A Google Problem

Seenu S. Reddi
ReddiSS at America Online dot Com
October 15, 2005

The following problem is supposedly posed at examinations administered by Google to select their prospective employees. Define a google function g(n) to be the number of ones encountered while writing the numbers from 0 to n. Thus we have g(1) = 1, g(2) = 1, .., g(10) = 2, g(11) = 4, g(12) = 5, g(13) = 6, .. The question posed is to find the next largest number such that g(n) = n. It is highly improbable that I would have solved it in a reasonable time to qualify for employment at Google. Leaving aside the efficacy of problems like these in selecting candidates who are long term contributors to a company, the problem is interesting in its own right (or found to be interesting after struggling for a week or so with the help of computers) and also brings certain facets about numbers which we as humans think they are not capable of. We generally associate numbers to be equitable and equally probable in the sense that any general solution involving numbers will have equal distribution of integers 0 through 9 and they do not play favorites by selecting top four numbers 1, 2, 3, and 4 (ignoring zero for a minute). Solution for this particular problem proves us (or at least me) wrong.

First we will extend the definition of the google function (I would like to change the name once the person responsible for this problem identifies himself) to integers other than 1. We define $g_d(n)$ as the number of times number d is encountered while writing numbers down 0 to n. Thus we have $g_2(10) = 1$ (since 2 is encountered only once from 0 to 10) and $g_2(20) = 3$. Writing a simple program to evaluate these functions, we note:

TABLE 1.

n	9	99	999	9999	99999
g_0	1	10	190	2890	38890
g_1	1	20	300	4000	50000
g_2	1	20	300	4000	50000
g_3	1	20	300	4000	50000
g_4	1	20	300	4000	50000
g_5	1	20	300	4000	50000
g_6	1	20	300	4000	50000
g_7	1	20	300	4000	50000
g_8	1	20	300	4000	50000
g_9	1	20	300	4000	50000

and it is relatively easy to find the following function for g_1 through g_9:

$$g_m(n) = ((n + 1) * \log_{10}(n + 1)) / 10, m \int 0, n = 9, 99, 999, 9999, ..$$
$$g_0(n) = ((n + 1) * \log_{10}(n + 1)) / 10 – yo(n), n = 9, 99, 999, 9999, ..$$

where the yo function (borrowing from Craps parlance) is defined as:

yo(9) = 0, yo(99) = 10, yo(999) = 110, yo(9999) = 1110, yo(99999) = 11110, ..

or equivalently:

yo(n) = (n / 9) – 1, for n = 9, 99, 999, 9999, ..

One can see intuitively that g(n) behaves as n*log(n) and eventually n cannot keep up with its logarithmic growth. It should be pointed out that one can figure out the above g functions without the need of a computer but this depends on the mindset of the individual – whether you want the computer to do the groundwork or rely on the analytical brain skills as the mathematician prefers to do.

So far everything is equitable, g functions look the same for all digits except for 0 (this is attributed to the way we write numbers suppressing leading zeros). Continuing the computer simulation, we note the numbers n for which g(n) = n, n > 1. TABLES 2 and 3 give the g numbers for n up to 20,000,000,000 (twenty billion).

The smallest g_1 after 1 is 199981. (In the table x-- indicates a sequence of ten numbers starting with x). One may expect even distribution of g numbers for 1 through 9 but this is not the case. Initially g_1 is predominant but it stops at 1111111110 (is it the largest ?) but later we find g_8 is predominant. g_5 appears majestically and very sparingly at ten billion and twenty billion. Ten billion has g numbers from 2 through 9, twenty billion has g numbers from 3 through 9 and one may speculate for thirty billion g numbers for 4 through 9 (?). g_0 never appears so far and it is interesting to conjecture whether it ever will (a mathematical proof would be nice). There is also a presence of "round" numbers like 500 million, 10 billion and 20 billion which share multiple g numbers. A mathematical analysis of the distribution of g numbers will be extremely interesting from a theoretical perspective.

From TABLE 1 one may deduce an upper bound for g numbers as10^{100}. (For N = 10^{100}, the g function will be around 10N, and for 10N, g function will be 100N thus implying that N can never catch up with its g function). Tighter bounds will make the search easier to find the rest of the g numbers and resolve the issues raised above.

This document can be found at www.rspq.org/pubs.

g_1	g_2	g_3	g_4	g_5
199981--				
200000				
200001				
1599981--				
2600000				
2600001				
13199998				
	28263827			
35000000	35000000			
35000001				
35199981--				
35200000				
35200001				
117463825				
	242463827			
		371599983--		
			499999984--	
500000000	500000000	500000000	500000000	
500000001				
500199981--				
500200000				
500200001				
501599981--				
502600000--				
513199998				
	528263827			
535000000	535000000			
535000001				
535199981--				
535200000				
535200001				
1111111110				
	10000000000	10000000000	10000000000	10000000000
	10028263827			
	10035000000			
	10242463827			
		10371599983--		
			10499999984--	
	10500000000	10500000000	10500000000	
	10528263827			
	10535000000			
		20000000000	20000000000	20000000000

TABLE 2.

g_5	g_6	g_7	g_8	g_9
		9465000000	9465000000	
		9471736170	9471736170	
			9486799989--	
			9497400000	
			9498399989--	
	9500000000	9500000000	9500000000	
	9628399986--			
		9757536170		
			9882536171	
		9965000000	9965000000	
		9971736170		
			9986799989--	
			9997400000	
			9998399989--	
10000000000	10000000000	10000000000	10000000000	10000000000
		19465000000	19465000000	
		19471736170		
			19486799989--	
			19497400000	
			19498399989--	
	19500000000	19500000000	19500000000	
	19628399986--			
		19757536170		
			19882536171	
		19965000000	19965000000	
		19971736170		
			19986799989--	
			19997400000	
			19998399989--	
20000000000	20000000000	20000000000	20000000000	20000000000

TABLE 3.