USAMO 1997/3

Evan Chen

Twitch Solves ISL

Episode 74

Problem

Prove that for any integer n, there exists a unique polynomial Q with coefficients in $\{0, 1, \ldots, 9\}$ such that $Q(-2) = Q(-5) = n$.

Video

https://youtu.be/rGoMTlJwq-I

External Link

https://aops.com/community/p343873

Solution

If we let
$$Q(x) = \sum_{k \ge 0} a_k x^k$$
then a_k is uniquely determined by $n \pmod{2^k}$ and $n \pmod{5^k}$. Indeed, we can extract the coefficients of Q exactly by the following algorithm:

- Define $b_0 = c_0 = n$.
- For $i \ge 0$, let a_i be the unique digit satisfying $a_i \equiv b_i \pmod 2$, $a_i \equiv c_i \pmod 5$. Then, define
$$b_{i+1} = \frac{b_i - a_i}{-2}, \qquad c_{i+1} = \frac{c_i - a_i}{-5}.$$

The proof is automatic by Chinese remainder theorem, so this shows uniqueness already. The tricky part is to show that all a_i are eventually zero (i.e. the "existence" step is nontrivial because a polynomial may only have finitely many nonzero terms).

In fact, we will prove the following claim:

Claim. Suppose b_0 and c_0 are any integers such that
$$b_0 \equiv c_0 \pmod 3.$$
Then defining b_i and c_i as above, we have $b_i \equiv c_i \pmod 3$ for all i, and $b_N = c_N = 0$ for large enough N.

Proof. Dropping the subscripts for ease of notation, we are looking at the map
$$(b, c) \mapsto \left(\frac{b-a}{-2}, \frac{c-a}{-5}\right)$$
for some $0 \le a \le 9$ (a function in b and c).

The $b \equiv c \pmod 3$ is clearly preserved. Also, examining the size,

- If $|c| > 2$, we have $\left|\frac{c-a}{-5}\right| \le \frac{|c|+9}{5} < |c|$. Thus, we eventually reach a pair with $|c| \le 2$.
- Similarly, if $|b| > 9$, we have $\left|\frac{b-a}{-2}\right| \le \frac{|b|+9}{2} < |b|$, so we eventually reach a pair with $|b| \le 9$.

this leaves us with $5 \cdot 19 = 95$ ordered pairs to check (though only about one third have $b \equiv c \pmod 3$). This can be done by the following code:

```
import functools
@functools.lru_cache()
def f(x0, y0):
  if x0 == 0 and y0 == 0:
    return 0
  if x0 % 2 == (y0 % 5) % 2:
    d = y0 % 5
  else:
    d = (y0 % 5) + 5

  x1 = (x0 - d) // (-2)
  y1 = (y0 - d) // (-5)
```

```

  return 1 + f(x1, y1)

for x in range(-9, 10):
for y in range(-2, 3):
  if (x % 3 == y % 3):
    print(f"({x:2d}, {y:2d}) finished in {f(x,y)} moves")
```

As this gives the output

```
(-9,  0) finished in 5 moves
(-8, -2) finished in 5 moves
(-8,  1) finished in 5 moves
(-7, -1) finished in 5 moves
(-7,  2) finished in 5 moves
(-6,  0) finished in 3 moves
(-5, -2) finished in 3 moves
(-5,  1) finished in 3 moves
(-4, -1) finished in 3 moves
(-4,  2) finished in 3 moves
(-3,  0) finished in 3 moves
(-2, -2) finished in 3 moves
(-2,  1) finished in 3 moves
(-1, -1) finished in 3 moves
(-1,  2) finished in 3 moves
( 0,  0) finished in 0 moves
( 1, -2) finished in 2 moves
( 1,  1) finished in 1 moves
( 2, -1) finished in 2 moves
( 2,  2) finished in 1 moves
( 3,  0) finished in 2 moves
( 4, -2) finished in 2 moves
( 4,  1) finished in 2 moves
( 5, -1) finished in 2 moves
( 5,  2) finished in 2 moves
( 6,  0) finished in 4 moves
( 7, -2) finished in 4 moves
( 7,  1) finished in 4 moves
( 8, -1) finished in 4 moves
( 8,  2) finished in 4 moves
( 9,  0) finished in 4 moves
```

we are done. □