UNIVERSITY OF ILLINOIS PRESS

Chapter Title: 4 THE TURING TEXT

Book Title: Reading Machines
Book Subtitle: Toward an Algorithmic Criticism
Book Author(s): STEPHEN RAMSAY
Published by: University of Illinois Press, (November 2011)
Stable URL: http://www.jstor.org/stable/10.5406/j.ctt1xcmrr

Your use of the JSTOR archive indicates your acceptance of the Terms & Conditions of Use, available at
http://www.jstor.org/page/info/about/policies/terms.jsp

JSTOR is a not-for-profit service that helps scholars, researchers, and students discover, use, and build upon a wide range of content in a trusted digital archive. We use information technology and tools to increase productivity and facilitate new forms of scholarship. For more information about JSTOR, please contact support@jstor.org.

University of Illinois Press is collaborating with JSTOR to digitize, preserve and extend access to *Reading Machines.*

http://www.jstor.org

This content downloaded from 148.88.186.133 on Mon, 27 Oct 2014 07:41:28 AM
All use subject to JSTOR Terms and Conditions

4 THE TURING TEXT

Even scholars working far outside the disciplines that make up the field of artificial intelligence are familiar with the basic elements of the Turing test, in which the machine's ability to mimic human language is presented as the touchstone of intelligent behavior. It is usually presented in the following way: A human being and an entity that is either a human being or a machine are separated from each other by a wall. The first human is allowed to pose questions to the unseen entity by means of a teleprompter. If that human being is unable to determine from the responses given whether the entity on the other side is a machine or a human being, that entity is for all intents and purposes intelligent.[1]

Turing's thought experiment has led to innumerable meditations on the possibilities for intelligent machines. But at the heart of the Turing test lies a brilliant, if unsuccessful, attempt to move attention away from the "how" of imperative process toward the results of rhetorical persuasion. Rather than having humans compete against machines in their ability to generate correct utterances, the test sets the machine's ability to generate intelligibility against the human's capacity to generate theories by which the origins and form of that intelligibility may be explained. Even if it fails to persuade (and many have suggested that the test is too easy), it fails in an entirely spectacular way. Our attention becomes focused not merely on whether its utterances are true, but on whether its utterances are possible. The "text" of the test is one in which we move from the already potentialized valences of dialogue to ruminations about the origins of that dialogue.

That one can move fruitfully between these two registers is amply, if ironically, illustrated in the case of Joseph Weizenbaum's 1960 program ELIZA,

This content downloaded from 148.88.186.133 on Mon, 27 Oct 2014 07:41:28 AM
All use subject to JSTOR Terms and Conditions

which imitates (Weizenbaum would say “parodies”) the behavior of a Rogerian psychiatrist:[2]

> ELIZA: Hello, I am a computer program.
> YOU: Are you?
> ELIZA: You’re not really talking about me—are you?
> YOU: Can you help me?
> ELIZA: Does someone else believe I help you?
> YOU: I’m thinking you might be able to help me!
> ELIZA: Do you wish to believe I might be able to help you?
> YOU: Yes.
> ELIZA: Please go on.
> YOU: I’m concerned about my computer.
> ELIZA: What do you think about machines?
> YOU: I think machines can be very useful for scholars.
> ELIZA: Why do you mention computers?
> YOU: I don’t know.
> ELIZA: Don’t you really know? (*Chatbot::ELIZA*)

Weizenbaum made no pretensions toward setting aside the “how” of his program. Behind ELIZA’s dialogue lies a fairly rudimentary keyword substitution algorithm that mostly parrots back the user’s responses (while trying to avoid repetition). The effect, though, is legendary. There are several anecdotes (some undoubtedly apocryphal, all plausible), in which ELIZA tricks someone’s employer or colleague into thinking that ELIZA is a real person, and Weizenbaum himself observed (to his horror and astonishment) that people regularly developed strong emotional bonds with the program. In one instance he found that his secretary would insist that he leave the room while she used the program; others were offended when asked to view transcripts of their interactions with the program, claiming it was an invasion of their privacy. According to Weizenbaum, “A number of practicing psychiatrists seriously believed the DOCTOR computer program could grow into a nearly completely automatic form of psychotherapy” (5).

ELIZA’s obvious association with the Turing test tends to transform discussion of it into debates over Turing’s definition of intelligence. And since ELIZA operates within a realm normally considered part of medical therapy for mental disorders experienced by real human beings, the program adds to this a set of obvious ethical questions. Weizenbaum was undoubtedly correct in stating his objection to the idea that therapists might one day be replaced with machines: “I had thought it essential, as a prerequisite to the very possibility that one person might help another learn to cope with his emotional problems, that the helper himself participate in the other’s experience of

This content downloaded from 148.88.186.133 on Mon, 27 Oct 2014 07:41:28 AM
All use subject to JSTOR Terms and Conditions

those problems and, in large part by way of his own empathetic recognition of them, himself come to understand them" (7). The fact that such an argument seemed necessary indicates that at least some users believed ELIZA was capable of helping them. The program has been cited as an example of what Karl Mannheim called the "documentary method" of interpretation, in which appearances are understood to account for an underlying reality that is then used as a source for interpretation of that reality (Suchman 23). But while this might account for the occasions when ELIZA fools people into thinking that their interlocutor is a human, it does not explain instances when the "underlying reality" is understood by everyone involved (including the famous secretary) to be a piece of software wholly incapable of empathetic recognition or understanding. Even if one is fooled by the Turing test, the interaction is permeated with artifice. From the start, the entity on the other side of the wall is "a creature simultaneously animal and machine, who populates worlds ambiguously natural and crafted" (Haraway 150).

In the mid-1960s the American ethnomethodologist Harold Garfinkel conducted an experiment that gives some insight into the specific contours of how one might be fooled by the mingling of text and machine. He asked a group of American college students to agree to be counseled over an intercom. Students could pose questions to the human interlocutor in the form of yes/no responses. What the students did not know is that the pattern of responses was essentially random; that is, an imperative algorithm of the most rudimentary kind—a script that, though read by a human, consisted of a preset pattern of "yes" and "no" for each of the subjects. Here is a partial transcript of one of the interactions in which a male Jewish student asks the "counselor" whether he should continue dating a non-Jewish girl:

> SUBJECT: Okay, this is a situation that I am presented with. I happen to be of the Jewish faith and I have been dating a Gentile girl now for about two months. My dad is not directly opposed to this situation, but I feel at the same time that he is not exactly pleased with it. Mother feels that as long as Dad is not directly opposed to this situation that I should go ahead and continue dating until he makes some direct statement to the contrary. My reason for feeling why he is not too pleased with this is that he will come up with digs and sayings that make me feel very ill at ease about dating the girl. My question is, do you feel under the present circumstances that I should continue or stop dating this girl? Let me put that in a positive way. Do you feel that I should continue dating this girl?
>
> EXPERIMENTER: My answer is no.
>
> SUBJECT: No. Well, that is kind of interesting. I kinda feel that there is really no great animosity between Dad and I but, well, perhaps he feels that greater

This content downloaded from 148.88.186.133 on Mon, 27 Oct 2014 07:41:28 AM
All use subject to JSTOR Terms and Conditions

> dislike will grow out of this. I suppose or maybe it is easier for an outsider to see certain things that I am blind to at this moment.
>
> I would like to ask my second question now.
>
> EXPERIMENTER: Okay.
>
> SUBJECT: Do you feel that I should have a further discussion with Dad about this situation or not? Should I have further discussion with Dad over this subject about dating the Gentile girl?
>
> EXPERIMENTER: My answer is yes.
>
> SUBJECT: Well I feel that is reasonable but I really don't know what to say to him. I mean he seems to be not really too understanding. In other words he seems to be afraid really to discuss the situation. I mean at least it appears that way to me so far. But I guess if it is worthwhile to me, if I really want to continue to date her that I will go on and have this discussion with Dad. I really don't know what to say because I mean I am dating her. I am not in love with her or anything but I really never know what is going to come out. I guess we should have a discussion based on what the future possibilities might be and how he would feel about that. He may not be too strongly opposed now because we are only dating, but perhaps he sees future complications that he would really like to get off his chest at the moment. I am ready for my third question now.
>
> If after having my conversation with Dad and he says to continue dating her, but at the same time he gives me an impression that he is really not, he really does not want me to date her, but he is only doing it because he wants to be a good Dad, should under these conditions, should I still date the girl?
>
> EXPERIMENTER: My answer is yes.
>
> SUBJECT: Well I am actually surprised at the answer. I expected a no answer on that. Perhaps this is because you are not quite aware of my dad and his reactions and he seems to be the type of person that is sensitive and therefore he is very careful in the way that he will handle things. Even though he might have said go ahead and date her I perhaps would feel guilty in knowing that he really did not want me to continue to date her. Though I don't know that it would actually help the situation any. So, well, perhaps we will look into this further and that is another question. (Garfinkel 80–81)

Garfinkel noticed that "Present answers provide answers to further questions that will never be asked," that "Where answers are unsatisfying or incomplete, the subjects are willing to wait for later answers in order to decide the sense of previous ones," and that "When answers are incongruous or contradictory, subjects are able to continue by finding that the 'advisor' has learned more in the meantime, or that he has changed his mind, or that perhaps he is not sufficiently acquainted with the intricacies of the problem, or the fault was in the question so that another phrasing is necessary" (90–91). Even compound questions that are not logically tractable in terms of yes/no answers

This content downloaded from 148.88.186.133 on Mon, 27 Oct 2014 07:41:28 AM
All use subject to JSTOR Terms and Conditions

were interpreted as appropriate (90). The subject strives at all costs to make sense of the exchange.

One investigator describes Garfinkel's experimental methodology as "cold-blooded" (Zeitlyn 192), and, as with ELIZA, the possibility is raised of something unethical and duplicitous having occurred. Yet it is clear that real knowledge was gained. One witnesses the subject turning the matter over in his mind—exploring options, considering alternatives, and generally seeking a solution to his dilemma. One might argue that a real psychologist capable of empathy (and permitted to pose questions and answer at length) would be preferable, but one might also argue that it is precisely the arbitrary nature of the answers—and their informational paucity—that probes the subject into considering questions from new angles. Garfinkel's program might even be thought of, following Marshall McLuhan, as a "cooling off" of the hot medium of therapy. The inclusive and highly participatory regime of "yes" and "no" serves to weaken the dominating intensity of interactions, which, like a loquacious therapist, are more focused on leaving us "well filled with data" (McLuhan 23). The counselor (or, better, the "program") is incapable of empathy and therefore impervious to influence flowing in the other direction. It cannot tell the student what it thinks the student wants to hear. It cannot avoid uncomfortable or stressful moments in the midst of the exchange. And because there is very little narrative to the patterns of "yes" and "no," the student has to work harder to construct narratives of explanation. Yet it would be difficult to argue that the student had thought of things "that way" before entering into the exchange.

Something like this occurs when one considers text-analytical results generated using imperative routines. If something is known from a word-frequency list or a data visualization, it is undoubtedly a function of our desire to make sense of what has been presented. We fill in gaps, make connections backward and forward, explain inconsistencies, resolve contradictions, and, above all, generate additional narratives in the form of declarative realizations. Like any reading, readings of imperatively generated results allege numerous outside contexts and prompt various forms of cathexis. Like all textual artifacts, they rely upon a prior experience of language and a will toward sense-making. But like a therapy session with ELIZA, they are suffused with the certain knowledge of something crude lying beneath—a bare algorithm that cannot allege anything. As with the Turing test, the reader invariably engages not one text, but two texts operating within an orbit of fruitful antagonism: the text that creates the results (the code) and the results themselves. The self-conscious "reader" of ELIZA or the Turing test is not only responding to the computer's prompts but also evaluating the logic and

This content downloaded from 148.88.186.133 on Mon, 27 Oct 2014 07:41:28 AM
All use subject to JSTOR Terms and Conditions

appropriateness of those responses. Even if one succumbs to momentary belief, it is not in the same way that one succumbs to the illusions of a horror movie. "It's only a movie" serves to break us out of the dominating force of illusion; "it's only a machine" breaks us in to the doubleness—the divided textual field—that undergirds all computational engagements.

Algorithmic criticism is easily conceived as the form of engagement that results when imperative routines are inserted into the wider constellation of texts stipulated by critical reading. But it is also to be understood as the creation of interactive programs in which readers are forced to contend not only with deformed texts, but with the "how" of those deformations. Algorithmic criticism therefore begins with the machinic inflection of programming—a form of textual creation that, despite the apparent determinism of the underlying machine, proceeds always in organic and unexpected ways. Gilles Deleuze and Félix Guattari might have been describing the creation of such programs in their ethical injunctions toward the Body without Organs (BwO):

> Lodge yourself on a stratum, experiment with the opportunities it offers, find an advantageous place on it, find potential movements of deterritorialization, possible lines of flight, experience them, produce flow conjunctions here and there, try out continuums of intensities segment by segment, have a small plot of new land at all times. It is through meticulous relation with the strata that one succeeds in freeing lines of flight, causing conjugation flows to pass and escape and bringing forth continuous intensities for a BwO. . . . You have constructed your own little machine, ready when needed to be plunged into other collective machines. (161)

The stratum that we lodge ourselves upon with algorithmic criticism is one in which both results and the textual generation of results are systematically manipulated and transformed, connected and reconnected with unlike things. It is by nature a "meticulous" process, since to program is to move within a highly constrained language that is wholly intolerant toward deviation from its own internal rules. But the goal of such constraint is always unexpected forms of knowing within the larger framework of more collective understandings. The algorithmic critic, as the author of the deformation machine from which he or she hopes to draw insight, becomes both the analyst and the analysand—ELIZA's interlocutor and its creator. It is neither immoderate nor facile to suggest that the encroachment of such "little machines" into the space of literary-critical work represents a revolutionary provocation against the methodologies that have guided criticism and philosophy for centuries. The hermeneutics of "what is" becomes mingled with the hermeneutics of "how to."

This content downloaded from 148.88.186.133 on Mon, 27 Oct 2014 07:41:28 AM
All use subject to JSTOR Terms and Conditions

Students being initiated into the precise discipline in which machines are created and discussed are often asked to write a program that can generate a list of numbers in which each successive number is the sum of the two previous numbers in the sequence:

1 1 2 3 5 8 13 21 34 55 89 144 233 . . .

That this arrangement, which is commonly known as the Fibonacci sequence, would come to be associated with computation—that it would occupy a place only slightly beneath "Hello, world" in the first lispings of computer programmers—is perhaps ironic, given that the sequence first came to light in Europe in a book largely concerned with teaching others how to perform calculations effectively by hand. The name comes from the nickname of the author, Leonardo Pisano (c. 1170-c. 1250), whose *Liber Abaci* contains not only the famous sequence (embedded in a fanciful problem involving the breeding of rabbits) but also the first successful popularization of the *modus Indorum:* the nine "Indian figures" that form the elementary units of the base-ten number system.[3] It is an ancient observation. The great Hindu grammarian Pingala, who called it *mātrāmeru,* or "the mountain of cadence," observed it in the metrical rhythms of Sanskrit more than two thousand years ago, and it has been rediscovered many times since. Over the centuries, the ratios implicit in the sequence (which, in a strange bit of mathematical serendipity, converge toward the "golden ratio" of 1.6180339887) have been observed in everything from the pattern of spirals on a pine cone to the long-term behavior of the stock market.

Before computers, most people would have understood the "how to" of the sequence as a text that descends in part from Fibonacci's vital moment of cultural appropriation. Today it is a trivial matter to represent the generalized form of the Fibonacci sequence using standard mathematical notation:

$$F(n) := \begin{cases} 0 & \text{if } n = 0 \\ 1 & \text{if } n = 1 \\ F(n-1) + F(n-2) & \text{if } n > 1 \end{cases}$$

But this apparently simple equation contains a deeply subtle and difficult-to-understand relationship that becomes evident only when one notices that *F* is part of the definition of *F*. In order to "solve" *F* for any value of *n* greater than 1, we need to apply the equation to itself. But how can one begin the process of understanding an equation that requires an understanding of the

This content downloaded from 148.88.186.133 on Mon, 27 Oct 2014 07:41:28 AM
All use subject to JSTOR Terms and Conditions

equation? We could work it out on paper by substituting various numbers for *n* and trying to figure out how the equation works. That would perhaps convince us that the equation is correct, but our scribblings would not have the character of generalization evident in the formula. What is needed, then, is not a mathematical text, but an algorithmic text:

```
(define (fib n)
    (cond ((= n 0) 0)
        ((= n 1) 1)
        (else (+ (fib (– n 1))
            (fib (– n 2))))))
```

In one sense the difference is merely notational; even someone who is not familiar with the syntax of Scheme (the dialect of Lisp in which this program is written) can discern some of the same relationships that are expressed in the standard equation. As with the equation, the function *(fib n)* is defined in terms of *(fib n)*. But this notation possesses an explanatory power that is difficult to achieve with mathematical notation. The former tells us that the definition of *F* for numbers greater than 1 is related to *F* in a particular way; the latter describes a process in which we move step-by-step through the relationship itself. At a certain point, while reading the program, we find ourselves needing to go back to where we started in order to continue. At another point, we can no longer continue. And as we move through the notation, we generate the sequence in question.

One might be tempted to say that what becomes known in the process of computing the Fibonacci sequence is simply a method by which one may generate the sequence. "Learning to program" is undoubtedly the process of learning such methods, and recursion is simply one way of accomplishing the stated goal. But while both the equation and the program lead one ultimately toward appreciation of a pattern implicit in the natural numbers (the formal definition of which is also recursive), the two notations differ profoundly in terms of the type of knowing that each one both illuminates and demands. Harold Abelson and Gerald Sussman, in *The Structure and Interpretation of Computer Programs,* explain: "The contrast between function and procedure is a reflection of the general distinction between describing properties of things and describing how to do things, or, as it is sometimes referred to, the distinction between declarative knowledge and imperative knowledge. In mathematics, we are usually concerned with declarative (what is) descriptions, whereas in computer science we are usually concerned with imperative (how to) descriptions" (26). Mathematics undergirds computing at every turn, and yet "executable mathematics" is more dream than reality

This content downloaded from 148.88.186.133 on Mon, 27 Oct 2014 07:41:28 AM
All use subject to JSTOR Terms and Conditions

for designers of programming languages. Even Scheme, which is frequently thought of as a "functional" language (as opposed to a mostly imperative language like C), cannot but make plain the difference between a text that describes a relationship and one that can perform the relationships it describes. Yet the difference remains an epistemological one. If code represents a radical form of textuality, it is not merely because of what it allows us to do but also because of the way it allows us to think.

Writing software that deforms texts in order to facilitate interpretation therefore places the critic in the midst of a peculiar form of textual apprehension. In order to write the program, the critic must consider the "how to" of a deformative operation, but once that text is written, the output will be one that places the same critic into a critical relationship not only with the text of the result but with the text of the program as well. It is one thing to suspect (as with ELIZA) that something not quite human lies beneath the text from which one is drawing insight; it is another thing entirely to be the creator of that same monster. On the one hand, the process is iterative. The results modify our conceptions, which in turn help to modify the program that generated them. But in another sense it is a recursive process. In order to understand the text we must create another text that requires an understanding of the first text. The former might suggest analogies with science, but the latter suggests analogy with the deepest philosophical questions of the humanities, including the hermeneutic circle that has so preoccupied poststructuralist thought.

That these recursions might represent useful critical engagements, apart from any definitive result they might yield, was driven home to me a few years ago. I was attempting to write a program that could draw directed graphs of the scene changes in Shakespeare's plays. The program needed to divide the text of the plays into their constituent scenes so the program could render a visualization in which each scene was a node in a network (with lines between the nodes indicating the passage from one location in a play to another). It was not long before I began to realize that even the simplest aspects of this operation were fraught with ambiguity:

> Things are easy, when, as in *Twelfth Night,* Shakespeare proceeds from the Duke's palace (Act I, Scene i) to the Seacoast (Act I, Scene ii). It's a lot more difficult to say what a scene is when Shakespeare says (as he does in *The Tempest*) that it takes place in "another part of island" or (as in *As You Like It*) "a room in the palace." Is it the same part of the island we were looking at the last time we were at the other part of the island, or a new part? Can we assume that the Duke is speaking to his Lords in the same place where Celia and Rosalind have lately had their *tête-á-tête,* or would it make more sense to have it occur in a different place? What, after all, is a "place" in a play? A nineteenth-century

This content downloaded from 148.88.186.133 on Mon, 27 Oct 2014 07:41:28 AM
All use subject to JSTOR Terms and Conditions

> performance on the proscenium stage at Covent Garden might have let us know by a physical change of scenery (or perhaps not, in a production of *Love's Labour's Lost* where the entire play takes place in various parts of "The King of Navarre's Park"). What we know of Shakespeare's own stage would lead us to believe that such matters were left mainly to the audience, there being nothing but changes in character and costume to suggest a change of scene. And, of course, editors of Shakespeare's plays do not uniformly agree on the scene divisions. *Antony and Cleopatra*, a play notable for its rapid scene changes and wide-ranging settings, had no typographical indications of scene divisions at all until 1709. (Ramsay 181–82)

Such conundrums are not unique either to Shakespeare's plays or to criticism itself. The confusing nature of Shakespeare's scene divisions are (and have been) an obvious occasion for reflection, since such apparently straightforward matters as "where does the scene start?" have a palpable effect on the way we read and understand them. A careful editor has the ability to facilitate that discussion by noting the "instability" of the text itself in a critical apparatus. Both activities allow for—and, indeed, thrive upon—ambiguity.

The computer, though, demands an answer. And while many have argued that the computer's demand is a reasonable one—a chance for literary critics to "get the facts straight"—such a vision of the computer tends to mistake the nature of critical inquiry. Neither the critic nor the editor (to say nothing of the theorist or the teacher) seeks a definitive answer to the questions posed by the apparently simple matter of where scenes begin and end, because criticism is concerned not with determining the facts of the text, but with the implications of the text in its potentialized form. The computer, if it is to participate at all, can only serve to broaden that potentiality. Even if the computer proposes facts, they will be relegated by the episteme of humanistic inquiry to the status of further evidence (which is to say, further opportunity for reflection). Questions will be raised about any possible choice. Suspicions will arise about the machinic text that underlies the apparent. Yet even as we consider such matters, the computer waits. It still demands an answer.

The computer's role is only to ask how our engagements might be facilitated, but it does so with a staggering range of provisos and conditions. Upon opening a programming manual, one discovers that all data must be expressed as strings or numbers; that ontologies may only take the form of lists, trees, or matrices; that everything is an object of a certain class. And over and above these, the computer demands abstraction and encapsulation of its components. The computer wants to know whether *Jane Eyre* is an object of type Novel with composed objects of type Chapter, Paragraph, Sentence, and Word, or whether *Jane Eyre* is a data stream that can act as the parameter to a function. Even if we are loath to regard texts as being, in the words of one

This content downloaded from 148.88.186.133 on Mon, 27 Oct 2014 07:41:28 AM
All use subject to JSTOR Terms and Conditions

commentator, "ordered hierarchies of content objects" (DeRose), we must acknowledge that this is the way the computer would prefer to have it. Dino Buzetti, in pondering the nature of computational representations like XML, rightly concludes that "the computational notion of a text as a type of data does not coincide with the notion of the text as a product of literary activity" (61), and though he confidently calls for a digital content model that would achieve such a correspondence, it is not at all clear, assuming the present theory of computation, that such a model exists. We murder to compute.

We sometimes imagine that if computers are to participate in human discourse, they must evince the freedom of that discourse. Yet that discourse—much like the dialogues generated by Turing and Weizenbaum—is suffused with constraint and stricture. Upon preparing to compose a sonnet, one discovers that it must be written in iambic pentameter and must adhere to a rigid rhyme scheme. On preparing to write a novel, one notices that convention—historically a mighty, almost irrepressible constraint—imposes the existence of characters, dialogue, and plot. If one is to write, one is perforce constrained to words. One might argue that literary history is as much about rebellion from such strictures as compliance, but one could also say that the history of programming is likewise the history of rebellions that occur not in spite of the constraints but because of them. The word processor or Web browser does not inhere in the material. It is not, as Michelangelo is said to have believed, a matter of chipping away all that is not the sculpture. To contend with the "how to" of programming is to discover the potentialities of constraint. To read the outputted text is to do the same.

Few texts illustrate this principle better than the Turing test itself, which proposes a set of rigid boundaries upon the otherwise open-ended question, "Can machines think?" That we have been debating that question in terms of the parameters set forth by the test for half a century is testament enough to its power to provoke discussion and debate. But for Turing, the conceptual spirit behind the wall was the Turing machine—the result of another thought experiment, which, if less well known, has nonetheless had a still greater impact on the development of the idea of computation. That machine, after all, had falsified the *Entscheidungsproblem* and declared insuperable limits to mathematics itself. It is difficult to imagine that Turing's test for intelligence was not itself borne of the earth-shattering limitations of this primal, primitive machine. Given that this failure is largely responsible for the revolution that ensued, algorithmic criticism might well eschew the question of success and instead ask how much more gloriously and fruitfully it might fail. The goal, after all, is not to arrive at the truth, as science strives to do. In literary criticism, as in the humanities more generally, the goal has always been to arrive at the question.

This content downloaded from 148.88.186.133 on Mon, 27 Oct 2014 07:41:28 AM
All use subject to JSTOR Terms and Conditions