Virtual Muse

Charles O. Hartman

Published by Wesleyan University Press

>>>> / 6 /

AUTOPOET

Far beyond the simple RanLines program on the Sinclair ZX81, "Monologues of Soul and Body" raised questions about authorship. Of course I was the author: I wrote most of the sections, gave them their titles, rewrote the published program, laboriously chose eight from a morass of computer outputs, and composed the whole thing as a poem, giving it its title and epigraphs.

But the "Soul" sections are full of quotations; and selecting from among the scrupulously unedited productions of a computer program hardly seems like *writing poetry*. Whatever this was, it wasn't exactly the sort of authorship we attribute to Homer or to Charlie Parker. On the other hand, there's a six-hundred-page dissertation demonstrating that Parker's incredibly inventive playing was based on a thesaurus of melodic formulas—as was Homer's singing.

Questions like this are woven into the fabric of twentieth-century art, as I've pointed out by talking about juxtaposition and collage and composition. The source of some material in my poem was fairly novel, but the poem's overall method wasn't unfamiliar. Even the idea of using a source text as the basis for a poetic text had been around for some time. Jackson Mac Low, Rosmarie Waldrop, William Burroughs, and dozens of others have used various methods to produce one text out of another for years, often with uncannily beautiful results.

"Monologues" is partly *about* the problems of its origin. The "Soul" sections dwell on computers acting like people and vice versa. The poem shows thought devolving into mechanism and a machine struggling toward what looks like thought. It worries about the limits

of knowledge and how very close to home they sometimes fall. It mixes facts and other sorts of fictions and expresses distrust about the relation between games like chess and realities like wars.

So I had finally employed a computer in the construction of a poem I found genuinely interesting (and have since embraced by publishing it, first in the experimental magazine *Tyuonyi*, and then in my book of poems from Wesleyan University Press, *Glass Enclosure*). What next?

Coleridge called poetry "the best words in the best order." Glimpsed from this particular angle, a poem is nothing but a selection and arrangement of words from the dictionary. This seems like a potentially mechanizable process. The poet's task can be seen as a problem of scale: there are so very many possible combinations of words to choose among. This gives the poet an enormous practical problem. But aside from that, it's an idea of poetry that makes the poet's chief job *judgment*, not exactly *creation*. This sounds more like an eighteenth-century poet than a nineteenth-century one, more Neoclassical than Romantic. But the twentieth century offers, if nothing else, a choice of ancestors.

What I wanted to do next was to resurrect the Scansion Machine and turn it into a productive engine, rather than a strictly analytic one. If I could find a way to generate candidate lines of verse, the scansion procedure could filter out unmetrical ones. Then I could build a metrical poem out of the victors. It would still be me doing the building. But the computer would be producing the poem's language on its own.

I was seeking a common ground between what the computer (in my amateur programming hands) could realistically do and what I could plausibly define as writing poetry. After "Monologues," how far would I have to go in a different direction to make these ends meet again?

So a program in Pascal began to grow out of the Scansion Machine. At some early stage I named it "Natural Selection." The metaphor I had thought about while writing "Monologues" was still unsatisfied, still looking for expression. Vaguely, I entertained the vision of a horde of lines, a whole population competing for the sustenance of

attention—but whose attention? I neither expected nor wanted to write myself out of the picture. Presenting the reader with all the computer's combinations was out of the question. I anticipated making any final selection myself. The question was what it would take, in the way of computer filtering, to reduce an indefinitely large number of possible lines to a flock small enough for me to cull.

The initial problem was to generate the candidates. My first method was to string together words at random until I'd collected the right number of syllables (nine to thirteen, say) for an iambic pentameter.

What words? The commonest words are very common indeed. Without a good proportion of them, no stretch of imitation English has much chance of sounding sensible. So I began by building a dictionary of the few thousand most common words. I used the word-frequency list compiled by Kuçera and Searle from their sample of a million printed English words.

My dictionary added three pieces of information to this list: the number of syllables in the word, the stressed syllable if there was more than one, and the part of speech. These are the same details that the Scansion Machine elicited from the user for each word it came across in the poem to be scanned. This kind of information isn't usually included in the computerized dictionaries used to check spelling; I had to build my own.

The output tended to run like this:

> investigation of the guy the stay
> wrote great the seeing the blue particular
> wonderful services repeated remember
> the summer more vision with the wet past
> division in the none traditional
> universe appeared generally day
> settled early the complex feel the dropped
> the early reality is nuclear

and so on. Disappointing, to say the least. Obviously, mere random selection wasn't enough.

In fact I had already introduced heavy statistical weighting into the

program's random choices. Words from the first group of five hundred were chosen far more often than any of the rest. Within the first group the choice was still further guided by an algebraic function with a very steep curve (one over x squared, plus one) that roughly approximates the curve of frequencies in actual English. I also added a group of five hundred "specials," words I hand-picked for interest. These were inserted unusually often.

None of it helped. The nonsense factor was working far too well. Even the most willing reader—and I was still pretty willing, even after thrashing around in the output of Travesty—couldn't make much sense of these lines, especially in combination. For the next step I'd have to add syntax.

I have great faith in syntax. Language is sentences, not words, and not simple word frequencies. I've heard (though I've heard other linguists dispute it) that children learn syntax even before they learn vocabulary. Certainly, my son uttered little tunes that contained no recognizable words but sounded like speech. I could almost, but not quite, understand him. He was meaning to speak. Writing and reading poems drives home the conviction that it's not the words alone that create voice and image, power and meaning, but the relations among them. Meter makes one kind of relation among words, but syntax relates them in a far more pervasive and subtle way. Where there's syntax, there ought to be meaning. Or at least, to state my hope more exactly, enough syntax ought to tempt a reader to help make sense.

But the syntactical system of English is complex. Major research efforts by computer science labs and by linguists, together and separately, haven't come up with a complete formulation of the rules of English syntax. Researchers don't even agree about what such a formulation would look like if we had it. Where those angels were treading on each other's toes, I wasn't fool enough to intrude.

Actually, I realized, what I needed wasn't anything like that. Those researchers were all trying to understand English; I merely wanted to write it. The Imitation Game sharpens your eye for shortcuts. In this case, I was playing the much easier half of the game. Unlike almost all computer programs, the one I envisioned would have no input. It

would just utter its language like an oracle, leaving the job of meaning making to the reader's instinctively intelligent questioning.

Notice, by the way, that in this game the computer reverses the human tendency. We can gather a pretty good "passive vocabulary" in a language long before we become fluent speakers. But it's a lot easier to make a computer talk than listen.

At first I tried piecemeal fixes. If the last word chosen at random was an article, make the next one a noun. If a pronoun comes up, follow it with a verb. These patches didn't work well. For example, the rules I've just given exclude all article-adjective-noun combinations (like "a rude awakening"). I tried a variation in which I recast all the rules in negative terms. It was better but not good enough. I needed a real grammar.

The available literature on artificial intelligence is beginning to reach a level the determined nonspecialist can understand. How-to books have accumulated particularly around the language called Prolog, which is especially suited to AI projects like handling what are called grammar networks. My next step should be to vary and extend some of the example programs given in these books. (This is what I'd done with Travesty, altering it toward my purposes. It's how a lot of computer investigations begin. Poets, too, often prime the pump by loosely imitating earlier poems.)

I was learning Prolog at the same time and trying to preserve the work I'd already done. The result of this tinkering was truly a monster, a program written in two pieces in two very different languages that could communicate only with the greatest discomfort. For low computer comedy, here's the headnote to that version: "This version of the Natural Selection program collaborates with (indeed runs as a subtask of) a Prolog program whose function is to create sentence templates. Communication between the programs is by means of a disk file called 'TEMPLATE.' The main parts of the complete system are interrelated [as shown in figure 1.]"

Oddly, the poor thing worked; it would produce one or two lines per minute. Dr. Moreau, too, in the horror story by H. G. Wells, got his beasts to walk on two legs.

I had fallen victim to a kind of purism. Programmers live within an

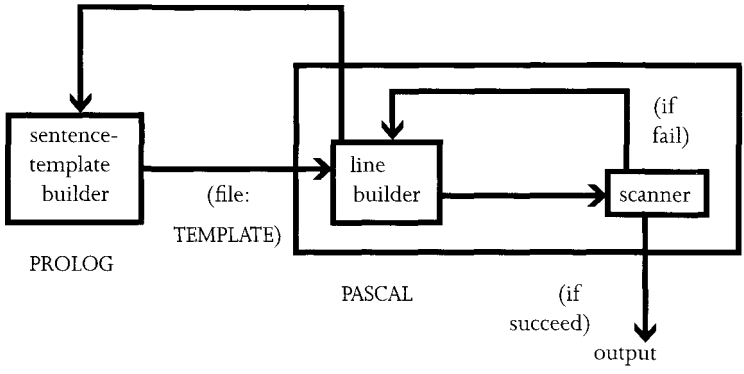

Figure 1

intricate class system of computer languages. Prolog was the Right Language for the template-generating job, so I used Prolog. But my program wasn't big enough to demand that kind of efficiency. I was not doing AI research. Better to fool around in a usable language like Pascal.

So I tore the thing apart and rebuilt it. How it manipulated grammar rules is something I'll describe in more detail in the next chapter because it's the main feature I preserved in the Prose project that finally made something useful out of this mess. But the rest of the program, now called AutoPoet, worked straightforwardly:

1. Create a syntactical skeleton or template—a list of parts of speech in sentence order (e.g., determiner, adjective, noun, verb).
2. Pick words at random from lists of determiners, adjectives, nouns, verbs, and so on, to build a sentence on this template (e.g., "this fatal strop commutes").
3. When the words have accumulated to about the length of a pentameter, test it for metricality.
4. If it passes the test in step 3, print it. If it doesn't, go back to step 2.

The only remaining trick is to preserve the state of the program somewhere around step 2 so that step 4 can return to that point

without losing the grammatical thread of the sentence. That is, we want the sentence to continue even when the line has been successfully completed. This will have the additional advantage of *enjambing* many lines, making them run over in sense. Enjambment gives metrical lines much of their supple liveliness. To accomplish this, there's a lot of information to preserve and, if necessary (if a trial line fails the meter test), to restore: the template, the point reached in the template before the failing candidate line was built, the then-current state of plurality, person, number, and so on. Furthermore, since either a line can end while a sentence is still incomplete or a sentence can end partway through a line, restarting a line can get complicated.

Did it work? No, not very well:

> The garden of steel—place—had figured in
> this. When I am every afternoon,
> how can't the last teacher write? But I
> was art without my play between a result
> and the metabolism, and the night
> of language toward a story between the part
> and any light (the thin subject) remains.
> Unless their jazz among so national
> a center burned to practice, history
> is a machine's afternoon. So sure a plane:
> hotel. The hell of day determines her.
> Because they turn to someone, history
> is so thin a science. The club—so democratic
> a list—is the earth of length. Because the fight
> of clay has met his willow, whom is so straight
> a line determining? While I might compute
> its night, I am the room, and that day plane
> (that garden) had based practice. Will its second
> kill so professional a game? So black a
> jazz stopped to think sound for the range of women,
> and voice—the gas of night through your device—
> was working.

Variations suggested themselves. I even wrote one elaborate version of the program that passed its sentence template through a ver-

sion of Travesty before filling in the words at random. The result was somewhat interesting—but only if you knew where it had come from.

That was the fundamental problem. The fact that my program worked at all was a little surprising. But it didn't work well enough. I could remain intrigued as the programmer but not as a poet. I had arrived at a barrier where many computer poetry experiments have died of ennui. The Imitation Game is hard to play on humans' home turf, language. All my metrical and syntactical drubbing of the language did only a little to drive the random words toward sense.

Maybe I should have taken this as a victory. As a human poet, I could still do something my computer proved simply incapable of doing. Emerson had said that "it is not meters, but a meter-making argument that makes a poem." I should have listened to him and known better. But it's a pretty cheap victory. If that were the point, I would never have tried the experiments in the first place. If the result is merely a round of congratulations that the human club remains closed to outsiders, all these efforts are pointless.

I was sure I had had a point. I began to think that the fault lay not so much in the computer program but in what I was trying to make it do.

AutoPoet embodied an inappropriate idea of poetry. As long as the goal was the imitation of a human poet—or as long as the poem's reader was encouraged to think that was the goal—I wasn't likely to get any farther. What's wrong with the AutoPoetry I've quoted here (and all the other reams of it the machine would produce until it was turned off) is exactly that it's *imitation poetry*. All our habits of reading are called upon, all the old expectations, and then let down. "Monologues of Soul and Body" had worked because its "body" sections were so *different* from human poetry. It had successfully demanded its own way of reading. To go on from there, once again I needed a new idea.