That's Numberwangcoin!

Robert J. Simmons, Calculemus LLC, rob@calculem.us, Not From Somerset

Abstract

We present a new design for The Blockchain. This attempts to solve several problems, including boredom, lost coins, shouty bits, speculative investment, and the number 2 which you may remember from school is deadly to humans.

Background: The Blockchain, Hashes, and Difficulty

Every block in a The Blockchain can be seen as the jamming together of three things:

1. The hash of the previous block
2. Some stuff you care about (The Ledger (TM)). In its simplest form, The Ledger (TM) involves a bunch of addresses (big numbers) that transfer value to one another; everyone can see The Ledger (TM) and compute the current value of every address.
3. Some random stuff

Your job, as a miner in a The Blockchain, is to come up with random stuff over and over, jam it together with the other bits and compute its hash. A hash takes data and turns it unpredictably into a string of, say, 256 bits. Then the hash is evaluated to see if it's Good™.

Being "Good™" is something that everybody working on the same The Blockchain has to agree on: everybody has to be able to look at your three parts, concatenate them themselves, compute the hash, and say, "Yep, Julie's random stuff caused the jamming together to have a hash that is Good™. Julie is a worthwhile member of society and deserving of scarce resources."

Presumably Julie just found the Good™ hash by picking new random stuff over and over until one of the versions of the random stuff was Good™. Picking random stuff is like pulling the arm on a slot machine: it produces some random output and that output might be Good™ news for you.

A fundamental design aspect of any The Blockchain is *difficulty*. It needs to become harder or easier to accidentally generate a Good™ block in order to keep the rate of newly solved blocks roughly consistent across a The Blockchain. In Bitcoin's The Blockchain, difficulty is recalibrated every 2016 blocks, with the goal of making some contestant able to randomly come up with a jackpot Good™ random value once every ten minutes.

In most The Blockchain, a Good™ hash has a lot of zeroes at the front. This can lead to an important but subtle misconception that Good™ness is a property of how many zero bits are at the start of the hash, and difficulty is tweaked by calibrating how many zeroes there are at the beginning of the hash.

Too easy: `hash & f800000000000000000000000000000000000000000000000000000000000000` is zero

Realistic: `hash & fffffffffff0000000000000000000000000000000000000000000000000000` is zero

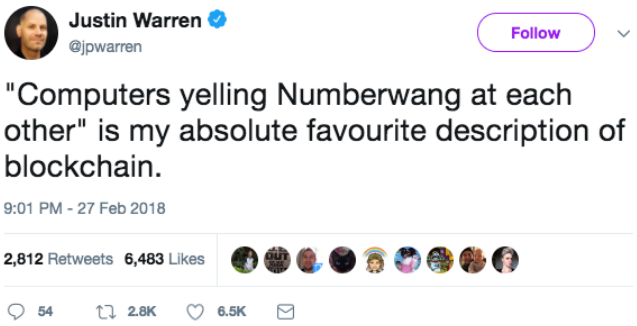
Unrealistic: `hash & ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff000` is zero

In the "too easy" example, we would expect that it would only take 32 guesses at random stuff before one of our hashes would be Good™.

The only problem with saying that the difficulty is the number of zeroes is that that the difficulty can then only get twice as hard or twice as easy by adding or removing a bit-that-must-be-zero. The better idea is to to say a Good™ hash is numerically smaller than some *target*; lowering the target by a small amount increases difficulty a small amount, in general.

Shouting Numberwang At Each Other

We turn to the problem we're solving. Specifically, the problem that this isn't true enough:



So let's make it truer, and make Numberwangcoin in the process. Computers in boring The Blockchain are actually shouting at each other about programs with rather low hashes (hashes that are below the target). Can we have them shout Numberwang?

Numberwang, Yes, But is it Numberwang Enough?

We could make the computer shouty bits a little more Numberwang by requiring the inevitable zeroes in front of a hash to be the ASCII representation of "NUMBERWANG NUMBERWANG NUMBERWANG" (all caps. Remember: they're supposed to be shouting). This is a string with the following 256-bit representation:

```
4e554d42455257414e47204e554d42455257414e47204e554d42455257414e47
```

The most uniform way to enforce this is to say that a computed hash must be XORed against this shouty value before it is compared against the target. Therefore, a Good™ hash becomes not the one that is smallest in an absolute sense, but the one that is the Most Numberwang. The hardest possible hash to come up with is no longer zero, it is Numberwang (Numberwang Numberwang).

We encounter a problem, though. Even with a resources at a global scale devoted to the problem of shouting boring Bitcoin low-value hashes, the current target only has 18 leading zeroes.

In other words, if the Bitcoin protocol were based on our proposed design, computers would regularly be shouting "NUMBERWAN" at each other, but not necessarily "NUMBERWANG". At present, they would shout a full "NUMBERWANG" about once every 32 transactions, though, so we are close. But a truly climate-altering amount of computational resources are being devoted to this shouty computer process. We need to figure out how to make Numberwang with less.

Here we make the observation that we're using the ASCII encoding, which falls in the range 0-127, wasting one bit per character. If we truncate the first character, it becomes over a thousand times easier to produce an actual full Numberwang, giving us an XOR value of

```
9d566c28b4abc19d1d04eab36145a55e0ce8e827559b0a2d2af06747413aacd8
```

This makes it 1024x easier to come up with Numberwang, and also allows us to store an additional "num" and 4 bits of "b" in the shout string. However, shouting a full "NUMBERWANG" in every new addition to The Blockchain still corresponds to a hash difficulty[1] that was only reached in late 2016 on the Bitcoin network.

[1] Roughly 270 trillion, for those following along at home. This means that at the lowest difficulty setting, 1 in 270 trillion blocks would have a full Numberwang.

It's evident that we need to go further. We will compress even further using the predictable "`"A"` is zero, `"B"` is one…" encoding, in which we need five bytes, instead of seven, per letter.

```
Char:   N    U    M    B    E    R    W    A    N    G    N    U    M    B    E    R    W    A    N
Ord:    13   20   12   1    4    17   22   0    13   6    13   20   12   1    4    17   22   0    13
Binary: 01101101000110000001001001000110110000000110100110011011010001100000010010010001101100000001
Hex:    6   d   1   8   1   2   4   6   c   0   6   9   9   b   4   6   0   4   9   1   b   0   1
```

This encoding allows us to encode "NUMBERWANG" in 50 bits. Given that we have 256 bits to work with, and given that the last six bits represent a truly astronomical difficulty, we will leave the last six bits zero, meaning that our shouty XOR-with-the-hash value that is:

`6d181246c0699b460491b01a66d181246c0699b460491b01a66d181246c06980`

I Can't Think Of Any More Numbers

Getting a single numberwang in this encoding represents a difficulty[2] reached in early 2011, way before anyone gave a crap about any of this. Still a bit high, though: at the lowest difficulty setting, only 1 in every 260 thousand blocks would be expected to shout a full "Numberwang."

We choose to live with this reality, and turn "lemons" into Wordwang. Observe that the difficulty recalibration interval represents a natural time demarcation (corresponding to roughly two weeks in Bitcoin's The Blockchain). If, between two recalibration steps, no full 50-bit numberwang appears, we enter Sudden Death.

In Sudden Death, we re-hash the hash of the previous board recalibration block to get the Deadly Number Gas Hash (Deadly Gash). The address with a non-zero balance that matches the most bits of the Deadly Gash, starting with the least-significant digit, will have its balance replaced by 2 WangerNumb. If there's a tie, all first place winners lose.

The Sudden Death protocol will make adoption of Numberwangcoin faster, because the more people mine Numberwangcoin, the faster the difficulty will rise to a level that makes Sudden Death at first unlikely, and then, in time, impossible.

Let's Rotate The Board!

The most important and lasting effect that happens along with difficulty recalibration is Rotating the Board. When we Rotate the Board, every address with a nonzero balance is put in numerical order by *address* (not balance). We then rotate value from an address to its next highest address. The highest address wraps around and transfers to the lowest.

If an account has more than a thousand WangerNumb, then the balance is rounded down to the next power of 10, and one-one-thousandth of that amount is rotated to the next address on the board. If an account has less than a thousand WangerNumb, then one full WangerNumb is rotated to the next address on the board until the balance becomes zero and it leaves the rotation.

This wealth redistribution mechanism ensures that no Numberwangcoin value will ever truly leave circulation. It also makes Numberwangcoin a terrible mechanism for long-term investment, charging approximately 2.5 percent in redistributive taxation every year. These fees are quite trivial if one is holding Numberwangcoin for a short period of time as a medium of free exchange, bringing The Blockchain back to the purpose that I, um I mean Satoshi, intended.

The Maths Coin That Simply Everyone Is Talking About

The genesis block for Numberwangcoin, along with a state-of-the-art browser-based miner, will or will not be available from https://numberwangco.in/ on March 30, 2018.

[2] About 260k for those following along at home.