The Secret Behind the Luhn-ie

A look at the Luhn algorithm and how it is used in the 21st century for error detection.

By Broderick Causley

DOI: 10.1145/2331042.2351917

The newest hit is about to be released, and you have to pre-order your copy now. Being as excited as you are, you rush to enter your 16-digit credit card number, and sure enough you make a typo. Wait a minute? How did the system know what your credit number should be, before submitting? What information was needed (your name, address, phone number, bank account numbers) to be able to pull this off? The surprising answer is none of these. The secret behind this technology is the Luhn Algorithm.

Hans Peter Luhn served as a communications officer in the German Army during WWI before moving to the United States. In 1960, while working at IBM, he patented a device called a *Computer For Verifying Numbers* [1]. This “computer” (See illustration) was made to verify large lists of manufacturing parts. The algorithm is a counting technique similar to the checksum-10 method, but with more perks.

Here’s how it works. Each string of numerical digits is added from right to left, where starting from the rightmost digit, every second entry is doubled. One restriction is upon doubling where if acquiring a number 10 or greater, you must add the two digits (or a “digital root”). A simple way to code this is to subtract 9 each time this happens. When the summation is complete, the total is divided by 10 hoping to see a remainder of zero. If this happens, the string of digits passes the check, with no finite number too large to use. Take the example 12345678 in Figure 1.

Now we will use Java to test the number 12345678 in Figure 2.

Figure 1.

1	2	3	4	5	6	7	8
x2	x1	x2	x1	x2	x1	x2	x1
--	--	--	--	--	--	--	--
2	2	6	4	10	6	14	8
2	+	2	+	6	+	4	+
				(1+0)	+	6	+
						(1+4)	+
							8
							= 34
(Does not check)							

Figure 2.

```
public class luhn {
    public static void main(String[] args) {
        long x = 12345678;
        System.out.print(x + " ");
        long total = 0;
        while(x != 0) {
            total += (x % 10) + (2 * ((x / 10) % 10));
            if(((x / 10) % 10) > 4) {
                total -= 9;
            }
            x /= 10;
        }
        if((total % 10) == 0) {
            System.out.print("passes the Luhn check.");
        } else {
            System.out.print("does not pass the Luhn check.");
        }
    }
}
```

Figure 3. Luhn's original creation.
US2950048, August 1960.

Some patterns arise which are interesting. If you had to pick a single digit from 1-9 and repeat it until the Luhn-check passes, 8888 is the smallest and 66666666666666 is the largest. Also, every single digit error is found regardless of the value or position. This answers the question if someone could “accidentally” steal your identity (ie. Canadian Social Insurance Number) with a number almost the same as yours. At least, changing one digit will never be possible. You won’t be able to find another credit card number with a single digit error either. These techniques are true with the checksum-10 method as well, but an extra perk is that almost all transpositions of two adjacent digits are found. The only undetected case is when switching the position of 0 and 9, which computes to approximately 2 percent of all cases.

0	9			9	0
x2	x1			x2	x1
--	--			--	--
0	9			18	0
0	+	9	=	9	(1+8) + 0 = 9

Since a credit card number has 16 digits, will we ever run out of numbers that work? Theoretically, yes, but you will be surprised how many are possible. Because there are 16 digits, we can create the first 15 digits any way we like and assign the final digit to work with the previous 15 numbers. Now we know that there are 10^{15} ways (1 quadrillion) to do this—more than enough for every living man, woman, and child to have one hundred thousand credit cards in their wallet.

Now, how does this apply to the Canadian dollar? You can find Luhn’s algorithm in Canadian Social Insurance Numbers, Health Card Numbers, GST Registration Numbers, and many others as well. The Canadian organizations and government agencies that are using Luhn’s technique are not trying to be sneaky, but to help with the most common data entry errors.

Some of these common entry errors would be when entering your identification number or a tax receipt when completing your taxes, numerous banking applications, every documented visit to the hospital, or even employment membership cards. The benefits of Canadian data entry are very high—a time saver if you will.

So don’t panic. The secret in your wallet is innovative, but this secret is a helpful one.

Reference

- [1] Luhn, H.P. [1960]. U.S. Patent No. 2,950,048. Washington, D.C.: United States Patent Office. Retrieved March 14, 2012 from <http://www.google.com/patents/US2950048>.

Biography

Broderick Causley is currently an undergraduate at Algoma University, Ontario, Canada majoring in mathematics. He co-founded AU Prime Thinkers math club at Algoma University, and also enjoys snowboarding and listening to music. His interests are partial differential equations and complex analysis.

© 2012 ACM 1528-4972/12/09 \$15.00