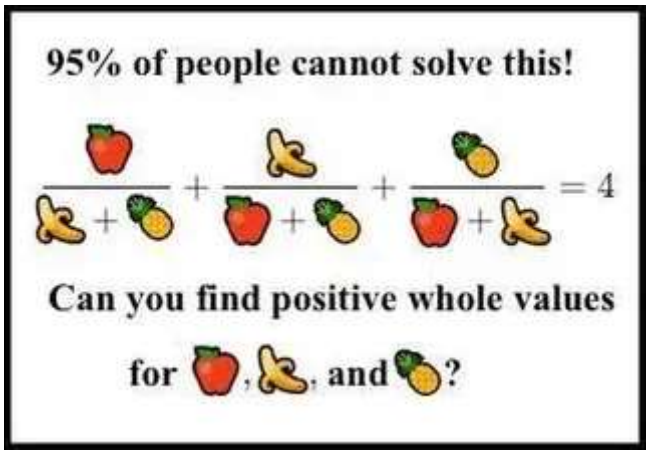
A simple explanation of a/(b+c) + b/(c+a) + c/(a+b) = 4

2025 May 11 See all posts

You may have at some point seen this math puzzle:



The puzzle has gained some degree of notoriety on the internet because it *looks* like it must be either simple, impossible, or a trick question (solve P = NP? Hahaha, the answer is N = 1, got you!), but in reality it's none of those three. The problem is exactly what it seems. The problem is solvable. And yet the smallest solution is:

```
a = 154476802108746166441951315019919837485664325669565431700026634898253202035277999
b = 368751317941299998271978115652254748254929799689719709962831374716372246340555 79
c = 437361267792869725786125260237139015281653755816161361862143799337842346777 2036
```

What the heck is going on??! If you search on the internet, you will see long-winded explanations about how the problem is actually connected to elliptic curves, and other fancy

buzzwords from algebraic geometry that you've never heard of even if you think you know everything about elliptic curves because you're a ~~cryptographer~~ crypto enthusiast.

The goal of this post will be to give a maximally elementary explanation of this puzzle that does not rely on any pre-existing knowledge of these concepts.

Solve without requiring positive values first, then find a positive solution

First, let us start off by looking at a *relaxed* version of the problem. Specifically, let's remove the requirement that the three values must be positive. It turns out that you can just try a whole bunch of possibilities and get two answers by pure human brute force: $(-11, -4, 1)$ and $(11, -5, 9)$. You can get more solutions from either of these two by re-arranging the numbers, flipping signs and multiplying by constant factors (eg. $(-2, 8, 22)$ is also a valid solution), but we'll treat those as being the same.

Now, we get to the interesting part. What if we can come up with a way to *combine* two solutions and get a totally new *third* solution? Often, this is possible in mathematics: if you have two multiples of 5, their sum is also a multiple of 5. More nontrivially, if you have two rational points on a circle (ie. $(\frac{p_1}{q_1}, \frac{r_2}{s_2})$ and $(\frac{p_2}{q_2}, \frac{r_2}{s_2})$ satisfying $(\frac{p_i}{q_i})^2 + (\frac{r_i}{s_i})^2 = 1$), you can use point addition laws to make a third rational point on the circle: $(\frac{p_1p_2}{q_1q_2} - \frac{r_1r_2}{s_1s_2}, \frac{p_1r_2}{q_1s_2} + \frac{r_1p_2}{s_1q_2})$.

If we can come up with such an algorithm for *our* problem, then we could just use it over and over again, until eventually we get a point that happens to be all-positive by pure luck. This will be our solution path.

Combining two solutions to find a third

Let us simplify the problem by making it only have two variables, a and b. Note that the equation is *homogeneous*: it has the property that scaling all the inputs by the same number does not change the answer. Let's take advantage of this to set $c = 1$:

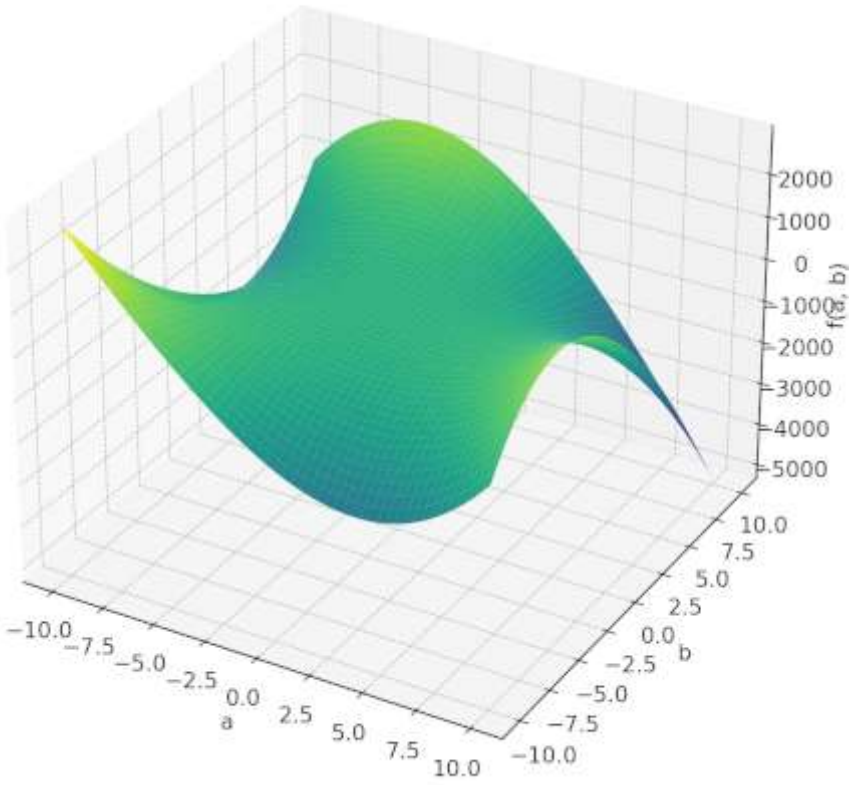
$$\frac{a}{b+1} + \frac{b}{a+1} + \frac{1}{a+b} - 4 = 0$$

Any solution to this two-variable equation with *rational* $a = \frac{p}{q}$ and $b = \frac{r}{s}$ can be scaled into an *integer* solution to the original three-variable equation: $(a * qs, b * qs, qs)$. We also moved the 4 to the left, this will make things more convenient for us later.

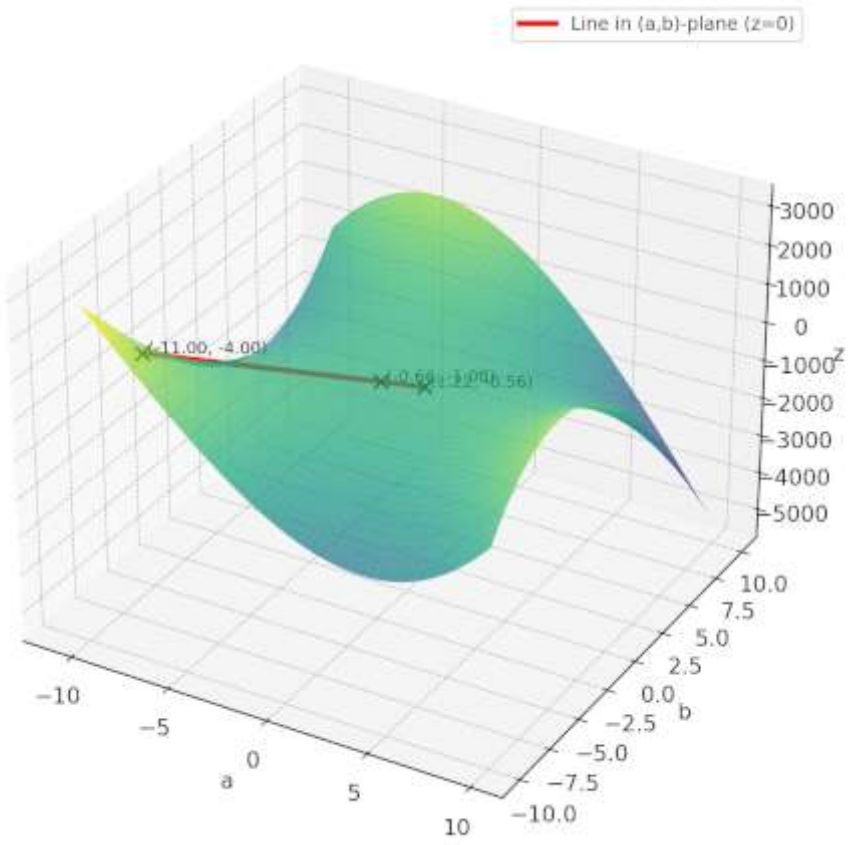
Now, let's multiply by the denominators, to make the whole equation a pure polynomial:

$$a(a+1)(a+b)+b(b+1)(a+b)+(a+1)(b+1)-4(a+1)(b+1)(a+b)=0$$

This introduces a few extraneous solutions, eg. $(a=1,b=-1)$. But we just ignore them. We can plot the above expression as a function; it looks like this:



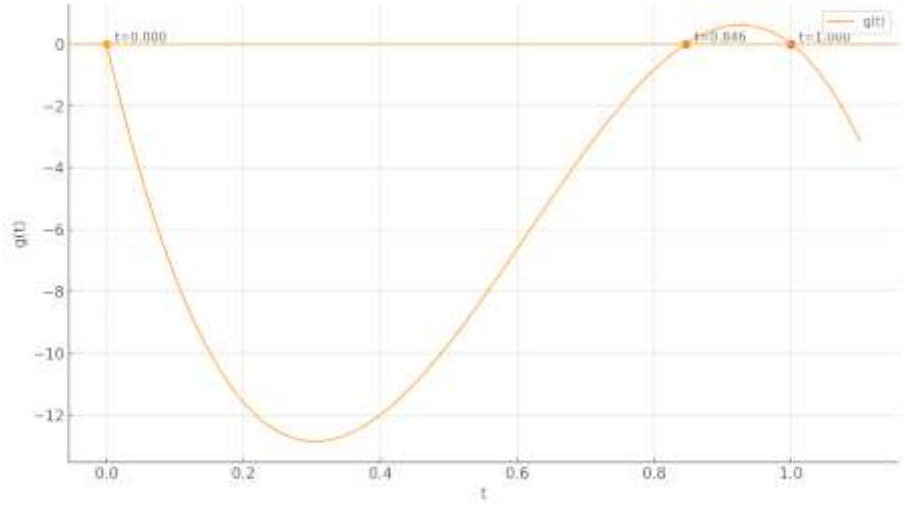
Now, let's draw a line through the two points where we know this curve intersects the $z=0$ plane: $(-11,-4)$ and $(\frac{11}{9},\frac{-5}{9})$. We derive these by starting from the two solutions to the three-value equation, and converting $(a,b,c)\to(\frac{a}{c},\frac{b}{c})$.



And as we can see, the line has a third point of intersection. And because the line is along the $z = 0$ plane, the third point must also be along the $z = 0$ plane, ergo it must also be a solution. Now, let us prove that this point is *rational*, so it still corresponds to a valid solution to the original problem.

We can parametrize the line as $(a = -11 + (\frac{11}{9} + 11) * t, b = -4 + (\frac{-5}{9} + 4) * t)$. $t = 0$ gives the first point, $t = 1$ gives the second point. Then, we can look at how $a(a+1)(a+b) + b(b+1)(a+b) + (a+1)(b+1) - 4(a+1)(b+1)(a+b)$ behaves *along the line* by substituting a and b with their expressions based on t.

This gives us the curve: $y = \frac{9071}{81} * t^3 + \frac{16748}{81} * t^2 + \frac{7677}{81} * t$. Here's a plot of that curve, showing all *three* intersections:



You can then take the new t and plug it into the line and get back your new point: $(\frac{-5951}{9071}, \frac{-9841}{9071})$.

The underlying reason the new t (and hence the new point) is rational is: the curve is a degree-3 polynomial in t, and if a degree-3 polynomial has three solutions, then it fully factors into $k(x-a)(x-b)(x-c)$. The degree-2 term of this equals $-k(a+b+c)$. Hence, if the degree-2 and degree-3 terms are rational (which they are, because we constructed the polynomial out of an equation with rational coefficients), and two of the solutions are rational, the third solution must be rational as well.

From here, we can brute-force our way to getting more solutions. Unfortunately, we can't directly keep going with the three points we have: if you were to repeat the above process to generate a new point starting from any two of the three points we now have, you would just get back the third point. To get past this, we'll use a clever trick to generate even more solutions: convert (x, y) into (y, x).

Now, with this in mind, we just brute-force, repeatedly using the "find the third point on the line" algorithm and coordinate flipping to get as many new solutions as possible. Here's the python code:

```
from sympy import symbols, Poly, solve, simplify, Rational, expand
from math import lcm

def find_third_point(a1, b1, a2, b2):
    """
    Find the third intersection point of a line through points (a1, b1) and (a2, b2)
    on the cubic curve a(a+1)(a+b) + b(b+1)(a+b) + (a+1)(b+1) = 4(a+1)(b+1)(a+b).
```

```
    Args:
        a1, b1: Coordinates of the first point (rational numbers)
        a2, b2: Coordinates of the second point (rational numbers)

    Returns:
        Tuple (a3, b3): Coordinates of the third intersection point
    """
    # Define symbolic variables
    t, a, b = symbols('t a b')

    # Parameterize the line: a = a1 + t*(a2 - a1), b = b1 + t*(b2 - b1)
    a_expr = a1 + t * (a2 - a1)
    b_expr = b1 + t * (b2 - b1)

    # Define the cubic curve equation:
    # a(a+1)(a+b) + b(b+1)(a+b) + (a+1)(b+1) = 4(a+1)(b+1)(a+b)
    left_side = a * (a + 1) * (a + b) + b * (b + 1) * (a + b) + (a + 1) * (b + 1)
    right_side = 4 * (a + 1) * (b + 1) * (a + b)
    equation = left_side - right_side

    # Substitute the line parameterization into the equation
    poly_t = equation.subs({a: a_expr, b: b_expr})

    # Simplify and convert to a polynomial in t
    poly_t = expand(poly_t)
    poly = Poly(poly_t, t)

    # Get the coefficients of the cubic polynomial
    coeffs = poly.coeffs()

    # The polynomial is cubic (degree 3), so coeffs = [c3, c2, c1, c0]
    # For a cubic c3*t3 + c2*t2 + c1*t + c0, the sum of roots is -c2/c3
    while len(coeffs) < 4:
        coeffs.append(0)
    if len(coeffs) != 4:
        raise ValueError("Unexpected polynomial degree. Expected cubic polynomial.")

    c3, c2, c1, c0 = coeffs

    # The known roots are t=0 (for point 1) and t=1 (for point 2)
    # Sum of roots: t1 + t2 + t3 = -c2/c3
    # Since t1 = 0, t2 = 1, we have 0 + 1 + t3 = -c2/c3
    t3 = -c2 / c3 - (0 + 1)
```

```
    # Compute the third point coordinates
    a3 = a1 + t3 * (a2 - a1)
    b3 = b1 + t3 * (b2 - b1)

    # Simplify the results to ensure rational output
    a3 = simplify(a3)
    b3 = simplify(b3)

    return (a3, b3)

# Verify the a point is on the curve
def is_on_curve(a, b):
    left = a * (a + 1) * (a + b) + b * (b + 1) * (a + b) + (a + 1) * (b + 1)
    right = 4 * (a + 1) * (b + 1) * (a + b)
    return left == right

def is_too_big(x, y):
    xn, xd = x.as_numer_denom()
    yn, yd = y.as_numer_denom()
    return max(abs(xn), abs(xd), abs(yn), abs(yd)) > 10**200

def find_smallest_all_positive_point():
    points = [(Rational(-11, 1), Rational(-4, 1)), (Rational(11, 9), Rational(-5, 9))]
    queue = [(points[0], points[1])]

    def accept(x, y):
        print(x, y)
        for (x2, y2) in points:
            queue.append(((x2, y2), (x,y)))
        points.append((x, y))

    while len(queue) > 0:
        print(len(queue))
        (x1, y1), (x2, y2) = queue.pop()
        x3, y3 = find_third_point(x1, y1, x2, y2)
        if not is_too_big(x3, y3):
            if (x3, y3) not in points:
                accept(x3, y3)
            if (y3, x3) not in points:
                accept(y3, x3)

    print(points)
    eligible = [(x,y) for (x,y) in points if x > 0 and y > 0]
```

```
    return min(
        eligible,
        key=lambda x: lcm(x[0].as_numer_denom()[1], x[1].as_numer_denom()[1])
    )
```

This is horribly inefficient, but it does the job. And out we get:

```
(154476802108746166441951315019919837485664325669565431700026634898253202035277999/43736126
```

This is a positive rational solution to the two-equation formula; add $c = 1$ and you get a positive rational solution to the three-equation formula. And from here, we multiply by the denominator, and get the original solution:

```
a = 154476802108746166441951315019919837485664325669565431700026634898253202035277999
b = 368751317941299998271978115652254748254929799689719709962831374716372246340555 79
c = 437361267792869725786125260237139015281653755816161361862143799337842346777 2036
```

This does *not* prove that this is the smallest solution: for that you actually would have to go into the much deeper elliptic curve theory that I tried hard to avoid. But it gives you a solution to the problem, and the underlying math actually is elliptic curve math in disguise: "find the third intersecting point on the line and flip the coordinates" is exactly the same thing as the elliptic curve addition law, except flipping across the diagonal axis instead of the x axis. This "addition law" we came up with even satisfies associativity.