Joe Loughry
Oxford University Computing Laboratory
Wolfson Building
Parks Road
Oxford, OX1 3QD
England

August 30, 2012

Professor Donald E. Knuth
Room 215
Oxford

Dear Professor Knuth:

Here is a real-world problem; I call it the banker's problem. My father asked about it one day; he used to run the cheque processing department of a bank downtown. Every day, he would take in millions of items—cheques and deposit slips—and run them through high-speed sorting machines. Occasionally, a few slips of paper got eaten. At the end of the day, it's not unusual to be a few million dollars out of balance. Someone, usually a clerk, has the job of matching a pile of loose cheques and deposit slips to a thick green-bar printout of daily transactions. As a manual process, it usually takes someone until ten o'clock at night.

The simplest situation, of course, is when one ($5,000) deposit slip matches another $5,000 cheque. But more often it takes a few items to make a match; for example, these sets of items:

$$\overbrace{\{\$127.14, \$100, \$5000, \$312, \$1000, \$100\}}^{\text{credits}} \text{ and } \overbrace{\{(\$539.14), (\$5000), (\$1100)\}}^{\text{debits}}$$

yield solutions:

$$\$127.14 + \$100 + \$312 = (\$539.14)$$

$$\$5000 = (\$5000)$$

$$\$1000 + \$100 = (\$1100)$$

The preferred solution is the smallest number of credits that match one debit.

Here is what I came up with to solve my father's problem: to consider candidate subsets of credits in order by the number of elements in the subset.

All the 1-subsets:

0001
0010
0100
1000

All the 2-subsets:

0011
0101
0110
1001
1010
1100

All the 3-subsets:

0111
1011
1101
1110

All the 4-subsets:

1111

As you can see, the binary sequence is unlike lexicographic order; in fact, *lexicographic order just plain won't work* on a set of a few thousand credits, because the computer spends all its time fiddling with stupidly large subsets of the first few elements in the set before it even gets around to looking at the last one. This way, all the items in the set are seen early on, and the most likely solutions—the small subsets—are looked at early.

Back in 1998, I couldn't find anything like it in the *Encyclopædia of Integer Sequences*. I wrote an unpublished note about it, and a few years later Jano van Hemert and L. Schoofs contacted me with news that it was a solution to a constraint satisfaction problem of theirs. Jano offered a much better

recursive algorithm than the one I had written, so we put all our names on the second version of the paper. I'm afraid it's still never been published, but it's been cited in a few places [Feder 2007, Kandaswamy *et al.* 2008, Maez 2007]. You can imagine my excitement when you began publishing fascicles for Volume 4. Is this possibly something you can use?

The enclosed paper has some pretty graphs contrasting the proposed method with lexicographic and Gray code ordering, and an algorithm for generating the binary sequence.

Respectfully,

Joe Loughry
D. Phil. PRS
Computing Laboratory

References

Elie Feder and David Garber. Towards the computation of the convex hull of a configuration from its corresponding separating matrix, September 2007. `arXiv:0709.2555`.

Gopi Kandaswami, Anirban Mandal, and Daniel A. Reed. Fault tolerance and recovery of scientific workflows on computational grids. In *Eighth IEEE International Symposium on Cluster Computing and the Grid*, pages 777–782, Lyon, France, May 19–22, 2008.

Dave Maez. Perl distinct and complete anagram generator, December 6th, 2007. `http://dharmadevil.com/code/anagram.pl.html`.

Encl: 'Efficiently Enumerating the Subsets of a Set'