Sierpinski’s Triangle and the Lucas Correspondence Theorem

Julian Trujillo, and George Kreppein

Western Washington University

November 30, 2012
Sierpinski’s triangle also known as Sierpinski’s gasket was first described in 1915 by Waclaw Sierpinski. However it had been seen as early as 13\textsuperscript{th} century at the cathedral of Anagni. Sierpinski triangle has also appeared in our current culture. The Tri-force from legend of Zelda is the second iteration of Sierpinski’s Triangle.
Sierpinski’s Triangle and the Lucas Correspondence Theorem
Sierpinski’s Triangle and the Lucas Correspondence Theorem
Sierpinski’s Triangle and the Lucas Correspondence Theorem
Sierpinski’s Triangle and the Lucas Correspondence Theorem
Sierpinski’s triangle is a self similar set.
The area of Sierpinski’s triangle is zero in Lebesgue measure.
There is a three dimensional analogue of Sierpinski’s triangle called a Tetrix.
Someone in our class made a shale using a Sierpinski’s triangle pattern.
There are several ways to construct Sierpinski’s triangle.
Construction

There are several ways to construct Sierpinski’s triangle:

- Deletion method
- Pascals triangle mod 2
- Chaos game
- Bit wise operations (made possible by Lucas Correspondence Theorem)
Sierpinski's Triangle and the Lucas Correspondence Theorem

```
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
1 6 15 20 15 6 1
1 7 21 35 35 21 7 1
```

```
1
1 1
1 0 1
1 1 1 1
1 0 0 0 1
1 1 0 0 1 1
1 0 1 0 1 0 1
1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0
```

```
0
0 0
0 1 0
0 0 0 0
0 1 2 3 0
0 0 2 2 0 0
0 1 0 1 0 1 0
0 1 0 1 0 1 0
```

Lucas Correspondence Theorem

Theorem

Lucas Correspondence Theorem If $p$ is prime and $M$ and $N$ are integers such that $M = M_0 + M_1p + M_2p^2 + \cdots + M_kp^k$ and $N = N_0 + N_1p + N_2p^2 + \cdots + N_kp^k$ with $0 \leq M_r < p$ and $0 \leq N_r \leq p$, then $inom{M}{N} \equiv \binom{M_0}{N_0} \binom{M_1}{N_1} \binom{M_2}{N_2} \cdots \binom{M_k}{N_k} \pmod{p}$.
Proof of Lucas Correspondence Theorem

**Proof.**

Consider the generating function with \( \binom{M}{N} \) as the coefficient of the \( N^{th} \) term. Then we have:

\[
\sum_{N \geq 0} \binom{M}{N} x^N = \sum_{N=0}^{M} \binom{M}{N} x^N
\]

\[
= (1 + x)^M
\]

\[
= (1 + x)^{M_0 + M_1 p + M_2 p^2 + \ldots + M_k p^k}
\]

\[
= \prod_{r=0}^{k} [(1 + x)^{p_r}]^{M_r} \equiv \prod_{r=0}^{k} (1 + x^{p_r})^{M_r} \pmod{p}
\]

\[
= \prod_{r=0}^{k} \left( \sum_{s_r=0}^{M_r} \binom{M_r}{s_r} x^{s_r p_r} \right)
\]

\[
= \left( \binom{M_0}{0} x^{0 \cdot p^0} + \binom{M_0}{1} x^{1 \cdot p^0} + \binom{M_0}{2} x^{2 \cdot p^0} + \ldots + \binom{M_0}{M_0} x^{M_0 \cdot p^0} \right)
\]

\[
\left( \binom{M_1}{0} x^{0 \cdot p^1} + \binom{M_1}{1} x^{1 \cdot p^1} + \binom{M_1}{2} x^{2 \cdot p^1} + \ldots + \binom{M_1}{M_1} x^{M_1 \cdot p^1} \right)
\]

\[
\vdots
\]

\[
\left( \binom{M_r}{0} x^{0 \cdot p^{M_r}} + \binom{M_r}{1} x^{1 \cdot p^{M_r}} + \binom{M_r}{2} x^{2 \cdot p^{M_r}} + \ldots + \binom{M_r}{M_r} x^{M_r \cdot p^{M_r}} \right)
\]

\[
= \sum_{N=0}^{M} \sum_{k=0}^{M} \prod_{r=0}^{k} \binom{M_r}{s_r} x^N
\]
Proof.

Where the inner sum is taken over all sets \((s_0, s_1, \ldots, s_k)\) such that
\[
\sum_{r=0}^{k} s_r p^r = N.
\]
But \(0 \leq s_r \leq M_r < p\), so there is at most one such set and given by \(s_r = N_r(0 \leq r \leq k)\) if every \(N_r \leq M_r\); otherwise, the sum is zero. We see that the theorem follows by equating coefficients.
Bitwise operations

- Bit: a true false value or in our case one or zero
- Not: takes one bit and change to opposite value T goes to F, F goes to T
- And: compares two bits and returns true (1) iff both bits are true (1), otherwise returns false (0)
The Connection to bitwise operations

So we can calculate \( \binom{M}{N} \) really quickly by comparing the \( M \) and \( N \) in base \( p \). If any \( N_r \) is larger than the corresponding \( M_r \) then \( \binom{M}{N} \) will be zero. If we let our \( p = 2 \) we see how bit wise operations come into play. We will use the composition of the not and and operations. Which will be and(not(\( m \)),n). We will color the square if our operation yields zero else we leave our square blank. More generally, we color squares that have Zero outputs one color and color squares that have non-zero outputs a different color.
Why bitwise

- Why not.
- Faster to run
- Easier to program (I’m guessing it would be)
Questions

Questions, now is the time to ask.
Sources

http://mathworld.wolfram.com/SierpinskiSieve.html