Factoring, random maps, and polynomial maps

Derek Garton

Portland State University

November 5, 2013

Factoring is a thing that number theorists do

Example

RSA-768 =

1230186684530117755130494958384962720772853569595334
7921973224521517264005072636575187452021997864693899
5647494277406384592519255732630345373154826850791702
6122142913461670429214311602221240479274737794080665
351419597459856902143413 =
3347807169895689878604416984821269081770479498371376
8568912431388982883793878002287614711652531743087737
814467999489 · 367460436667995904282446337996279526322
7915816434308764267603228381573966651127923337341714
3396810270092798736308917

Sizes of numbers

Computers work in binary.

$$\underbrace{\overbrace{\_\_\ \_\_\ \_\_ \cdots \_\_}^{b \text{ blanks (or bits)}}}_{2^b \text{ possible numbers represented, from 0 to } 2^b-1}$$

The "size" of a number is the number of bits required to store it; ie, the size of a number n, if $2^{b-1} \leq n \leq 2^b - 1$, is

$$b(n) = \lceil \log_2 (n+1) \rceil \approx \log_2 n$$

Trial division

Suppose n is a large composite number, with smallest factor p.

Very naive factoring method (Trial Division)

Does 2 divide n?
Does 3 divide n?
Does 4 divide n?
Does 5 divide n?
Does 6 divide n?
$\vdots$

Terminates by finding p in about p steps:

$$p = 2^{\log_2 p} \approx 2^{b(p)},$$

so the runtime of Trial Division is *exponential* in $b(p)$.

Trial division

Suppose n is a large composite number, with smallest factor p.

~~Very~~ naive factoring method (Trial Division)

Does 2 divide n?
Does 3 divide n?
~~Does 4 divide n?~~
Does 5 divide n?
~~Does 6 divide n?~~
$\vdots$

Terminates by finding p in about $\frac{1}{2}p$ steps:

$$\frac{1}{2}p = \frac{1}{2}2^{\log_2 p} \approx 2^{b(p)-1},$$

so the runtime of Trial Division is (still) *exponential* in $b(p)$.

Pollard's "rho" algorithm

Pretend that I want to factor 41831.

First I need a "pseudorandom" sequence mod 41831, say

$$(x_i) = 0, 1, 2, 5, 26, 677, 40020, 16904, 39487, 14476, 23098, 5031, 3207, 36255, 11344, 14181, 19145, 7804, 38312, 1386, 38602, 10523, 6873, 10931, \ldots$$

Pollard's "rho" algorithm ($0 \to 1 \to 2 \to 5 \to 26 \to \cdots$)

What I do: compute $\gcd(41831, |x_{2i+1} - x_i|)$ for $i = 0, 1, 2, \ldots$

$\gcd(41831, |1 - 0|) = 1,$

$\gcd(41831, |5 - 1|) = 1,$

$\gcd(41831, |677 - 2|) = 1,$

$\gcd(41831, |16904 - 5|) = 1,$

$\gcd(41831, |14476 - 26|) = 1,$

$\gcd(41831, |5031 - 677|) = 1,$

$\gcd(41831, |36255 - 40020|) = 1,$

$\gcd(41831, |14181 - 16904|) = 1,$

$\gcd(41831, |7804 - 39487|) = 59.$

What's going on (Birthday Theorem)

We won when $\gcd(41831, |7804 - 39487|) = 59$; ie, when

$$x_{17} \equiv x_8 \pmod{59}.$$

Theorem (The Birthday Theorem)

Suppose you have p numbers. If you choose $\left\lceil \sqrt{\log 4} \cdot \sqrt{p} \right\rceil$ *of these numbers, with repetition, then the probability that two are the same is over* $\frac{1}{2}$*.*

So modulo 59, the odds are you'll get a repeat after

$$\left\lceil \sqrt{\log 4} \cdot \sqrt{59} \right\rceil = 11 \text{ steps.}$$

- Random number generators are not allowed
- Can't store all terms of a sequence, and wait until a repeat.

What's going on (Collision time)

The sequence (x_i) is not a random sequence. In fact, we choose a "pseudorandom" function ϕ modulo 41831, and set

$$x_0 = 0 \qquad x_i = \phi(x_{i-1})$$
$$x_0 = 0 := \phi^0(0) \qquad x_i = \phi^i(0) := \underbrace{\phi \circ \cdots \circ \phi}_{i \text{ times}}(0).$$

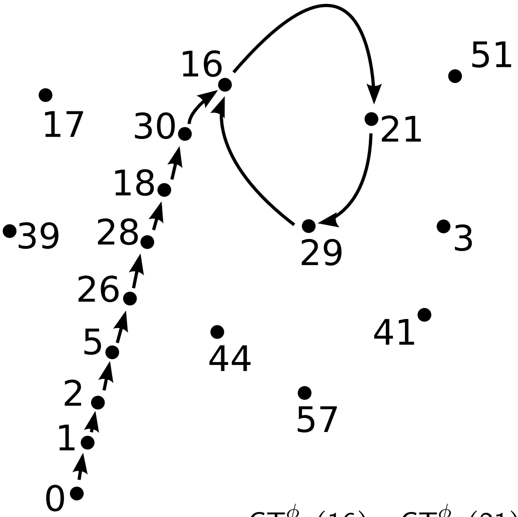
What do I mean by "collision time"?

$\mathrm{CT}^{\phi}(x_0)$ = the smallest i such that $x_i = x_j$ for some $j \le i$.

Well, we found a factor when $x_8 \equiv x_{17}$ mod 59; ie, when

$$\phi^8(0) \equiv \phi^{17}(0) \pmod{59}.$$

What's happening mod 59? $(0 \to 1 \to 2 \to 5 \to 26 \to \cdots)$



$$\mathrm{CT}^{\phi}_{59}(0) = 11$$
$$\mathrm{CT}^{\phi}_{59}(18) = 5$$
$$\mathrm{CT}^{\phi}_{59}(16) = \mathrm{CT}^{\phi}_{59}(21) = \mathrm{CT}^{\phi}_{59}(29) = 3$$

Collision time (continued)

Theorem (Collision time)

The average collision time of random functions on size p set is $\sqrt{p}$. More precisely, there are p^p functions on $\{0, \ldots, p-1\}$ and

$$\frac{1}{p^p} \cdot \sum_{f:\{0,\ldots,p-1\}\to\{0,\ldots,p-1\}} \left(\frac{1}{p} \cdot \sum_{a\in\{0,\ldots,p-1\}} \mathrm{CT}_p^f(a) \right) \sim_p \sqrt{p}.$$

And now something amazing happens

Let $\phi(x) = x^2 + 1$.

Floyd's cycle-finding algorithm

Algorithm is deterministic.

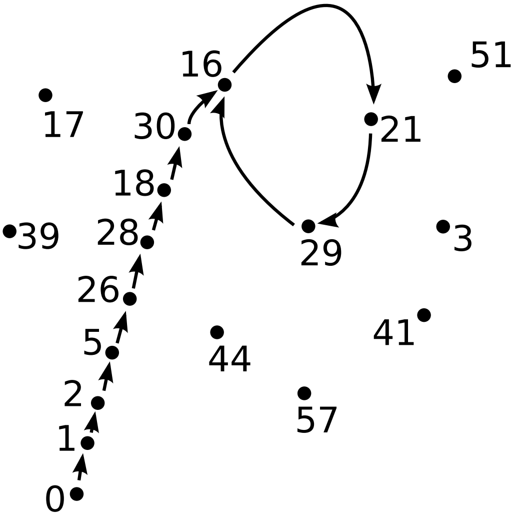
How do we find collisions? (Recall, we can't just "remember" everything that has happened.)

We use Floyd's cycle-finding algorithm:

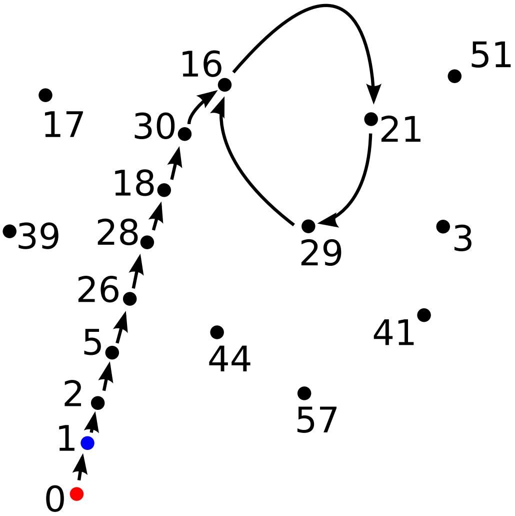
Algorithm (Floyd's cycle finding algorithm)

Compute $|x_{2i+1} - x_i|$ *for* $i = 0, 1, 2, \ldots$

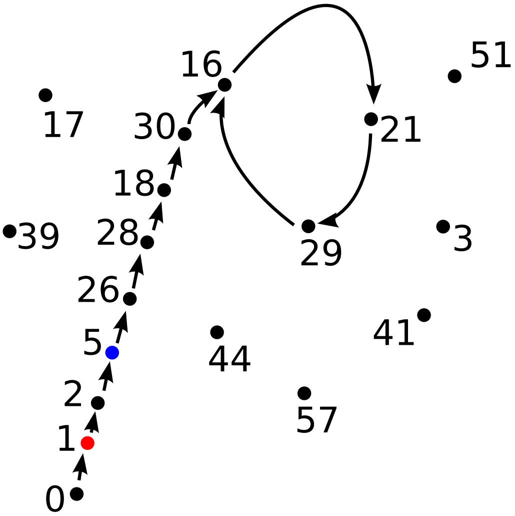
What's happening mod 59?



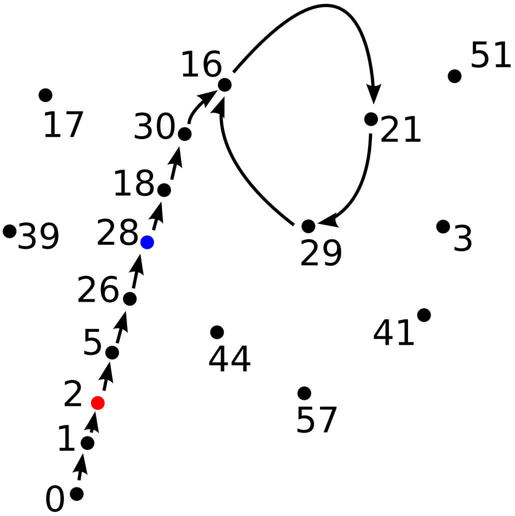
What's happening mod 59?



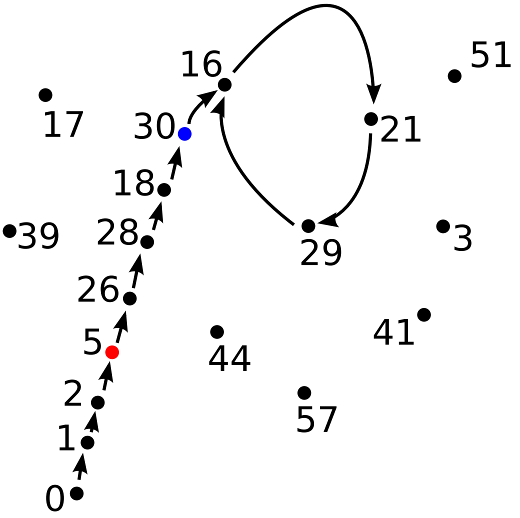
What's happening mod 59?



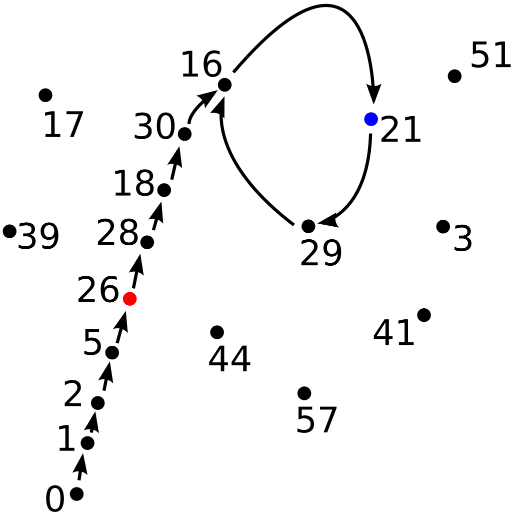
What's happening mod 59?



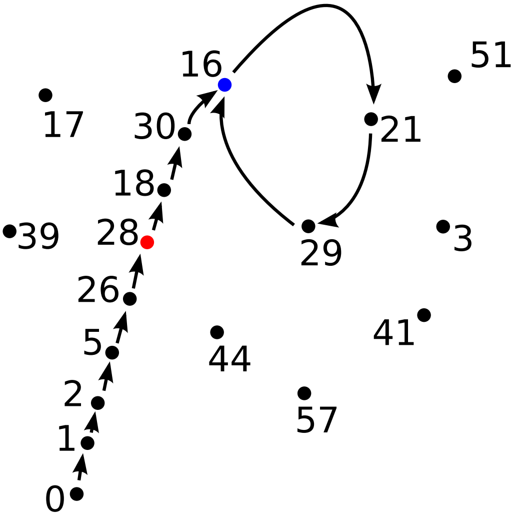
What's happening mod 59?



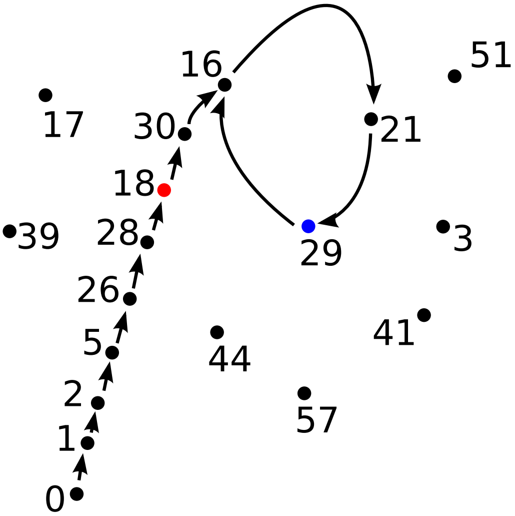
What's happening mod 59?



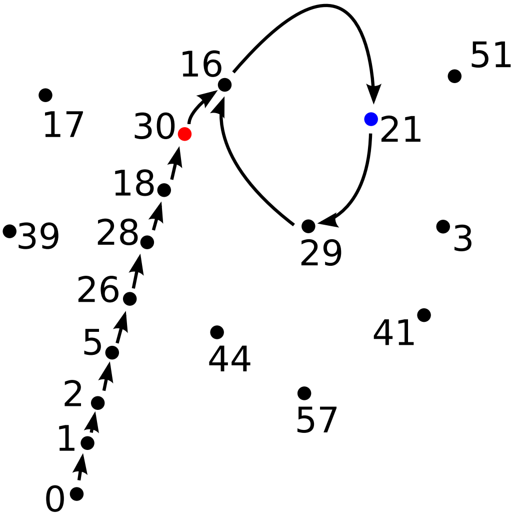
What's happening mod 59?



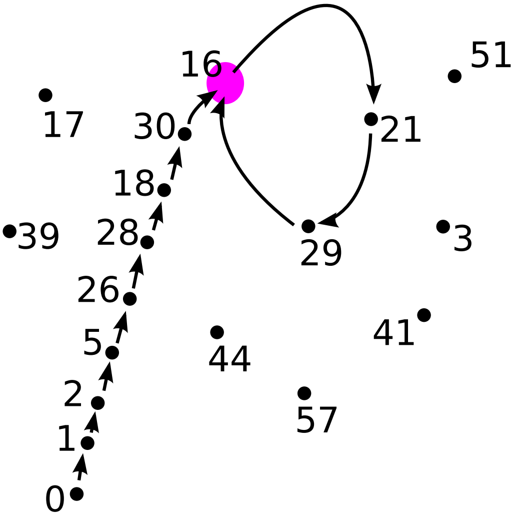
What's happening mod 59?



What's happening mod 59?



What's happening mod 59?



Failure (and conjectural success)

Pollard's "rho" algorithm can fail, just try to factor 4:

$$0 \to 1 \to 2 \to 1 \to 2 \to 1 \to 2 \to 1 \to \cdots$$

Success Conjecture (Silverman)

Let $\phi(x)$ be the polynomial $\phi(x) = x^2 + 1$, and let $\epsilon > 0$. Then.

$$\lim_{X\to\infty} \frac{|\{p \leq X \text{ with ``correct'' CT}\}|}{|\{p \leq X\}|} = \lim_{X\to\infty} \frac{\left|\left\{p \mid \mathrm{CT}_p^{\phi}(0) \leq p^{\frac{1}{2}+\epsilon}\right\}\right|}{|\{p \leq X\}|} = 1$$

Runtime

Success Conjecture

So when the "rho" algorithm works, its runtime is about

$$\sqrt{p} = 2^{\log_2 \sqrt{p}} = 2^{\frac{1}{2}\log_2 p} = 2^{\frac{1}{2}b(p)} = \sqrt{2}^{b(p)}.$$

Better!

Part II: The Arizona Winter School 2010

- Week-long number theory workshop every March.
- It's warm there.
- 2010 was the "dynamics" year.
- Silverman was my project leader.
- Silverman said, "Remember how every polynomial gives you a graph $\bmod p$ for every prime p?"
- "Can you say anything about the average number of components of these graphs?"

Comparing polynomials to random functional graphs

Facts about random functional graphs graphs

- *The average number of components of a random functional graph on p numbers is* $\sim_p \frac{1}{2}\log p$.
- *The average number of periodic points of a random functional graph on p numbers is* $\sim_p \sqrt{p}$.
- *The average collision time of a random functional graph on p numbers is* $\sim_p \sqrt{p}$.

There are two ways to show polynomial functions "act randomly":

- Fix a polynomial ϕ and consider ϕ mod p for all primes p.
- Fix a prime p, and consider all polynomials ϕ (of a fixed, hopefully small, degree d) over $\mathbb{F}_p$.

The "other way" to average polynomial maps

Goal

Fix a prime p and a degree d. Compute

$$\frac{1}{\left|\{\phi \in \mathbb{F}_p[x] \mid \deg(\phi) = d\}\right|} \cdot \sum_{\substack{\phi \in \mathbb{F}_p[x] \\ \deg(\phi) = d}} \left|\{\text{components of } \Gamma_\phi\}\right|.$$

In particular, is the above average anywhere close to $\frac{1}{2} \log p$?

Randomness result

Theorem (Flynn and G.)

$$\frac{1}{\left|\left\{\phi \in \mathbb{F}_p[x] \mid \deg(\phi) = d\right\}\right|} \cdot \sum_{\substack{\phi \in \mathbb{F}_p[x] \\ \deg(\phi)=d}} \left|\left\{\textit{components of } \Gamma_\phi\right\}\right|$$

$$> \log\left(\min\left\{d, \sqrt{p}\right\}\right) - \frac{1}{4}.$$

In particular, if $d \geq \sqrt{p}$, then

$$\frac{1}{\left|\left\{\phi \in \mathbb{F}_p[x] \mid \deg(\phi) = d\right\}\right|} \cdot \sum_{\substack{\phi \in \mathbb{F}_p[x] \\ \deg(\phi)=d}} \left|\left\{\textit{components of } \Gamma_\phi\right\}\right|$$

$$> \frac{1}{2}\log(p) - \frac{1}{4}.$$

A sample computation

Any polynomial looks like of degree d or less has the form

$$\phi(x) = a_d x^d + a_{d-1} x^{d-1} + \cdots + a_1 x + a_0,$$

so there are

- p^{d+1} polynomials of degree d or less,
- p^d polynomials of degree $d-1$ or less, and
- $p^{d+1} - p^d$ polynomials of degree d.

We try to compute

$$\frac{1}{p^{d+1} - p^d} \cdot \sum_{\substack{\phi \in \mathbb{F}_p[x] \\ \deg(\phi) = d}} \left| \left\{ \text{components of } \Gamma_\phi \right\} \right|.$$

Lagrange interpolation

Lagrange interpolation

Suppose you have k data points in $\mathbb{F}_p$; ie, a set $S = \{(a_1, b_1), \ldots, (a_k, b_k)\}$.

Then there exists an interpolating polynomial ϕ_S with $\deg(\phi_S) < k$;
ie, such that $\phi_S(a_i) = b_i$ for all $i \in \{1, \ldots, k\}$.

This is the analog of the "theorem" that two points determine a line; ie, given (x_1, y_1) and (x_2, y_2), that line is

$$y - y_2 = \frac{y_2 - y_1}{x_2 - x_1}(x - x_2)$$

so $\phi_S(x) = \frac{y_2 - y_1}{x_2 - x_1}(x - x_2) + y_2$.

Examples of Lagrange interpolation

- Suppose $S = \{(2,2),(4,6),(5,2)\} \subseteq \mathbb{F}_7 \times \mathbb{F}_7$.
  Then $\phi_S(x) = 5x^2 + 3$, and

$$\begin{aligned} \phi_S(2) &= 2 \\ \phi_S(4) &= 6 \\ \phi_S(5) &= 2. \end{aligned}$$

- Suppose $S = \{(1,3),(2,0),(5,3),(6,1)\} \subseteq \mathbb{F}_7 \times \mathbb{F}_7$.
  Then $\phi_S(x) = x^2 + x + 1$, and

$$\begin{aligned} \phi_S(1) &= 3 \\ \phi_S(2) &= 0 \\ \phi_S(5) &= 3 \\ \phi_S(6) &= 1. \end{aligned}$$

Idea of proof

Instead of counting "components per polynomial", count "polynomials per component" instead.

That is,

$$\sum_{\substack{\phi \in \mathbb{F}_p[x] \\ \deg(\phi)=d}} \left|\left\{\text{components of } \Gamma_\phi\right\}\right|$$

$$= \sum_{\substack{\text{possible} \\ \text{components } C}} \left|\left\{\text{polynomials } \phi \text{ interpolating } C\right\}\right|.$$

Observation

Cycles are in 1-1 correspondence with graph components, so let's count cycles instead.

Making the counting problem easier

- Count the number of possible cycles of length k.
- Count the number of polynomials interpolating each k-cycle.
- Then

$$\begin{aligned}
&\sum_{\substack{\phi\in\mathbb{F}_p[x]\\ \deg(\phi)=d}} \left|\left\{\text{components of } \Gamma_\phi\right\}\right| \\
&= \sum_{\substack{\text{possible}\\ \text{components } C}} \left|\left\{\text{polynomials } \phi \text{ interpolating } C\right\}\right| \\
&= \sum_{\substack{\text{possible}\\ \text{cycles } C}} \left|\left\{\text{polynomials } \phi \text{ interpolating } C\right\}\right| \\
&= \sum_{k=1}^{p} \left(\begin{matrix}\text{number of cycles}\\ \text{of length } k\end{matrix}\right)\left(\begin{matrix}\text{number of polynomials}\\ \text{interpolating a } k\text{-cycle}\end{matrix}\right).
\end{aligned}$$

The final step of the proof (and recap)

$$\frac{1}{\left|\left\{\phi \in \mathbb{F}_p[x] \mid \deg(\phi) = d\right\}\right|} \cdot \sum_{\substack{\phi \in \mathbb{F}_p[x] \\ \deg(\phi)=d}} \left|\left\{\text{components of } \Gamma_\phi\right\}\right|$$

$$= \frac{1}{p^{d+1} - p^d} \cdot \sum_{\substack{\phi \in \mathbb{F}_p[x] \\ \deg(\phi)=d}} \left|\left\{\text{components of } \Gamma_\phi\right\}\right|$$

$$= \frac{1}{p^{d+1} - p^d} \cdot \sum_{\substack{\text{possible} \\ \text{components } C}} \left|\left\{\text{polynomials } \phi \text{ interpolating } C\right\}\right|$$

$$= \frac{1}{p^{d+1} - p^d} \cdot \sum_{\substack{\text{possible} \\ \text{cycles } C}} \left|\left\{\text{polynomials } \phi \text{ interpolating } C\right\}\right|$$

$$= \frac{1}{p^{d+1} - p^d} \cdot \sum_{k=1}^{p} \binom{\text{number of cycles}}{\text{of length } k} \binom{\text{number of polynomials}}{\text{interpolating a } k\text{-cycle}}$$

$$> \frac{1}{p^{d+1} - p^d} \cdot \sum_{k=1}^{\min\{d, \sqrt{p}\}} \left(\frac{p^k}{k}\right)\left(p^{d-k+1} - p^{d-k}\right) = \sum_{k=1}^{\min\{d, \sqrt{p}\}} \frac{1}{k}. \quad \square$$

The theorem

Theorem (Flynn and G.)

$$\frac{1}{\left|\{\phi \in \mathbb{F}_p[x] \mid \deg(\phi) = d\}\right|} \cdot \sum_{\substack{\phi \in \mathbb{F}_p[x] \\ \deg(\phi) = d}} \left|\{\textit{components of } \Gamma_\phi\}\right| > \log\left(\min\{d, \sqrt{p}\}\right) - \frac{1}{4}.$$

In particular, if $d \geq \sqrt{p}$, then

$$\frac{1}{\left|\{\phi \in \mathbb{F}_p[x] \mid \deg(\phi) = d\}\right|} \cdot \sum_{\substack{\phi \in \mathbb{F}_p[x] \\ \deg(\phi) = d}} \left|\{\textit{components of } \Gamma_\phi\}\right| > \frac{1}{2}\log(p) - \frac{1}{4}.$$

Further work

We only count k-cycles for $k \leq d$. The long cycles are mysterious.

What about when $d + 1 < k < \sqrt{p}$?

- *Well, we bound the average below, by* $\sum_{k=1}^{d} \frac{1}{k} \approx \log d$.
- *But we want a better bound!*
- *How many degree d polynomials interpolate a k-cycle when $k > d$?*
- *The odds that three points are on a line is* $\frac{1}{p}$.
- *The odds that four points are on a line is about* $\frac{1}{p^2}$.
- *The odds that k points are on a degree d polynomial... is*
$$p^{d+1-k}?$$
- *Do these probabilities carry over to cycles?*
- *Do these probabilities give the right answer?*

More further work

What about when $k > \max\{\sqrt{p}, d+1\}$?

- *We only bound the average from below, what about from above?*
- *When $k > \sqrt{p}$, the number of cycles is "about" zero.*
- *Is it "zero enough" to give an upper bound, assuming the guess on the previous slide?*
- *We could carry out computations to test the hypothesis that degree 2 polynomials act like random functional graphs.*

Thank you slide

Thank you!