IN QUEST OF A PANGRAM (PART 1)

LEE SALLOWS
Nijmegen, The Netherlands

> *In the August 1991* **Word Ways**, *Douglas Greenwood mentions Kee Dewdney's widely-referenced Scientific American account of my search for a self-descriptive pangram [1]. In fact, Dewdney's column was based on a long and rather technical article of mine that appeared subsequently in Abacus [2]. The following is a shortened adaptation of the first part of the original.*

The Pangram Problem

In 1983, a Dutch newspaper carried an astonishing translation of a rather tongue-in-cheek sentence of mine that had previously appeared in one of Douglas Hofstadter's Scientific American columns [3]. The translation was by Rudy Kousbroek, a well-known journalist in Holland (two chapters of his 1984 book, De Logologische Ruimte, appeared in the November 1986 and November 1987 Word Ways). Here is the original sentence:

> Only the fool would take trouble to verify that his sentence was composed of ten a's, three b's, four c's, four d's, forty-six e's, sixteen f's, four g's, thirteen h's, fifteen i's, two k's, nine l's, four m's, twenty-five n's, twenty-four o's, five p's, sixteen r's, forty-one s's, thirty-seven t's, ten u's, eight v's, eight w's, four x's, eleven y's, twenty-seven commas, twenty-three apostrophes, seven hyphens and, last but not least, a single !

Composing such self-descriptive sentences can be exacting, to say the least. The process has points in common with playing a diabolically conceived game of patience. How does one begin? My approach is to decide first what the sentence is going to say and then make a flying guess at the frequencies of each sign. Writing out this provisional version, the real totals can be counted up and the initial guess updated. The process is repeated, trial and error leading to successively closer approximations. This opening soon shades into the middle game. By now all of the putative totals ought to have been corrected to within two or three of the true sums. There are, say, 9 f's but only seven being claimed, and 27 real t's where twenty-nine are declared. Switching seven with the nine in twenty-nine corrects both totals at a single stroke. Introducing further careful changes with a view to pulling off this sort of mutual cancellation of errors should eventually lead to the final phase.

The endgame is reached when discrepant totals number some four or less. The goal is in sight, but, as in a maze, proximity can be misleading. For instance, suppose painstaking effort has at last yielded a near perfect specimen: only the x's are wrong.

Instead of the **five** claimed, in reality there are 6. Writing **six** in place of **five** will not merely invalidate the totals for e, f, s, and **v**; the **x** in **six** means that their number has now become 7. Yet replacing **six** with **seven** will only return the total to 6. What now?

Paradoxical situations of this kind are the norm in this activity. Interlocking feedback loops magnify tiny displacements into far-reaching upheavals; harmless truths cannot be stated without disconfirming themselves. It seems the only hope of dehydrating this Hydra and getting every snake-head to eat its own tail lies in doctoring the text accompanying the listed items. In looking at the above case, for example, only a fool will fail to spot instances where style has been sacrificed to arithmetic. This is what made Kousbroek's translation so stunning. Totals excepted, his rendering not only adhered closely to the original in meaning, it was simultaneously a self-descriptor in Dutch!

Or at least, so it appeared at first sight. Counting up, I was amused to discover a few incorrect totals. So I wrote to the author pointing out these discrepancies. This resulted, a month later, in a second article in the same newspaper. Kousbroek wrote of his dismay on being caught out by the original author of the sentence, "specially come over from America, it seems, to put me right." The disparities I'd pointed to, however, were nothing new to him. A single flaw had been spotted in the supposedly finished translation on the very morning of submitting his manuscript. But a happy flash revealed a way to rectify the error in the nick of time. Later a more careful check revealed that this brainwave had in fact introduced even more errors elsewhere. He's been awaiting "the dreaded letter with its merciless arithmetic" ever since. The account went on to tell of his titanic struggle in getting the translation straight. The new version was included; it is a spectacular achievement. However, the tail concealed a sting. At the end, Kousbroek threw out a new self-descriptor of his own:

> Dit pangram bevat vijf a's, twee b's, twee c's, drie d's, zesenveertig e's, vijf f's, vier g's, twee h's, vijftien i's, vier j's, een k, twee l's, twee m's, zeventien n's, een o, twee p's, een q, zeven r's, vierentwintig s's, zestien t's, een u, elf v's, acht w's, een x, een y and zes z's.

A finer specimen of logological elegance is scarcely conceivable. The sentence is written in flawless Dutch beyond hope of refinement. It says, "This pangram contains five a's, two b's, .., one y, and six z's." Following this came a devilish quip in my direction: "Lee Sallows will doubtless find little difficulty in producing a magic English translation of this pangram." Needless to say, I didn't manage to find a single error in this sentence of his!

Computerized Construction

Rudy's playful taunt came along at a time when I was already thinking about computer-aided construction of self-descriptors. (Recall this is 1983; advertisements of the "You just bought a personal WHAT???" kind are still in the future.) At first I envisaged no more than an aid to hand composition: a program to count letters so as to yield feedback on the results of keyboard-mediated surgery performed on a sentence displayed on screen. Later I began wondering about a program able to cycle through the list of totals and make automatic corrections along the way. Could solutions be evolved through a repetitive process of mutation and selection? Experiments were made, but processing always became trapped in an endless loop of repeated exchanges. What seemed to be needed was the ability to look ahead so as to evaluate and compare the results of prospective word changes.

I was pondering this problem when Kousbroek's challenge presented itself and sent me off on a different tack. Its sheer hopelessness caught my imagination. But was it actually impossible? What a comeback if it could be brought off! The task was to complete a sentence beginning "This pangram contains..". A solution, were it discoverable, must in a sense already exist out there in the abstract realm of logological space. It was like seeking a number that satisfies certain mathematical conditions. And nobody -- Kousbroek least of all -- knew whether it existed or not. The thought of finding it was tantalizing. Reckless of long odds, I put aside programs and launched into a resolute attempt to find it by hand trial. It was a foolhardy quest, a search for a needle in a haystack without even the comfort of knowing that a needle was concealed there. In the end, I had only the consolation prize of a near solution: all totals correct save one, 21 t's instead of the 29 claimed. With a trick it could even be fudged:

```
ttttt
  t
  t
  this pangram contains five a's, one b, two c's, two d's, twenty-
  seven e's, six f's, three g's, five h's, eleven i's, one j, one
  k, two l's, two m's, twenty n's, fourteen o's, two p's, one q,
  six r's, twenty-eight s's, twenty-nine t's, three u's, six v's,
  ten w's, four x's, five y's, and one z.
```

To the purist in me, that single imperfection was a hideous fracture in an otherwise flawless crystal. However, a promising new idea now presented itself. The totals in this near solution must be pretty close to those in the real solution, assuming it existed. Why not use it as the basis of a systematic computer search through neighbouring combinations of number-words? Each total above could be seen as centered in a range of, say, 10 consecutive possibilities within which the perfect total was likely to fall. With these ranges defined, a program could generate and test every possible combination. The test would consist in comparing sets of potential totals with the computed letter frequencies they gave rise to, until an exact match was found. Or until all cases had been examined. Blind searching might succeed where cunning was defeated. Work on the program began at once.

It isn't necessary to test all 26 totals, since in English there are just 10 letters which never occur in low valued cardinals: A,B,C,D,J,K,M,P,Q,Z. Totals for these letters can thus be determined from the initial text and filled in directly:

> This pangram contains five a's, one b, two c's, two d's, ? e's, ? f's, ? g's, ? h's, ? i's, one j, one k, ? l's, two m's, ? n's, ? o's, two p's, one q, ? r's, ? s's, ? t's, ? u's, ? v's, ? w's, ? x's, ? y's, and one z.

This leaves 16 critical totals. Counting up shows there are already 7 e's, 2 f's, .., 1 y: 16 constants which must be added to those letters occurring in the trial list of totals. Similarly, in the program, number words are replaced by profiles or 16-element lists representing their letter content. The profile for twenty-seven would then be:

E	F	G	H	I	L	N	O	R	S	T	U	V	W	X	Y
3	0	0	0	0	0	2	0	0	1	2	0	1	1	0	1

Profiles for all numbers up to fifty were stored in memory and a label associated with each. These labels were simply the corresponding decimal numbers; the label for **twenty-seven** would thus be 27.

Starting with the lowest, a simple algorithm could now generate successive combinations of labels (i.e., numbers) drawn from the 16 predefined ranges. Each set would be used to call up their associated set of profiles. The 16 profiles would then be added together element for element, and the resulting sums in turn added to the above set of constants so as to form the complete **sentence profile**. All that remained was to check whether the numbers in the sentence profile coincided with the present set of labels. If so, the self-descriptor had been found. If not, generate the next combination and try again.

At length, the program was finished and set running as a batch job on a mainframe computer at the University of Nijmegen. Even so, the 16x16 additions needed to calculate each sentence profile made for slow processing; average speed was only about 10 candidates per second. Every morning I would hasten to call up the job file, running my eye down the screen in search of EUREKA! But as day succeeded day without result, the question of how long it would take to exhaust all possibilities gradually loomed in importance. It was a matter I had never seriously considered. Alas, the calculation is an absurdly simple one and even now I blush to recall first seeing what it implied. The time needed is 10^{15} seconds. A pocket calculator soon converts this to intelligible units. There seemed to be something wrong with the one 1 was using. Every time I worked it out the answer was ridiculous: 31.7 million years!

I was so unprepared for the blow contained in this revelation that initially I could hardly take it in. The whole object of turning to a computer had been to win speed. Now that the truth had dawned, I began cursing my naivete in ever embarking on such a fool's errand. Doubtless a faster program could be written, but even checking one million candidates a second, it would still take 317 years to run through the 10-deep ranges!

Yet "millions" put me in mind of megahertz, which in turn brought my thoughts to electronics (my profession). This in turn prompted an idea, a fanciful notion, for the first few days no more than an idle phrase repeated in the head, a good title perhaps for a science fiction story: The Pangram Machine. lnitially whimsical, suddenly the vague intuition began to crystallize. ln a flash, l saw how a central process in the program could be simulated electronically. Taking this as a starting point, l tried translating other aspects of the algorithm into hardware. lt worked; it was easy. A few hours later I was thrilled to find the broad outlines of an actual design already clear in my mind.

The Phoenix now emerging from the ashes of the Pangram Quest soared serenely to the sky, smoothly circled, swiftly swooped, and soon bore me off, a helpless prisoner in its relentless talons. For the next three months I would be pouring all my energy into the construction of a high speed electronic Pangram Machine.

The Pangram Machine

How seriously should a word puzzle be taken? Though only the size of a smallish suitcase, the apparatus to emerge from three months' concentrated activity packed more than two thousand components onto thirteen specially designed printed circuit cards. Foresight of this complexity may have dissuaded me from starting. ln the event, the completed machine turned out to involve a good deal more circuitry than originally intended.

As indicated, machine operation mimics that of the program. Readers interested in technical details should consult my original article [2]. For now, suffice it to say that electronic switching speed plus shortened range lengths tailored to individual letters brought running time down to about a month. Technical problems beset the design, but at last came a day when the rocket was ready for launching into logological space. Lift-off came in October 1983, some eight months following Kousbroek's audacious challenge. The ensuing period found me hovering nervously over the machine. There was the nagging worry of reliability. What guarantee was there of faultless operation over so long a period? None, of course. After a while the suspense became nerve-racking. Mornings were worst. On waking, the first thought in consciousness would be: has it halted? It took nerves of iron to go through the morning's ablutions before going down to the living room where the machine was installed on my writing bureau. Opening the door, there would be the flickering gleam of light-emitting diodes signalling the state of the counters as they switched their way through $2.71\text{x}10^{12}$ trial sentences. One million a second for 32 days. It was a torturing experience. The novelty of watching the machine soon wore off, but a single second's distracted attention was accompanied by the thought that another million chances had already elapsed, so perhaps NOW??? .. and my glance would be wrenched back to the twinkling array.

At length, the grains of sand ran out. lt looked like defeat. There never had been a needle in the haystack! But much had happened during the long weeks of waiting. Already, detailed plans were in hand for a Mark II machine in which automatic number-word selectors were to slice running time to a mere two hours! And alternative translations remained to be explored, top of the list being "This pangram comprises..". So it was, with this reconstruction work complete, that some weeks later found me sitting before the machine during its eighth run, following seven previous trials with different verbs, every one of which had ended in failure! How long could this go on? However, suddenly the EUREKA! lamp came on and my stomach turned a somersault. Decoding the diode display into the set of number-words represented, a careful check then verified the following perfect pangram:

> This pangram lists four a's, one b, one c, two d's, twenty-nine e's, eight f's, three g's, five h's, eleven i's, one j, one k, three l's, two m's, twenty-two n's, fifteen o's, one p, one q, seven r's, twenty-six s's, nineteen t's, four u's, five v's, nine w's, two x's, four y's, and one z.

Despite a hangover from the celebration, next morning saw me back at the machine. And the zenith of glory was still to come. Changing **and** to **&** in the rendering of Kousbroek's pangram, a last desperate bid for a perfect magic translation had finally met with success. The Quest for the pangram had ended in triumph!

> This pangram contains four a's, one b, two c's, one d, thirty e's, six f's, five g's, seven h's, eleven i's, one j, one k, two l's, two m's, eighteen n's, fifteen o's, two p's, one q, five r's, twenty-seven s's, eighteen t's, two u's, seven v's, eight w's, two x's, three y's, & one z.

REFERENCES

[1] Dewdney, A.K. Scientific American, October 1984, pp 18-22.
[2] Sallows, L.C.F. Abacus, Vol.2, No.3, Spring 1985, pp 22-40.
[3] Hofstadter, D. Scientific American, January 1982, pp 12-17.

A 275,000-WORD COMPUTER DATABASE

Edward Clarke, Menanhyl, Trenance, Newquay, Cornwall TR8 4DA (tel (0637)860575) has developed Wordsworth, a computer-readable databse of about 275,000 words and phrases of 2 to 28 letters. Shareware registration in the UK is £22 for the program and as much of the database as will fit on a 3.5 floppy; the remainder of the database occupies a further five floppies, priced at £5 apiece. In addition, he has just started a quarterly newsletter which he hopes to sell for £7 per year. The first issue (Nov 1991) outlines his unsuccessful search for a 10-square using his database.