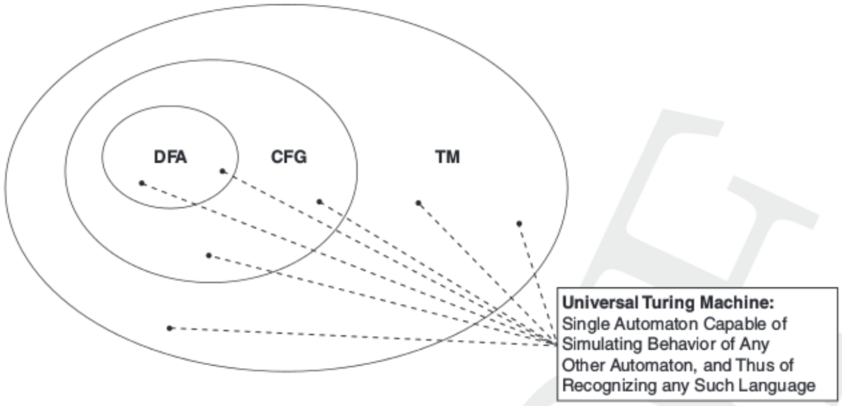


Figure 29.3 Relative scope of languages recognized by different kinds of automata

regular languages, which are essentially all languages recursively definable in terms of three simple functions. Loosely speaking, the *union* of strings from any two regular languages is itself a regular language; any possible *concatenation* of strings from any two regular languages is itself a regular language; and any possible *iteration* (or "self-concatenation") of strings from any regular language is itself a regular language. More carefully, if L_1 and L_2 are two languages recognizable by a DFA (and thus regular languages), the following languages are also recognizable by a DFA (and thus regular languages): L_1 U L_2 = {the set of all strings w, where w is a string in L_1 or L_2}; L_1 o L_2 = {the set of all strings w_1w_2, where w_1 is a string in L_1 and w_2 is a string in L_2}; and L_1^*= {the set of all strings $w_1w_2w_3 \ldots w_k$, where k >=0 and w_i is a string in L_1}. For example, if L_1={a,b} and L_2 ={c}, L_1 U L_2={a,b,c}, L_1 o L_2={ac,bc}, and L_1^*=[e,a,b,ab,ba,aa,bb,abb,aab,…}, where e is the empty string (that is, the string with no characters). And so on, recursively, for languages like (L_1 o L_2) U L_1, (L_1 U L_2)*, and L_1^* U (L_1 o L_2). In this way, with three relatively simple functions, and some primitive notions like empty strings and singleton languages (or languages with only one string, itself consisting of a single character), one can build up languages with great complexity. More complicated (and useful) regular expressions include: …, …, and … Practical applications that implement DFAs include password checks, word searches, swearword censors, and simple spam-filters; as well as devices like automatic doors, traffic lights, and elevator traffic controls.

CFGs not only recognize all regular languages, they also recognize languages like {the set of all strings *w*#*w* | where *w* is itself a string of any length over some alphabet}, which require an infinite amount of memory that is only accessible in a relatively restricted fashion (essentially a kind of "last-written, first-read" form of storage). In particular, in contrast to DFAs, the domain of the transition function of a CFG turns on not just the current state of the device, and the character currently being read from the string, but also (potentially) the character currently being read from the top of the "stack" (its restricted memory). And the output involves not only updating the state of the device, and moving to the next character on the string, but also (potentially) writing some other character onto the top of the stack. When understood as generating languages (as opposed to recognizing them), CFGs should be immediately familiar to linguistic anthropologists in terms of the rewrite rules (or "tree structures") of formal models of language. For example, a particular set of rules (such as S => NP-VP; NP => DET-ADJ-N; VP => V-NP; DET => *a*, *the*; ADJ => short, tall; N => boy, girl; V => pinched, ticked) may be understood to generate a particular language (which would include the following strings: *the short girl pinched the tall boy*, *the tall girl tickled the short girl*, and so forth). Such languages may exhibit another kind of recursion, as when the range or output of a rule ultimately makes reference to the same variable that constitutes its domain or input (for example, PP => Prep NP, NP => N PP). Practical applications that implement CFGs include most parsers (involved in compiling or interpreting the computer programs that are run on universal Turing Machines, and thus the texts that specify their transition functions), as well as many applications that either simulate or process natural languages. Indeed, as will be discussed at length in section 29.4, many of Chomsky's early intuitions about language (themselves a key foil for functional linguists and linguistic anthropologists for the last fifty years) were grounded in the structure and logic of CFGs (and related kinds of automata).[10]

Finally, TMs not only recognize all languages recognized by CFGs (and thus, all languages recognized by DFAs), but also languages like $\{w \mid$ where w is an integer root of the polynomial $x^3+3x^2+8x=0\}$.[11] Indeed, the Church–Turing Thesis postulates that such devices are definitionally equivalent to algorithms: they can recognize any language that can be specified in terms of a finite deterministic procedure (loosely speaking, an iteratively applied, easily followed, and simply stated set of rules for undertaking a longer and more complicated calculation).[12] Not only do they have an infinite amount of memory but, in contrast to CFGs, their memory is unrestricted in its accessibility. Finally, as already mentioned, a *Universal Turing Machine* (essentially a modern-day computer with infinite memory) is an automaton that can be programmed (by giving it a string that encodes the set of instructions that specify its transition function) to model the behavior of any particular Turing Machine. In some sense, it is the one automaton that can take the place of any other automaton. Or, to invoke a comparison that will need some unpacking, and should echo Marx's (1967[1867]) notion of universal money (as well as Benjamin's 1968b notion of empty homogeneous time), it is akin to a universal language: the one language whose expressions can be used to translate the meaning of any expression from any other language (Kockelman 2006b: 100).

29.5 Universal Turing Machines: Universal grammar and linguistic relativity

Crucial to the theoretical imaginary surrounding Turing Machines is the fact that various adjustments to a TM's basic capacities do not affect its functioning (or the set of languages it can recognize) in any important way. For example, there are TMs that can stay in place at any transition (in addition to moving left and right); there are TMs that use more than one tape (where characters may be read or written); there are TMs that move in two dimensions rather than one (and thus accept two-dimensional "swatches" of text rather than one-dimensional strings of text); there are TMs that enumerate languages rather than recognize them; and so on, and so forth. And not withstanding such differences, all of these devices can be shown to be *equivalent* to the others. Put another way, with certain caveats, all the different kinds of computers out there, and any of the different programming languages used on any particular computer, are equivalent (or "commensurable"). For example, any program written in LISP can be written in Java or C; and any program run on one machine can be run on another. Because TMs are so incredibly "robust" in this way, computer scientists consider the class of languages that they recognize (i.e., the set of algorithmically solvable problems) to be relatively "natural" (Sipser 2007: 128–33). This is another way of framing the claim, introduced above, that such devices are "universal."

One issue of fundamental importance to linguistic anthropologists is closely related to this fact: the tension between universal grammar and linguistic relativity. To see how, let me both elaborate and extend a claim made by Sapir (1949[1924]; and see Lucy 1992a, 1992b; Sidnell and Enfield 2012): while all languages are arguably "formally complete," in that they are able to represent the same set of experiences (*qua* reference), each has its own "secret," which involves not only a way of orienting to a referent (*qua* "sense") but also an associated feeling of orientation (*qua* "sensability").[13] While one may or may not be particularly committed to this claim, it is worthy of careful consideration because of the foundational tensions it brings to light.

To use an example from geometry, note that both Cartesian coordinates (x,y) and polar coordinates (r,O) may be used to represent the same set of points (all points in a two-dimensional plane). See Figure 29.4.

Any expression in either system may thereby be translated into the other system (through equations like x = r cos O and y = r sin O). But that said, the equations of particular entities may be more or less aesthetically elegant when expressed in one system rather than the other (e.g., lines are relatively simple entities in Cartesian coordinates, whereas circles are relatively simple entities in polar coordinates). As physicists know (Arfken and Weber 1995), certain problems may be more or less easy to solve in one system rather than the other (insofar as the symmetry of the problem

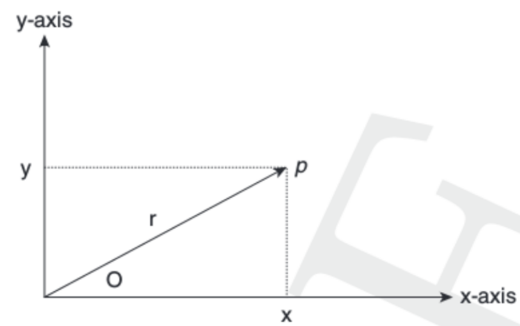


Figure 29.4 Comparison of Cartesian and Polar coordinate systems

matches the symmetry of the system). And finally, as a function of such symmetry and solvability, the intuitions of problem solvers may be more or less enabled or constrained. In this way, while the two systems are equivalent at the level of reference, they are non-equivalent at the level of sense and sensibility. More generally, and perhaps most colorfully, while different systems may allow us to "touch" the same worlds, the worlds so touched may be nonetheless "felt" in distinctly different ways.

Understood in such terms, three points may now be made. First, while Sapir was, of course, talking about natural languages, such claims may also be understood to hold for artificial languages. In particular, while any program written in any programming language may be written in any other programming language (as per our discussion above), it is likely that different programming languages (not to mention interfaces, architectures, and so forth) have different "secrets" – different symmetries built into them (that make certain problems easier or harder to solve), and different sensibilities disciplined by them (as embodied in those who habitually program in them). While this claim is a low-hanging fruit, it is worth making insofar as it shows another site where classic techniques of linguistic anthropology can be applied to classic objects of computer science – the texts that generate computation (*qua* programs) as much as the texts generated by computation (*qua* languages, in the technical sense discussed above).

Second, given the relation between the languages generated by context-free grammars and natural language, given the "naturalness" of the class of languages recognizable by Turing Machines, and given the strong referential equivalence of natural languages (not to mention the close initial disciplinary linkage between computer science, cognitive science, and formal linguistics), it is not difficult to empathize with the desire of early generative linguists to discover the "universal grammar" (a kind of ur-language) underlying all natural languages. Nor is it difficult to empathize with their intuition that it should be equally discoverable through any particular language if analyzed closely enough. In other words, if all coordinate systems (*qua* systems of signs) are equivalent (in that any one may relate to the others as interpretant to sign), why not just use one of them to understand the world (of objects) so referred to?

And third, just as it is easy to foreground equivalence of reference (as a disciplinary focus), it is also easy to foreground non-equivalence of sense and sensibility. In particular, having just characterized some key ideas underlying the analytic imaginary (or disciplinary culture) of generative linguists, it is easy to see where some of linguistic anthropology's own contrastive commitments come from. For example, if universal grammar may be understood to foreground equivalence of reference, linguistic relativity may be understood to foreground non-equivalence of sense and sensibility. And while early linguistic anthropologists like Whorf (1956 [1939], 1956[1937]) and Sapir (1985[1927], 1985[1945]) could comfortably shift between both perspectives (indeed, not withstanding contemporary readings of them, Sapir and Whorf highlighted linguistic invariance as much as relativity, as should be clear in the work just cited), the latter has been given center stage. For example, Hill and Mannheim (1992 ; and see Lee 1996), have gone so far as to argue that linguistic relativity should be understood as an "axiom" of linguistic anthropology rather than an hypothesis. In this way, two sets of scholars have passionately rallied around flags of complementary colors, themselves placed in contiguous and often overlapping terrains that were originally staked out by the same surveyors.

29.6 Conclusion

One way to reframe some of the foregoing claims is as follows: *computation is the enclosure of interpretation*. In part, this means that computation is a species of interpretation that is relatively regimented as to its use-value, truth-value, and exchange-value (Kockelman 2010). In part, this means that it is a species of interpretation that has been relatively mediated by technology, science, and economy (ibid.). And in part, this means that the values in question (be they signs, objects, or interpretants) become relatively portable: not so much independent of context, as dependent on contexts which have been engineered so as to be ubiquitous, and hence seemingly context-free (Kockelman 2007b, 2013a). In effect, the mediation is so great that it appears to be unmediated – and thus a mere intermediary. For the average denizen of such an environ, thirdness often goes about as secondness (and vice versa).

This claim may be easily generalized. While the focus has been on the relation between computation and interpretation (and thus the input–output, or sign–interpretant relation per se), we could also focus on the sign–object relation, and argue that information is the enclosure of meaning (Kockelman 2013a). And we could focus on the signer–interpreter relation, and argue that infrastructure is the enclosure of interaction

(Kockelman 2011). In this way, we could focus on *a set of concomitant processes whereby semiosis gets not only automated, but also formatted and networked*.

Hand in hand with the real-time practices and *longue durée* processes though which this occurs is a kind of reflective understanding of its occurrence – itself usually radically refracted (or so the story goes). In each kind of enclosure, a great degree of agency (power, flexibility, meta-reflexivity, progress, etc.) seems to be gained – and so there is celebration and speculation. And a great degree of authenticity (context-dependence, historical uniqueness, cultural specificity, etc.) seems to be lost – and so there is nostalgia and mourning.[14] Recall the admonishments of Ruskin.

Needless to say, such refracted reflectivity should be all-too-familiar to anthropologists, as they are themselves grounded in a particular imaginary that is found again and again in critical theory (Kockelman 2007c): from Aristotle and Marx (e.g., substance and form, quality and quantity), through Maine and Toennies (e.g., status and contract, community and society), to Levi-Strauss and Bourdieu (e.g., raw and cooked, practice and structure). Indeed, anthropology has always been, in part, the disciplinization of precisely such refracted reflections: in its more sophisticated variants it proposes them; in its less sophisticated variants it presupposes them.

With these core claims and key caveats in mind, we may now sketch some future topics and techniques for a linguistic anthropology of automatized (formatted and networked) languages, one which focuses on *intermediation* (or embeddedness) rather than constantly trying to counter intermediaries by reference to mediation, and thus one which seeks to empathize with machines (and their makers), as much as with humans (and their makings). Recall Figure 29.2 and Table 29.1.

For example, an obvious topic of interest to linguistic anthropologists is the *Turing Test*, and attempts to make computers speak (and interact more generally) in ways that are more or less indistinguishable from human speech and interaction (Turing 2004[1950]; and see French 2000 and Saygin *et al.* 2001). One relatively direct route to this topic is through the lens of ontology (our assumptions as to the kinds that constitute a particular world, and how these assumptions license particular inferences) and transformativity (how such assumptions change through our indexical encounters with such kinds), as these relate to the machine-human relation (Kockelman 2013a, 2013b). And one relatively indirect route to this topic would be to study the intersection of several text-building processes. First, the texts (*qua* computer programs) used to make computers speak. Second, the texts (*qua* human–machine dialogues) generated through interactions between these programs and people. And third, the texts (*qua* meta-language) by humans (and perhaps machines) about these dialogues and programs (describing them, theorizing about them, categorizing them, evaluating them, commodifying them, vilifying them, and so forth). More generally, these kinds of inter-textual processes are at work in a multitude of natural language-processing projects: voice recognition, spam filtering, dialog censoring, machine translation, etc. And so there are ample opportunities for linguistic anthropologists who want to study the tensions among such texts (and their underlying processer of textualization).[15]

One important relation that shows up again and again in computer science, among other places, might be called *ontological isomorphisms*, *cross-domain diagrams*, or even, "real imaginaries." By this is meant that a set of relations found in one domain is found in another seemingly disparate domain, such that insights from each domain may be used to generate insights about the other (often licensing large-scale theoretical and technological innovation). For example, just as Boole (1958[1854]) worked out the relation between binary numbers and truth conditions (and thus math and logic), Shannon (1936) worked out the relation between truth conditions and electrical circuits (and thus logic and engineering).[16] And actual material instantiations of Turing Machines, such as the standard desktop computer we now have, itself not much different from the architecture initially proposed by von Neumann (Ceruzzi 2000; Petzold 2000), exploit precisely these relations. While closely related to metaphor (and thus able to be studied, in part, using the techniques of trope analysis), these mappings are not metaphors (in the conventional sense of, say, Lakoff and Johnson 1980) for two crucial reasons: first, there is no distinction between concrete and abstract domains (each domain is on equal par, as it were); and second, it is not, strictly speaking, a linguistic or conceptual phenomenon (the parallels exist in the domain of reference, and may be pointed to with any kind of sign). But that said, while certain mappings may be well founded and referentially motivated, other mappings may be more whimsical or social – licensed by particular imaginaries (concepts) and symbolics (signs), as much as by particular reals (reference).[17] In particular, a key kind of relation between relations to study is the relation between these real imaginaries (i.e., cross-domain diagrams) and symbolic imaginaries (i.e., metaphors in their more conventional sense, and textual and technological aesthetics more generally). These are key sites where the promises and pitfalls of automatized languages, as well as the interfaces they present to the world, and the infrastructures they depend on in the world, get refracted in reflection. And again, linguistic anthropologists, who have long been interested in the relation between culture and diagrammatic iconicity, and the relation between poetics and pragmatics more generally, are perfectly poised to analytically tap into these issues.

Intersecting the phenomenon of cross-domain diagrams is *multiple encoding* (or perhaps, "multiple interpreters"): the way a given computer language (or technology more generally) is subject to the demands and abilities of the machines that compute with it as much as of the humans that interpret with it.[18] For example, the texts that tell computers what to do (i.e., "programs"), such that they may generate further texts (i.e., "languages") can be more or less easily "read" by humans and, concomitantly, less or more easily "read" by machines. And so, there are programming languages like Assembly which stay very close to the structure of the machines that run them; and there are programming languages like Python and C, which are generally more amenable to the intuitions of people who write in them, and which have to be interpreted or compiled before they can be run by a computer.

Framed more generally, to serve a single function (or have a particular object) a given sign must be amenable to the ontologies (capacities, codes, habits, cultures, etc.) of several interpreting agents at once. Crucially, this means that each kind of interpreting agent might understand it in different ways, and so there can be issues of translation, the division of labor, relative perspicacity of encoding, and so forth, with all the usual tensions. And, needless to say, this is related to a set of more timeless topics: how a tool is crafted to the demands of the world, the body, and the mind at once; or how a practice is regimented by cultural norms and natural causes simultaneously; or how a dream may be interpretable in regards to both its manifest and latent content at the same time; and so forth.

To put this in a more critical perspective, this leads to a set of tensions that were first foregrounded by Marx: the degree to which an instrument is designed to fit the requirements of a user (e.g., a "tool"); or a user is disciplined to fit the requirements of an instrument (e.g., a "machine"). And this relation may itself be reframed in semiotic terms: the degree to which a sign (and, concomitantly, a signer) takes on features of its object (iconicity, via Saussure and Peirce); or an object takes on features of its sign (projection, via Sapir and Whorf). Again, the relevance and reach of these issues for linguistic anthropologists should be clear, especially given the discussion of reference, sense, and sensibility in section 29.3.

Another relatively obvious, but nonetheless key move would be to simply apply the usual linguistic anthropological toolkit to the study of *programming languages*. For example, descriptive grammars of such languages are waiting to be written (i.e., grammars of actual usage patterns, as opposed to language specifications, instruction manuals, and so forth). There is discourse analysis of the real-time writing of texts, *qua* programming languages – and not just coders in dialogues with other coders and users, but also coders in dialogue with CPUs and worlds so to speak (via the shell, debugging applications, and so forth). There are ethnographies of communication (Hymes 1962; Bauman and Sherzer 1975) to be written about the sub-cultures surrounding different programming languages (text editors, etc.) in their language use (e.g., users of Perl versus users of Lisp, users of VI versus users of Emacs). Cross-cutting all of these, there is an immense "historical archive" of intertextual relations that has yet to be touched regarding the multiple corpora of code that have been written: not just issues related to intellectual property, authorship, riffing, disclosure of source code, voicing, borrowing, and theft; but also simply the use of libraries, genres, well-known sub-routines, algorithm implementation, and of course the data generated by the programs themselves (e.g., all the information out there on each and every one of us). There is the relation between language ideologies (Silverstein 1979) and programming languages: for example, the politics, pragmatics, and poetics of why one language is picked up or rallied around rather than another (which turns closely on the kinds of issues discussed above). There is the relation between Goffman's interaction order (1981, 1983) and human–machine interaction, as well as human–human interaction when mediated by machines (not to mention machine–machine interaction when mediated by humans). As developed in section 29.3, there are also the classic Sapir–Whorf issues regarding the linguistic relativity of programming languages (and, more generally, of interfaces, architectures, platforms, and so forth).[19] And finally, there is a lengthy and ever-evolving set of concepts in computer science and engineering that have direct parallels and strong resonances with key concepts in linguistic anthropology: pointers, name-spaces, unmarked values, files, protocols, libraries, platforms, and so forth. While all of these are relatively obvious topics to take up (in the sense that one just applies already developed analytic concepts to new objects), they are not low-hanging fruits (insofar as they will require a great deal of expert knowledge, and thus specialization, in the domains at issue). Moreover, because of the cross-fertilization of concepts, they are likely to yield analytic insights (for both disciplines) as much as empirical knowledge.

And finally, there is a set of topics that might be best considered as staple goods: on the one hand, they are the most readily available to anthropologists (in terms of the tools they already have); and, on the other hand, they are of the most central interest to anthropologists (given the topics they are used to). Indeed, the literature on such topics is already enormous, and growing larger every day. (See Gershon and Manning, this volume, Chapter 22, for a careful and creative review.) For example, there is the culture and identity of hackers, and any other sub-community closely linked to computer science (e.g., their beliefs and values in relation to language, technology, politics, and so forth). There is the mediation of natural language via new channels and interfaces (e.g., what happens to English grammar, discourse patterns, and genre styles in the context of text-messaging, cell-phones, and so forth). More generally, there is the mediation of semiotic practices and social relations via new channels and interfaces (e.g., identity and interaction in the context of social networking sites, dating applications, the crowd-sourcing of artistic creation, etc.). There is the mediation of language and, in certain cases, talk-in-*intermediated*-interaction, in virtual worlds, video games, and so forth.

In short, there is an infinite list of topics (most of which fall under the rubric of "digital anthropology") that might be characterized as *old concerns in light of (rapidly changing) new media*. It is easy to predict that such relations will be found to change quickly enough over time, and be different enough in different places, such that anthropologists should be able to generate (!)

article after article, and dissertation after dissertation, on their local mediation somewhere: hacker culture in the former Soviet Union; diasporic identity in the context of Facebook; love in the time of relational databases; Japanese slang in NSM; property and personhood (not to mention anonymity, enmity, and amicability) in virtual worlds; gender (race, sexuality, ethnicity, etc.) in video games; space (time, value, etc.) in chat rooms; new political imaginaries and counter-cultures in the wake of new channels; and so forth. In particular, not only will the ethnographic details differ as a function of time and place, but the mediating media themselves will be constantly changing (emails, cell-phones, NSM, Skype, . . .; Myspace, Friendster, Facebook, Diaspora, . . .; mainframe, desktop, laptop, smartphone, . . .; and so on, and so forth), such that if linguistic anthropologists aren't strategically lazy, they will find themselves perpetually busy – passionately tracking potentially superficial topics whose ultimate empirical and theoretical allure is, I'm tempted to say, analytically *akin to* commodity fetishism. (But that is another essay.)

While there are many more potentially interesting topics (sieves in general, self-automatizing languages, data explosions, spam-filters, noise, the relation between capital and computation – for example, electronic trading –, the nature and culture of automated creativity, self-replicating automata, the relation between evolution, computation, and interpretation, Unicode, etc.), the foregoing list of potential frames (and possible pitfalls) should be enough to illuminate some of the space of possibilities, and some of the analytics of approach, for a linguistic anthropology of language automata.

And so there is only one last methodological point to be made: *the tools and techniques used by linguistic anthropologists to study such domains will, in part, have to partake of precisely the objects in such domains*. This means that the usual tools (Shoebox, Elan, etc.) won't be enough. Linguistic anthropologists will not only be writing texts (programs) to study such texts, they will also probably be writing them to write up their studies. More generally, semiotics will have to go hand in hand with statistics, or meaning with mathematics. In particular, the ap-perceiving and ap-intending ears and mouths, eyes and hands, brains and bodies, of linguistic anthropologists do not afford nearly enough power, leverage, speed, or space for the task ahead. To study automatized (networked and formatted) languages, linguistic anthropologists will need to automatize (network and format) languages.

Notes

1. Though, as is well known from information theory, complicated alphabets are not necessary, in that a simple binary alphabet like {0,1} can be used to represent the characters from all other alphabets and, indeed, the strings from all other languages.

2. There is a technical sense of deterministic, as used by computer scientists (Sipser 2007), which is not being invoked here.
3. While linguistic anthropologists have long been nervous about postulating rules to understand behavior (witness the success of practice-based approaches in anthropology), large collections of such rules can exhibit behaviors that appear – and, thus, for all intensive purposes *are* – highly flexible. (And, indeed, one can write rules for a device that enables it to update its own rules – depending, say, on the environment it finds itself in.)
4. Indeed, the claims are themselves recursively applicable. For example, within the domain of critical theory (or continental philosophy), Foucault is to Deleuze what structure is to practice.
5. Ironically and again recursively, in the domain of cognitive psychology, Chomsky is to Skinner as mediators are to intermediaries, and so sits on both sides of this contrast depending on whom he is contrasted with.
6. Linguistic anthropology has always been more formal than its cultural anthropology cousins; and so I'm hoping that they are resolute enough in their commitment to the right-hand side (thirdness), such that they can delve deeply into the left-hand side (secondness), and thus investigate intermediation, which should necessarily obviate the distinctions.
7. Phrased another way, while the input–output relation is deterministic, the "meaning" such devices have and the "function" they serve only make sense in terms of the interests of some agent and the features of some object, which themselves are usually only partially determined, and are always dependent on the placement of such a device in a particular context (a context which includes the device's own input).
8. Insofar as transition functions presume that such devices can read and write inputs which are tokens of a particular type, such devices exhibit the hallmark of digital process (Haugeland 1998b). And, as per the nature of digitality, the types in question, as well as the states and positions, are necessarily discrete: there are no partial types, quasi-states, or half-positions. Needless to say, the discreteness (or "digitality") of the mechanism, like the discreteness of the alphabet, closely aligns it with classic Saussurean understandings of the "symbolic": value (*qua* typehood of any token) only adheres in difference. Loosely speaking, a particular character can be instantiated however we like, so far as it is distinguishable, in both reading and writing, from the other characters with which it contrasts. Crucially, as discussed above, this does not entail that the meaning of such devices is "arbitrary" or conventional (as opposed to "natural" or motivated). Such devices are usually highly iconic and always highly indexical. But again, it's precisely the tension between such grounds (and the ideological claims to one or the other) that make them interesting objects of analysis. Loosely speaking, where we draw the line between the symbolic and the iconic-indexical (or the arbitrary and the

motivated), is itself grounded in convention (or so say the culturalists), which is itself grounded in nature (or so say the realists).

9. Key works on these three kinds of automata, the history of automata-theory more generally, and the nature of programming languages, include: Church (1941); McCulloch *et al.* (1943); Shannon (1949); Chomsky (1956); Kleene (1956); Rabin and Scott (1959); McCarthy (1960); Thompson (1968); Kernighan and Ritchie (1988[1978]); Abelson *et al.* (2001); Piccinini (2004); Turing (2004[1936]); Friedl (2006); Sipser (2007); Jurafsky and Martin (2008); Bird *et al.* (2009).
10. On the one hand, anyone who has ever tried to specify the grammar of a language in any detail will recognize the need for something like a generative capacity (in the unmarked sense): a finite number of words and rules gives rise to an infinite number of actual sentences. And so the idea of a construction (in Bloomfield's original sense) which incorporates parts which are themselves wholes with parts (potentially recursively), and so on, indefinitely, is necessary for describing syntactic processes. On the other hand, these same people will all quickly recognize that this is not sufficient. Formalists may push for further and further refinements (under banners like transformations, government and binding, principles and parameters, minimalism). And functionalists may simply accept a kind of Bloomfieldian minimalism, and then get to work on context-dependent grammars.
11. Left aside are issues related to recognizing versus deciding a language. Famously, as discussed in Sipser (2007), a TM can recognize but not decide a language like {p | p is polynomial with integer roots}.
12. Or, as famously defined by Hilbert, a "process according to which it can be determined in a finite number of operations" (quoted in Sipser 2007).
13. As Sapir put it, "All languages are set to do all the symbolic and expressive work that language is good for, either actually or potentially. The formal technique of this work is the secret of each language" (1949[1924]: 155).
14. For example, celebrations of cyborg futures are just as misplaced as lamentations about authenticities lost.
15. Also key are the framing processes that are involved in linking distinct and potentially distal "texts" and "contexts" (across different points in space-time, so to speak) as much as constituting any particular "text" or "context" (at some particular point in space-time). See, for example, Halliday and Hasan (1976); Lucy (1993); Silverstein and Urban (1996); Enfield (2009); Kockelman (2011; 2012: 202–3).
16. Curiously, Peirce had made similar claims forty years earlier (Chiu *et al.* 2005).
17. A more recent move with comparable importance is probably *graph theory*: a mathematical field that is used to account for the sieving patterns of automata as well as the interconnections among agents, and a field that is encoded by machines as much as imagined by people.

18. And not only expert populations of humans, but also lay populations.
19. Many of the most famous interface designers make reference to Whorf. See, for example, Englebart (1962), Victor (2006). And see Stephenson for an inspired engagement with Whorf-like ideas (1999).

References

Abelson, Harold, Gerald Jay Sussman, and Julie Sussman. 2001. *The Structure and Interpretation of Computer Programs*. Cambridge, MA: MIT Press.

Arfken, George B., and Hans J. Weber. 1995. *Mathematical Methods for Physicists*. 4th ed. San Diego: Academic Press.

Bauman, Richard, and Joel Sherzer. 1975. The Ethnography of Speaking. *Annual Review of Anthropology* 4: 95–119.

Benjamin, Walter. 1968a. The Work of Art in the Age of Mechanical Reproduction. In *Illuminations*, ed. Hannah Arendt, 217–52. New York: Schocken Books.

1968b. Theses on the Philosophy of History. In *Illuminations*, ed. Hannah Arendt, 253–64. New York: Schocken Books.

Benkler, Yochai. 2007. *The Wealth of Networks*. Cambridge, MA: Harvard University Press.

Bird, Steven, Ewan Kleine, and Edward Loper. 2009. *Natural Language Processing with Python*. Sebastopol: O'Reilly.

Boole, George. 1958[1854]. *An Investigation of The Laws of Thought*. New York: Dover.

Bourdieu, Pierre. 1977[1972]. *Outline of a Theory of Practice*. Cambridge: Cambridge University Press.

Ceruzzi, Paul E. 2000. *A History of Modern Computing*. Cambridge, MA: MIT Press.

Chiu, Eugene, Jocelyn Lin, Brok McFerron, Noshirwan Petigara, and Satwiksai Seshasai. 2005. *The Mathematical Theory of Claude Shannon*. Unpublished ms.

Chomsky, Noam. 1956. Three Models for the Description of Language. *IRE Transactions on Information Theory* 2: 113–24.

1965. *Aspects of the Theory of Syntax*. Cambridge, MA: MIT Press.

Church, Alonzo. 1941. *The Calculi of Lambda-Conversion*. Princeton, NJ: Princeton University Press.

Dreyfus, Hubert L. 1991. *Being-in-the-World: A Commentary on Heidegger's Being and Time, Division 1*. Cambridge, MA: MIT Press.

Enfield, N. J. 2009. *The Anatomy of Meaning: Speech, Gesture, and Composite Utterances*. Cambridge: Cambridge University Press.

Engelbart, Douglas C. 1962. *Augmenting Human Intellect*. SRI Summary Report AFOSR-3223.

French, Robert M. 2000. The Turing Test: The First 50 Years. *Trends in Cognitive Science* 4(3): 115–22.

Friedl, J. 2006. *Mastering Regular Expression*. Sebastopol: O'Reilly Media, Inc.

Gibson, James. 1986[1979]. *The Ecological Approach to Visual Perception*. Boston, MA: Houghton Mifflin.

Goffman, Erving. 1981. Footing. In *Forms of Talk*, 124–59. Philadelphia: University of Pennsylvania Press.

1983. The Interaction Order. *American Sociological Review* 48(1): 1–17.

Gombrich, E. H. 1979. *The Sense of Order*. Ithaca, NY: Cornell University Press.

Halliday, M. A. K., and Ruqaiya Hasan. 1976. *Cohesion in English*. London: Longman.

Haugeland, John. 1998a. The Intentionality All-Stars. In *Having Thought: Essays in the Metaphysics of Mind*, 127–70. Cambridge, MA: Harvard University Press.

1998b. Mind Embodied and Embedded. In *Having Thought: Essays in the Metaphysics of Mind*, 207–37. Cambridge, MA: Harvard University Press.

Hayles, N. Katherine. 1999. *How We Became Posthuman*. Chicago: University of Chicago Press.

Hill, Jane H., and Bruce Mannheim. 1992. Language and World View. *Annual Review of Anthropology* 21: 381–406.

Hymes, Dell. 1962. The Ethnography of Speaking. In *Anthropology and Human Behaviour*, ed. T. Gladwin and W. C. Sturtevant, 13–53. Washington, DC: Anthropology Society of Washington.

Inoue, Miyako. 2011. Stenography and Ventriloquism in Late Nineteenth Century Japan. *Language and Communication* 31: 181–90.

Jakobson, Roman. 1990a. The Speech Event and the Functions of Language. In *On Language*, ed. L. R. Waugh and M. Monville, 69–79. Cambridge, MA: Harvard University Press.

1990b. Shifters and Verbal Categories. In *On Language*, ed. L. R. Waugh and M. Monville-Burston, 386–92. Cambridge, MA: Harvard University Press.

Jurafsky, Daniel, and James H. Martin. 2008. *Speech and Language Processing*. 2nd ed. Englewood Cliffs, NJ: Pearson Prentice Hall.

Kernighan, Brian, and Dennis Ritchie. 1988[1978]. *The C Programming Language*. 2nd ed. Englewood Cliffs, NJ: Prentice Hall.

Kittler, Friedrich A. 1989[1986]. *Gramophone, Film, Typewriter*. Palo Alto, CA: Stanford University Press.

1996[1993]. The History of Communication Media. Online at www.ctheory.net/text_file.asp?pick=45.

Kleene, S. C. 1956. Representations of Events in Nerve Nets and Finite Automata. In *Automata Studies*, ed. C. Shannon and J. McCarthy, 3–41. Princeton, NJ: Princeton University Press.

Kockelman, Paul. 1999. Poetic Function and Logical Form, Ideal Languages and Forms of Life. *Chicago Anthropology Exchange* 29: 34–50.

2006a. Residence in the World: Affordances, instruments, actions, roles and identities. *Semiotica* 162(1): 19–71.

2006b. A Semiotic Ontology of The Commodity. *Journal of Linguistic Anthropology* 16(1): 76–102.

2007a. Agency: The Relation between Meaning, Power, and Knowledge. *Current Anthropology* 48(3): 375–401.

2007b. Enclosure and Disclosure. *Public Culture* 19(2): 303–5.

2007c. From Status to Contract Revisited. *Anthropological Theory* 7(2): 151–76.

2010. Enemies, Parasites, and Noise: How to Take Up Residence in a System Without Becoming a Term in It. *Journal of Linguistic Anthropology* 20(2): 406–21.

2011. Biosemiosis, Technocognition, and Sociogenesis: Selection and Significance in a Multiverse of Sieving and Serendipity. *Current Anthropology*, 52(5): 711–39.

2013a. Huckleberry Finn Takes the Turing Test: The transformation of ontologies, the virtuality of kinds. *Language and Communication* 33: 15–154.

2013b. The Anthropology of an Equation: Spam, sieves, agentive algorithms and ontologies of transformation. *Hau: The Journal of Ethnographic Theory* 3(3): 33–61.

Kripke, Saul A. 1982. *Wittgenstein: On Rules and Private Language*. Cambridge, MA: Harvard University Press.

Lakoff, George, and Mark Johnson. 1980. *Metaphors We Live By*. Chicago: University of Chicago Press.

Latour, Bruno. 2005. *Reassembling the Social*. Oxford: Oxford University Press.

Lee, Penny. 1996. *The Whorf Theory Complex: A Critical Reconstruction*. Amsterdam: John Benjamins.

Levi-Strauss, Claude. 1969[1949]. *The Elementary Structures of Kinship*. Boston, MA: Beacon Press.

Lucy, John A. 1992a. *Language, Diversity, and Thought*. Cambridge: Cambridge University Press.

1992b. *Grammatical Categories and Cognition*. Cambridge: Cambridge University Press.

ed. 1993. *Reflexive Language: Reported Speech and Metapragmatics*. Cambridge: Cambridge University Press.

MacKay, Donald M. 1969. *Information, Mechanism and Meaning*. Cambridge, MA: MIT Press.

Marx, Karl. 1967[1867]. Capital, Vol. 1. New York: International Publishers.

McCarthy, John. 1960. Recursive Functions of Symbolic Expressions and their Computation by Machine. *Communications of the ACM* 3(4): 184–95.

McCulloch, Warren S., and Walter Pitts. 1943. A Logical Calculus of the Ideas Immanent in Nervous activity. *Bulletin of Mathematic Biophysics* 5: 24–56.

McLuhan, Marshall. 1996[1964]. *Understanding Media*. Cambridge, MA: MIT Press.

Mirowski, Philip. 2002. *Machine Dreams*. Cambridge: Cambridge University Press.

Peirce, Charles Sanders. 1955a. Logic as Semiotic: The Theory of Signs. In *Philosophical Writings of Peirce*, ed. Justus Buchler, 98–119. New York: Dover Publications.

1955b. The Principles of Phenomenology. In *Philosophical Writings of Peirce*, ed. Justus Buchler, 74–97. New York: Dover Publications.

1955c. Pragmatism in Retrospect: A Last Formulation. In *Philosophical Writings of Peirce*, ed. Justus Buchler, 269–89. New York: Dover Publications.

Petzold, Charles. 2000. *Code: The Hidden Language of Computer Hardware and Software*. Redmond, WA: Microsoft Press.

Piccinini, G. 2004. The First Computational theory of Mind and Brain. *Synthese* 141(2): 175–215.

Rabin, M. O., and D. Scott. 1959. Finite Automata and their Decision Problems. *IBM Journal*, April: 114–25.

Sapir, Edward. 1949[1924]. The Grammarian and His Language. In *Selected Writings in Language, Culture, and Personality*, ed. David G. Mandelbaum, 150–9. Berkeley: University of California Press.

1985[1927]. The Unconscious Patterning of Behavior in Society. In *Selected Writings in Language, Culture, and Personality*, ed. David G. Mandelbaum, 544–59. Berkeley: University of California Press.

1985[1945]. Grading: A Study in Semantics. In *Selected Writings in Language, Culture, and Personality*, ed. David G. Mandelbaum, 122–49. Berkeley: University of California Press.

Saussure, Ferdinand de. 1983[1916]. *Course in General Linguistics*. La Salle, IL: Open Court Press.

Saygin, Ayse Pinar, Ilyas Cicekli, and Varol Akman. 2001. Turing Test: 50 Years Later. *Minds and Machines* 10: 463–518.

Serres, Michael. 2007[1982]. *The Parasite*. Minneapolis: Minnesota Press.

Shannon, Claude E. 1936. *A Symbolic Analysis of Relay and Switching Circuits*. Cambridge, MA: MIT Department of Electrical Engineering M.S. MIT Institute Archives.

1949. Communication Theory of Secrecy Systems. Declassified Document.

1963[1949]. The Mathematical Theory of Communication. In *The Mathematical Theory of Communication*, 29–125. Urbana: University of Illinois Press.

Sidnell, Jack. 2010. *Conversation Analysis: An Introduction*. London: Blackwell.

Sidnell, Jack, and Nick Enfield. 2012. Language Diversity and Social Action: A third locus of linguistic relativity. *Current Anthropology* 53: 302–33.

Silverstein, Michael. 1976. Shifters, Linguistic Categories and Cultural Description. In *Meaning in Anthropology*, ed. K. Basso and H. Selby, 11–55. Albuquerque: University of New Mexico Press.

1979. Language Function and Linguistic Ideology. In *The Elements: A Parasession on Linguistic Units and Levels*, ed. Paul R. Clyne, William

F. Hanks, and Carol L. Hofbauer, 193–247. Chicago: Chicago Linguistic Society.

Silverstein, Michael, and Greg Urban, eds. 1996. *Natural Histories of Discourse*. Chicago: University of Chicago.

Simon, Herbert A. 1996. *The Sciences of the Artificial*. Cambridge, MA: MIT Press.

Sipser, Michael. 2007. *A Theory of Computation*. New Delhi: Cengage Learning.

Stephenson, Neal. 1999. *In the Beginning Was the Command Line*. New York: Perennial Press.

Suchman, Lucy A. 2007. *Man-Machine Reconfigurations*. Cambridge: Cambridge University Press.

Thompson, Ken. 1968. Programming Techniques: Regular Expression Search Alogorithm Communications. *ACM* 11(6): 419–422.

Tomasello, Michael, and Josep Call. 1997. *Primate Cognition*. New York: Oxford University Press.

Turing, Alan. 2004[1936]. On Computable Numbers, with an Application to the Entscheidungsproblem. In *The Essential Turing*, ed. B. Jack Copeland, 58–90. Oxford: Oxford University Press.

2004[1950]. Computing Machinery and Intelligence. In *The Essential Turing*, ed. B. Jack Copeland, 433–64. Oxford: Oxford University Press.

Van Valin, Robert D., and Randy LaPolla. 1997. *Syntax*. Cambridge: Cambridge University Press.

Victor, Bret. 2006. Magic Ink: Information Software and the Graphical Interface. Available at: http://worrydream.com/MagicInk/.

Wardrip-Fruin, Noah and Nick Montfort, eds. 2003. *The New Media Reader*. Cambridge, MA: MIT Press.

Whorf, Benjamin L. 1956[1937]. Grammatical Categories. In *Language, Thought, and Reality: Selected Writings of Benjamin Lee Whorf*, ed. John B. Carroll, 87–101. Cambridge, MA: The MIT Press.

1956[1939]. The Relation of Habitual Thought and Behavior to Language. In *Language, Thought, and Reality: Selected Writings of Benjamin Lee Whorf*, ed. John B. Carroll, 134–59. Cambridge, MA: The MIT Press.

Wittgenstein, Ludwig. 1958[1953]. *Philosophical Investigations*. Englewood Cliffs, NJ: Prentice Hall.

1961[1921]. *Tractatus Logico-Philosophicus*. London: Routledge and Kegan Paul.