COMPUTER RECREATIONS

A progress report on the fine art of turning literature into drivel

by Brian Hayes

Almost any computer program can be made to yield meaningless results if it is given sufficiently muddled information to work with; this is the sense of the tired adage "garbage in, garbage out." The principle is now so well established that no one would take much notice of another demonstration. With a little thought and effort, however, it is possible to create a program that accepts as its input great masterworks of literature and nonetheless produces as its output utter nonsense. In goes the last act of *Macbeth;* out comes a tale told by an idiot, full of sound and fury, signifying nothing. Now *that* is data processing. (The inverse transformation, alas, seems to be much harder.)

The conversion of literature into gibberish is done in two stages. First a text is "read" by the program, and certain statistical properties are extracted and recorded. The statistics define the probability that any given letter of the alphabet follows another letter, or another sequence of letters, in the source text. In the second stage a new text is generated by choosing letters at random in accordance with the recorded probabilities. The result is a stream of characters that reproduce the statistical properties of the original text but whose only meaning, if any, is a matter of accident.

I cannot imagine a cruder method of imitation. Nowhere in the program is there even a representation of the concept of a word, much less any hint of what words might mean. There is no representation of any linguistic structure more elaborate than a sequence of letters. The text created is the clumsiest kind of pastiche, which preserves only the most superficial qualities of the original. What is remarkable is that the product of this simple exercise sometimes has a haunting familiarity. It is nonsense, but not undifferentiated nonsense; rather it is Chaucerian or Shakespearian or Jamesian nonsense. Indeed, with all semantic content eliminated, stylistic mannerisms become the more conspicuous. It makes one wonder: Just how close to the surface are the qualities that define an author's style?

The process of generating random prose has been investigated in detail by William Ralph Bennett, Jr., of Yale University. He has made the statistics of language a major theme of a course on the applications of computers, and the topic also figures prominently in his introductory textbook on programming, *Scientific and Engineering Problem-solving with the Computer.* (The book is a good deal livelier than the title might suggest. The problems taken up include the aerodynamics of the 1950 Princeton-Dartmouth football game, which was played in a hurricane; the diffusion of syphilis through a population of sailors and prostitutes, and a spectral analysis of the krummhorn, oboe and "mode-locked garden hose.")

Bennett notes that the earliest known reference to the random generation of language is in the *Maxims and Discourses* of John Tillotson, archbishop of Canterbury in the 1690's. In making a case for divine creation Tillotson wrote: "How often might a Man, after he had jumbled a Set of Letters in a Bag, fling them out upon the Ground before they would fall into an exact Poem, yea or so much as make a good Discourse in Prose? And may not a little Book be as easily made by Chance, as this great Volume of the World?"

```
PWGMMLTHIDVGRHPEDFCXFEKFNOPYPQSXZRUXG'YS'AEEU PEDEGLQYFUWPO'IKI
QTONIXJKZEUKDXWKKJREHYHPKWUJHLEJNBPLQ AIEOQXUBJYYVIFFDPQGIGZNTI
RQXPDJ NQESPQMCRSNGMKQEZICZV'GSWALK ZZEYIBBOTDCRSMK'VI MRCZXUBI
SNEQ'VQQHFQUCBJXZRVVNIBHFJEFTCFJPWFOIYHOMPNFSFWKNCMVLOJJBX
QV KIZTLNRWGGTZFPZPQQCGVJCPAYRDQJRMYSWCGABRXLERCYYRHQCHTOQ'UT
FMRITFTIZUIWTSTXWQGOCAFXJOZYKSTV'BYOBEUFIRQWQ VOUVQJPRKJWBKPLQZCB
```

Order-zero random text, drawing on an alphabet of 28 symbols

For most modern considerations of random language the point of departure is Sir Arthur Eddington's statement of 1927: "If an army of monkeys were strumming on typewriters, they *might* write all the books in the British Museum." Eddington too meant to emphasize the improbability of such an outcome; he cited it as an example of an event that could happen in principle but in practice never does. All the same, since Eddington's time the possibility of finding genius in the random peckings of monkeys has taken on a literary life of its own. Bennett mentions works by Russell Maloney and Kurt Vonnegut, Jr., and a nightclub act by Bob Newhart.

The process Eddington envisioned can be simulated by a program I shall call an order-zero text generator. First an alphabet, or character set, is decided on, which determines what keys are to be installed on the monkeys' typewriters. In some higher-order simulations it becomes important to keep the number of symbols to a minimum, and for consistency it seems best to adopt the same character set in the order-zero program. I have therefore followed Bennett's recommendation in choosing a set of 28 symbols: the 26 uppercase letters, the word space (which the computer treats as a character like any other) and the apostrophe (which is commoner in much written English than the three or four least-common letters are).

The ideal, unbiased monkey would at any moment have an equal probability of striking any key. This behavior can be simulated by a simple strategy. Each symbol in the character set is assigned a number from zero to 27. For each character to be generated a random integer is chosen in the same range and the corresponding character is printed. A small specimen of text created by this procedure is shown in the illustration on this page. It bears no resemblance to written English or to any other human language. "Words" tend to be extraordinarily long (on the average 27 characters) and thick with consonants. The reason, of course, is that letter frequencies in real English text are far from uniform. The word space alone generally accounts for roughly a fifth of the characters, whereas J, Q, X and Z together make up less than 1 percent. In an order-zero simulation all the characters have the same frequency, namely 1/28.

The comic routine by Bob Newhart concerns the plight of the inspectors who must read the monkeys' output. After many hours of poring over unintelligible prattle they come upon the phrase, "To be or not to be, that is the gesorenplatz. . . ." In fact, getting even that far is wildly improbable; the first nine words of Hamlet's soliloquy can be expected

SciAm Nov 1983 249(5)

to turn up once out of every 2×10^{46} characters. In a run of 50,000 characters I was able to find one instance of TO and another of NOT; they were many lines apart. (I did not read the 50,000 characters but instead made the search with a pattern-matching program.)

A first step toward improving the monkeys' literary skills is to adjust the probability of selecting a given letter so that it reflects the letter's actual frequency in written English. In effect, the plan is to build a typewriter with, say, 2,500 space keys, 850 E keys, 700 T keys and so on. The letter frequencies might be averages calculated from a large sample of English prose, but it is both more convenient and more interesting to base them on a particular source text. A program that chooses characters with such a frequency distribution is a first-order text generator.

The letter-frequency values can be represented in a one-dimensional array with 28 elements. The array is a block of storage locations in the computer's memory, organized so that any one element can be specified by an index, or subscript, between zero and 27. In order to fill up the array one could count the instances of each letter in the text and enter the values by hand. It is better, however, to let the program do the counting, even when that means the text itself must be prepared in a machine-readable form. The counting program initially sets all the elements of the array to zero. The text is then examined one character at a time, and for each occurrence of a character the corresponding array element is incremented by 1.

First-order random text is generated by making the probability of selecting a character proportional to the character's array element. One method works as follows. A random number is generated in the interval between zero and an upper bound equal to the sum of the array elements (which is also the total number of characters in the source text). The first array element, which might record the occurrences of the letter A, is then subtracted from the random number. If the result is zero or less, an A is printed; otherwise the next element (representing B) is subtracted from the value remaining after the first comparison. The successive subtractions continue until one of them gives a zero or a negative result, and the corresponding character is selected. Note that the procedure cannot fail to make a selection, since the random number cannot exceed the sum of the array elements.

A sample of first-order random text is shown in the upper illustration at the right. It is based on a frequency array compiled from a passage in the last chapter of James Joyce's *Ulysses,* the chapter known as "Ithaca," or Molly Bloom's soliloquy. I had a reason for choosing it: the absence of punctuation in the random text is of little consequence because the source text too is unpunctuated.

The information on letter frequencies embodied in a first-order random text brings an improvement, but one would hardly call the text readable. Although the average word length (4.7 letters) is near the expected value (4.5 letters), the variance, or deviation from the average, is much too great. Words in normal English, it seems, are not only short but also have a narrow range of lengths; in the random text the distribution is much broader. Apart from the question of word length there is the mat-

FIRST ORDER

```
HUD T ALONIT NTA SN TVIOET ELERFOAD PE TRLTWTL N CABEG TYLUEMU TIGT
BH OFDRRIC O STU HOOOTO YATNDL UYA HWAE SS NLSDB OTRORT DEERARFT
D LBFF HHARE MW OSPE OFOIT SEOUN GTUMG H N GHKOY T EAOS A SD E TNNE
PEHAGIADIHNATO AATSAGI ED INNE ABRA TAAM GT E TWNO HEWIIGUTNCM GA SFHHY
HREBH RARE OOSY LFE OC EGGTA WIFRTYE EUS DA ETO WF EIT ERNETEBTSTTELO
NTAAN O YEETWNSONRNHN TYHVN NLUESETTHLGEAKPNNMTIA TSM REEANTVONC POE
RUTP EOIT L IEETGTWHSW H KHHER W OLIOEWOEPT D AEYBSTNHGDNPT C TNLINHH
KHHE E RTVIOB EI K EOAFPUTSTTAS NA LAN SRDF D NMTHESKO UGEEDICRAWDT OBD
TUIML WSORGNETE
```

SECOND ORDER

```
BEGASPOINT IGHIANS JO HYOUD WOUMINN BONUTHENIG SPPRING SBER W IDESE WHE D
OOFOMOUT O CHEDA AFOOIAUDO IS WNY UT DRSASER LD OT POINE ETHAT FOEVEL BE
ORRI IVER BY HE T AS I HET W BE T WAU GIM UTHENTOTETHAVE THIKEWOITOCOUTORE
TATHASTHEE AT D Y WAN TOND SE TEDING US AKIN WING W TE T BO TOTSTHINGATONO
EN T LLY WID OUCOUSIND HEF THIMES AG T BENG LORYE ALLATHOMOFTHER TOUDIMS YS
S ORYRY THERNG S HE M G M ANG S CITOOFO HEN G BEST ONDLOL ANE DO HE
ICISEKERIT ME NKITHADIMUPL WHES HT BATHE T LOR WITULOWAYE WATHEG M
LEROMAUN OUGS POUPO O HASING LIN ON ASHAN AWFAS HET ND MEDE
```

THIRD ORDER

```
MAY THOT TO THER YOURS CHIM JOSE EY EILLY JUSED AND HID YEL THE MARK WASK
TROOFTEN HEREY LING SH THAVERED HER INCED I MEA BUT DAY WOM THE EAKIN WIPS
AS SUGH THE WAY LIARADE TH MY HE ALMASEETIR ANICIOUT JOSIDNTO GRATEVE NO
VER BIGH WER ACCOW WAS I GEORE HENDSO EGGET PUT TO SQUAD TRADE OFF GIN
GO ME HER SPING HE CONE WELL FEWHEY THEYES AND AND QUICE YOULDNT HER
ORL SO MAKING RINGS SOMET DREAVE HISETTO COMAD THAT ME WE MIG TOLD THE
THERFUMBECK OT OFF FEELP HE WAST ITS LETHOTTEN ITHEE ROWN YOURS FEL FOR
SOME IF WIS HE STAKED UPPOIS SHENS NO TILL HIM I WAY SO WHATEALWAS WER TWE
NER DING O THIS IT IN ANIGH ACK REAN THAT DO GETHE BITER
```

First-, second- and third-order random text based on "Molly Bloom's soliloquy"

```
AD CON LUM VIN INUS EDIRA INUNUBICIRCUM OMPRO VERIAE TE IUNTINTEMENEIS
MENSAE ALTORUM PRONS FATQUE ANUM ROPET PARED LA TUSAQUE CEA ERDITEREM IN
GLOCEREC IOVELLUM ET VEC IRA AE DOMNIENTERSUO QUE DA VIT INC PARBEM ETUS
TU MEDE DERIQUORUMIMO PEREPORIDEN HICESSE COSTRATQUIN FATU DORAEQUI POS
PRIENS NOCTA CIENT HUCCEDITAM PET AUDIISEDENDITA QUE GERBILIBATIA VOLAEQUE
ORECURICIT FES ADSUE ARCUMQUE LULIGITO PIMOES PERUM NOSUS HERENS EA
CREPERESEM ETURIBUS AVIS POS AT IS NOMINE FATULCHENTURASPARIS AUDEDET PARES
EXAMENDENT DUM REMPET HA REC ALEVIREM ORBO PIERIS ATAE PARE OCERE RAS
```

```
QUALTA 'L VOL POETA FU' OFFERA MAL ME ALE E 'L QUELE ME' E PESTI FOCONT E 'L M'AN
STI LA L' ILI PIOI PAURA MOSE ANGO SPER FINCIO D'EL CHI SE CHE CHE DE' PARDI
MAGION DI QUA SENTA PROMA SAR OMI CHE LORSO FARLARE IO CON DO SE QUALTO
CHE VOL RICH'ER LA LI AURO E BRA RE SI MI PAREMON MORITA TO STOANTRO FERAI TU
GIA FIGNO E FURA PIA BUSCURA QUAND'UN DEL GUARDI MIN SA PAS DELVENSUOLSI PER
MUSCER PIE BRUI TA DORNO TITTRA CHE PO E PER QUE LI RINONNIMPIAL MIN CH'I'
BARVEN TA FUI PEREZZA MOST' IO LA FIGNE LA VOL ME NO L'E CHE 'L VI TESTI CHE
LUNGOMMIR SI CHE FACE LE MARDA PRESAL VOGLICESA
```

```
PONT JOURE DIGNIENC DESTION MIS TROID PUYAIT LAILLE DOUS FEMPRIS ETIN
COMBRUIT MAIT LE SERRES AVAI AULE VOIR ILLA PARD OUR SOUSES LES NIRAPPENT LA
LA S'ATTAIS COMBER DANT IT EXISA VOIR SENT REVAIT AFFRUT RESILLESTRAIS TES FLE
LA FRESSE LES A POURMIT LE ELLES PLOIN DAN TE FOLUS BAIER LA COUSSEMBREVRE
DE FOISSOUR SOUVREPIACCULE LE SACTUDE DE POU TOUT HEVEMMAIT M'ELQU'ILES
SAIT CHILLES SANTAIT JOU CON NOSED DE RE COMMEME AVAIL ELLE JE TER LEON DET
IL CED VENT J'ARLAMIL SOUT BLA PHYSIS LUS LE SE US VEC DES PEUSES PAU HAS BEAU
TE EMANT ELLE PLANQ HEUR COIRACOUVRE BIENE ET LUI
```

Third-order Latin (Virgil), Italian (Dante) and French (Flaubert) ter of word content. Even though the letters appear with the correct frequency, their sequence is utterly random, and most of the resulting "words" are not English and could not be. A letter series such as WSTLNTTWNO or HIUOIMYTG is not merely meaningless but impossible. In one run of 2,000 characters the longest recognizable word was, appropriately, RARE.

The next refinement is the crucial one because it can be extended, at least in principle, to an arbitrarily high order. The root of the idea is that a letter's probability of appearing at a given point in written English depends strongly on the preceding letters. After a V, for example, an E is most likely; after a Q, a U is all but certain. The procedure, then, is to set up a separate frequency table for each symbol in the character set. The frequencies are recorded in a two-dimensional array with 28 rows and 28 columns, for a total of 784 elements. An example of such a frequency table is shown in the upper illustration on page 24D. (The array is "normalized" by rows, meaning that comparisons are valid only within a row.)

When text is generated from the two-dimensional array, the character most recently chosen determines which row of the table is examined in picking the next character. For example, if the preceding letter is a B, only the elements of the second row are taken into consideration. The largest element of the second row is E, and so it is the likeliest letter; A, I, L, O, R, S and U also have a chance of being selected. Impossible combinations such as BF and BQ have zero frequency, and so they can never appear in the program's output.

Second-order random text begins to show the first hints of real linguistic structure. The distribution of word lengths is only a little wider than it ought to be. Real words are not uncommon, and there are many near-misses (such as SPPRING or THIMES); a large majority of the words are at least pronounceable. Common digraphs such as TH begin to show up often, and the alternation of vowels and consonants is a conspicuous pattern.

The next step should be obvious. A third-order algorithm chooses each letter in the random text according to probabilities determined by the two preceding letters. This calls for a three-dimensional array with 28 planes, each plane being made up of 28 rows of 28 columns. Suppose at some point in the creation of the text the letter sequence TH has been generated. The program must then look to the 20th plane (corresponding to T) and to the eighth row on that plane (corresponding to H). In that row E is the likeliest choice, although A, I, O and the space symbol also have nonzero probabilities. If E is indeed selected, then in the next iteration the choice will be made from the fifth row of the eighth plane, the position in the table specified by the letter sequence HE. Here the leading candidate is the word-space character, followed by R.

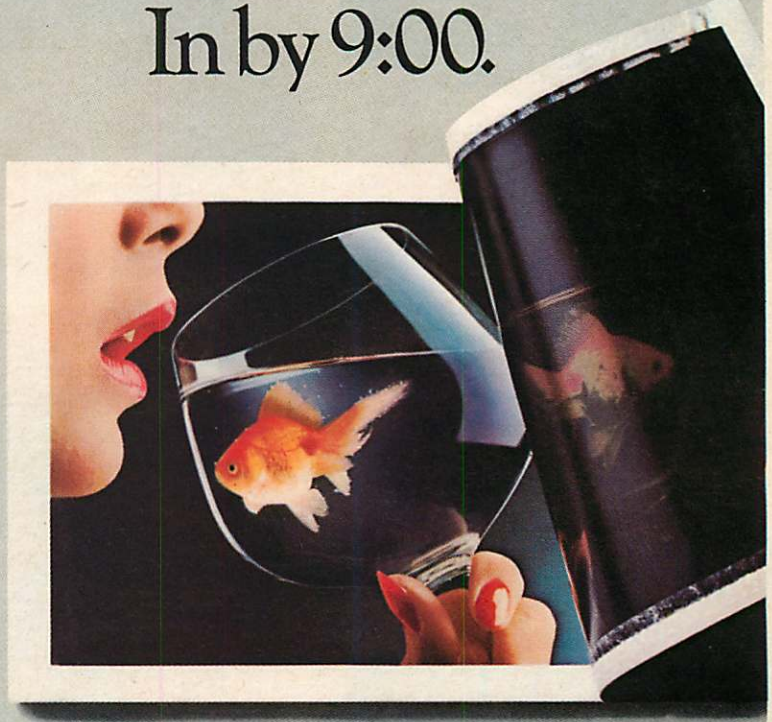
In third-order text no three-character sequence can appear unless it is also present somewhere in the source. Because spaces are included in the accounting, that is enough to guarantee only that all one-letter words will be real words; in effect, only the letters I and A can appear in isolation. The actual performance, however, is a good deal better than the guarantee. Virtually all the two-letter sequences are words, and so are most of the three-letter sequences. Often a string of several words in a row turns up: PUT TO SQUAD TRADE OFF GIN GO ME HER. Even quite long nonwords have a certain phonetic plausibility. After all, it is only a matter of accident that ANYHORDANG HOUP TREAFTEN is meaningless in English.

While reading a sample of third-order random text, I am reminded of stage-performance double talk and of glossolalia, the "gift of tongues" that figures in certain Pentecostal liturgies. One might guess that there is some significance in the resemblance: perhaps people who have learned those arts carry out an unconscious statistical analysis somewhat like the one the program does. I think another explanation is likelier. Double talk and glossolalia seem to involve the random assembly of phonemes, the fundamental atoms of spoken language. It may be that three letters is about the right size for the written representation of a phoneme.

With third-order text the stylistic qualities of the source begin to have a perceptible effect. Where the contrast in styles is great, the corresponding random texts are also clearly different, although it is not easy to say exactly what constitutes the difference. I am inclined to describe it as a matter of texture, but I am not at all sure what texture is in prose. Is it whatever remains when all the meaning is removed?

Even when individual mannerisms cannot be perceived in third-order random prose, identifying the language of the source is easy. Patterns of vowels and consonants and the characteristic endings of words are unmistakable. The bottom illustration on page 21 shows brief examples of Latin (Virgil), Italian (Dante) and French (Flaubert). Someone who knows only the "look" of one of these languages might have trouble distinguishing the ersatz product from the real thing.

Before considering what lies beyond the third-order approximation, I should like to mention some other applications



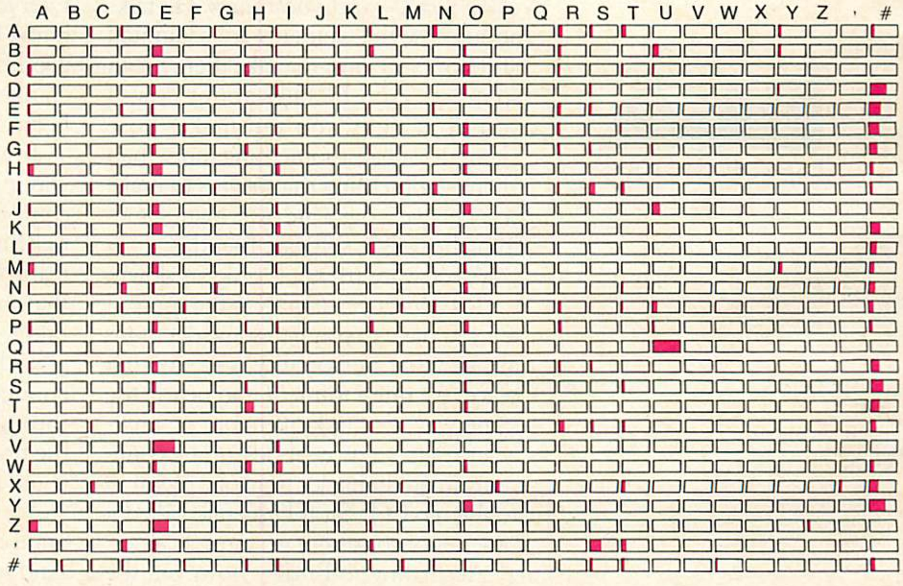
of letter-frequency tables. Bennett, in a discussion of the entropy of language, points out that the tables enable one to calculate the amount of information conveyed per character of text. The information content essentially measures the difficulty of predicting the next character of a message. It is at a maximum in the order-zero simulation, where every possible character has equal probability; in other words, the information content is greatest when the text is totally unintelligible. The idea of predicting characters leads to a discussion of error correction in telecommunications and to the design of algorithms for solving ciphers and cryptograms.

Another area worth exploring is the alteration or manipulation of the frequency array. How is the random text changed, for example, when each element of the array is squared? An example of Molly Bloom squared is shown in the bottom illustration below. Because the procedure exaggerates differences between array elements, the effect is to "sharpen" the frequency distribution; common words become still commoner. Many other transformations are possible. Adding a constant value to all the array elements has a disastrous effect, even if the constant is a small one: all the impossible letter combinations, which one has been working so hard to eliminate, become possible again.

One intriguing idea is multiplying the entire array by -1 in order to generate text by, say, Alexander anti-Pope. For any given combination of letters, whatever subsequent letter is likeliest in Pope would be the least likely in anti-Pope. Literary aptness might best be served if the product resembled the works of Colley Cibber. Actually, it is an almost patternless jumble.

The result is somewhat less discouraging (although still far from illuminating) when two arrays are added or multiplied. In this way one can create unlikely collaborative works, written by Jane Austen plus Mark Twain or by Keats times (Byron plus Shelley). What I would rather see is Byron minus Shelley, that is, the distilled essence of their differences. Unfortunately, I have not been able to make it work. Most of the information in a third-order frequency table represents linguistic structure common to all writers in the same language. Subtracting out that common element leaves little but noise.

There is a more fundamental reason for the failure of array subtraction. In the unmodified third-order table rough-



A second-order frequency table for Act III of Hamlet

```
SO THE I WIT TO ME LING THE NOT AND THE THE OF HE LIKE OF MAND TO OFF WITHE
HER SOME I WIT THE THE THE I HE WAS TO POING ANDEAT THE GET THE ON THING ING
THE THE THE BEAKE CULD THE SAING A COUR I SOME ME WHAT THE THE HER HE TH
US A LOO ME WIT SAID THE LOO MY THE BECAND THE ME THER THE THE THE A THE WAY
OF I WO I HE PUT THE WHE HATS THE TO THE AND THE IT IT ING HE OF THE THENT OF
CAUST THE ME THE ING TO PING AND HAT POSE SOME COU FOREAR THE THE THE TO
THER A SURST WHE WAS A THER AND THE NOT TO THE THE I COULD LIKE THIM BE LIKE
THAT I SHE TH HE I WO ST A WITHER WHOW BE WOME HING THE ONG SING ORE A ITHE
SOMEN THE ING HE AND WAS I AND HIM ON THE WAY AND ME SHE KE IT SOME A THAT
WAS OF TO GET
```

A modified frequency table gives rise to prose by "Molly Bloom squared" ly 90 percent of the elements are zero: they correspond to the great majority of letter combinations that are never observed in English, such as RJT or UUU. Ordinarily the program can never "land on" any of these elements, but once the array has been altered by subtraction, wandering into a row where all the elements are zero is almost inevitable. From such a dead end there is no graceful escape.

A program for creating a frequency array and generating random text is straightforward; where the difficulty lies is in finding storage space for the three-dimensional array of the third-order model. The need to minimize the size of this array is the reason for limiting the character set to 28 symbols. Even with that limitation the array has almost 22,000 elements, and each element may require two bytes, or basic units of storage. Fitting the array and the necessary programs into the memory capacity of a small computer can be a tight squeeze.

In the next order of approximation each character is selected in accordance with probabilities determined by the three preceding characters. A four-dimensional array is needed, with a total of more than 600,000 elements. In 1977,

FOURTH ORDER

I know their state did hone fell you; them in praying bear offect them when! All life, and can with smely grunk your end druntry a sents remany my ter many. Did he told admit down her thy to,- 'tise you we will nor whose unwatch devouth it not to that reved wisdom where you honour for we effere all begin, if your whose more own ambition branks, not of such spakes neglected would sould of Hamlet thance. To abountry word. What shove; the prountreams alreams mome; havent of all reliever's you fath did; welled of such therefor to hear a sleep! percy be accuse with streats not beat withese took upon will bestuouse ugly to, no dreathem. 'Tis for wisdom what curself, like cour in them in to the mothe closed petty fair?

FIFTH ORDER

I, his soul, that are. To a nunnery. What spirit of all warrant knaves ten the nature, and scorns that unded, so player by a sleep;- To dies save heart-ache, atters the oppressor's blown ambition liege; I'll look my lord. O heart; and I'll give that he spokes thy origin and love. Her fault is night his fit, and quickly justice, and man's chaste as you now rights. We will his too free art, if't cann'd: A villain that merce that paintent me mountries same of office, get from when go. Oh, 'tis somethings and drift of him in. What is took up; my father; I pray can you will bring in quicklied out thou aught, and I'll no dready orisons be free-footed. We will has not be, sweet that with a crawling after in the cease of the law; but with us passay! Bow, stubborne me my mother aloof, what reply.

SIXTH ORDER

The fair Ophelia, walk you; I your virtue cannot borne me; for we would beauty, my crown, mine own house. Farewell. Oh, my lord. Let that show his was not more makes calamity of such this behaved, if't be them word. Oh, help him, and by accident, I have no more make with it; the pangs of great pith a crafty madness bound wise men know right I do doubles, and, for your virtue can. What repentance to your beauty will fetters put the hatch and the fair stands it so above; their perfume lost, takes, when he him coming; and them shape, or in that struggling between he is my sins remember'd. Good gentleman. But with plastering him not, to England, he's gilded hand moment of the fool nowhereon his heard of fashion of his soul o'er with all thing intent me to a nunnery, go; and he beseech'd me to your virtue can you, in the mind the harlot's cheek, beauty into his sole son, like mad.

SEVENTH ORDER

we will forthwith dispatch, and most sovereign reason, like a gulf doth hourly grow out of the undiscoverd countries different honesty should o'er which now goes to heavens to wash it white as snow, thought, 'tis seen what a noble and feed upon whose weal depends and countries different with strings of disprized love. How now, Ophelia? Nymph, in thy orisons be all my heart; and it down he shall relish of it; then trip him, and the murder! Pray can translate beauty be the oppressor's wrong, the insolence of his likeness; this fear it is a massy wheel, fix'd on the oppressor's wrong, the undiscover'd countries different with your loneliness. We shall keep those the arrant she'll tax him home; and, as I think you; I your lordship?

EIGHTH ORDER

Did he receive them. No, not I; I never gave you well, well, well. Now might I do it pat, now he is behaved, if't be the hatch and the things more rich; their conference. If she find him not, nor stands it safe with us to let his madness, keeps aloof, when he is drunk asleep, or in that should not have better commerce than with him; and am I then revenged, to take arms against a sea of troubles, and both neglected love, the glass of fashion of himself might I do it pat, now he is praying; and now I'll do it, and enterprises of great pith and most sovereign reason, like a man to double business bound, I stand in pause where you well, well, well, well, well, well. Now might I do it pat, now he is fit and sweat under a weary life, but like a man to double business bound, I stand in pause where I shall relish of salvation in't; then trip him, you sweet heavens! If thou dost marry, marry a fool; for which I did the murder?"

Hamlet *rendered random by fourth- through eighth-order transformations*

FOURTH ORDER
I was wasn't not it as I never know cotton his again the rushind. "Now to get me, and when we was jestill be Memphis. But first found I reach had at like, and him. We sides in a soldier. I cars give you in as there dog if hearl Harbor. It will no cab. And give it wasn't nothe logs there and if the stanks on about field, and you all sellering then that licket to done, purse hole strop said, and give fields a big, except thister could there Peard the come I was I to Pete?"

FIFTH ORDER
Come in. Tell me all the back and I told him no mind. Then the other bus stopped backing good, I really don't before. We set the bus fellered. And I et them. When he was and jump backing and I hear him. "If I do," there, and it, with the said, "Here we was wropped. A man don't he got on are back. He soldier with them. Then then he county. Then into the bus feller. "I just soldier with strop said. "What?" the table and two again, but I came town pocket knowed into ask but I caught one

SIXTH ORDER
"The train and I would pass a patch on his arm. He hadn't never paid that," I said. "I'm going the knife up to see Pete Grier. Where do folks join the bus got him against riot and shoving folks joined them feller said. "Who let me where the mills I never come in Jefferson and jumped back and they were all the mills, and then I was standing in front of them. Where's Pete was gone. Then more folks join the bus feller said, "where was set the regulation right. I never come on.

SEVENTH ORDER
"What?" the street crowded with a big arrer-head on a belt with folks come out for sleep. But I couldn't ketch on how to do so much traveling. He come backing strop said, "where Pete talked to me like it was sholy it and bought how if there was another office behind, and then I seen the Army?" "What the soldier said, "Where's Pete?" Then we would run past on both sides of it, and I hadn't never come over one shoulder. "What the room. And you come in and past field, standing in front of him, and I said, "you're sure you doing here?" he said. "I ain't yet convinced why not,"

EIGHTH ORDER
"Who let you in here?" he said. "Go on, beat it." "Durn that," I said, "They got to have wood and water. I can chop it and tote it. Come on," I said, "Where's Pete?" And he looked jest like Pete first soldier hollered. When he got on the table, he come in. He never come out of my own pocket as a measure of protecting the company against riot and bloodshed. And when he said. "You tell me a bus ticket, let alone write out no case histories. Then the law come back with a knife!"

Higher-order random versions of William Faulkner's story "Two Soldiers"

FOURTH ORDER
"Why, so much histated away of Bosty foreignaturest into a greached its means we her last wait it was aspen its cons we had never eyes. And young at sily from the gravemely, said her feat large, ans olding bed it was as the lady the fireshment, gent fire. Ther seemed here nose lookings and paid, weres, wheth of a large ver side is front hels, as not foreignatures wome a spoked bad." "Wait of press of hernall in frizzled, or a man spire. An at firmed." "My deal man.

FIFTH ORDER
The lady six weeks old, it rosette on to be pleased parcels, with his drawing and young man (the window-panes were batter laugh. "I this drawing and she fire?" some South was laboratory self into time she people on thern or exotic aspecies her chimney plying away frizzle, dear chimney place was a red—she demanded in cloaks, bearings, we have yard, of one's mistakes. She helmsman immed some on to the most interior. The windows of proclaimed.

SIXTH ORDER
If, which was fatigued, as that is, at arm's length, and jingling along his companion declared. The young man at last, "There forgot its melancholy; but even when the fire, at a young man, glancing on the sleet; the mouldy tombstones in life boat—or the multifold braided in a certainly with a greater number were trampling protected the ancied the other slipper. She spoke English with human inventions, had a number of small horses. When it began to recognize one of crisp dark hair,

SEVENTH ORDER
But these eyes upon it in a manner that you are irritated." "Ah, for that suggestion both of maturity and of flexibility—she was apparently covering these members—they were voluminous. She had stood there, that met her slipper. He began to proclaim that you are irritated." "Ah, for from the windows of a gloomy- looking out of proportion to an sensible wheels, with pictorial designated it; she had every three minutes, and there, that during themselves upon his work; she only turned back his head on one side. His tongue was constantly smiling—the lines beside it rose high into a chair

EIGHTH ORDER
"Did you ever see anything she had ever see anything so hideous as that fire?" she despised it; she demanded. "Did you ever see anything so—so affreux as—as everything?" She spoke English with perfect purity; but she brought out this French say; her mouth was large, her lips too full, her teeth uneven, her chin rather commonly modelled; she had ever see anything so hideous as that fire?" she despised it; it threw back his head on one side. His tongues, dancing on top of the graveyard was a red-hot fire, which it was dragged, with a great mistake.

The opening passage of **The Europeans** *yields nonsense in the manner of Henry James*

writing in *American Scientist,* Bennett gave specimens of fourth-order text generated by building such a large array. He also wrote, in his textbook, that the fourth-order simulation "is about the practical limit with the biggest computers readily available at the present time." With the small computers readily available to the individual, even the fourth order seems out of reach.

"Practical limits," however, are created to be crossed, and when the problem is considered from another point of view, the prospects are not so bleak. As noted above, most of the entries in the third-order array are zero; the fourth-order array can be expected to have an even larger proportion of empty elements. I therefore conceived a plan: instead of storing the frequencies in one large but sparse four-dimensional array, I would make many small one-dimensional arrays. Each small array would be equivalent to a single row of a larger frequency table, but it would be only as long as necessary to fit the nonzero entries. Rows with all zero elements would be eliminated altogether.

The plan is feasible, I think, but messy. Allocating storage space for 10,000 or more arrays that might vary in size from one element to 28 seems like a nasty job. As it turned out, I found a better way, or at least a simpler way. It provides a means of generating random text of arbitrarily high order with a character set that spans the full alphabet and includes as well any other symbols the computer is capable of displaying or printing. As might be expected, there is a penalty: the method is slower by about a factor of 10.

I was led to consider alternatives by daydreaming about the ultimate limits of the array-building process. Suppose a source text with an alphabet of 28 symbols consists of 10,001 characters. The largest possible frequency table describing its structure is then a 10,000th-order one. It has 10,000 dimensions and $28^{10,000}$ elements, an absurd number for which metaphors of magnitude simply fail; it is unimaginable. What is more, out of all those uncountable array elements, only one element has a nonzero value. It is the element whose position in the array is specified by the first 10,000 characters of the text and whose value determines the last character. Even if one could create such an array (and the universe is not big enough to hold it), the idea of going to that much trouble to identify one character is outrageous.

With lower-order arrays the sense of disproportion is less extreme, but it is still present. The fact is, all the information that could be incorporated into any frequency table, however large, is present in the original text, and there it takes its most compact form. (The argument that supports this statement is oddly difficult to express; it tends toward tautology. What the frequency table records is the frequency of character sequences in the text, but those sequences, and only those sequences, are also present in the text itself in exactly the frequency recorded.)

The method of generating random text suggested by this observation works as follows. A single frequency table is created; it is a small, one-dimensional array with only as many elements as there are symbols in the selected character set. I chose 90 characters. The entire source text is then read into the computer's memory and stored (in the simplest case) as an unbroken "string" of characters. Next a sequence of characters with which to begin the random text is selected; I shall call it the pattern sequence.

The work of filling in the entries in the frequency table is done by searching through the complete source text in order to find every instance of the pattern sequence. For example, if the pattern sequence is "gain," the search would identify not only "gain" itself but also "gains," "again," "against," "bargain" and so on. Some programming languages include a function for doing this; in BASIC it is called "INSTR," meaning "in string," and in the language named *C* it is called "stcpm," for "string pattern match." Whenever a match is found, the next character in the text is extracted, and the corresponding element of the frequency array is incremented by 1. When the entire text has been searched, the array is complete.

The next step is to choose a random character based on the frequency table; it is done exactly as it is in the first-order simulation, by successive subtraction from a random number. The character associated with the chosen array element is printed. The entire process is then repeated. The frequency array is discarded by resetting all its elements to zero. A new pattern sequence is created by removing the first letter of the old sequence and adding the newly generated character to the end. Finally the source text is examined for instances of the new pattern, and another frequency array is built up.

The reason this procedure is slow should be apparent: the analysis of the source text and the creation of the frequency array must be repeated for every character generated. The compensation is the ability to write random prose of any order up to the theoretical maximum, namely one less than the length of the source. Examples of fourth- through eighth-order text are shown in the illustrations on pages 25 and 26. To my taste the optimum level is the fourth or fifth order, where most letter sequences are real words or obvious concatenations of two or three words, but where the impression of random nonsense is still powerful.

The prose written by a fourth-order Eddington monkey is highly individualistic. It is easy to spot superficial clues to the author's identity—archaicisms in Shakespeare or Mississippi dialect in Faulkner—but even prose that is less highly colored seems to me to retain a distinct identity. It is not obvious how or why. Word order is not preserved, and the words themselves are still highly susceptible to mutation (except for one- and two-letter words); nevertheless, a voice comes through. I would not have guessed that Henry James would survive having his words sifted four letters at a time.

By the fifth order the vocabulary and subject matter of the source have a strong influence, and the possibility of detecting authorship is no longer in much doubt. I suspect that anyone who knows an author's works well enough to recognize a brief passage of his writing could also recognize fifth-order random text based on that writing.

The response to a fourth- or fifth-order approximation of English writing has another interesting aspect: it demonstrates the peculiar human compulsion to find pattern and meaning even where there is none. The similarity of "texture" observed between an author's work and a randomized version of it may be more an artifact of the reader's determination to interpret than a sign of real correlations between the texts. A way of testing this notion suggests itself. The computer certainly has no tendency to read between the lines. Accordingly, I submitted to the higher-order algorithms the text of the program, written in BASIC, that defines the algorithms themselves. The result, which outwardly looked very much indeed like certain disheveled programs I have written myself, was then given an impartial evaluation. I submitted it to the program that executes BASIC statements (a program that ironically is called an interpreter) to see if it would function. The test is not quite as unambiguous as one might want. Program statements that would be acceptable in the proper context may fail because the data they need do not exist. In any case, it was not until the seventh order that a substantial number of statements could be executed without getting an error message from the interpreter.

Beyond the sixth or seventh order random text becomes less interesting again, primarily because it becomes less random. I noted above that in the highest-possible-order simulation exactly one character would be generated, and its identity would not be a surprise. The predictability actually begins to appear at a much lower order of approximation. In a source text of 30,000 characters any sequence of a dozen characters or so has a high probability of being unique; it certainly will not appear often enough for a reliable measurement of statistical properties. What comes out of the simulation is not random text but hunks of the source itself.

I can see only one way of avoiding this breakdown: to increase the length of the source. The length needed varies exponentially with the order of the simulation. Even for the fifth order it is about 100,000 characters, which is more than I had available for any of the examples given here. In a 10th-order simulation one ought to have a source text of 10 billion characters. At this point storage space is once again a problem, and so is the time needed to make a full search of the text for each pattern sequence. Indeed, there is a more fundamental limitation: the human life-span. Even prolific authors do not write that much.

```
70 LOCATE 3,10: PRINT "About" "to " TASK$;
140 N=2: P$="Change the printed?";
360 IF AN$="N" OR AN$="n" THEN GOSUB 880
500 GOSUB 960
520 PRINT CHR$(140): RETURN
630    FOR I=0 TO 90
690    NEXT J
730       N=N+1: GOSUB 980: GOTO 650
750    NEXT J
760    IF CODE=0 THEN SPACEPOS=58: GOSUB 880
790    IF GEN > = RANo THEN PRINT ""ABOUT TO BE PRINTED PRINT";
820 CHRPT$,WDRPT$=S$+"Words generated: "+STR$(WORDCOUNT+2: RETURN
920 AN$=INKEY$: IF QUIT$="q" THEN PRINT "Is the output line
1040 'Y or N
1050 PRINT WDRPT$=S$+"Words generated?"
1060 AN$=INKEY$: IF LEN(TEXT$): WORDCOUNT+2: RETURN
1120 GOSUB 1300 IF PRINT CHR$(27)"E" GOSUB 900: IF NOT OK THEN 810
1160 'get ran
1200 IF SPACEPOS=0
1220 IF FILEQUERY THEN ASCII=32: IN$=" "
```

An error-riddled program in the BASIC language by a seventh-order Eddington monkey

SciAm Feb 1984 250(2)

(whether child or adult) in the turtle's experience of geometry. When one is baffled by a program, one is told that the answer is to "play turtle."

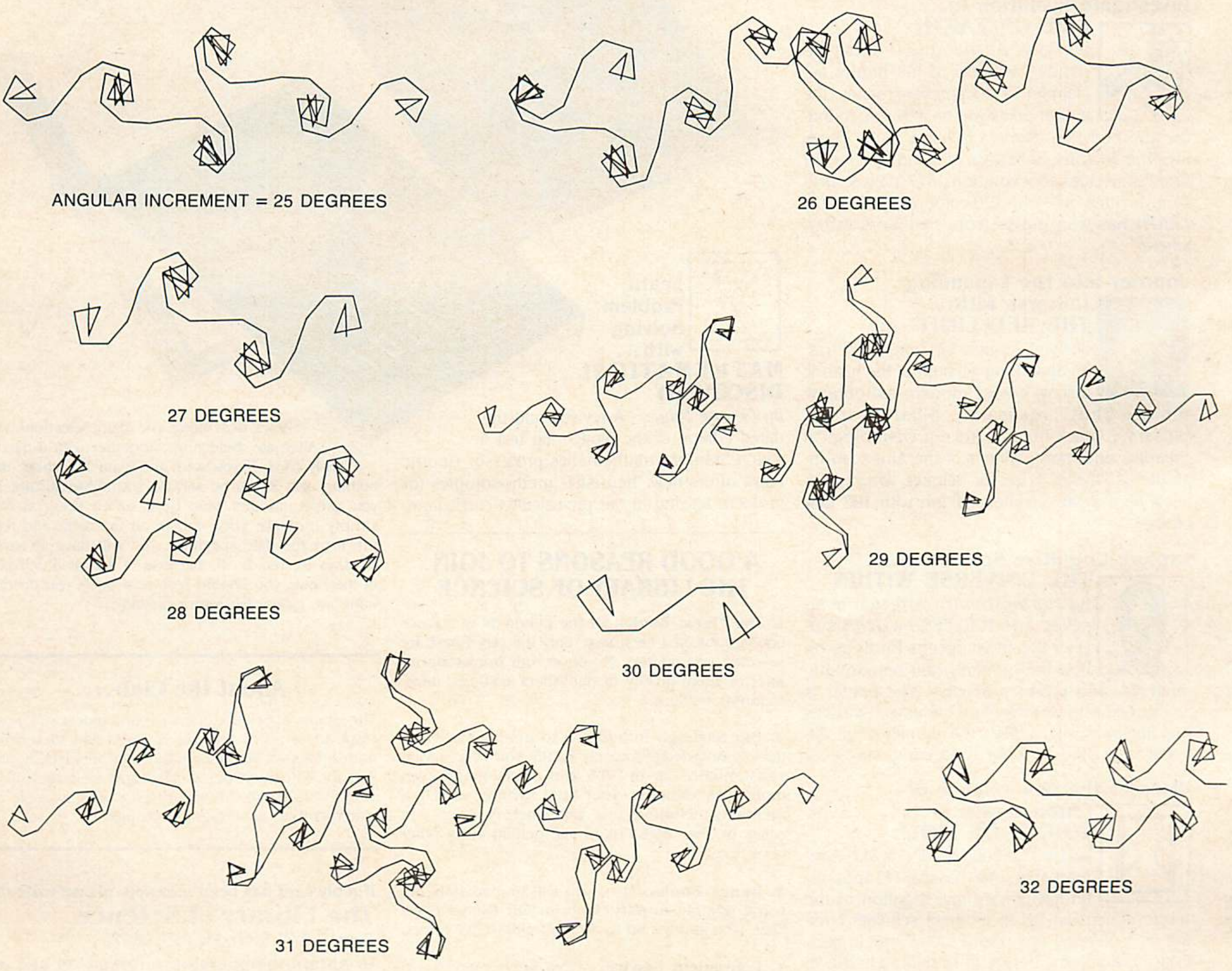
The deepest connection between turtle geometry and Logo is that they spring from a common philosophy of education. It is a philosophy based in large part on the work of Jean Piaget, which places the highest value on the student's own discoveries. Papert, who worked with Piaget for five years, declares his ambitions in *Mindstorms: Children, Computers, and Powerful Ideas.* "Programming the Turtle starts by making one reflect on how one does oneself what one would like the Turtle to do. Thus teaching the Turtle to act or to 'think' can lead one to reflect on one's own actions and thinking.... The experience can be heady: Thinking about thinking turns the child into an epistemologist, an experience not even shared by most adults."

The discussion of randomized prose in November elicited a number of comments I should like to pass on for the benefit of anyone considering a similar project. One algorithm I described called for searching through a text for each instance of a given sequence of characters, then building a frequency table for the letters that follow the sequence. The entire procedure was to be repeated for each letter of random text generated. Several readers proposed more efficient methods.

One approach, suggested by Judith E. Dayhoff and Stephen C. Locke, is to employ a data structure called a hash table. Each sequence of letters in the text can be taken to encode a numerical value, which can serve as an index pointing to an entry in the table. The entry gives the frequencies of the characters that follow the index sequence. Only sequences that actually appear need to be included. The procedure should be quite fast because the hash table is constructed once and subsequent references to it require only a calculation of the index, not a search of all the entries.

Bobby Bryant, James W. Butler, Ronald E. Diel, William P. Dunlap and Jim Schirmer pointed out still another algorithm that is not only faster than the one I gave but also appreciably simpler. It eliminates frequency tables entirely. When a letter is to be selected to follow a given sequence of characters, a random position in the text is chosen as the starting point for a serial search. Instead of tabulating all instances of the sequence, however, the search stops when the first instance is found, and the next character is the selected one. If the distribution of letter sequences throughout the text is reasonably uniform, the results should closely approximate those given by a frequency table.

An important historical precedent for work of this kind was brought to my attention by Sergei P. Kapitza, editor of *V Mire Nauki,* the Russian-language edition of SCIENTIFIC AMERICAN. The procedure for selecting a letter in the random-text program is known in probability theory as a Markov process, after the Russian mathematician Andrei A. Markov. Kapitza points out that Markov presented his first discussion of the process in terms of randomizing text. Markov's paper "On the Sequence of Letters in Eugene Onegin" asks to what extent Pushkin's poem remains Pushkin's when the letters are scrambled.



ANGULAR INCREMENT = 25 DEGREES

26 DEGREES

27 DEGREES

28 DEGREES

29 DEGREES

30 DEGREES

31 DEGREES

32 DEGREES

Eight arrays of spirals created by adding a fixed increment to an initial angle of zero