INVENTING COMPUTATIONAL RHETORIC

By

Michael W. Wojcik

A THESIS

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Digital Rhetoric and Professional Writing — Master of Arts

2013

ABSTRACT

INVENTING COMPUTATIONAL RHETORIC

by

Michael W. Wojcik

Many disciplines in the humanities are developing "computational" branches which make use of information technology to process large amounts of data algorithmically. The field of computational rhetoric is in its infancy, but we are already seeing interesting results from applying the ideas and goals of rhetoric to text processing and related areas. After considering what computational rhetoric might be, three approaches to inventing computational rhetorics are presented: a structural schema, a review of extant work, and a theoretical exploration.

For Malea

ACKNOWLEDGEMENTS

Above all else I must thank my beloved wife, Malea Powell, without whose prompting this thesis would have remained forever incomplete. I am also grateful for the presence in my life of my terrific stepdaughter, Audrey Swartz, and wonderful granddaughter Lucille.

My thesis committee, Dean Rehberger, Bill Hart-Davidson, and John Monberg, provided me with generous guidance and inspiration. Other faculty members at Michigan State who helped me explore relevant ideas include Rochelle Harris, Mike McLeod, Joyce Chai, Danielle Devoss, and Bump Halbritter. My previous academic program at Miami University did not result in a degree, but faculty there also contributed greatly to my theoretical understanding, particularly Susan Morgan, Mary-Jean Corbett, Brit Harwood, J. Edgar Tidwell, Lori Merish, Vicki Smith, Alice Adams, Fran Dolan, and Keith Tuma. Then too there are the many graduate students I have known and worked with over the past 21 years—too many to list.

Finally, I would like to express my gratitude to and affection for my parents and siblings, whose intelligence and intellectual curiosity informed my childhood and put me on the road to academia.

TABLE OF CONTENTS

Chapter 1

Introduction:

{Rhetoric, Computation, Invention}

Considered as a Vector Subspace

It might be observed, as a general principle of academic writing, that the length of titles ought to be inversely proportional to the length of the documents they announce. While "Keats's use of 'Pip-Civilian': An Antedating for the OED Entry for 'Pip' N2.3" (Edgecombe) is dealt with comfortably in some six hundred words, a label like "Developmental Psychology" nearly demands at least a shelf of sturdy volumes. The more qualified (and so longer) the title, in other words, the more specific the topic; and thus the less space required to illuminate its mysteries in a satisfactory fashion.

The alert reader will have noticed that the title of the present study is merely three words. They are not idle words, to be sure; back to front, we have a noun of some specificity, a qualifying adjective to bring some focus to the matter, and a gerund to suggest an orientation to the subject. Certainly things would be worse if the title were "Doing Unlimited Everything."[1] Nonetheless, I feel the first order of business ought to be

[1] M. Proust, forthcoming.

defining the subject of this thesis rather more precisely, at least as a consolation to the reader.[2]

My definitions for each of these three terms—*rhetoric*, *computation*, and *invention*—will appear momentarily, but I'll preface with the suggestion that each one be considered in terms of what it points toward—an angle of view, if you will, onto the landscape of meaning. That is because I will ask the reader to imagine the subject of this thesis as a volume or subspace in the universe of possible intellectual work. Consider rhetoric, computation, and invention as three not-entirely-independent vectors, straining away from one another in this space of our cognition but entangled at their roots; sketching out a nebulous region of uncertain extent, interpenetrating other fields (classical and modern rhetoric, computational linguistics, and so on) but nonetheless with a discernible shape of its own. In this project I hope to go some way in defining this space as an area for further work and play.

Rhetoric

"Rhetoric" has enjoyed many definitions, from the relatively narrow and specific variations on Aristotle's "the faculty of observing in any given case the available means of persuasion" to broader ones, many introduced by contemporary rhetoric scholars arguing for an "epistemic" or "constitutive" status for rhetoric (Scott, White). I may be numbered among the latter camp, having elsewhere proposed a definition of rhetoric as "ideoplasty," the process of constructing durable assemblages of ideas (Wojcik "Defining Rhetoric"[3]). For me rhetoric is the study of how ideas can be shaped and shuffled to fit together—in

[2]This is a lie; I never aim to console the reader. My purpose is always to cause as much intellectual turbulence as necessary, so that in accordance with the theory of ideational thermodynamics I can encourage the cognitive system as far from its present equilibrium as possible.

[3]Much of the following discussion of ideoplasty is taken or adapted from this document, an informal essay I published on my own website.

particular, how a set of ideas can be coaxed into fitting onto the structure of existing ideas held by each member of the audience. (In this context it's important to remember that we are all audiences for our own ideas, that the creation of an elaborated, articulated system of ideas involves an internal—and often externalized—process which phenomenologically feels and behaves like a discussion or debate, a working-through of concepts embedded in language. Thus we are all rhetors to ourselves.)

Ideas are plastic: they're malleable, sticky manifolds with complicated shapes, protuberances and concavities, that fit together eagerly but awkwardly into the complex, unstable, but strangely resilient accretions that are systems of ideas, theories and philosophies and ideologies. Rhetoric is the chemistry of ideas, the protein-folding of language and other representations.

The plasticity of ideas is not only the result of rational meaning-production. In the thinking subject, emotion and the unconscious have equal, if not greater, weight; it's well-known that emotional reactions in particular come prior to the interventions of the conscious mind. The physical world prompts ideas through its effects on the senses, and constrains them as they more or less succeed or fail in modeling and predicting it (or offering attractive alternatives to it). Myriad social forces affect the assembling of ideas. By ideoplasty, then, I mean the total construction of ideas and idea networks upon the substrate of a meaning-producing network of heterogeneous agents, using the pressures that can be applied through semiotic systems such as language. And rhetoric is the study and practice of ideoplastic techniques.

Ideas are networks of meaning-relations, which is part of what gives them their flexibility: each relation is a potential pivot.[4] Yet meanings in turn are networks of ideas. There are self-similar layers of progressive abstraction all the way from the general negative networks of sign-systems to the most specific, individual, contingent narratives.

[4]This idea of pivoting relations is similar to one expressed by Stuart Hall in explaining his adoption of the term "articulation." See Hall 1986.

Ideoplasty must operate across levels of abstraction; rhetoric must account for language games and the feints of the individual unconscious but also for effects at the largest scales, like national ideologies and the macroeconomics of global commerce.

Ideoplasty holds that significant ideas (models and theories, generalizations aggregated from small, ephemeral, highly-contextual ideas) are formed upon and by networks of agents; they don't arise or reside solely in individual minds. Thus ideas are not communicated more or less accurately through some semiotic medium between individuals. Instead, semiotic pressures may lead interacting meaning-producing networks to arrive at similar meanings (which means they arrive at meanings that cause agents to produce similar symptomatic effects—the only rubric we have for comparing meanings). The rhetor as ideoplast mills ideas, sorting and shaping, refining and combining, creating tools to coax networks of meaning into configurations likely to have the desired outcome.

Under this description, then, rhetoric is an orientation toward the meaning-making process that emphasizes the production and consumption of idea-systems by networks of heterogeneous agents. For this study, this has a number of ramifications. Most importantly, it acknowledges that technologies may participate in rhetorical processes, and not simply serve to transmit rhetorical force. And so the rhetoric vector looks toward, yes, persuasion; but in general the use of symbolic and semiotic systems to propagate ideas and shape cultural structures; and even more broadly the reification of meaning as an interpersonal act, to promote some accord in understanding, intent, and action. It considers matters in the light of communicative action on ideas, the attempt to share our interior representation of the world.

Computation

Ken Wilson's famous description of computation as the new "third leg" of science, supplementing theory and empirical work, captures the importance of computational approaches

to such fields as physics, biology, and linguistics (Wilson, quoted in Denning). Theory and experiment (or controlled observation, for empirical questions that aren't amenable to experiment) complement one another; theory guides the experimental program and produces interpretations from data, and experiment challenges and endorses theory. Each goes where the other cannot, but remains relevant to its complement as it does so. Wilson's insight is that the computational fields have mobilized machine computation in an equivalent fashion, achieving what neither theory nor conventional empirical research can do, while remaining consistent with the scientific episteme.

In the following chapters I describe in more detail just how computation does that; the key point here is that adding computational methods to scientific research protocols produced a qualitatively different scientific approach that enlarged the domain of questions amenable to investigation. Could the same be true of rhetoric? Obviously this thesis is predicated on the assumption that it is, though I will return to the subject of the essential limitations of computational rhetoric. For now, I will simply claim that computation extends my field of investigation because it enables rhetorical projects that are impossible without it.

Of course we have a long intellectual history of hostilities, or at least mutual suspicion, between our intuitive, experiential understandings of human activity (what becomes the humanities in an academic context) and those areas of understanding where we have attempted to minimize subjectivity—mathematics and the sciences. From the earliest constructions of separate disciplines of knowledge to C. P. Snow's "two cultures" scholars have described the humanistic and the computational as opposed or even incommensurable. But the past few decades of research into computational approaches to human activities have shown that the gulf between them is not so vast after all.

The computation vector is the reducing and purifying glance of abstraction. It seeks formalisms that mime the quintessential shapes beneath the messiness of the world's phenomena. At its core, computation is not concerned with meaning—a human gloss—

but with information, the accretion of the unlikely and unexpected, and tries to expect it. Computation is not the antithesis of the humanities (among which rhetoric is assuredly placed) but its suspension, a bating of our awareness of the subjective and specific and elusive. By permitting rhetorical projects to operate temporarily and in part within that cold and airless space, we may see new features crystallize and become available for our eventual production of meaning.

Invention

Invention, my third vector, begins quite cozily alongside rhetoric, since it is primarily in the rhetorical sense that I'm employing the term. My "inventing" refers in part to the nascent status of computational rhetoric as a field, but it will continue to apply long after computational rhetoric is an established part of the discipline. That is, I do not mean "inventing" so much in the common sense of presenting a *sui generis* intellectual feat—I am by no means the first to suggest computational rhetoric—but as the rhetorical canon: What are some of the moves we might make as we try to imagine what computational rhetoric could become, and how to bring that into being?

I'm invoking the spirit of invention, then, because my field of inquiry here is itself a kind of inquiry: I want to consider how we may consider yoking computation and rhetoric. The conceptualization of rhetoric—the consideration of rhetoric in the abstract, as a whole—asks us to look at the world of human activity from one perspective. The idea of computational technique has us cast our gaze in another direction. Invention proposes a third from which we can survey the combination of the first two. In other words, conjoining rhetoric and computation raises many sorts of questions, and I'm mostly concerned with the ones around invention.

Though I'm using invention in primarily a rhetorical sense here, I am not talking about rhetorical invention *per se*: how I generated ideas for this essay, for example, or what in-

vention strategies might be enabled by computation (other than perhaps in passing, as an example of computational rhetorical practice). So my "rhetoric" and "invention" vectors really are separate aspects of inquiry.

These three vectors, then, extending in their not-entirely-separate directions, unfold a space of possibility for intellectual work. What can we do to imagine ways to introduce computational methods into rhetorical investigation and practice? Obviously this space is far too large to survey even cursorily in a single thesis; but my intent is to describe some trajectories and points of interest within it, to illustrate some of its potential. In other words, in what follows I do not invent a computational rhetoric; I describe an approach to inventing them.

Of the chapters that follow, the first explores the nature of computational fields, and lays out the groundwork for the remainder of the thesis. Chapter three describes in detail the primary invention strategy I'm proposing, while the fourth chapter contains brief reviews of several computational-rhetorical projects with an emphasis on invention. The last part of the thesis, which is actually an *arsis* (the opposite of a thesis, a "lifting up"), offers provocations and suggestions for further work.

Chapter 2

Computational Rhetoric and How to Invent It

Recent years have seen the growth of "computational" branches in a number of academic and scientific fields. Though some of these, such as computational biology, date back nearly to the introduction of programmable digital computers,[1] for the most part computational sciences became prominent starting in the 1980s (as personal computers became available and scientific workstation computers dropped in price), and have continued to grow in importance.[2] And while the sciences formed the vanguard of computational research, the humanities are rapidly developing their own computational branches. That fields such as linguistics, which straddles the sciences/humanities divide, or sociology, with its longstanding interest in statistical and other quantitative analytic approaches, were quick to adopt computational methods is unsurprising; perhaps more remarkable is the growth in self-described computational history, computational law, computational

[1] As Hagen notes in his abstract: "Computers emerged as important tools in molecular biology during the early 1960s. A decade before DNA sequencing became feasible, computational biologists focused on the rapidly accumulating data from protein biochemistry."

[2] See for example the reports from the National Science Foundation Blue Ribbon Panel on Simulation-Based Engineering Science and the Council on Competitiveness.

philosophy, computational approaches in literary studies—even computational approaches in the arts.[3]

To date, however, there has been no organized effort to establish a computational branch in the discipline of rhetoric. That is not to say that no one has done work in rhetoric that could be called "computational"—far from it, and I will review some of that work later—or even that no one has done work under the aegis of "computational rhetoric" (again, examples will be provided in a later chapter). But as a field we generally lack regular conference panels (much less entire conferences), newsletters and journals, graduate concentrations, email or online forums, research groups, and the rest of the apparatus that would indicate that computational rhetoric is an established and vibrant branch of rhetorical research. It is budding; it has yet to flower.

What follows are suggestions and provocations for inventing computational rhetoric.

What is a computational field?

The computational fields are clearly united by their emphasis and reliance on modern information and communications technology (ICT)—that is to say computers, in various guises and roles. But this is also clearly insufficient to define them, since computers are now ubiquitous in scholarly and scientific research; and computational rhetoric in particular cannot be defined simply by its use of information technology, because rhetorics have always been founded on the information technologies of their era, going back to the first information technology, language.[4] What distinguishes the computational fields, and will distinguish computational rhetorics as they are developed, is not just the emphasis

[3]Two recent examples of computational humanities are computational journalism (Cohen) and computational folkloristics (Abello).

[4]I'm using "language" here loosely, to mean any systematic attempt to create external representations of information with the goal of encouraging congruent meaning-production in an audience. Besides spoken and written languages *per se*, this would include representational gestures and the like.

on ICT but how it is used, and more importantly its epistemic role. In the computational fields, information technology is treated not simply as an adjunct to the processes of research methods (say as a means to record and transmit data and results, or a piece of laboratory apparatus), but as the very center of those methods, the mill that threshes the meaningful grain from the noisy chaff.

To better illustrate the distinctive qualities of the computational fields, and how those arise from and reveal ICT as their epistemological ground, I want to consider some of the specific, concrete affordances of computing technology, how those are typically used in computational fields, and how those uses are different from the general employment of computers in scholarly work. Of course, the division between computational and traditional methods is not so firm, and work in the real world often includes computational and non-computational elements.[5] Ultimately, a computational project is one that puts some emphasis on computational methods—though what degree of "some" is sufficient will depend on the observer.

Calculation

From the common-sense notion of "computation" as manipulating numbers (or from its etymological meaning of reckoning or settling accounts) we might first associate computational research with calculations of some sort. And indeed the earliest computational science—computational physics—began employing computer technology to perform large and tedious calculations. Von Neumann's design notes for the EDVAC use physics calculations as guides for such details as the size of a machine word, for example, and computational physics arguably got its start in the Manhattan Project, when Frankel built a scientific calculator from IBM business tabulators (Feynman 108–114). Those tabula-

[5] Anyone looking for a more formal way to treat this ambivalence might wish to consider it in terms of Zadeh's fuzzy sets: a computational project might have 0.7 membership in the computational set, and 0.3 membership in the traditional.

tors were electromechanical digital computers, predecessors of today's electronic digital computers; and prior to digital computers, analog ones had been employed to solve such physics problems as computing ballistic tables and predicting astronomical events. So number-crunching—performing repetitive and complex calculations, often on large data sets—is one of the characteristics of a computational field.

It may not be obvious what role, if any, number-crunching might play in typical research in the computational humanities, lacking as we do a calculus of aesthetics or a quantitative model of the human condition. Nonetheless, computational work outside the hard sciences still often involves some relatively ambitious applied mathematics, often in the form of probability or statistical analysis, as I describe below.

Big data

Calculation was the original primary role of ICT in computational research, but the roughly exponential growth in data storage capacities over the past few decades[6] has made the storage, retrieval, and processing (particularly searching and pattern analysis) of large data sets perhaps the most common task for computing hardware in the computational disciplines. In the humanities this has recently been encouraged by grant initiatives such as the Digging into Data Challenge created by the National Endowment for the Humanities and other granting organizations, in effect endorsing so-called "big data" projects as the quintessential work of the computational humanities. (I discuss

[6]Because data storage devices come in many types, arrangements, and prices, it's possible to pick almost arbitrary sets of data points when trying to discern actual growth patterns. However, while some aspects of storage—particularly the speed of reading and writing permanent storage—have not grown as quickly as, say, CPU processing power, it's reasonable to argue that the overall curve is a roughly exponential one, doubling over a period of between one and three years. Consider that 1991 saw the first gigabyte-capacity single hard drive (the IBM Corsair), 1992 the 2.1 GB Seagate Barracuda, 1997 the 16GB IBM Titan, and 2003 the 37GB Western Digital Raptor (see Farrance); while those are only four data points, and other factors (size, speed, price) should also be examined, they do fit a rough exponential curve.

some of these projects in more detail in a later chapter.)

Of course calculation and big data often go hand-in-hand, particularly in ambitious physical-science research experiments like those conducted with the Large Hadron Collider and similar apparatus. Such projects generate vast quantities of data that could never be examined directly by human researchers, and monitor phenomena in indirect ways that can only be brought to bear on the hypotheses being tested by extensive mathematical manipulation. In the humanities, however, big-data experiments may consist primarily of searching large data sets in comparatively simple ways, such as graphing the relative frequency of two phrases across a corpus of text (a comparative n-gram search—these days a recreational pastime, thanks to the Google Books Ngram Viewer). Indeed, though I don't know of any study on the subject, I suspect most computational humanists get their start doing casual searches—*ad hoc* big-data experiments—in some of the many humanities databases available at research libraries.

Machine sensing and data collection

Obviously the use of various information technologies as extensions of the senses, to detect and record data about the physical world, is far older than digital computers; Muybridge's photographic motion studies are an obvious example, and one could argue for including ancient technologies such as noisemakers attached to animal traps. Modern ICT has greatly broadened automatic data collection, combining a vast array of sensors (many of which respond to signals inaccessible to human senses) with high-frequency, high-capacity recording—which often produces big data, of course.

While machine sensing and data collection are most often associated with the physical sciences and engineering, they have found applications in the humanities. Psychologists and HCI researchers have used eye-tracking equipment to study how people use visual data, such as visual components of user interfaces, for example. Computerized audio

recording and analysis have applications in linguistics[7] and speech therapy.

Probabilistic analysis and modeling

Because big data sets are, more or less by definition, not amenable to analysis in their entirety, many computational projects use various forms of probabilistic analysis. These include random sampling techniques (Monte Carlo, reservoir sampling, etc) and statistical models derived from them. Heuristic searching techniques are often employed to identify a small subset of data of particular interest, especially in studies where "interesting" is not well-defined. More-sophisticated probabilistic models, such as simple Bayesian models, Hidden Markov Models, and relatively esoteric mathematical abstractions such as lattice algebras, are also commonly constructed using large data sets. These models can be used for *decoding* (that is, to assign meaning to new data based on the model derived from the existing data set), *prediction* (determining what the next datum is likely to be, given a sequence of input), *classification* (grouping sequences of input), and similar tasks.

Because of the difficulties of dealing with human languages, much natural language processing (NLP) research—and applications, including commercial ones—employs multiple probabilistic models, both for basic operations such as determining sentence boundaries, identifying parts of speech and semantic roles for given words, and syntactic parsing, and for higher-order analysis like determining the relationship among actors named in a text. The application of these techniques to rhetorical work should be obvious.

[7] In a later chapter I'll describe a computational-rhetoric project, Speech Made Visible, which uses audio analysis to extract prosodic information from speech. SMV is built on Praat, a well-known prosody-analysis tool that's seen wide use in computational linguistics.

Approximating human abstractions

The true promise of probabilistic modeling for much of the humanities, and particularly for computational rhetoric, might lie in constructing meaningful approximations of (currently) unquantifiable human abstractions. Concepts like emotion, reason, sense, persuasion, truth (in a non-formal sense), and good are the currency of the humanities, but because we have no formal models of them, it's difficult to construct bridges between problem domains that exceed human capacity for direct, personal investigation—precisely those that computational approaches address—and explanations that appeal to such subjective, difficult-to-specify ideas. We may do a close reading of a poem to construct a meaning that explores its affective power; it's not feasible to do comparative close readings of a million poems. When we want to survey emotional impact or persuasive power, say, across a huge corpus of texts, we need a heuristic computational model that accords in some way with our intuitive and theoretical understandings of human abstractions.

A prominent example in existing computational research (and an example of work that could be seen as computational rhetoric, even though it's not generally described that way) is sentiment analysis. Attempting to model the sentiment, or content that reflects the attitude and emotions of the author, in written text (and in a smaller but important body of research, speech) is a major area of both research and commercial NLP. Sentiment analysis has become a significant business because it has considerable value in marketing, public relations, and evaluating the success of products; an obvious and common application is in determining sentiment trends in online customer-contributed product reviews. Beyond such mercantile uses, sentiment analysis is important for semantic analysis, since the emotive content of an utterance conditions how the factual content is meant to be understood[8]—consider in this regard tone, or any use of rhetorical

[8]For the sake of brevity here I'm assuming something of an intentionalist model ("is meant to be understood") of meaning—one which I actually believe poorly represents the mechanisms of truth-representation and meaning-production in linguistic interaction.

tropes to turn from the obvious (most probable, or surface) emotional reaction.

Computational epistemology

Of course roles like the ones I've outlined here are just a convenience; actual projects always contain each of these in some measure, as they're all simply aspects of the essential functions of computation. They emerge from gathering, storing, retrieving, filtering, transforming, and transmitting data—that is, the basic computer operations, outlined by researchers such as von Neumann at the dawn of the digital-computing era. And these operations are themselves really only interpretations of the fundamental processes of computing hardware (the manipulation of bits, or more concretely the routing of electrical and other signals); and those processes are no more than physical approximations of various equivalent formalisms (Turing machines, Post machines, untyped lambda calculus, etc.). Or taken in another direction, all of these operations can be reduced in principle to arithmetic (since the entire state of an ICT system can be considered as a single large binary number) or compression (it's possible to describe any use of ICT as a process of discovering the interesting elements of a data set; this is a rearrangement of information entropy, which is simply a formal definition of data compression).[9]

My point here is that identifying a computational field is not a Linnæan matter of discerning functions like those named above in computer applications used by researchers; these are merely examples of ways to interpret how computational researchers have used ICT, because all computer use is in the end of a single kind. Ultimately, as I suggested earlier, what distinguishes a computational field is an epistemological orientation toward

Intentionalist and anti-intentionalist understandings of language, and the discrepancies between them, raise various problems for the possibilities for computational rhetoric, but space does not permit reviewing any of them at this point.

[9]Taking this argument a bit further, through information thermodynamics, we can see that all computation is simply a process of moving heat around. Thus computational rhetoric is really nothing new—just rhetoricians producing more hot air.

machine computation: an intellectual project that endorses the notion that there are meanings which can only be discovered or justified by applying information technologies to operate on data in ways or volumes that human beings could not achieve without such assistance. Computational fields, ultimately, are simply those branches of scholarly and scientific disciplines that accept a computational epistemology, and consequently accept information technology as central to their projects. This is in some ways a dramatic shift in the philosophy of intellectual work; for most of human history, information technologies seemed to be regarded as adjuncts and conveniences for recording and conveying the results of work which took place entirely in the human mind.[10] While computational fields do not make an oracle of the computer—*meaning* is still ultimately produced by human subjects, interacting with the information distilled by the machine—it mediates the scholarly/scientific relationship between the object of study and the investigator, and as such becomes the irreplaceable fulcrum of knowledge.

So what might such an orientation mean for rhetoric? And if we were to adopt one, how might we proceed?

These are the questions I want to begin to address in the chapters that follow. I will do so in ways that I admit are tentative, partial, and unsubtle. Many issues theoretical and practical will go unaddressed and even unacknowledged. My hope, though, is to suggest a number of points of departure for further inquiry into the possibilities both for computational rhetorics in the abstract, and for specific projects in that field.

Before that, however, I wish to briefly address a handful of minor issues: how I intend

[10]I admit this claim is speculative; if there's historical or anthropological work that supports (or challenges) it, I'm afraid I'm unaware of it. And, of course, the modern era has seen a series of challenges to notions of individual genius and recognition of the roles of society and environment in intellectual work; it's now generally accepted that the "creative mind" is a gloss for a complex network of interactions that extend far beyond the individual. Nonetheless, not only admitting that information technology can exceed human intellectual capacities but eagerly pursuing the implications seems to me to be a distinctly modern phenomenon.

to approach the problem, why it needs to be addressed (and why it needs to be addressed now), and just how far I expect to get.

Invention strategies

There are innumerable invention strategies, from carefully delineated ones like Aristotle's *topoi,* to the vague aphorisms of writing handbooks like "write what you know" or "paint a picture for the reader", to those idiosyncratic behaviors—rambling countryside walks, visits to the local coffee shop, and the like—that each writer likely discovers to free his or her personal flow of words. And no doubt any of us could spend a bit of time researching computational fields and then sit back and generate any number of ideas for possible computational rhetorics with nothing more definite to go by.

Rather than let this project be taken over by random speculation (however entertaining that might be), I chose three approaches for my attempt to invent computational rhetoric: one structural, one empirical, and (more briefly in the final arsis) one philosophical. The first is a schema, where I break computational rhetoric down in a series of categories and attributes. This is a classically structuralist approach, and while it commits all the sins of structuralism—oversimplifying, homogenizing, eliding difference, excluding middles, ignoring circulations across boundaries, and so forth—it can be very productive, both by following it as a sort of scaffold or checklist to expand the parameters of an idea, and by resisting its strictures to see what might be overlooked. Like Donna Haraway, I find "structuralist meaning-making machines" useful (318), provided we do not give them the last word. The schema also provides a vocabulary and something of a taxonomy for the remainder of the thesis.[11]

[11]Some might be tempted to label it an "ontology," following that term's new popularity in some branches of computer science and informatics. I'm reluctant to further dilute that useful technical term by casual application.

My empirical invention strategy is simply to look at some interesting examples of extant work. That chapter includes work that is explicitly described as computational rhetoric, but because the phrase is not yet widely used, I also review other rhetorical work that I consider computational, work in cognate research fields such as linguistics, and applications of information technology that I argue effectively embody rhetorical practices. This strategy is not a literature survey, though some of what I consider is in the form of traditional scholarly articles; it's a rumination on what computational rhetoric might be, inspired by a handful of examples of what it currently is.

In the final pages, among other notes on possibilities for for future work, I turn to what I've called a philosophical invention strategy. Here I suggest we can continue to invent computational rhetoric by interrogating it as a concept. This is literally a theoretical approach, but I believe (indeed I believe it is obvious) that theory can bear directly on practice.[12] This involves raising (and briefly considering, only to leave unanswered) a number of questions, regarding topics such as the philosophy of mechanical rhetoric (what happens when we reduce a rhetorical task to mechanical steps, along the lines of Searle's "Chinese Room" argument[13]), computational complexity theory (what sorts of

[12]Some of course disagree, for example Stanley Fish in such essays as "Truth but No Consequences". I have explained why I disagree with Fish elsewhere ("Inventing Theory" 5 *passim*), but to summarize, I believe he ignores a number of what in the information-security field we refer to as "side channels" by which philosophical beliefs can shape the substantive beliefs that we then refer to when reviewing possible courses of future action. And in some cases people may make a deliberate attempt to derive practices from abstract theoretical commitments; as an anecdotal example, I'll recall Bill Hart-Davidson mentioning on more than one occasion that the design of Eli, a software package used to facilitate peer review in writing classrooms (about which I have more to say in chapter four), was based on ideas about language derived from post-structuralism, and in particular from Derrida's work (personal conversations, *circa* 2010).

[13]The Chinese Room argument was first published in "Minds, Brains, and Programs;" by that time Searle had presented it in various talks, and the article includes his summaries of assorted objections, and his responses to them. I have also benefitted from some of the many discussions of and responses to it, such as in Hofstadter (631 *passim*) and Penrose (18–21).

tasks require more operations than real computers can perform in reasonable time), the problem of algorithmic reduction of semantic layers (how much information, and of what sort, do we lose as we reduce language acts to abstractions and separate them into planes of structure, reference, and so on), and the limit posed by the phenomenological horizon: since we only now understand, and perhaps will ever understand, only a portion of how our own minds work when performing rhetorical tasks, can we hope to create machines that can implement any more than that understanding?

Why do we need computational rhetoric? And why *now*?

I admit it's tempting to simply ignore this question, or to argue that we should investigate computational rhetoric simply because we can. Like everyone else, rhetoric scholars today have unprecedented access to computational resources: an iPhone has roughly one-and-a-half times the computing power of a 1980s supercomputer (the Cray-2), and storage and networking capacities have grown even faster while prices have plunged. A few thousand dollars' worth of commodity hardware will give you a system equivalent to hundreds of the computers used for early work in computational linguistics and natural language processing. (An illustrative comparison can be found in Olds.) A slightly more sophisticated version: if this is a potentially valuable area of research which is newly open to rhetoric, then the discipline has an obligation to devote some effort to investigating it.

And while computational rhetoric in no way replaces other forms of rhetorical research and practice, aside from certain specialized tasks (such as coding data and compiling statistics) in certain kinds of projects, neither does it duplicate established fields such as computational linguistics. It's worth briefly describing ways in which computational rhetoric is not computational linguistics or NLP, though there are certainly projects which straddle the division or investigate what might be considered rhetorical questions in a lin-

guistic framework.[14] First, linguistics is fundamentally concerned with the operation of language (*langue*) as a phenomenon in itself, and treats instances of language use (*parole*) as the surface which reveals its object of study. Rhetoric, traditionally, has viewed *langue* as a fundament, ground, or set of *topoi* of rhetorical action, and *parole* as a scene or locus of rhetorical acts; for rhetoric *parole* is the primary object of study. Rhetoric also has a fundamental teleological bent, whether that's conceived of as persuasion or the production of shared meaning or some other end, with which linguistics is often relatively unconcerned. Second, much of linguistics, including most of computational linguistics, appeals to formal (tautological) truth; computing the f_0 pitch train of recorded speech, for example, is not a subjective operation (though its interpretation is). Computational rhetoric is, in its overall range and on average, more provisional than computational linguistics. And a third: computational rhetoric, as I've conceived it here, operates at a higher (potentially much higher) level of abstraction than even such linguistic fields as sociolinguistics or pragmatics, and the "more is different" principle applies.[15] Nor is computational rhetoric a new twist on "digital rhetoric." Digital rhetoric has focused on examining rhetorical affordances of digital information and communication technologies,

[14] As an example of an NLP effort that is directly relevant to rhetoric, consider the "Conversation Entailment" project being conducted by the Language and Interaction Research (LAIR) group at the College of Computer Science and Engineering of Michigan State University. Conversation entailment is a problem in deriving knowledge from a conversation (*i.e.* an exchange, rather than a single unified instance of discourse): does the conversation manifest a particular state, such as a belief or topic? Since classical rhetoric is fundamentally concerned with the interaction between rhetor and audience, conversational exchange is an important dimension of rhetorical activity.

[15] The "more is different" principle was defined by P. W. Anderson in his eponymous 1972 article. In essence, it claims that quantitative differences in scale produce qualitative differences in kind, specifically for scientific epistemology. Thus chemistry is not simply "applied physics," and biologicy is not simply "applied chemistry," and so on; though chemistry deals with effects that arise from the underlying physics of the elements, it cannot be reduced to them. Anderson's argument hinges on the non-equivalence of the reductionist argument (complex systems can be reduced to the interactions of simpler ones) and the constructionist position (the behavior of complex systems can be inferred from that of the simpler ones they are composed of).

adapting traditional rhetorical theories to those domains and creating new ones, using new digital media in reconsidering rhetorical ideas, and so on. Computational rhetoric is different in that it seeks to formulate computable (if usually heuristic) algorithms from rhetorical theory and implement them, using ICT to reify formulations of those theories and apply them in ways infeasible without it; not simply rhetorical computing but computing rhetoric. *Computational rhetoric is, or can be, a new thing.* It can produce new kinds of rhetorical understandings.

But the most important reason, I believe, for a broad, energetic, and sustained engagement with computational rhetoric now is that it is already happening, even if it's rarely acknowledged as computational rhetoric. In academia, industry, and government we find many projects that are employing information technology in the service of goals we as rhetoric scholars would recognize as part of our domain. My colleague Douglas Walls has rightly cautioned me against the temptation to claim the work of others as part of my intellectual domain—to say "they do what we do, they just don't know it"—and we must be aware of this sort of colonizing maneuver. When such projects arise from other disciplines and knowledges, other schemes of meaning, other sets of goals, we do them a disservice by reframing them in our terms, and ourselves a disservice by seeing everything through our own preferred lens.[16] That said, we also must acknowledge that these are projects (I discuss some in chapter four) with consequences for the study and practice of rhetoric. Often they have other, sometimes disturbing, consequences as well; computational rhetoric offers many tools to the burgeoning surveillance state, for example, and its private-sector equivalents. I claim we (as a discipline, if not each of us as individuals) have an ethical obligation to confront this work: to understand, critique, anticipate, challenge, and participate in it.

[16]I should note here that not all of my colleages agree with this position.

The scope of the thesis, or Just how long *is* this thing?

Obviously this document is a very limited space in which to consider these questions. I am constrained by genre (no one wants to see a master's thesis stretch on into hundreds of pages) and by resources of many categories—though relevant information is not one; there's no lack of that. One area I will largely exclude is specific technologies that can be applied to computational rhetoric, for example. My primary focus will be on the sample invention strategies I outline in the next three chapters and the two parts of the arsis—directions for further research and theoretical musings. So the brief survey of computational rhetoric in practice (chapter four) is not intended to be particularly representative, much less comprehensive, and does not go into any great detail; I simply want to point out some interesting features of the landscape. Similarly the theoretical chapter that follows aims more to raise questions than to answer them. I've conceived of this thesis as motivation for a more vigorous and informed conversation, not a manifesto or paradigm.

Chapter 3

A Schema for Computational Rhetoric

My first and most important invention strategy is to lay out a schema for classifying possible varieties of computational rhetoric. This is not a formal taxonomy, and I don't imagine it's rigorous or exhaustive. Rather I'm proposing some possible attributes of computational-rhetoric projects to inspire consideration of what it might mean to combine those attributes in various ways, and spur the imagination to ask what else might be possible under the definition I've proposed for computational rhetoric. In this exercise, I've found it productive to define a handful of axes as dichotomies (even though they often actually describe ends of—or perhaps merely points on—continua, and though many projects will incorporate both of a pair of qualities in a hybrid approach): analytic/synthetic, fixed/adaptive, and so on.

As I've already noted, this is a structural approach, and I'm entirely sympathetic to the critiques mounted by post-structuralism and other theoretical interventions into structuralist meaning-production apparatuses. But at this early stage in establishing computational rhetoric as a distinct field, I think it's most useful to use a relatively simple and mechanistic approach like this to sketch the perimeter, before delving into the incongruities of the interior.

On schemata

Schemata, or schemas, are widely used in computation and the subject of much research and development in computer science and related fields such as informatics. (Indeed they sometimes seem to be the very soul of informatics, an area which well deserves the label "discipline", as it strives to bring order to the tumult of unstructured data.) A schema is simply an arrangement of labels or categories, a device for on the one hand grouping the elements of some set, and on the other expressing relationships among those groups. Schemata have a long history in rhetoric, and indeed can probably be identified throughout the history of language use, though their deliberate construction and employment likely only appears when cultures have sufficient leisure to start reflecting philosophically on their own language practices. The classical rhetors offer many a schema, such as Aristotle's division of the modes of argument into *logos*, *ethos*, and *pathos*, or the classical list of the seven rhetorical canons. A schema can be used to store and retrieve information (the Dewey Decimal System), to regularize terminology (Linnæan taxonomy), to provide a framework for representation (DITA). And as I'll show below, it can also be used to invent.

In critical theory,[1] schemas are easy to find in the work of Structuralists such as Lévi-Strauss, whose famous abstraction of kinship relations, and particularly the incest taboo and exogamy mandate which he saw organized around combinations of the four roles of father / son / brother / sister, was so influential in the development of Lacan's psychoanalytic theory of the Symbolic. Other examples are Greimas and his semiotic square,

[1]The term "critical theory" is often associated historically with the work of the Frankfurt School, and so some writers use it specifically for theories of social organization and cultural production and consumption informed by Marxism and other prominent philosophical perspectives of inter-war continental Europe. I'm using it more broadly, as is typical in contemporary rhetoric scholarship, to refer to the gamut of theoretical perspectives that rhetoric and other disciplines under language studies have engaged in the past few decades, particularly since the rise of "high theory" in U.S. literary studies during the 1980s.

which takes two concepts, treated as oppositional, and their antitheses, and then analyzes a subject in terms of various combinations (A and B, not-A and B, etc); Frye's archetypal theory of literature, which proposes four categories for literary works, each with several corresponding attributes; Burke's dramatic pentad; or the more complex case of Propp's formalist approach to *marchen*, with its assortment of actors and functions. A schema of this sort is often employed mostly for its descriptive and rhetorical power, and tends to be discretionary: there are few or no absolute rules for classifying objects, and assignment within the schema is rather subjective and debatable, with many examples of entities straddling boundaries or occupying multiple positions. The schemata of information technology, on the other hand, are in many cases derived from formal principles and have material consequences, in terms of the work required to perform various operations under them (and so in the resources consumed by that work). Some schemas are promulgated by standards bodies, such as the Document Type Definitions (DTDs) defining HTML and XML produced by the W3C, or the TEI (Text Encoding Initiative) schema widely used in digital scholarship in literature, or the DITA schema of document organization popular in technical writing, developed by IBM and now maintained by the OASIS standards body.[2] Other schemata are *ad hoc* creations for specific applications, though they too are often constrained by formal rules; so for example a relational database schema arises from the applications it serves, but often will be designed in accordance with some degree of "normal form" so that certain useful attributes that derive from relational algebra hold true.[3]

[2]These are also examples of a hierarchy of schemata that is typical of information technology. HTML and XML are applications of SGML, the Standard Generalized Markup Language, which is a schema for defining schemas for formal languages; and both TEI and DITA in turn are applications of XML. See <http://www.w3.org/> for HTML and XML, <http://www.tei-c.org/> for TEI, and <http://dita.xml.org/> for DITA.

[3]"Normal form" is a set of descriptions of constraints on the organization of data in a database, from "first normal form", the weakest, through successively stricter "second normal", "third normal", and so on. A database that follows a normal form has behaviors that are useful for applications, such as consistency under various operations. In practice, most real-world non-trivial databases are to some extent "denormalized", with specific

Other application-specific schemata in IT are a blend of standard and local elements, as is typically the case for the schema defined for an LDAP repository.[4] The schema I'm offering here is of the sort typical in the humanities, purely of my own conception (of course influenced by ideas from many others) rather than originating in a mathematical model or published standard.

A final observation about schemata is that some aspire to completeness in their domain of representation, while others are explicitly partial. The standard LDAP schema defines many types of information about people and organizations, and even about certain kinds of property, but makes no attempt to anticipate other types—except by including its own extension mechanism. Linnæan taxonomy on the other hand is complete, designed to label every conceivable species of organism. HTML (up through version 4[5]) is complete: in a conforming document, every element is one of the types specified by the standard. XML is partial; it does not in itself include enough information to specify even a single (non-trivial) document, so every document extends XML, and XML incorporates optional schemas for defining those extensions (such as DTDs and XML Schema). Aristotle's three-part division of rhetoric is intended to be complete (all appeals are of one of the three

violations of normal form to improve performance, and additional controls so that the reliability of the data is not jeopardized.

[4]LDAP, the Lightweight Directory Access Protocol, is a set of standards that used by many directories of organizational data, such as employee information. So an organization's LDAP repository might contain each employee's name, title, office, contact information, and so on; it will also often contain information peculiar to the organization such as an employee number, records of company-owned equipment issued to the employee, and security information like how recently the employee signed on to the corporate network. The former is standardized by X.400, the international standard LDAP is descended from, while the latter are defined as extensions to the standard schema. A large part of the LDAP schema is actually a schema for defining those extensions to itself.

[5]As of this writing, the published standard version of HTML is 4.01. There are two competing standards: XHTML, an XML version of HTML produced by a different group in the W3C, and HTML 5, a draft "standard" (though it's debatable whether it deserves that term) originally promulgated by WHAT-WG and now taken up by the W3C. I'm referring here to HTML 4.01 Strict, which is the only direct successor of the historical line of HTML standards.

types); his list of invention strategies is not. I hope it's obvious that the schema I'm presenting here is partial, and at the end of this chapter I have some brief comments on how it might be extended.

The axes

Here are four of the axes I see structuring the possible space of a computational rhetoric (CR). Each of these can be thought of as a line from the former concept to the latter. A given computational-rhetorical project can likely be described as occupying some position on each of these lines—though in some cases it might seem more accurate to say a project stretches across an extent on the line, or that parts of a project occupy different positions. Or conversely, we might pick a position on each axis and speculate about what such a project might look like: an analytic application in the computational domain with interactive behavior and adaptive algorithms, for example, might be a program that helps a reader parse Wikipedia entries into lists of claims and refines its operation by remembering corrections made by the user.

Synthetic / Analytic

From a certain very high perspective, rhetorical work—under a classical definition of persuasion, or one closer to my own of shaping and promoting idea-systems—has two modes. The rhetor may be engaged by the production of rhetorical product or by the examination of it, assembling arguments or taking them apart. Both of these modes have theoretical and practical sides, and both can refer to a specific rhetorical work or to rhetorical praxis in the abstract. Of course, each of these modes necessarily incorporates some of the other: when we study an argument we must argue for our conclusions, even if only to ourselves, and an unexamined rhetorical foray—argument by accident, as it were—can hardly be called rhetorical at all. Nonetheless I think we can broadly divide

rhetorical work into a primary orientation toward arguing, on the one hand, and an orientation toward questioning arguments on the other.

For computational rhetoric, I'm referring to these two modes as the synthetic and the analytic.[6] Analytic CR is the algorithmic analysis of rhetoric in incoming data. One technical example I'll describe in a later chapter is my Ethos Estimator project, which tries to identify authoritative authors in an email corpus. Synthetic CR, on the other hand, seeks to employ information technology in crafting effective arguments (or idea-forms); my own example for it is the Speech Made Visible project.

Today much of the work that could be considered computational rhetoric, or akin to it, is on the analytic side of the house. Particularly in industry (and quasi-industry, such as the burgeoning apparatus of the surveillance state), there is great interest in distilling trends and making predictions from "online"[7] text, part of the current fascination with processing large corpora of unstructured or semi-structured data. (Recall the discussion of "big data" in the first chapter.) Alongside that trend, much of the interest, and funding, in cognate fields such as natural language processing (NLP) has been in the machine interpretation of human text and speech, often as a user interface (consider Apple's Siri voice-recognition search system) but also for such applications as machine translation. Synthetic CR is not unknown, however, with a history stretching back some decades, for example in the work of Hugh Burns (about which I'll have more to say in the next chapter). With user-facing computing technology[8] becoming a nearly-continuous part of many

[6]Some of my readers might recognize these as the labels the Cubists assigned to their two major representational modes, and I admit that was an influence in my terminology, though in a way too indirect to really describe.

[7]"Online," which was once a term of art in computing, has become broadened nearly to the point of meaninglessness in common parlance. Here I'm afraid I'm employing it in the rather vague popular sense, which means something like "readily available from an Internet-connected server using a straightforward protocol, with no significant access restrictions"—though many of the people who are so keen to explain the attributes and consequences of "online" activity would have difficulty providing or explaining such a definition.

[8]In the industry, "user-facing" technologies are those that end users interact with di-

people's lives, consumers ever more comfortable with delegating aspects of communication tasks to machines,[9] academics and industry researchers eager to explore the potential of new technologies, and an optimistic industrial sector ready to provide first and attempt to monetize later, I believe we will be seeing more and more software that aims to help its users produce arguments. Such software, I suspect, will in most cases be designed to operate as seamlessly (automatically, continuously, and unobtrusively) as possible: it will be the power-steering of argument.

Computer-Assisted Rhetoric / Computational-Domain Rhetoric

When I ask myself how ICT can be employed rhetorically, I imagine two kinds of roles for IT in the network of rhetorical production. One is as a communication technology: helping a rhetor perform communicative acts, much like other authoring technologies (paper, printing press, mass production and distribution of books, postal systems, etc), and enabling a rhetor to find, gather, and receive information useful to the argument, which is also a communication function. (That is, the information in question is already in the world, and ICT is used to locate and replicate it.) This is what I'm calling computer-assisted rhetoric (CAR), by analogy to, for example, computer-assisted design, and it could even include such familiar applications as word processing, email, and web servers—

rectly. Even ten years ago, even in the most technologically-sophisticated cultures, most people did not consciously use computing devices as a large part of their daily activities, except in some cases in the workplace. Computer use was a partitioned activity: people would set aside time to "get online," read and write email, view web pages, and so on. With the rise of laptop computers (particularly the small "netbook" form factors), smartphones, and tablet computers, deliberate ICT use has become pervasive—often too pervasive, as any number of people have argued. Of course for quite a few years people in industrialized societies have lived with pervasive computing, but most of it has been hidden "embedded" computing in devices we use for other purposes, such as cars and household appliances.

[9]Examples of this delegation trend range from macro-scale (the rise of social-networking websites) through group and institutional (any of several excessive or inefficient email practices spring to mind) to micro-scale (relying on an auto-correct feature when composing text, sometimes with rather hilarious results).

though obviously the term "computational rhetoric" becomes rather less useful if we let it include everything done with computers that's in some way related to rhetorical activity. A more direct example might be the use of a search engine or Wikipedia to find material to support a claim. These are not dedicated rhetorical computer applications, and they have non-rhetorical uses (very broad definitions of "rhetoric" aside); but when used to enable rhetorical activity, they are examples of CAR. There are also some clear examples of programs that are explicitly designed to assist rhetors, some of which I look at in the next chapter.

Less-realized, I think, is the area of computational-domain rhetoric (CDR).[10] Where CAR is essentially a labor-saving innovation, a set of ways to employ IT in relieving the rhetor of repetitive and essentially unrhetorical overhead tasks, CDR truly attempts to realize rhetorical insights in the information-processing domain. So CDR would include such things as the machine analysis of arguments.

At times the distinction may be difficult to draw. Often it's conceivable that a CDR project could be performed by humans, however onerous that might be. (Of course, it's always true that any computation can in theory be performed by humans in pencil-and-paper fashion, given sufficient time and determination.) Thus the CAR/CDR distinction is ultimately a subjective one, and many projects lie somewhere in the middle (or combine the two). A rubric for distinguishing between them might ask questions such as: Is it feasible to do this work without ICT? Is the software written as an implementation of rhetorical theory? Are the rhetorical processes visible to the software developer (for example, looking for a fixed set of keywords), or do they emerge in unpredictable ways in

[10]Readers who are familiar with the famous and influential LISP programming language will recognize a technical pun: in LISP, *car* and *cdr* are two of the most important and basic programming operations. My acronyms are a bit more than an inside joke. *car* returns the first item in a list, and *cdr* returns the remainder of the list; and in the history of ur-computational-rhetoric, CAR is to some extent what came first, and CDR what will follow it. It's easy to make too much of the analogy, which is why I've relegated it to a footnote; but I still find it felicitous.

the operation of the software (for example, with an adaptive algorithm such as a Hidden Markov Model)?

I believe CDR projects relatively rare but becoming less so, and it's tempting to see a historical shift from an almost purely CAR environment in "primordial" computational rhetoric to a dawning new age of CAR and CDR. It would be misleading to view this as a teleology, though. In part it's due to the vast increases in computing facilities available to researchers at every level, from the individual scholar and desktop computer to massive grid and cloud systems trawling through terabytes of data. Simply put, there are kinds of computation available now which were not worth speculating about ten years ago. Also, CDR is inherently a more technical and specialized field, so fewer rhetoric scholars will have delved into it; thus we can expect it to grow more slowly than CAR. But computational-domain rhetoric does not supersede computer-assisted rhetoric, and there is no end to the exciting work that can still be done in the latter area.

As part of my schematic invention strategy, I think this axis is useful in a couple of ways. In many cases, it's productive to shift the concept of a project from one pole to the other. There is a great deal of software (too much, I fear) that attempts to assist writers with matters of style, by identifying areas that might be problematic; this is a classic CDR application. What if such software actually rewrote the text to eliminate the (supposed) problems? Regardless of whether that's desirable (not so much for writing instructors, I suspect, but perhaps more so in the eyes of long-suffering readers), it certainly seems *interesting*. On the obverse, many projects originally conceived of as purely CAR prove too ambitious for current technologies and algorithms, and sometimes realigning them to more of a support role in conjunction with human rhetors could produce better overall results.[11] The other primary way I find this axis useful is simply as a reminder of the

[11]Sometimes an application which is strictly speaking computer-assisted can appear to the end user as if it were purely computational. A fascinating example is Matt Richardson's "Descriptive Camera," an art artifact which is a camera that produces a text description of the scene it photographs, rather than a picture. The Descriptive Camera works

breadth of computing. It's easy, when focusing on a project or a handful of projects, to unconsciously identify the kinds of work they perform as the natural way to employ ICT. CAR puts the computer in the role of a hand tool; CDR in that of an apprentice. Those are quite different relations to the machine, and they possess different requirements and possibilities.

Interactive / Automatic

Another axis for distinguishing computational-rhetorical applications is the model they present for interacting with users. Interactive computational rhetoric provides users with some sort of collection of tools that can be applied to the workpiece or input data; users observe the results and continue until they're satisfied. Interactive applications could take forms similar to, say, Photoshop or other common user-facing applications. Automatic computational-rhetorical applications, on the other hand, would work in a continuous or batch mode on input data, and users would query the results.

This might seem a rather artificial distinction (though of course they all are), and certainly it's often the case that a software package can be used in both interactive and automatic ways, so this axis can be a matter of how users employ CR software rather than what algorithms are employed or how they're implemented. (That said, there are constraints on what approaches are suitable for interactive use, because people demand a certain degree of responsiveness from their tools.[12]) Nonetheless it can be useful to con-

by uploading the image as a job for sale on Amazon's Mechanical Turk work-exchange site. A worker views the image and writes a description (for the princely sum of $1.25); the description is returned to the camera, which prints it out for the user.

[12]Precisely what "a certain degree" might mean in this context varies. While some usability studies have tried to determine what constitutes minimum responsiveness from interactive software—Jakob Nielsen's famous rule of thumb states that a one-second delay is the maximum to avoid users losing their train of thought, and ten seconds the maximum before users will switch to other tasks (Neilsen cites Miller [1968] and Card [1991])—usability is surely strongly predicated on user expectation, and different users have different expectations. In particular, and though I don't have data to support (or re-

sider how a particular CR project or application is situated on the automatic/interactive axis, because the two approaches have different advantages.

For computational rhetoric, interactive analysis and (probably even more so) synthesis has at least two striking advantages over purely automatic mechanisms. The first is that it can incorporate more human expertise. While natural language processing (NLP) has made tremendous strides over the past few decades, we are still very far away from a rigorous algorithmic understanding of language, much less how meaning is produced in the processes where it is used. Nor does any extant computing system have a model of the world that approaches the depth and richness of any human's. Interactive users can continuously interpret the data and guide the software using the vast amounts of auxiliary information, and the specialized neurological equipment for evaluating it, at their disposal. The second great benefit of interactive approaches is that they permit *ad hoc* experimentation: users can make intuitive leaps based on the results currently being produced, follow whims, turn away from unpromising avenues. Automatic software has to follow some sort of prescribed plan, even if that plan is too complex for people to predict its outcome. It can be introspective, in the sense of allowing preliminary results to influence further processing, but it cannot have flashes of insight or be distracted.

What automatic processing *can* do, and interactive processing cannot, is operate on huge amounts of data. Those may be precompiled corpora or incoming streams or a combination of the two. In itself this is a function that human rhetors simply cannot perform on the same scale, and so if there's any substantial chance that interesting results can be produced by working at that scale, automatic CR promises a new kind of rhetorical work. Indeed much of the work that might be considered computational rhetoric, a precursor to it, or a cognate endeavor does operate with big data—there's a widespread assump-

fute) this, my experience suggests that programmers are more prepared to tolerate longer delays in interactive applications than users who have less insight into what the machine is doing.

tion that such products will produce (are producing) new insights. Also, from an NLP point of view, operating on large amounts of data has algorithmic advantages: there are mathematical arguments that it lets models converge on more-accurate representations of language use. While there are potential theoretical problems with training language models on vast (and consequently noisy and irregular) corpora—a model that becomes sensitized to average language use will be correspondingly less sensitive to exceptional use—in practice this is helpful for the vast majority of applications. And finally, automatic algorithms sometimes identify correlations in their input data that are not obvious to human observers, precisely because they're not being guided by human judges. A computerized process that is ignorant of the meanings most plausibly produced by the text it's analyzing cannot make assumptions about those meanings, assumptions which could lead an interactive investigation to favor particular kinds of connections over other possibilities.[13]

Of course it's possible to have projects that employ both the interactive and automatic modes. One interesting possibility in this area is using interactive sessions for *supervised training*, the process of setting the parameters of a language (or other) model by providing it with sample data that has already been decoded and labeled by human judges. Traditionally most supervised training has been done in an automated, batch-processing fashion: humans mark up data, which is then fed *en mas* to the program. But it's also possible to have software observe users while they interactively explore data, and let the software build a model of what users do in response to particular kinds of data; then the

[13]One example would be the models produced by Latent Semantic Analysis (LSA), an NLP algorithm invented at Bell Labs in the 1980s (Deewester). LSA employs a fairly straightforward linear-algebraic technique (singular-value decomposition) to create a probabilistic model of a piece of text. What's interesting here is that this model will often group words that do not have any meaningful association in actual language use, because the algorithm does not include any information about the semantic values (meanings) of words. It derives the model entirely from a high-dimensional representation of how words are associated with one another. Thomo provides a short introduction that includes a useful diagram.

software could emulate the user while automatically processing large amounts of input.

Fixed-rule / Adaptive

It might also be useful to group computational-rhetoric implementations broadly by various technical criteria. This helps suggest what areas of computer science seem to offer promise, or may have been neglected for this sort of application. There are also of course many practical consequences to technical choices: what classes of algorithms make best use of the computing resources at hand, how they meet the performance criteria of the project, how much effort will be required to implement them (particularly when some suitable software is already available[14]), how comfortable the development team is with the theory behind a given algorithm. As an example of a technical criterion I offer the distinction between algorithms using fixed rules—a set of unchanging, predetermined functions patent in the program source—versus those that take an adaptive approach. A simple fixed-rule algorithm might search texts for specific words that often signal emotional content, for example, to identify likely appeals to pathos. Adaptive algorithms might range from statistical models such as Bayesian filtering (used by many email spam filters), to "learning" mechanisms such as neural networks, to highly stochastic approaches with competing machine-generated algorithms (using "genetic" and other simulated-evolution mechanisms).

The majority of NLP research these days seems to focus on adaptive algorithms. This is hardly surprising: we don't have anything even approaching a comprehensive set of fixed rules for how natural languages work, but we do have a great deal of language data, so it makes sense to turn adaptive algorithms loose on the data and let them construct their own models of it. Historically, though, some fixed-rule systems have done quite well in

[14]So, for example, the existence of the well-known OpenNLP open-source project, which implements the Maximum Entropy Markov Model algorithm, makes it tempting to select Maximum Entropy Markov Models as the language-modeling mechanism.

limited domains. Weizenbaum's famous ELIZA program was very successful at convincing users of the day that they were communicating with a human interlocutor, when in fact they were facing a program that simply implemented a fairly short set of rules for imitating simple Rogerian therapy (which consists of turning the patient's responses into questions for amplification, while maintaining a frame of empathy and approval). In contemporary computing, one perhaps surprisingly productive area of work has been in simply using large data sets to solve NLP problems. The most prominent example is Google's language-translation application, which searches web corpora of translated documents to try to map phrases from the source language to the target—basically one quite straightforward fixed rule ("search for this text") applied to a vast database.

On the other hand, it's not hard to see why adaptive algorithms are so popular with researchers. For one thing, there are so many choices: Bayesian filters, Hidden Markov Models (HMMs), Maximum Entropy Markov Models (MEMMs), Support Vector Machines (SVMs), neural networks, and so on. While in a theoretical sense many of these are equivalent, [15] they lend themselves to different kinds of problem statements. For example, if the project can be decomposed, in a straightforward way, into a *feature vector* (a list of attributes that a given input may or may not have: "does this phrase contain a gerund?", "is this phrase more than three words long?"), then a binary-classifier algorithm like MEMM or SVM is amenable to the task. If you need to sort a lot of inputs quickly into categories using a probabilistic model, a simple Bayesian filter will likely get the job done. And adaptive algorithms are intrinsically interesting (at least for those who are interested in that sort of thing), in their abstractions of relationships and formalizations of probability

[15] That is, given an algorithm A of one class and an algorithm B of another class, where the two classes have equal discriminatory power, and the two algorithms have the same information available to them, then it should be possible to find a homomorphism mapping A to B. The underlying computing mechanism sets an upper bound on the computational complexity of any algorithm that can be implemented on a computing device, and a number of the algorithms that are popular these days can be shown to be completely general within those limits.

distributions and clever little mathematical lemmas. Perhaps most importantly, though, adaptive algorithms avoid the danger of early interpretation: while a fixed-rule approach embodies assumptions about what the data expresses and how it's expressed, an adaptive one derives the interpretation from the structure of the data as it's processed, at least to some extent. In this sense adaptive approaches are more open to surprise.

For computational rhetoric, both fixed rules and adaptive models produce interesting results. One advantage of fixed rules is that they can be quite easy to implement, so they're useful for explorations, prototypes, and small-scale projects (for example in the classroom). Of the four of my own class projects that I discuss in the next chapter, three are fixed-rule programs. Fixed rules also may be easier to implement where computing resources are limited, which is why they appeared in much of the early consumer NLP software such as grammar checkers; so they may be useful for CR smartphone apps and the like. And certainly over its long disciplinary history rhetoric has seen an abundance of rules offered by its proponents—there's no small amount of material to be mined here for computational heuristics.[16] On the other hand, we have the tremendous amount of work being done with adaptive algorithms, much of which is on the cusp of what would traditionally be considered "rhetorical" analysis—if it is not there already. It's important that the field of rhetoric include a cohort of scholars who are engaged in that work and have a sophisticated understanding of such algorithms.

Adjunct schema: disciplinary dichotomies

The four dichotomies above define the theoretical schema I employ as an invention strategy for possible directions in computational rhetoric. In the next chapter, however, I look at some existing projects which represent computational rhetoric in practice or work relevant

[16]Thus we have, for example, the interesting branch of NLP called Rhetorical Structure Theory (RST), an attempt to derive a language model from rhetorical theory. I'll talk more about RST in the next chapter.

to it. There is a wide array of such work, and though I only have space to consider a handful of examples, I found it useful to partition them into four categories determined by two additional dichotomies.

So in reviewing possible contributions to my budding conception of computational rhetoric, I distinguished on the one hand between those that are explicitly located in the field of rhetoric and those that locate themselves in cognate fields such as linguistics; and on the other hand between those that explicitly refer to the sorts of computation I identified in my first chapter, and those that seem to me applicable to such computation but do not refer to it directly. That gives me the following four disciplinary categories: computational rhetoric *per se*, computational theories in cognate fields, rhetorical research applicable to computational rhetoric, and work in other fields where I can argue for a latent relevance to computational rhetoric.

Though we might be wary of reinscribing disciplinary boundaries in a field which is explicitly interdisciplinary, I would argue for two advantages to breaking my examples up into these disciplinary categories at this stage. The first, which applies to any of my dichotomies, is that it makes it easier to recall a specific example or locate it for reference. We are trained to think in terms of disciplines, so associating examples with disciplines provides them with additional mnemonic markers. The other advantage might be described as a negative one. I hope that explicitly labeling work from fields outside rhetoric or computation as such will serve as a reminder to avoid naive "tourism"—the too-easy habit of borrowing ideas generated within and in reference to other intellectual contexts, and lifting them out of those contexts in a fashion that strips them of critical nuance. Distinguishing a project as of interest to rhetoric but not *of* rhetoric reminds us that it should not simply be taken up as a lens for rhetorical critique.

Employing the schema

That, then, is my little schema for computational rhetoric: four dichotomies with which to describe the possibilities of CR. What might someone do with it?

I want to emphasize once again that my purpose here is invention: I want to help open the conceptual space, not constrain it. My suggestions at this point are generative rather than specific. In the next chapter I do discuss a number of actual applications in computational rhetoric, but that sort of survey is a different invention technique than the one I'm developing here. Instead I'll suggest various protocols for employing the schema in the inventive process. These protocols are similar to some classical invention strategies (such as "compare and contrast") or to the regulations of simple genres like the five-paragraph essay—they provide a framework in which a CR researcher, or group of researchers,[17] can arrange, combine, and elicit ideas.

For inventing with the schema I can suggest a number of protocols; no doubt my readers could come up with others. Here is one: identify an existing or proposed project that has a computational-rhetorical bent. Analyze it in terms of the schema, positioning it on each of the axes. Then invert one of the terms (much the maneuver that enjoyed some popularity in literary criticism as poststructuralism began to penetrate it). As an example, it's common for search features (such as web search engines) to return results sorted by "relevance"—a quasi-rhetorical metric. Of course it's not always clear to the user how this relevance is measured (indeed it's usually a black box), and users often adjust their searches to try to move results *they* would find more relevant to the front. I'd analyze such a search function as analytic, computer-assisted, automatic, and generally

[17] I think some of these protocols are particularly well-suited to group invention, and in general I believe that the demands of a fully-realized field of computational rhetoric (particularly its combination of technical specialties that historically have belonged to different groups and training regimens) will require extensive collaborative work, much of it cross-disciplinary. But of course individual scholars can also employ them to interesting ends.

fixed-rule. Note that while the overall process (user enters search terms, examines results, edits search terms) is interactive, the search function itself, and particularly the ranking by "relevance," is an automatic process; the user isn't involved in that ranking. Now invert the automatic/interactive term and consider how the relevance-ranking might be made interactive, letting the user help the search function identify what results are useful. Perhaps the user interface could let the user drag results to change their ranking, and recompute the ranking after each move made by the user, adjusting the parameters of the ranking process based on those moves.

A protocol that does not start with an existing solution, only a desired result, is to pick a position in the schema (at random, or intuitively, or based on some other criteria such as existing software capabilities), and then try to design an approach around it. Alternatively, create a grid from all sixteen possible combinations of the schema, then consider what a solution might look like at each point. This sort of procedure may seem rather forced, but as, I think, with many invention strategies, its very artificiality often leads to surprising results—not infrequently because of resistance from participants, or surprising insights generated from the cognitive dissonance of trying to force a solution to a problem using incongruous means.[18]

Another: Take inspiration from theoretical discussions of computational rhetoric and its cognate fields, or from arguments about CR-related technologies in the technical or popular press. Analyze these arguments in terms of the schema, to see what portions of the space defined by the schema are covered by a given argument; for example, does the argument assume the computational domain? And if so, what happens to it when transposed into the realm of computer-assisted rhetoric? Or, starting from the side of practice rather than theory, examine a CR application and attempt to map it into a

[18]A moment's reflection suggests that activities of this last sort—trying to solve problems using a skewed or artificially restricted toolset—can engage our best faculties. After all, it's the essence of many of our entertainments: games, sports, puzzles, and so on.

different part of the schema. An example might be recommendation systems used by online merchants (famously by Amazon and Netflix), web-advertisement brokers (most notably Google), and the like. Imagine such a system that modeled the intended audience of a piece of writing and prompted the author during the invention process: "this audience responds strongly to arguments regarding long-term environmental costs" or "this audience rates personal anecdote highly." And this example works from the theoretical side as well, since much has been written from any number of perspectives on recommendation systems, due to their importance to online commerce. If online advertisement selection based on data harvested from user behavior is an intrusion on privacy, as some have claimed, is using such data and algorithms to fine-tune an argument even worse? Or is it simply a convenience to assist a rhetor in performing a traditional rhetorical task?

Invention protocols can be ornate affairs of multiple steps and restrictions—sometimes working within constraints provides a productive focus—or minimalist, like many of the Aristotelian *topoi*. Using my schema for invention could be as simple as adopting its terminology for a nomenclature, or sketching it on a whiteboard for a brainstorming session. My point here is really that the schema is not so much an invention strategy as a ground on which invention strategies can be constructed, by augmenting it with protocols. In a sense, it's material for inventing invention strategies.

Moving on

The schema is the most specific, complex, realized, and coherent invention strategy I will present in this thesis. And though it has a certain brute clumsiness in its insistence on one term or another, though it's terribly general and unsophisticated, I find it quite useful, along the lines I've just described. Nonetheless, having described it I feel compelled to touch briefly on how to improve it or move beyond it completely.

Certainly the schema could be extended, for example by adding more axes. Consider

the distinction often drawn in NLP research between "trained" algorithms that require bootstrapping with sample data annotated by human experts, versus "untrained" ones that can be turned loose on real-world data with no preparation. A more subtle potential axis would be between what we might call "mechanical" algorithms, those that operate on their input simply as a signal to be processed, and "knowledgeable" ones which can draw upon representations of such contextual domains as vocabulary, semantics, conventions of language, or the world described by the input data. Or the structure of the schema could be broadened, taking it beyond dichotomies to n-term sets; or some ambitious researcher could attempt to build it out to a complete schema, with some option for describing any possible manifestation of computational rhetoric. Then again it could be extended in its use, for example by opening it up to enhancement by a large audience of users, perhaps turning it into a user-extensible taxonomy ("folksonomy").

And of course the proposal of a schema implies its converse, the turn away from structured conceptualizations and the attempt to partition knowledge. This is the "post" move (poststructuralism, postcolonialism)—which is also a "pre" move and an "else" move, responding to ways of thinking that were undervalued by the form of scientism that dominated European intellectual activity during the Enlightenment and shaped modern structuralism. Thinking schematically should always remind us to then rethink in other modes, employing association and narration and all our other faculties. No invention strategy will produce all possible stories, and getting the full benefit of any strategy eventually requires giving it up.

Chapter 4

Computational Rhetoric in Practice

Much of what might be considered applied computational rhetoric, or akin to it, or work that could be put to computational-rhetorical use, has been done outside the academic discipline of rhetoric itself. Of course there are also numerous projects which are computational and are explicitly labeled as rhetoric, or associated with the discipline. In this chapter, I briefly review some from both categories (those we might read as potentially CR and the explicitly rhetorical), as an illustration of the concepts of the preceding chapters, as a map of some of the extant field, and as another invention strategy: extrapolating from examples.[1] I will look first at four prototype projects I contributed to; these are useful not because they are in any sense major contributions, but as demonstrations of how easy it is to begin working in CR, and of course as a contributor I understand something of the history of invention in these projects. Then I look at a handful of other academic work, some from rhetoric and some from cognate fields, and finally offer a few remarks on the

[1] It's worth recalling again the dangers of coopting outside work as "rhetoric under another name," and of seeing everything through the CR lens. I don't think it's necessary to demonstrate how a great many projects only related tangentially at best to computational rhetoric, as I've defined it, could be construed as CR with a bit of rationalization. In a review like the one I offer in this chapter, projects might range from those explicitly associated with rhetoric to ones coming from related fields, and from the explicitly computational (*i.e.* software-based, or including algorithms, etc) to merely amenable to automation.

use of (something like) computational rhetoric in industry.

Four prototypes

In the course of my classwork for the MA in Digital Rhetoric and Professional Writing at Michigan State University, I was a primary contributor on four prototype projects that I would describe as computational rhetoric. These are academic, but not research projects proper; they were done in response to course assignments and are demonstrations of concepts rather than tools suitable for real-world application. They do demonstrate, however, that the hardware and software resources necessary for experimenting with CR are readily at hand, at least for problems of small or moderate size.[2] I use these partly to show that CR demonstrations can be bootstrapped quickly and with relatively little effort (and I mean "show" literally—all four were demonstrated as part of various conference presentations), and also because I'm a privileged reader of these projects and can speak to conception and intent.[3]

In 2007–2008, I experimented with a simple approach for ranking the ethos of contributors to an email list, resulting in a small software application I called the Ethos Estimator.

[2]A small digression. There is much talk these days, in academia, industry, and the popular media, about "data science" and big data, and some of the work that I refer to in various parts of the thesis—sentiment analysis, for example—is often associated with big data, as I've noted previously. That might lead some readers to wonder at what point a computational-rhetorical project becomes a "big data" project and requires ICT resources beyond what's available to the individual researcher. Today, as a rule of thumb, I think it's fair to say that 100 GB of data is not "big": it fits on the hard drive of a personal computer, can be easily manipulated with free software, and so forth. Even static databases in the multiple-terabyte (or tebibyte, if you prefer the new SI prefixes for power-of-two values) range are considered quite normal and tractable in the industry. A true big-data project involves either vast amounts of static data, or streams of continually-arriving data that are too voluminous to process by conventional means.

[3]By "intent" I mean my intent in developing the technology, insofar as I understand it. As an anti-intentionalist I do not impute any intent to the work itself, or mean to suggest that other observers might not construct different meaning from it.

The Ethos Estimator is built on top of another prototype, Textmill, which I developed as an experimental system for helping researchers create their own text-processing modules with relatively little effort. Textmill provides some infrastructure for plug-in text processing. Both Textmill and Ethos Estimator are written in Javascript, in part because it's a relatively well-known, convenient, and widely-available language, and in part so they can process text on web pages, which are an interesting corpus for analysis (Wojcik "Estimating Ethos" and "TextMill and Estimating Ethos"). The web-page visual-rhetoric project I discuss below has similar motivation.

Ethos Estimator is a very simple program for heuristically ranking participants in an email list, particularly a list that is mostly used for posing and responding to questions. (The input I used for my demonstration was an excerpt from the archives of the CVSNT-Users list, which handled queries from users of the CVSNT software package.) The algorithm it uses is trivial: for every email sender (participant) in the corpus, it increases its estimate of the sender's ethos when that sender responds to another message, on the grounds that on a technical Q&A list, experts are most likely to be the most common respondents. Despite its simplicity, the algorithm did select as its top authorities the same contributors who would likely be chosen by human judges, thanks in large part to the consistency (signal strength) of the input; but the real value of the project is in demonstrating how easy it is to try out this sort of simple processing on easily-available data sources and use the results to speculate about possible CR processing.[4]

In terms of my schema, Ethos Estimator is analytic, computational-domain, automatic, and fixed-rule. The Textmill infrastructure, though, could in principle be used in adaptive-algorithm projects, and with some effort modified to support interactive and synthetic applications. Again, as a prototype, it's unlikely anyone would take it up as it is to try to build something more ambitious, but some of the ideas that it embodies are

[4]For a far more sophisticated analysis of reputation and ethos in a technical mailing list see Ducheneaut.

worth revisiting. It informed the web visual rhetoric project I discuss a bit further down.

How were Textmill and the Ethos Estimator invented? In 2007, as part of an independent study with the Michigan State University Writing in Digital Environments (WIDE) Research Center, Bill Hart-Davidson suggested I investigate the possibility of estimating ethos in student peer responses to student writing, as a possible enhancement to the Eli writing environment.[5] Within Eli there is metadata—authors' responses to the reviews they receive, for example—which provides a strong signal for estimating each reviewer's ethos. However, examining the problem soon led me to consider the more general problem of ethos estimation, and rather than work on a solution specifically for Eli, I came to focus on how to build a more general framework for experimenting with textual analysis of web-based documents. In schematic terms, this was a shift in my invention process from fixed-rule to adaptive, and a move from the automatic end of the interactive/automatic axis toward the middle—from a specific set of requirements to a broader conception of the problem. That led to the Textmill prototype and only back around to Ethos Estimator when I needed something to exercise and demonstrate Textmill. And because my proposal for the Computers & Writing conference was accepted a few months into the project, I needed a quick and clear way to demonstrate the potential of my approach; that dictated the form of the Ethos Estimator as it currently exists.

Around the time that the Textmill / Ethos Estimator project was drawing to a close, Kristen Flory and I embarked on a small project, under the guidance of Danielle Devoss, to study some of the aspects of visual rhetoric in the web pages of candidates for the 2008 U.S. Presidential election. We were looking specifically at typography (Kristen's

[5]Eli, a web-based application for coordinating and assessing writing review, provides writers, typically students in writing classes, with an environment in which they can read and review one another's writing, and then provide feedback on the reviews they receive in turn, so they can become better reviewers as well as better writers. The Eli developers were interested in seeing whether a computational estimate of the ethos of student reviewers would provide useful information about their success in assisting their peers.

focus) and color (mine), and to supplement our theoretical study of those elements and empirical analysis of the sites as they existed, I created some Javascript scripts that could be used to modify the fonts and colors used on the campaign web pages.[6] These allowed us to demonstrate, in a literally graphic fashion, how the rhetorical impact of these web pages (which presumably had been carefully designed by a team of professional graphic-design, communication, and marketing experts) could be radically altered—and generally diminished—by modifying their font and color choices. As with Ethos Estimator, most of the value of these scripts was as a prototype proof-of-concept, but we also felt they had some pedagogical value, as students could easily use them to experiment with existing web pages. (We had some expressions of interest from instructors; whether any went on to use the scripts in the classroom I can't say.)

Returning to the schema, these scripts are analytic (though they can be used to experiment for creative purposes, which would slide them back toward the synthetic end), computer-assisted, interactive, and fixed. They grew out of simpler scripts which were automatic rather than interactive and purely analytic—scripts which simply displayed information about the font and color choices on the pages into which they were injected. The idea for such scripts arose in discussions among Kristen, Danielle, and me, where initial suggestions of analyzing the campaign sites led to speculations about the possibility of performing some aspects of that analysis programmatically. As Kristen and I experimented with the scripts, we agreed it would be far more interesting to make them interactive and have them actually modify the pages. It was in recalling moves such as

[6]These scripts—which we named Type Reset and Color Coordinator—were injected into the pages as they were loaded by the browser using the Greasemonkey add-on for Firefox. The scripts and explanatory material can be found in Wojcik and Flory, "Visual Rhetoric Greasemonkey Scripts." We also maintained archives of the sites for each of the major campaigns, downloaded periodically with WinHTTrack, as they changed frequently and we wanted to be able to refer to specific incarnations in our discussions. The project was presented at the Watson and CCCC conferences (Wojcik and Flory "Candidates, Color, and Type").

this during the development of my simple computational-rhetoric projects that I began to feel a schema like the one I've proposed here was a useful invention technique: it can quickly provide a set of possible pathways down which a project might develop.

The third classroom project in computational rhetoric I contributed to came the following year, in the fall of 2009. I take no credit for the genesis of this project, Speech Made Visible, which was conceived by my colleague Matt Penniman.[7] I joined the problem after hearing Matt's presentation of the problem he was interested in addressing: graphically representing prosodic effects of speech, such as pitch and stress, in text. Matt was familiar with some of the notation systems that had historically been used for this purpose, for example to mark chants, but felt that the typographic advantages of computer-rendered text opened the possibility for a more "intuitive" display. As the project developed, we discovered it was also quite possible to have the computer perform simple prosodic analysis on recorded speech, and then (with some success, and assistance from the user) match the recording to a transcript, which it could then render in a style that made the prosodic changes visible. For example, changes in the speaker's loudness might be rendered using various degrees of weight (boldface), and changes in pitch by shifting words up and down from the baseline. The current incarnation of the Speech Made Visible software lets the user experiment with different style options for various prosodic dimensions.

SMV is basically an analytic project, but it suggests the possibility of a synthetic one, in the ancient and nowadays somewhat unappreciated rhetorical domain of oratory: the software could be used by a speaker practicing a speech to coordinate prosodic emphases with logical and emotional ones, for example. Potentially the software could even perform rhetorical analysis on the text and make suggestions for oratorical flourish. It is

[7]Joining us in the project was fellow student Chris Huang, who did not work on the software but assisted with discussion, design, and presentation, and served primarily as the documentor of the effort. The project was completed under the guidance of Bump Halbritter. Matt and I presented SMV at the Edges conference (Penniman and Wojcik "Speech Made Visible"). The SMV source code is available from Sourceforge. Matt and I also discuss SMV in "Creating Complex Web Applications."

computer-assisted rhetoric, from the user's point of view, but the underlying prosody-analysis technology is computational, albeit not specifically rhetorical.[8] It is largely interactive, though the initial analysis is entirely automatic; and the rendering phase has some adaptive aspect, though on the whole (as with most prototypes) the processing rules are fixed.

The invention of Speech Made Visible was strongly determined by three factors. One was Matt's initial concept, which motivated the entire project and gave us a specific goal (and thus made this project more focused than either of the previous two). During the initial exploratory phase, our work was influenced primarily by theory, in particular by our attempts to connect rhetorical theory with the theory and empirical study of prosody, from traditional perspectives of aesthetics and social communication and more modern ones arising from linguistics, information theory, and the like. As we began to develop software, the dominating factor became what tools were readily available and sufficiently easy to use for us to construct a useful demonstration in the time available.

The final project I'll describe here was also inspired by the Eli system that prompted the Ethos Estimator. It applies sentiment analysis, a technique which has become very popular in industry for such purposes as gauging customer reaction to a product by analyzing online reviews, to peer reviews in Eli. In a typical Eli workflow, authors (generally students) submit pieces of writing, which are reviewed (generally by student peers). Then the authors respond to the reviews in various ways, for example by rating the helpfulness of the reviews and adopting suggestions from them for their revision plans. Part of the value added by the Eli system is that it maintains a "helpfulness" metric for reviewers, based on how much reviewing work they do and how their reviews are received by authors. In discussions with Eli developers Bill Hart-Davidson and Mike McLeod, we decided it

[8]Prosodic analysis of speech recordings is performed by a custom script, written by me, for the Praat open-source speech-analysis software. It tracks changes in the fundamental frequency f_0 and the instantaneous impulse (loudness) of the waveform, and uses pauses to guess where word divisions occur.

would be interesting to perform sentiment analysis on the text portions of reviews, to see how review sentiment correlated to other variables such as the numeric score also given by the reviewer to the piece being reviewed (*e.g.* was a low score typically accompanied by a textual review with negative sentiment?) and to the author's response (*e.g.* were authors more likely to reject reviews with negative sentiment?). This coincided with a course I was taking in 2011 in natural language processing with Joyce Chai, so it became the course project (some related material can be found at Wojcik "CSE 842 Materials").

To some extent, the Eli sentiment analysis project unfolded like Ethos Estimator, as primarily an exploration of technology—specifically IBM's UIMA framework for processing unstructured data, the OpenNLP package's English-parsing models, and my own implementations of three algorithms for sentiment analysis. (I also created some *ad hoc* tools to extract and transform the Eli data into a form more amenable for my analysis, as well as post-process reporting tools to perform simple statistical analysis of the results.) Unlike Ethos Estimator, however, this project did work with actual Eli data; it was also developed more closely with the Eli team.

The Eli sentiment analysis prototype is computational rhetoric as a newcomer to the field might imagine it: analytic, automatic computational-domain processing. There's some affordance for interactivity in setting operating parameters, and the process can be run manually as well as part of batch or stream processing; and while the models are fixed during processing (they do not dynamically adapt to the data), they can be tuned using new training data sets. And this software *looks* like computational rhetoric when it's demonstrated: it runs under the Eclipse IDE (Interactive Development Environment) with all manner of esoteric technical controls in its user interface, and the output is files of numbers. While the analysis it performs is relatively superficial and far from the sophistication of Rhetorical Structure Theory or the work of Chrysanne DiMarco's group (both discussed below), it has a certain computational heft which, while it is no doubt intimidating to some rhetoric scholars more accustomed to our field's usual place in the

humanities, suggests some of the exciting potential of computational rhetoric. Simplistic as it is, it's performing mathematical analysis of a rhetorical aspect of a corpus of texts, with a view toward providing a useful metric to improve rhetorical effectivity. It's also worth noting that even on a typical laptop computer it can perform this analysis more or less in real time—suggesting that eventually a system like this could provide rhetorical hints to a writer during the composition process.

Computational rhetoric in academia

I have already noted that I'm far from the first to use the term "computational rhetoric." That specific phrase goes back at least as far as 1991, when Marzena Makuta-Gihlk used it in her MA thesis. Subsequently it has appeared in work done by Makuta, DiMarco (*e.g.* "Goal-Based Grammar"), and colleagues at the University of Waterloo (1993 to present); by Floriana Grasso (2002); and likely by others, before I began using it in 2008.[9] Before and besides such usage there are many academic projects which apply formal and computational methods to rhetoric and rhetorical problems. In this section I'll review a few examples. Since these are not projects I worked on, I can't comment on the invention processes that led to them; I'm presenting them as an invention strategy in itself, to promote the invention of new CR technologies by considering what has already been done.

Though he has not, to my knowledge, used the term "computational rhetoric," Hugh Burns has done work that is comfortably within the space sketched by my schema since his doctoral dissertation work on the use of computers for rhetorical invention (*Stimulating Rhetorical Invention*): "teaching invention with computers," as Burns put it in an

[9]The phrase "computational rhetoric" was also suggested by Alistair Cockburn in 1995, but Cockburn means something quite different by the term: the rhetoric of software itself, as written by programmers.

interview (Trauman).[10] Or in other words, from the same interview: "I wondered what it might be like for a computer to play the role of Socrates and walk along with a student engaged in writing a paper." Burns' work in this period is particularly notable for being interactive at a time when the interactive use of computers was relatively rare,[11] and for its explicit nature as a tool for synthetic computer-assisted rhetoric—an instrument to assist the rhetor in rhetorical practice. It is difficult not to feel some regret (particularly when struggling with a piece of writing) that this sort of application, where the machine engages with the writer and offers rhetorical heuristics at useful moments, has not seen more active development in the ensuing decades. Considering the substantial improvements in natural-language processing algorithms and vast increases in the ICT capabilities available to even the most casual computer users, writing might have by now become an essentially different activity from what it was even half a century ago, perhaps closer to some of the plastic arts where many sorts of tools are readily at hand to shape the final output. (That said, it's clear that this sort of facility requires a light touch and careful design, as interrupting the writer is rarely welcome. One need only look at the considerable hostility that greeted Microsoft's "Office Assistant" feature when it was introduced in 1997. The despised "Clippy" was discontinued ten years later, apparently unmourned.[12])

Though not all of the promise of Burns' work, or indeed of many experiments with

[10]Tangentially, it's also interesting to note that Burns has offered suggestions for inventing new ways of applying computing technology to composition and rhetoric, much as I'm trying to do here, for example in "Four dimensions of significance."

[11]In 1979 the personal-computer industry was beginning to appear. S-100 systems such as the MITS Altair 8800 had been available since 1975; the Apple I appeared a year later, and by 1979 a number of models such as the Apple II, Commodore PET, and Tandy TRS-80 Model I were commercially available. Even prior to the PC boom, interactive use of larger computer systems through time-sharing had become relatively common, as the model popularized by MIT's CTSS was adopted by a range of mainframe and minicomputer operating systems. However, the idea of using a computer interactively to assist in a task not normally conceived of as computational was still quite unusual, computers not yet having come into their own as everyday appliances.

[12]Hesitant though I am to cite Wikipedia, for this sort of popular-cultural phenomenon it's as good a source as any ("Office Assistant").

writing-assistance tools, has been realized, the affordances of ever-growing computing resources have proven an irresistible lure for those interested in augmenting the writing process. Consider, for example, Bradley Rhodes' "Remembrance Agent" (RA), a mid-1990s open-source add-on for the Emacs text editor. RA combined an indexing and content-retrieval system with an "unobtrusive" (Rhodes' term), continuously-updated window in the editor. The indexer ran nightly, indexing content from all of the user's files that were in any of several supported formats. During edit sessions, the RA agent continuously tracked the last several words typed by the user, found the best matches for that phrase in the index, and displayed the names of the associated documents in its window. At any point, the user could pause while writing to view the documents located by the retrieval system. By integrating continuous search into the writing process, RA sought to minimize the impedance between text generation, on the one hand, and supporting tasks such as reference and citation.

Of course in the years since the use of content-retrieval (in the form of web searches and the like) while writing has become ubiquitous, though rarely with the fluidity offered by RA even in its relatively primitive form. The potential to integrate continuous search into Internet-aware, continuously-parsing editor applications has not been forgotten, however. In 2007, Alex Chitu wrote about a new Google Labs project, Google Writer, which would suggest topics, titles, quotations, and other materials for blog posts, and even automatically generate content; he wrote that Writer "learns your style from ... previous articles." Chitu's post was an April Fools joke ("Google Writer"). The humor was tempered a bit by the fact that everything Chitu described was well within Google's capabilities, and three years later Google Labs turned the tables when they introduced Google Scribe. Scribe, since discontinued along with most Labs projects, was essentially an expanded version of the prompting system used by the Google search page: as the user entered text, Google would prompt with the closest match it could find. Chitu noted the similarity with his "Google Writer" parody but was enthusiastic about the service, writing that it "works

surprisingly well." Kelley Fiveash, writing for the IT industry journal *The Register*, was rather less complacent: "Google has created an atmosphere of McCarthyism and a postage stamp-sized image of what the hell the world will look like if us hacks use the firm's latest creepy tool, dubbed Scribe." Fiveash disliked the fact that Scribe could be used for fraudulent purposes—the obvious objection—but also its potential for more Google data-harvesting.[13] Nonetheless, RA and Scribe are both interesting examples of some possible directions for interactive, synthetic, computer-assisted computational rhetoric.[14]

Hugh Burns is a composition and rhetoric scholar, and his work can certainly be considered computational rhetoric arising from that discipline. Rhodes and and the Google Labs developers, on the other hand, are computer scientists first, turning their attention to enabling writing as a task for computing. There is a long tradition of approaching computational intervention into writing and rhetoric from other disciplines, and from computational linguistics and natural-language processing we have Rhetorical Structure Theory (RST), initially developed by Mann, Thompson, and Matthiessen beginning around 1983 (Mann and Taboada). Of course RST is interesting in this context in part simply because its name is an explicit acknowledgment of a role for rhetorical theory in the computational manipulation of language, but it is relevant to computational rhetoric both in theory and in practice. Rhetorical Structure Theory originally emerged from research into automatic text generation (a synthetic application of CR), but it is broadly applicable to analysis of existing text as well. RST describes, and attempts to identify, extra-sentential structures

[13]One interesting facet of Scribe, also noted by Fiveash, is that it could be enabled on any web page, simply by activating a "bookmarklet," much like the visual-rhetoric Greasemonkey scripts I mentioned above. Whatever the virtues or faults of Scribe, this mutability of web technology, and the ability for users to assert control over content and affordance in web-based documents and applications, certainly creates many opportunities for computational rhetoric.

[14]Readers may have noted that with these past few examples I haven't discussed the fixed-rule/adaptive axis of my schema. I feel it's not particularly interesting in these cases, as the essence of their operation, from the user's point of view, is probably not affected by it.

in text that create "coherence"—relationships in the text that justify the inclusion of its various statements. One such relation, for example, is "evidence," indicating one passage in the text (the "satellite") provides supporting evidence for a claim made in another passage (the "nucleus"). The catalog of relations includes many basic rhetorical devices: providing background, offering comparisons, explaining motivations, summarizing, and so on.

Without going into further theoretical details, I'll note three aspects of Rhetorical Structure Theory that I believe can be useful for inventing computational rhetoric. The first is that it is a theory of language that makes rhetorical maneuvers central. It provides a nomenclature, a notation system, and other tools; and it also provides evidence to support the argument that rhetoric is important in the operation of language. Second, as a theory that has received substantial attention it has accumulated a body of work that contains many useful ideas, and some computational tools, though I am unaware of any fully-automated tools capable of RST-style analysis of raw text. (Extant RST work I've examined employs human judges to identify relations, and then performs analyses from those.) Beyond these practical aspects, though, RST is important for CR because it emphasizes, and provides considerable theoretical and empirical arguments for, the idea that rhetorical relations themselves contain or produce meaning—that meaning is not solely inherent in the symbols and syntax of the text.

Floriana Grasso's 2002 paper "Towards Computational Rhetoric" notes that while there is a significant body of work analyzing the algorithmic extraction of logical argumentation from text, less attention has been paid to what Grasso calls "rhetorical arguments," defined as "heavily based on the audience's perception of the world, and concerned more with evaluative judgments than with establishing the truth or the falsity of a proposition." While many rhetoric scholars might find this definition of "rhetoric" excessively narrow, Grasso's focus does help to extend some of the NLP and computational-linguistics work relevant to computational rhetoric beyond techniques that in essence reduce texts to a

series of statements in propositional logic. As with RST, Grasso is interested here in developing a typology of rhetorical devices, in this case derived from the rhetorical theory of Perelman and Olbrects-Tyteca. While I don't find Grasso's project particularly interesting for my own purposes—partly for the dichotomy between rhetorical and propositional arguments, which I feel is unproductive, and partly because I am not persuaded by Perelman's theory of rhetoric[15]—it's valuable as a sustained attempt to move from a specific and complex rhetorical theory, to formalisms, to implementations of those formalisms for ICT. And, of course, it's of interest simply because of its use of the term "computational rhetoric" in the sense I'm employing, as a moment in the history of that idea.

No account of computational rhetoric in academia would be complete without some discussion of the work done over the past two decades at the University of Waterloo, by the Inkpot Natural Language Research Group, its forerunners, and affiliated researchers. These include some I've already mentioned such as Marzena Makuta (-Gihlk) and Chrysanne DiMarco, along with several other faculty members and graduate students.[16] I have already mentioned their work on goal-directed behavior in language use, which can certainly be considered a rhetorical approach, concerned as it is with discern-

[15]My objections to Perelman stem primarily from its account of language, which I believe relies on simplifications that distort the actual interpersonal processes of symbolic communication and meaning-production. Language in Perelmanian rhetoric appears to be observable, with little recognition of inaccessible psychological depths or the processes (which I believe are phenomenologically inaccessible) by which users of language produce discourse as audiences to themselves. Similarly, Perelman's insistence that argument begins with agreement between rhetor and audience strikes me as an unnecessary and implausible axiom. I am far more inclined to a Davidsonian-Rortian explanation of language as a series of exchanges between parties, understood by all participants as signifying *something* (probably never quite the same thing, but significance of some sort is the defining attribute of language), resulting in the production of meanings that, when communication is "successful," have similar pragmatic consequences. For Rorty's account of language, which he extrapolates from his understanding of Davidson's, see the first part of *Contingency, Irony, and Solidarity.*

[16]DiMarco maintains a web site that describes the group and its various projects at <https://cs.uwaterloo.ca/∼cdimarco/research/index.html>.

ing the motive of the speaker or writer. Under the auspices of the HealthDoc project, the group also worked on adapting the presentation of medical information to different audiences (*e.g.* DiMarco *et al.* "HealtDoc"), examining hedging and other cues in scientific citation (*e.g.* DiMarco and Mercer),[17] and other subjects of interest to CR. Their IN3SCAPE project focuses on identifying the implied pragmatic content of text, which DiMarco characterizes as "the 'hidden' meaning such as the attitudes of the writer toward her audience, the intentions being communicated, the intra-textual relationships between document objects, and so forth" ("The IN3SCAPE Project")—again, I think the broad applicability to computational rhetoric is clear.

Most of the participants in the various University of Waterloo computational-rhetoric projects find their disciplinary homes in computer science, mathematics, or linguistics, with some additional contributors from medicine and the sciences, likely lending domain expertise and motivation for specific projects, and a few from the humanities. It is, I think, fair to say that while I am approaching computational rhetoric from the rhetoric side (seeking to add computation to rhetoric), DiMarco and her colleagues come at it, broadly speaking, from that of computation (injecting rhetoric into computational linguistics). Some of their work may thus be a bit less accessible to those rhetoric scholars who have not studied computer science or natural language processing. But of course it is only to the benefit of computational rhetoric as a whole to see as many approaches as possible. These researchers have also made a significant contribution simply in promoting the term "computational rhetoric;" as with Grasso, their use lends it considerable credibility. DiMarco, for example, describes her research interests as "computational rhetoric and computational argumentation" ("Chrysanne DiMarco"); her colleague Randy Allen Harris in English lists "Computational Rhetoric" among his; and the phrase appears in various presentations and publications produced by researchers working there.

[17] This work might be considered another way of approaching the problem of measuring rhetorical ethos in text.

Computational rhetoric in industry

Earlier I alluded to the presence of something like computational rhetoric already at work in industry. Though it may not be called computational rhetoric, such application of computing technology to the rhetorical domain, not simply for research purposes but as an explicit attempt to improve profits (or satisfy other business demands, such as reducing risk, which is in effect an attempt to improve average profit over the long term), demands our attention. Throughout its history, the study of rhetoric has been informed by calls to attend to the public use of rhetoric for political purposes—the use of argumentation in the exercise of power—and often to examine its use in other spheres, such as the interpersonal, the legal, and the economic (all of which are of course intertwined with the political in any case). We would be deficient in our duties as rhetoricians if we did not examine the automation of rhetoric in commerce. Moreover, if someone will be developing and deploying rhetorical algorithms, rhetoric scholars ought to be participants, to demonstrate our relevance, bring theoretical rigor, keep (one hopes) ethical considerations in the fore, and not least reap some of the rewards.

For the first example it's useful to consider sentiment analysis, the area of research for my own "Estimating Sentiment" project mentioned previously. Sentiment analysis is a crude sort of rhetorical work; it's a very coarse metric for pathos, often in only one dimension ("does this text express positive or negative sentiment?"), indeed often reduced to a binary choice (positive or negative, with no sense of neutrality, ambivalence, or degree)—a single bit of rhetorical information. Nonetheless, it *is* rhetorical, and sentiment analysis has become very widespread indeed. In terms of my schema for computational rhetoric, sentiment analysis is analytic,[18] computational-domain, and automatic.[19] Often

[18]Though there seems to be a growing presence of sentiment *generation* algorithms—the seedy synthetic side of sentiment analysis—in use, automatically producing text that promotes the opinion of the wielder's choice. The Liu presentation cited below mentions this phenomenon.

[19]Often very much automatic. Given the amount of text subject to sentiment anal-

it employs fixed rules rather than more-sophisticated adaptive algorithms, simply because the tolerance for imprecision and error in many sentiment-measuring applications is high: a company wants a rough gauge of the general sentiment expressed about one of its products, for example, in which case misreading even a considerable fraction of the corpus will likely not alter the final result enough to influence whatever decisions might arise from it. I will not say much more about sentiment analysis here, as its rhetorical dimension is so narrow, but I believe it is part of the beginning of a turn toward rhetoric in the industrial processing of language.[20]

In industries where rhetoric plays a prominent role—or, more accurately, where the prominent role of rhetoric is generally acknowledged—there has been some interest in assisting rhetors with producing effective domain-specific arguments. Thus we have, for example, attempts to provide software to assist in public-policy debate and in legal argumentation (Rehg *et al.*, Verheij). These are tidy demonstrations of the commercial potential of synthetic computer-assisted rhetoric. On the analytic side, we see attempts to assist audiences looking to distinguish among arguments (or find grounds to counter them) with, for example, Indiana University's web site `truthy.indiana.edu`, which scanned for tweet patterns in an attempt to detect "Twitter-bombing" campaigns to promote weak arguments about political candidates and issues ("Truthy.indiana.edu").

There is also considerable interest in various industry segments in identifying preferable or unwanted expression. A computational analysis of popular films claims to have identified a formal attribute (the "$1/f$ fluctuation") that makes films more appealing, for example, and furthermore claims that some film genres are more suited to producing this pattern than others (Page, Herbert; based on research by James Cutting)—leading

ysis (one of the few genuine big-data problems in CR), there is considerable interest in minimizing the need for human interaction of whatever sort. See for example Brody and Elhadad, who offer a sentiment-analysis model which uses unsupervised learning, that is, does not need to be trained using evaluations produced by human judges.

[20]Liu is a good introduction to sentiment analysis. The "MPQA Resources" website has various materials useful for sentiment analysis.

to the depressing possibility that Hollywood's output might soon become even more homogenized and tiresome, guided by purely mechanical rules. (I have never claimed that computational rhetoric is not dangerous.) Conversely, the widespread enthusiasm for censorship in the ICT industry and its patrons in entertainment and government has led to any number of proposals for restricting "undesirable" expression. Apple, for example, applied for and was granted a US patent (Lee and Lee) to block incoming or outgoing text messages that contain "objectionable content", or remove said content from the message at transmission or reception (see "Jobsian fondle-slab"). These are all rhetorical applications insofar as they seek to shape discourse, or other modes of symbolic expression, toward or away from particular channels of persuasion.

While CR is only gradually being adopted in industry (unsurprisingly, given the normal lag between research and commercial implementation), I believe it can only become more widespread and prominent. Examples from industry are particularly useful in attempts to invent computational rhetoric, first because they answer to constraints (feasibility, profitability) that are not so present in academic research, and second because they are already to some extent consequential, and thus demand our attention.

Chapter 5

Arsis

There cannot be a conclusion to computational rhetoric. It is upon us, and having realized that our technologies of data, information, and knowledge are capable of surpassing the capabilities of humans (individually or even collectively) in many ways—though not yet, and perhaps not ever, of even approaching them in others—we cannot fail to continue to develop those technologies and apply them to rhetorical work, barring some cataclysmic failure of modern civilization.[1] So I will not attempt to offer a conclusion to this thesis. Instead I'll end by reversing course, suggesting a few of the many points from which other work could lift off from the small piece of ground I hope I've provided here.

Theoretical issues in computational rhetoric

The invention strategies I presented in the two preceding chapters can, I think, be considered fairly concrete and direct. Studying existing work for inspiration and following protocols to generate ideas are straightforward procedures with a clear teleology. A more abstract route for inventing computational rhetoric is to consider some of the theoretical

[1] Whether computational rhetoric could still be said to exist in the absence of its practitioners is a metaphysical question of some abstract interest. I'm not going to get into it.

problems raised by the very concept.

The Turing test and the Chinese room

It is interesting, and probably not coincidental, that two of the best-known arguments about the possibility of "strong AI" (the possibility of machines that "think like people," in some sense) are defined in terms of language use. The computing pioneer Alan Turing described his eponymous test roughly as follows: if a human judge, using only the exchange of text messages, is unable to distinguish between human and machine interlocutors, then the machine must be seen as thinking, because its replies contain all the discernible attributes of cognition. John Searle, on the other hand, rejected the strong-AI program (at least in the possibility of implementing it with a computer running a program) with a thought experiment generally referred to as the "Chinese room." Suppose a worker who is not literate in Chinese sits in a room. Into this room comes a stream of messages written in Chinese characters. The worker has a set of mechanical rules for manipulating these messages and generating responses, based on the shapes of the characters. Searle declares that this process is an accurate representation of any algorithmic natural language processing, but it does not involve any "understanding."

There are a number of problems with both experiments. Turing did not introduce the idea of the test as an actual practical evaluation for artificial intelligence (which in any event seemed a remote speculation in his era, which could not have anticipated the unprecedentedly rapid development of ICT technology); it was a philosophical position. That has not discouraged generations of computer scientists from holding Turing Test competitions, often with rather questionable methodology, some of which is implied in the description of the test itself (*e.g.* how many human judges are required before we declare the machine successful?). Robert French has recently suggested that we should discard the test, because while it has been both a useful intellectual exercise and a spur

for various sorts of interesting development, it is fatally flawed and actually not very interesting as a practical decision process.[2] Searle's Chinese Room argument, on the other hand, is an analytic-philosophical position (it might be paraphrased as "what do we mean when we say *thinking*") grounded on a subjective phenomenological evaluation (in effect it reduces to "that's not what I think I'm doing when I think"[3]). Searle cannot demonstrate, on the one hand, that his model represents any possible NLP algorithm, or on the other that it does not accurately represent the process of human thought at some level—more specifically, he cannot demonstrate that *interpretation* of language elements must inhere at the point of the manipulation of signifiers, rather than emerging at some other level of abstraction.

Perhaps more interesting is the difference in philosophical heritage. As I've already noted, I'd characterize the position advanced by Searle (an American) as close to some trends in English analytic philosophy, which often turned to questions like "what do we mean when we say x?". (Similarly-minded thinkers might include Ludwig Wittgenstein, Bertrand Russell, and Frank Ramsay.) His objection arises from a metaphysical epistemology: there is some undefined, perhaps ineffable, attribute of thought which distinguishes it from the appearance of thought. The British Turing, on the other hand, offers an experiment which smacks of the American pragmatic tradition (Charles S. Pearce, William

[2]One of French's critiques, for example, is that there are a great many trick questions that rely on facts such as obscure behaviors of human anatomy, which a machine could only answer by relying on a large database of information irrelevant to any useful AI program (75). In this sense the Turing Test is perverse—it creates an incentive to burden an AI system with unimportant data.

[3]I believe Searle has somewhere—in response to a critique of the Chinese Room—made a claim along these lines, but am presently unable to locate it. But he espouses similar phenomenological positions in the original "Minds, Brains, and Programs" essay, *e.g.* "ask oneself what it would be like if my mind actually worked on the principles that the theory says all minds work on," "it seems to me quite obvious in the example that I do not understand a word of the Chinese stories" (quotations from the online version of the article). And elsewhere: "On the outside it looks the same. On the inside I understand English and I don't understand Chinese" (Kreisler).

James, *et al.*). It ignores metaphysics and the more dubious aspects of epistemology in favor of a scientific procedure (that is, one which enforces a method intended to reduce the number of epistemological variables, such as the biases of observers), and it bases its results solely on the measurable physical attributes of the process—namely the statistical accuracy, or lack thereof, in the human judges' ability to distinguish the category of their interlocutors.

This tension has long appeared in various forms in the theory and practice of artificial intelligence research and development.[4] It is of course impossible to decide, since at least one half of the dichotomy is fundamentally untestable (like any metaphysical proposition). But it is useful for the theory of computational rhetoric. First, of course, it shows the centrality of language use in our thinking about our information technology and our interactions with it. Beyond that, it offers another axis to consider when generating or responding to theoretical arguments in CR. Regardless of whether a researcher is interested in the strong-AI program, any sufficiently inclusive theory of computational rhetoric must at some point grapple with the question of if and when a collection of rhetorical algorithms can be said to be doing rhetoric in a "human" manner.

The possibility of persuasive p-zombies

Though there are always liminal cases, in general it seems relatively easy to point to specific speech acts and discursive formations as attempts to advance or support a thesis; and so to identify these types of acts and forms as the material of argument. It's rather less certain whether they constitute an attempt at persuasion, since we can ask, in the

[4]Sometimes the forms it takes wander quite far afield. The noted mathematician Roger Penrose, for example, argued against strong AI on the basis of his belief that the human mind is inherently nondeterministic, through the use of quantum-mechanical effects. The appearance of nondeterministic "quantum" computers—still in their infancy and not practical for real-world applications, but real and under development—would seem to pose rather a problem for that theory, even if Penrose is correct about human cognition, which I must admit strikes me as a bit unlikely.

philosophy of rhetoric, if persuasion requires holding an opinion (whether sincerely or speculatively). That in turn leads us to philosophy of mind, and what we mean by *hold an opinion*.

In the philosophy of mind, *qualia* (singular *quale*) are mental conscious experiences of particular subjective states. The statement "I'm hungry" expresses the quale of perceived hunger, for example. Holding an opinion would appear to be a quale—at least if the opinion is consciously recognized, which presumably we hope, as rhetoric scholars, will happen while the rhetor is formulating and advancing a thesis.[5] A p-zombie is a hypothetical entity that lacks qualia but manifests the outwardly-detectable attributes of their presence; so a p-zombie might declare itself hungry without actually being able to experience the conscious mental subjective state of hunger.

Would, or might, a synthetic, computational-domain, automatic CR system be a p-zombie? That is, might such a system produce arguments that imply the system holds an opinion, even though the system is not actually capable of that state? This can be seen as a more sophisticated version of the Chinese-room argument. A mechanical procedure can have a *telos* (indeed it must, in the real world, courtesy of thermodynamics), but is this equivalent to having a motive? If such a system is (or might be) a p-zombie, does that diminish the force of the arguments it produces? Should it? For example, can *ethos*—either in Aristotle's sense of the "good speaker," or in a more general one of a reputation that endorses the source—attach to a p-zombie? What if a human rhetor is persuaded by an argument generated by such a system and endorses or repeats it—does that change the status of the argument? Does it become unethical at some point to treat a sufficiently able CR system as a p-zombie? (Now we're in an analogue of Turing's ar-

[5]Of course it is not only possible but commonplace—indeed this is the normal human mode of operation—to react to situations without consciously recalling the existing opinions we hold on relevant subjects. But I hope I am not too controversial if I suggest that a key aspect of the motivation for rhetoric itself is to bring such opinions to the surface so they can be examined, interrogated, and justified.

gument.) At some point, does this become a question of whether we can accurately say that a machine *believes* something? And to what extent do any of these questions matter?

There are any number of other issues that can be posed for theories of computational rhetoric. Can we resolve the tension between mechanical/pragmatic conceptions of rhetoric and subjective/humanist ones? If our rhetorical models are to some extent phenomenological, based on our perception of human cognition, how do we account (or do we account at all) for the possibility of the unconscious? How can we retain the insights of poststructuralism? And so forth. Here I only want to suggest—again as an invention strategy—how even concise theoretical objections can burgeon into substantial interrogations into the possibility and constitution of possible computational rhetorics.

Practical issues in computational rhetoric

I believe purely theoretical projects always have some value; at the very least, they help describe the space of our understanding and suggest directions for further investigation. Nonetheless, computational rhetoric seems bound to be a field where most of the effort is directed, immediately or eventually, toward application. Thus it's worth jotting down a few notes about practical issues in CR and how they might lead to interesting projects.

There are a handful of universal barriers to any sort of computation, and CR projects can certainly be expanded or scaled up to the point where they run into one or more of them. The first is computability. I expect most of my readers are familiar, at least in passing, with Turing's proof (the "Halting Problem") of non-computability and non-decidability: essentially a generalization of Gödel's proof of incompleteness, Turing's work demonstrates that there are categories of problems (in fact it is the majority of problems) that have the form of computable tasks but cannot in fact all be computed, and it's impossible to prove in general whether a given instance of such a problem can be computed.

One perhaps surprising corollary of the Halting Problem is how many seemingly-unrelated tasks are isomorphic to it, and thus undecidable and for specific instances non-computable. This includes many potential approaches to computational-rhetorical tasks, even such straightforward examples as iteratively refining a system's "understanding" of a piece of text.

The second barrier is computational complexity. The cost of employing an algorithm typically grows in some relation to the size of its input, and different sorts of algorithms experience different patterns of growth. For many problems, the best known algorithms grow so fast that problems quickly become intractable if the input is of significant length, and for most of these problems it's widely believed that there cannot be any more-efficient algorithms. (This is the famous $P \stackrel{?}{=} NP$ problem.) Thus some possible approaches to CR problems may simply prove intractable in practice for all but the smallest inputs.

A third barrier, quite simply, is the human capacity to model sufficient levels of abstraction. In physics, we have some good computational models for simple problems (*e.g.* Newtonian mechanics), surprisingly effective models for some much more difficult problems (*e.g.* various statistical-mechanics approximations), models for even harder problems which prove adequate to answer some questions and not so useful for others (now getting into the realms of quantum interactions, on one end of the scale, and cosmology on the other), and problems which are still intractable. Natural language processing faces similar difficulties: we have made good strides with many facets of low-level language processing, and have interesting results in extracting information from large corpora, but systems that can handle the entire "stack" from parsing to identifying concepts to approximating meaning are still very much in early stages—the successes of, say, IBM's Watson system (impressive though it may be) notwithstanding. (They are also very expensive, by any measure.)

Some thoughts on work to be done

This thesis is a series of ruminations on inventing computational rhetoric; everything here could be seen as a suggestion for work that might be done in the field. Here in the final stage of my lifting off, though, I can't resist tossing out a few more suggestions.

Computational rhetoric is a technical field; it involves often-difficult formal concepts and complex technologies derived from them. What can we do to make those more accessible to rhetoric scholars? How, on the one hand, might we approach teaching these technical subjects to rhetoricians who may not be entirely comfortable with the abstractions of mathematics, computer science, and computational linguistics, to say nothing of the practical matters of programming? And on the other hand, what can we do to make those technologies available to scholars who may have valuable ideas for computational rhetoric but not the time or inclination to acquire the technical capabilities? Can we create interactive toolkits or the like for computational-rhetorical experimentation?

I suspect it would be useful to distinguish computational rhetoric from its cognate fields. It is natural for, say, the practitioners of computational linguistics or natural language processing to wonder what rhetoric scholars bring to the table, and whether "computational rhetoric" is simply a pretty name slapped on a clumsy borrowing of work from other disciplines. We do not need a repeat of the "science wars." Conversely, we need more collaborative work between rhetoric and its cognate fields; CR is inherently interdisciplinary, and as Waterloo's Inkpot group and other research groups and centers have shown, teams with an assortment of expertise can produce the most interesting and useful results.

Finally, while natural language processing is the obvious starting point for computational rhetoric, we must wonder about the potential for non-textual CR. Earlier I described the work I did with Kristen Flory, experimenting with altering the fonts and colors on web pages; that was a simple form of para-textual and non-textual compu-

tational rhetoric. Surely there are entire worlds of computational visual rhetoric, the computational rhetoric of other media such as video and music, applying computational rhetoric to facial-expression and gesture recognition, and so on. There may be rhetorical domains which are forever closed to computation, but for now, at least, it is not clear what they could be.

WORKS CITED

WORKS CITED

[1] Abello, James, Peter Broadwell, and Timothy R. Tangherlini. "Computational Folkloristics." *Communications of the ACM (CACM)* 55.7 (2012): 60–70.

[2] Anderson, P[hillip] W[arren]. "More Is Different: Broken symmetry and the hierarchical structure of science." *Science* 177 (1972): 393–396. <http://www.physics.ohio-state.edu/~jay/880/moreisdifferent.pdf>, accessed 10 August 2013.

[3] Aristotle. "The Art of Rhetoric." Trans. Lee Honeycutt. "Book I – Chapter 2 : Aristotle's Rhetoric." <http://www2.iastate.edu/~honeyl/Rhetoric/rhet1-2.html>, accessed 6 January 2011.

[4] Brody, Samuel and Noemie Elhadad. "An Unsupervised Aspect-Sentiment Model for Online Reviews." *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the ACL*. Los Angeles, CA. 2010: 804–812.

[5] Burke, Kenneth. *A Grammar of Motives*. Berkeley: U of California P, 1969.

[6] Burns, Hugh. "Four dimensions of significance: Tradition, method, theory, originality," *Computers and Composition* 21 (2004): 5–13, <http://www.cws.illinois.edu/IPRHDigitalLiteracies/Burns.pdf>, accessed 23 July 2013.

[7] —. "Stimulating Rhetorical Invention in English Composition Through Computer-Assisted Instruction." U Texas, 1979. PhD dissertation.

[8] Chitu, Alex. "Google Scribe." *Google Operating System*. 7 September 2010. Blog. Updated November 2011 to announce the discontinuation of Scribe. <http://googlesystem.blogspot.com/2010/09/google-scribe.html>, accessed 23 July 2013.

[9] —. "Google Writer." *Google Operating System*. 31 March 2007. Blog. <http://googlesystem.blogspot.com/2007/03/google-writer.html>, accessed 23 July 2013.

[10] Cockburn, Alistair. "Growth of human factors in application development." 1995. <http://alistair.cockburn.us/Growth+of+human+factors+in+application+development/v/slim>, accessed 23 July 2013.

[11] Cohen, Sarah, James T. Hamilton, and Fred Turner. "Computational Journalism." *CACM* 54.10 (2011): 66–71.

[12] Council on Competitiveness. *Benchmarking Industrial Use of High Performance Computing for Innovation*. 2008. <http://www.compete.org/publications/detail/486/advance/>, accessed 2013-08-03.

[13] Deerwester, Scott C., *et al*. *Computer information retrieval using latent semantic structure*. Bell Communications Research, Inc., assignee. Patent 4,839,853. 1989.

[14] Denning, Peter J. "The Profession of IT: Beyond Computational Thinking." *CACM* 52.6 (2009): 28–30.

[15] DiMarco, Chrysanne. "Artificial Intelligence Research Group | Chrysanne DiMarco." <https://cs.uwaterloo.ca/~cdimarco/cdimarco.html>, accessed 23 July 2013.

[16] —. "Artificial Intelligence Research Group | Research." "The IN3SCAPE Project." <https://cs.uwaterloo.ca/~cdimarco/research/citation.html>, accessed 23 July 2013.

[17] DiMarco, Chrysanne *et al.* "A Goal-Based Grammar of Rhetoric." 1993. <http://www.aclweb.org/anthology-new/W/W93/W93-0205.pdf>), accessed 23 July 2013.

[18] DiMarco, Chrysanne, Graeme Hirst, Leo Wanner, and John Wilkinson. "HealthDoc: Customizing patient information and health education by medical condition and personal characteristics." Workshop on Artificial Intelligence in Patient Education. Glasgow, Scotland. August 1995. Presentation. <https://cs.uwaterloo.ca/~cdimarco/pdf/publications/Glasgow1995.pdf>, accessed 23 July 2013.

[19] DiMarco, Chrysanne and Robert E. Mercer. "Hedging in scientific articles as a means of classifying citations." *Working Notes of the American Association for Artificial Intelligence (AAAI) Spring Symposium on Exploring Attitude and Affect in Text: Theories and Applications*. Stanford University. March 2004. <https://cs.uwaterloo.ca/~cdimarco/pdf/publications/AAAISymp2004.pdf>, accessed 23 July 2013

[20] Ducheneaut, Nicholas. "Socialization in an Open Source Software Community: A Socio-Technical Analysis." *Computer Supported Cooperative Work* 14 (2005): 323–368.

[21] Edgecombe, Rodney Stenning. "Keats's use of 'Pip-Civilian': An Antedating for the OED Entry for 'Pip' N2.3." *Notes and Queries* 57 (2010): 213–214.

[22] Farrance, Rex. "Timeline: 50 Years of Hard Drives." *PCWorld*. <http://www.pcworld.com/article/127105/timeline_50_years_of_hard_drives.html>, accessed 6 August 2013.

[23] Feynman, Richard. "*Surely You're Joking, Mr. Feynman!": Adventures of a Curious Character*. 1985. Ed. Edward Hutchings. New York: Norton, 1997.

[24] Fish, Stanley. "Truth but No Consequences: Why Philosophy Doesn't Matter." *Critical Inquiry* 29 (2003): 389–417.

[25] Fiveash, Kelly. "Google squirrels into human brains with Scribe experiment." *The Register*. 8 September 2010. <http://www.theregister.co.uk/2010/09/08/google_scribe/>, accessed 23 July 2013.

[26] French, Robert M. "Moving Beyond the Turing Test." *CACM* 55.12 (2012) 74–77.

[27] Frye, Northrop. *Anatomy of Criticism: Four Essays*. Princeton, NJ: Princeton UP, 1957.

[28] Grasso, Floriana. "Towards Computational Rhetoric." *Informal Logic* 22 (2002): 195-229. <http://www.phaenex.uwindsor.ca/ojs/leddy/index.php/informal_logic/article/view/2589/2030>, accessed 23 July 2013, but note that the link currently leads only to a truncated copy of the article that ends at page 204.

[29] Greimas, Algirdas J. *Structural Semantics: An Attempt at a Method*. Trans. Daniele McDowell, Ronald Schleifer, and Alan Velie. Lincoln: U of Nebraska P, 1983.

[30] Hagen, Joel B. "The Origins of Bioinformatics." *Nature Reviews Genetics* 1 (2000): 231–236.

[31] Hall, Stuart. "On Postmodernism and Articulation: An Interview with Stuart Hall." *Journal of Communication Inquiry* 10.2 (1986): 45–60.

[32] Haraway, Donna. "The Promises of Monsters: A Regenerative Politics for Inappropriate/d Others." *Cultural Studies*. Ed. Lawrence Grossberg *et al.* (New York: Routledge, 1992): 295–337.

[33] Harris, Randy Allen. "Randy Allen Harris." <http://www.arts.uwaterloo.ca/~raha/>, accessed 23 July 2013.

[34] Herbert, Wray. "Focusing on the Cinematic Mind." 18 February 2010. Blog. <http://www.psychologicalscience.org/onlyhuman/2010/02/our-household-is-rolling-alfred.cfm>), accessed 23 July 2013.

[35] Hofstadter, Douglas R. *Metamagical Themas: Questing for the Essence of Mind and Pattern*. New York: Bantam, 1986.

[36] "Jobsian fondle-slab in SEXY FILTHGRAM CRACKDOWN." *The Register*. 13 October 2010. <http://www.theregister.co.uk/2010/10/13/apple_filthgram_crackdown_patent/>, accessed 23 July 2013.

[37] Kreisler, Harry. "Philosophy and the Habits of Critical Thinking: Conversation with John R. Searle." 22 September 1999. Transcript. <http://globetrotter.berkeley.edu/people/Searle/searle-con4.html>, accessed 10 August 2013.

[38] Lacan, Jacques. *The Seminar of Jacques Lacan, Book IV: The Object Relation 1956–1957*. Ed. Jacques-Alain Miller. Trans. L.V. A. Roche. Unpublished manuscript. <http://www.scribd.com/doc/143580546/

The-Seminar-of-Jacques-Lacan-Book-4-The-Object-Relation>, accessed 10 August 2013.

[39] Language and Interaction Research Group. "Conversation Entailment." <http://links.cse.msu.edu:8000/lair/projects/conversationentailment.html>, accessed 10 August 2013.

[40] Lee, Matthew M., and Michael M. Lee. *Text-based communication control for personal communication device*. Apple Inc., assignee. Patent 7,814,163. 2010.

[41] Lévi-Strauss, Claude. *The Elementary Structures of Kinship*. 1949. Trans. James Harle Bell, Rodney Needham, and John Richard von Sturmer. Ed. Rodney Needham. Boston: Beacon, 1969.

[42] Liu, Bing. "Sentiment Analysis." *Fifth Annual Text Analytics Summit*. Boston, MA. 2009. Presentation. <http://www.clarabridge.comportals/Clarabridge/BingLiuSentimentAnalysis.pdf>, accessed 23 July 2013.

[43] Makuta-Gihlk, Marzena. "A computational rhetoric for syntactic aspects of text." Research Report CS-91-54, Faculty of Mathematics. U Waterloo, 1991. MMath thesis.

[44] Mann, William C. and Maite Taboada. "Rhetorical Structure Theory: Intro to RST." 2012. <http://www.sfu.ca/rst/01intro/intro.html>, accessed 23 July 2013.

[45] "MPQA [Multi-Perspective Question Answering] Resources." <http://mpqa.cs.pitt.edu/>, accessed 23 July 2013.

[46] National Science Foundation Blue Ribbon Panel on Simulation-Based Engineering Science. *Revolutionizing Engineering Science through Simulation*. 2006. <http://www.nsf.gov/pubs/reports/sbes_final_report.pdf>, accessed 2013-08-03.

[47] Nielsen, Jakob. *Usability Engineering*. San Francisco: Morgan Kaufmann, 1993. Excerpted at <http://www.nngroup.com/articles/response-times-3-important-limits/>, accessed 10 August 2013.

[48] "Office Assistant." <http://en.wikipedia.org/wiki/Office_assistant>, accessed 23 July 2013.

[49] Olds, Dan. "SUPERCOMPUTER vs *your* computer in bang-for-buck battle." *The Register*. 8 March 2012. <http://www.theregister.co.uk/2012/03/08/supercomputing_vs_home_usage>, accessed 27 July 2013.

[50] Page, Lewis. "Rom-coms, period dramas are rubbish: Mathematical proof." *The Register*, 24 February 2010. <http://www.theregister.co.uk/2010/02/24/chaos_theory_movie_analysis/>, accessed 23 July 2013.

[51] Penniman, Matt and Michael Wojcik. "Creating Complex Web Applications with Agile Techniques: Iterations as Drafts." *Designing Web-Based Applications for 21st Century Writing Classrooms*. Eds. George Pullman and Baotong Gu. Amityville, NY: Baywood, 2013: 69–92.

[52] —. "Speech Made Visible." Edges. Michigan State University, East Lansing, MI. February 2009. Poster and interactive demonstration. Some related materials are available at <`http://ideoplast.org/wra417/index.html#smv`>.

[53] —. "Speech Made Visible." <`http://sourceforge.net/projects/speechmadevis/`>. 2009. Software.

[54] Penrose, Roger. *The Emperor's New Mind: Concerning Computers, Minds, and the Laws of Physics*. New York: Penguin, 1991.

[55] Perelman, Chaim and Lucie Olbrechts-Tyteca. *The New Rhetoric: A Treatise on Argumentation*. 1958. Trans John Wilkinson and Purcell Weaver. Notre Dame, IN: U of Notre Dame P, 1969.

[56] Propp, Vladimir. *Morphology of the Folktale*. 1928. Trans. Laurence Scott. Ed. Svatava Pirkova-Jakobson. Bloomington: Indiana U, 1958.

[57] Rehg, William, Peter McBurney, and Simon Parsons. "Computer decision-support systems for public argumentation: Assessing deliberative legitimacy." *Artificial Intelligence & Society* 19 (2005), 203–229.

[58] Rhodes, Bradley J. and Thad Starner. "Remembrance Agent: A continuously running automated information retrieval system." *Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi Agent Technology* (PAAM). 1996: 487–495. <`http://xenia.media.mit.edu/~rhodes/Papers/remembrance.html`>, accessed 23 July 2013.

[59] Richardson, Matt. "Descriptive Camera." <`http://mattrichardson.com/Descriptive-Camera/`>, accessed 10 August 2013.

[60] Rorty, Richard. *Contingency, Irony, and Solidarity*. Cambridge, UK: Cambridge UP, 1989.

[61] Scott, Robert L. "On Viewing Rhetoric as Epistemic." *Central States Speech Journal* 18 (1967): 9–17. Rpt. in *Contemporary Rhetorical Theory: a Reader*. Ed. John Lucaites, Celeste Michelle Condit, and Sally Caudill. New York: Guilford, 1998. 131–139.

[62] Searle, John. "Minds, Brains, and Programs." *Behavioral and Brain Sciences* 3 (1980): 417–457. <`http://web.archive.org/web/20071210043312/http://members.aol.com/NeoNoetics/MindsBrainsPrograms.html`>, accessed 10 August 2013.

[63] Snow, C[harles] P[ercy]. *The Two Cultures and the Scientific Revolution*. London: Cambridge UP, 1959.

[64] Thomo, Alex. "Latent Semantic Analysis (Tutorial)." <http://www.engr.uvic.ca/~seng474/svd.pdf>, accesed 10 August 2013.

[65] Trauman, Ryan. "An Interview with Hugh Burns." "Theme: Artificial Intelligence as Machine-Rhetoric." *Kairos* 14.3 (2010). <http://kairos.technorhetoric.net/14.3/interviews/trauman/artificial.html>, accessed 23 July 2013.

[66] "Truthy.indiana.edu to search, identify smear tactics, Twitter-bombs through November election runup." 28 September 2010. <http://newsinfo.iu.edu/news/page/normal/15742.html>, accessed 23 July 2013.

[67] Turing, Alan. "Computing Machinery and Intelligence." *Mind* 59 (1950): 436–460.

[68] Verheij, Bart. "Dialectical Argumentation with Argumentation Schemes: An Approach to Legal Logic." *Artificial Intelligence and Law* 11 (2003), 167–195.

[69] Von Neumann, John. "First Draft of a Report on the EDVAC." 1945. Ed. Michael D. Godfrey. 1992. <http://virtualtravelog.net/wp/wp-content/media/2003-08-TheFirstDraft.pdf>, accessed 4 August 2013.

[70] Weizenbaum, Joseph. "ELIZA—A Computer Program For the Study of Natural Language Communication Between Man And Machine." *CACM* 9.1 (1966): 36–45.

[71] White, James Boyd. *When Words Lose Their Meaning*. Chicaco: U of Chicago P, 1984.

[72] Wilson, Kenneth G. "Grand Challenges to Computational Science." *Future Generation Computer Systems* 5 (1989): 171–189.

[73] Wojcik, Michael. "CSE 842 Materials." <http://ideoplast.org/cse842/>.

[74] —. "Defining Rhetoric." 2008. <http://ideoplast.org/rw/portfolio-2007-2008/rhetoric.html>.

[75] —. "Estimating Ethos." Computers & Writing Conference. Athens, GA. May 2008. <http://ideoplast.org/textmill/cw2008/presentation.pdf>. Presentation.

[76] —. "Inventing Theory." 2007. Unpublished.

[77] —. "TextMill and Estimating Ethos." 2008. <http://ideoplast.org/textmill/>.

[78] Wojcik, Michael and Kristen Flory. "Candidates, Color, and Type." Thomas R. Watson Conference. Louisville, KY. October 2008. Also Conference on College Composition and Communication (CCCC). San Francisco. March 2009. Presentation.

[79] —. "Visual Rhetoric Greasemonkey Scripts." 2009. <http://ideoplast.org/visrhet/>.

[80] Zadeh, Lotfi A. “Fuzzy Sets.” *Information and Control* 8 (1965): 338–353.