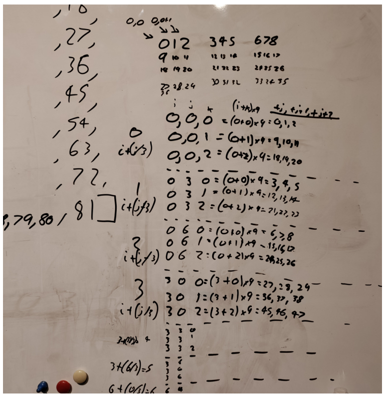
The Surprising Uses of Mathematics in Computer Science

by Joel Karet

I had been sitting at my desk for a few hours trying to code a program to solve a sudoku. I had reached a blockage in my work. I couldn't figure out at all how to solve the issue. The issue was thus: I needed to split the board into its 3 by 3 squares to check if there were any duplicated numbers. This might seem trivial, however I really didn't want to have to define all 81 squares into their own boxes. So instead I reverted to mathematical logic. For the rows and columns it was fairly easy; Put the first 9 numbers in a list for the first row, then the next 9 in a second list and so on. Then for the columns you just take every 9th number starting at the first number, then starting at the second number and so on. But for the 3 by 3 boxes? You need the first 3 numbers, then skip 6, then take 3 and skip 6, then take 3 again. Then you need to start at the 4th number, and repeat. You can see how quite quickly this doesn't seem to be much easier than just stating where all the numbers were.

So, what did I do? I got my whiteboard marker out, and thought of the board as a 2d plane. After a bit of writing, and working out how many loops I would need and how often I would need to repeat the loops, and increase each variable by after the loops, I got a beautiful mess of working on my board. After looking at it for a minute or so, I realised to myself that it looked surprisingly like multiple matrices multiplications.



This got me curious. Where else does mathematics appear in seemingly unrelated locations within computing? Not only will I have covered this sudoku example, but I will also cover 2 more examples I have found, namely graphic effects and hashing. Now I know what many of you are thinking. How is a sudoku an unrelated location for mathematics? It's like being surprised when pi inevitably reveals itself when looking at a circle. However, if you think about what a sudoku really is, the numbers inside have no relation to mathematics. They are just placeholders to represent differing objects. A sudoku could be filled with colours, shapes or letters.

A basic sudoku (i.e. ignoring variations such as killer sudokus) is just a boolean comparison of any 2 squares that see each other.

So, where does maths reveal itself in computing? One location that sprouted out to me was graphic effects. When making a render of a realistic image, it would seem that maths would be far away, as this is in the department of art. And yes, although this could be done purely by hand, stating the exact brightness of different pixels, it would take forever, and while possibly look realistic, it would not be realistic to do this for every frame possible while moving through a 3d world . Instead mathematical equations can be used to trace the line from the imaginary eye through each pixel in the screen and with the conjunction of a graphical processing unit (GPU), these calculations can be done on every ray that would affect the brightness of pixels, and create realistic shadows on your screen. The majority of this is done using 3d vectors, and seeing which rays would reflect off of objects and hit the light source. Some may point out that we should go from the light source as that is what happens in real life, however because we only need to render this in comparison from one point of view, the screen, it is much more efficient to only calculate those rays that would hit the imaginary eye behind the screen. This is because the other rays would not have any effect on the overall picture for the viewer.

A possibly less surprising use of mathematics within computer science could be the process of storing login information securely. A big company such as Google, that you have signed in to, does not, and should not, know your password. This is why if you forget your password, they can’t just send you an email containing the password, because they quite literally do not know it. So how do they know if you got your password correct? Well they use hashing of some kind. They store the output of your password inputted through a 1 way function, and when you input your password, they put that through the same 1 way function, and if the output is the same, then they know you inputted the correct password. A variation of 1 character before the hashing completely alters the hashed code, and there is no way to go from the hashed code back to the original password. This keeps your data safe, even if there is a data breach, as your password was never stored.

But how does a 1 way function work? It would seem that for a function f(x) there should always be a function $f(x)^{-1}$? Well while this is true, a 1 way function refers to a function that is incredibly quick and easy to compute going one way, however the reverse is incredibly slow to compute or a reverse function hasn't been found yet. Here is where the mathematics comes to shine beautifully. A great example of a one way function is with the use of prime numbers. Have a friend take 2 very large prime numbers and multiply them together. Now, without knowing those original prime numbers, find the prime factors of that number. How would you even do it? You would have to divide by every prime number until you get an integer out the other end, in which if you take large enough primes, would take impossibly long, given that either a huge list of known primes would be needed, or calculating primes on the way, which is incredibly resource intensive. This is just one way to create a 1 way function, in which there are many, but it shows how a function can be quick and simple, just one multiplication, in one direction, and almost impossible in a time efficient manner in the other direction.

I think it goes without saying that low levels of maths are required for computer science, as realised once again by me when programming a sudoku solver. However what I hope I have displayed here, is that with the use of more and more complex mathematics, so much more is possible in computing. Whether that be the beauty made via ray tracing, or the security ensured in hashing.