Mathematical Sciences Technical Reports (MSTR)   Mathematics

7-29-2011

The Elliptic Curve Discrete Logarithm and Functional Graphs

Christopher J. Evans
University of Arkansas

The Elliptic Curve Discrete Logarithm and Functional Graphs

C. Evans

Adviser: Joshua Holden

Mathematical Sciences Technical Report Series
MSTR 11-01

July 29, 2011

Department of Mathematics
Rose-Hulman Institute of Technology
http://www.rose-hulman.edu/math

Fax (812)-877-8333   **Phone (812)-877-8193**

THE ELLIPTIC CURVE DISCRETE LOGARITHM AND FUNCTIONAL GRAPHS

CHRISTOPHER J. EVANS

UNIVERSITY OF ARKANSAS, DEPARTMENT OF MATHEMATICS

ROSE-HULMAN REU, 2011

CC.JOHN.EVANS@GMAIL.COM

ADIVSOR: JOSHUA HOLDEN

ROSE-HULMAN INSTITUTE OF TECHNOLOGY

WITH SUPPORT FROM NSF GRANT DMS-1003924

JULY 29, 2011

ABSTRACT. The discrete logarithm problem, and its adaptation to elliptic curves, called the elliptic curve discrete logarithm problem (ECDLP) is an open problem in the field of number theory, and its applications to modern cryptographic algorithms are numerous. This paper focuses on a statistical analysis of a modification to the ECDLP, called the x-ECDLP, where one is only given the x-coordinate of a point, instead of the entire point. Focusing only on elliptic curves whose field of definition is smaller than the number of points, this paper attempts to find a statistical indication of underlying structure (or lack thereof) in the ECDLP. Using functional graphs, this translates into discovering certain structure in the functional graphs induced by the ECDLP on a wide range of elliptic curves, and comparing these statistics to what would be expected of a random functional graph. A lack of randomness may translate into an exploitable structure in the ECDLP that would undermine the security of cryptographic algorithms based on this problem. In addition to providing an extensive list of data for small primes, this paper details a generating function for the number of elliptic curve functional graphs (ECFGs).

1. INTRODUCTION

The rising importance of cryptography in modern society is a result of increased use of the web as a medium for transferring information. Fast, secure, and effective forms of encryption and identity assurance have become an important and broad area of research in computer science and mathematics. In terms of public-key cryptography, many protocols rely on the supposed difficulty of what is known as the discrete logarithm problem:

$$(1) \qquad\qquad g^r \equiv s \mod p$$

That is, given s, p, and g, calculate r. There is currently no known polynomial time algorithm for computing r, which means that for sufficiently large values of p, the value of r cannot be computed in any reasonable time. For example, to

ensure data security until the year 2041, the European Network of Excellence in Cryptology II (ECRYPT II) recommends a value of p on the order of 15,424 bits for discrete logarithm based encryption.[3] It is this lack of an efficient algorithm that underlies the security of cryptosystems utilizing (1).

The discrete logarithm problem has been adapted to elliptic curves in the hopes of providing even more security. The basic idea is that, for any prime p, there is only one field, $\mathbb{F}_p$. For elliptic curves, however, the number of possible elliptic curves over $\mathbb{F}_p$ is extremely large, even for small values of p. Furthermore, if one is looking at elliptic curves with N points, there are also a significant number of elliptic curves with N points. For example, for $N = 167$, there are 208,496 possible elliptic curves. This provides much more freedom than cryptosystems based on (1). In addition, the best known algorithms for solving the ECDLP are slower than the best known algorithms to solve the classical discrete logarithm problem. The extent of this increased difficulty is evident from the ECRYPT II recommendations for keysizes. Although algorithms based on the discrete logarithm require values of p around 15,424 bits, similar protection can be achieved through elliptic curve cryptography with only 512 bits. [3] Though the specifics will be dealt with later in the paper, a brief sketch of the elliptic curve discrete logarithm problem $ECDLP$ is as follows. It is a basic fact from the theory of elliptic curves that every elliptic curve admits an abelian group structure on the set of rational points. Let $E\left(\mathbb{F}_p\right)$ denote an elliptic curve taking values in $\mathbb{F}_p$, and $B \in E\left(\mathbb{F}_p\right)$ denote a point on the curve E. Then, given the additive structure of the points, we have the following notational convention:

$$kB = B + B + \cdots + B \,(k \text{ times})$$

With this in mind, we get the following definition of the ECDLP: Given a basepoint B, and elliptic curve E, and a point $P \in E\left(\mathbb{F}_p\right)$ such that

$$(2) \qquad\qquad k \mapsto P = kB$$

calculate the value of k. This paper is concerned with a slight modification of this problem, known as the x-ECDLP, in which one maps k to the x-coordinate of $P = kB$.

Any underlying structure to this mapping may suggest a method of quickly computing the constant k, and therefore undermining the security of cryptographic algorithms based on this mapping. This paper focuses on a graph-theoretic approach in which the mapping is turned into a functional graph, and a statistical analysis of the graph is undertaken in an attempt to find any potential structure. This reduces a complex algebraic problem into a graph-theoretical problem, which is in some ways easier to study.

2. Background

2.1. **Elliptic Curves.** In an attempt to clarify notational aspects, the following conventions will be used: E refers to an elliptic curve. $E\left(\Bbbk\right)$ denotes an elliptic curve taking values in a field $\Bbbk$ where we assume (for computational simplicity) that $char\left(\Bbbk\right) \neq 2, 3$. In fact, $\Bbbk$ will always be understood to be a field of characteristic greater than 3. $\mathbb{F}_p$ will denote the field $\mathbb{Z}_p$ for a p a prime.

With some basic notation out of the way, we will now cover the necessary aspects of the theory of elliptic curves defined over a finite field. Almost all proofs of these facts will be omitted, see one of the canonical references for the proofs [4]. An elliptic curve $E\left(\Bbbk\right)$ is defined by the following equation:

$$E\left(\Bbbk\right) : y^2 = x^3 + ax^2 + b \tag{3}$$

The values a, b are naturally elements of the field $\Bbbk$. This equation only works for fields of characteristic not equal to 2 or 3. For such fields, more general equations must be used. An equation of the form (3) will be referred to as a Weierstrauss equation, or as an equation in Weierstrauss form. Any elliptic curve defined over k is isomorphic (as an algebraic variety), to a curve given in Weierstrauss form. We will now discuss the most important aspect of an elliptic curve from our perspective, its underlying abelian group structure on the set of rational points.

Given a curve E defined by $y^2 = x^3 + ax + b$ and $P, Q, R \in E$ and letting $P = (x_1, y_1)$, $Q = (x_2, y_2)$, and $-R = (x_3, y_3)$ we arrive at the set of formulas:

(1) First, if $Q = \infty$, then $P + Q = -R = P$
   Any line that *seems* to intersect our curve at only two points will have a third point of intersection "at infinity"

(2) If $P = Q$ and $y_1 = 0$, then $P + Q = \infty$
(3) Now, if $x_1 \neq x_2$:
$$x_3 = m^2 - x_1 - x_2, \ y_3 = m\left(x_1 - x_3\right) - y_1 \text{ where } m = \frac{y_2 - y_1}{x_2 - x_1}$$

(4) If $P = Q$ and $y_2 \neq 0$, then:
$$x_3 = m^2 - 2x_2, \ y_3 = m\left(x_2 - x_3\right) - y_2 \text{ where } m = \frac{3x_2^2 + a}{2y_2}$$

(5) If $x_1 = x_2$ but $y_1 \neq y_2$, then $P + Q = \infty$.

These algebraic formulas have a geometric interpretation, which is given in Figure 1.

A well-known result in the theory of elliptic curves is that for any elliptic curve $E\left(\mathbb{F}_p\right)$, the associated group of rational points will be cyclic, or the direct sum of two cyclic groups. Since this paper is only considering elliptic curves with a prime
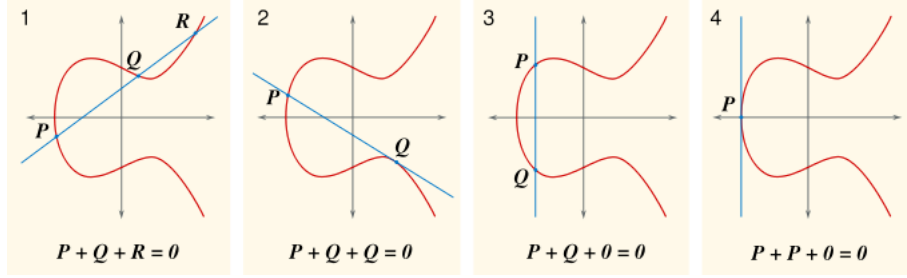
FIGURE 1. A geometric depiction of the group law on elliptic curves
Image taken from the Wikipedia article on elliptic curves

number of points, it follows that the underlying group will always be $\mathbb{Z}_q$ for some prime q.

2.1.1. *Counting Points on Elliptic Curves.* An important part of the methodology of the paper requires determining the number of points on a given elliptic curve. We will refer to the number of points on an elliptic curve $E\left(\mathbb{F}_p\right)$ as the order of $E\left(\mathbb{F}_p\right)$, denoted $\left|E\left(\mathbb{F}_p\right)\right|$. A number of algorithms exist for the efficient computation of the order of an elliptic curve. The interested reader should consult [4] and [5] for the relevant details. We will merely concern ourselves with a discussion of the theoretical background used in the computations for this paper. We will first define a modified version of the standard Legendre symbol.

Definition 2.1. *Given a finite field $\mathbb{F}_p$ let the symbol $\left(\frac{x}{\mathbb{F}_p}\right)$ be defined as follows:*

$$\left(\frac{x}{\mathbb{F}_p}\right) = \begin{cases} +1 \text{ if } t^2 = x \text{ has a solution } t \in \mathbb{F}_p^{\times}, \\ -1 \text{ if } t^2 = x \text{ has no solution } t \in \mathbb{F}_p, \\ 0 \text{ if } x = 0. \end{cases}$$

Theorem 2.1. (Hasse's Theorem on Elliptic Curves) *If N is the number of points on an elliptic curve $E\left(\mathbb{F}_p\right)$, then:*

$$|N - (p+1)| \leq 2\sqrt{p}$$

Proof. We refer the reader to Joseph Silverman's book [4] for the result. $\qquad\square$

.

Theorem 2.2. *Let E be an elliptic curve $y^2 = x^3 + ax + b$ over $\mathbb{F}_p$. Then,*

$$\left|E\left(\mathbb{F}_p\right)\right| = p + 1 + \sum_{x \in \mathbb{F}_p} \left(\frac{x^3 + ax + b}{\mathbb{F}_p}\right)$$

Proof. We refer the reader to Lawrence Washington's book [5] for the result. $\qquad\square$

2.2. **Functional Graphs.** Our approach to studying the ECDLP depends on the statistical analysis of functional graphs. Before the definition of a functional graph is introduced, a brief review of graph theory is in order.

Definition 2.2. *A* directed graph *$G = (V, E)$ is a graph where V denotes the set of vertices, or nodes, and $E \subset V \times V$ is a collection of ordered pairs such that $(v_1, v_2) \in E$ implies that there exists an edge directed from v_1 to v_2.*

Definition 2.3. *The* out-degree, *denoted $\deg^+ (v)$ of a vertex v is the number of vertices u such that $(v, u) \in E$.*

Definition 2.4. *Similarly, the* in-degree, *denoted $\deg^- (v)$ of a vertex v is the number of vertices u such that $(u, v) \in E$.*

Definition 2.5. *A* functional graph *is a directed graph $G = (V, E)$ such that for every $v \in V, \deg^+ (v) = 1$.*

Definition 2.6. *A* terminal node *is a vertex v such that $\deg^- (v) = 0$. Otherwise, v is called an* image node.

It is simple to notice that any function that maps a domain to itself induces an associated functional graph. Given any function f, let V be the domain of f. Then, an ordered pair (v_1, v_2) represents an edge in the graph if and only if $f(v_1) = v_2$. Conversely, every functional graph with vertex set V induces a function $f : V \to V$. For a function $f : V \to V$ we will denote the corresponding functional graph by $G_f = (V_f, E_f)$. Likewise, for a functional graph G, the associated function will be denoted by f_G.

The requirement that the out-degree of any vertex is 1 is equivalent to the requirement that a function map each element in the domain to precisely one element in the range. Similarly, the in-degree of a vertex represents the number of elements that map to that element under the associated function.

Definition 2.7. *A* binary functional graph *is a functional graph in which every vertex has in-degree 0 or 2.*

Definition 2.8. *Given a functional graph $G = (V, E)$ and its associated function f, a vertex $v \in V$ is called a* cyclic node *if there exists a $k \in \mathbb{N}$ such that $f^k (v) = v$, where $f^k (v)$ denotes the k-th iteration of $f(v)$. Otherwise, v is called a* tail node.

The usefulness of functional graphs lies in the transformation of a function-theoretic problem into a graph-theoretic problem. For example, given a function $f : V \to V$, instead of asking for which $v \in V$ we have $f(v) = v$, one considers the vertices of G_f such that there exists an edge $(v, v) \in E_f$. This is extremely beneficial when one considers the entire class of functions mapping a set V to itself, because the power of enumerative combinatorics can be used to consider the associated class of functional graphs.

When one considers the statistical properties of a *random* function from a set V to itself, these enumerative results become extremely important, since it allows normalization and statistical analysis. This will be the primary approach in this paper.

The functional graphs discussed in this paper will be constructed from elliptic curves as follows. Given a curve $E(\mathbb{F}_p)$ with N points, the vertex set V will be the set $\{0, 1, \ldots, N-1\} \cup \{\infty\}$. A possible issue with this is that considering the curve $E(\mathbb{F}_p)$ as embedded in projective space means that the point ∞ should technically *not* be included in the graph. Instead, since ∞ has x-coordinate 0, if $kB = \infty$, we would put an edge from k to 0. This paper will consider the curve $E(\mathbb{F}_p)$ as embedded in affine space, and therefore ∞ will be treated as a node separate from 0. In this case, ∞ will always map to itself. In terms of functional graphs, this means that ∞ will always be a fixed point. This will ensure that every functional graph will be binary (as discussed below), without losing or distorting any information about the mapping. Now, given a basepoint B for the curve $E(\mathbb{F}_p)$, we will draw an edge from 1 to $x(B)$. We will then draw an edge from 2 to $x(2 \cdot B)$. This process continues through the last node $N-1$. An example graph is shown in Figure 2.

3. Prior Work

To the knowledge of the author, little work has been done on elliptic curves using the approach of functional graphs. The notable exception is the paper by Aaron Blumenfeld that calculated several statistics on the family of ECFGs for a few small values. Some of the notable results obtained by Blumenfeld are listed below, and the associated proofs can be found in the paper by Blumenfeld [1].

Theorem 3.1. *For an elliptic curve defined over the field $\mathbb{F}_p$ for $p < N$ the induced ECFG is a binary functional graph.*

Theorem 3.2. *If $N > p$ is odd, then there will be $\frac{N+!}{2}$ terminal nodes, and the other $\frac{N+1}{2}$ nodes will be image nodes*

Theorem 3.3. *If b is not a square modulo p, then there will be a component consisting solely of $\{0, \infty\}$.*

Theorem 3.4. *A basepoint B and its negation, $-B$ generate the same functional graphs.*

As a final observation, the points $p, p+1, p+2, \ldots, N$ will always be terminal nodes, since they will never be the x-coordinate of a point defined over $\mathbb{F}_p$.

The primary obstacle in this statistical analysis was the generation of a sufficient number of elliptic curves and the associated functional graphs. As a solution, the author used a series of computer programs initially written by Blumenfeld, that have since undergone significant modifications. The original code for generating
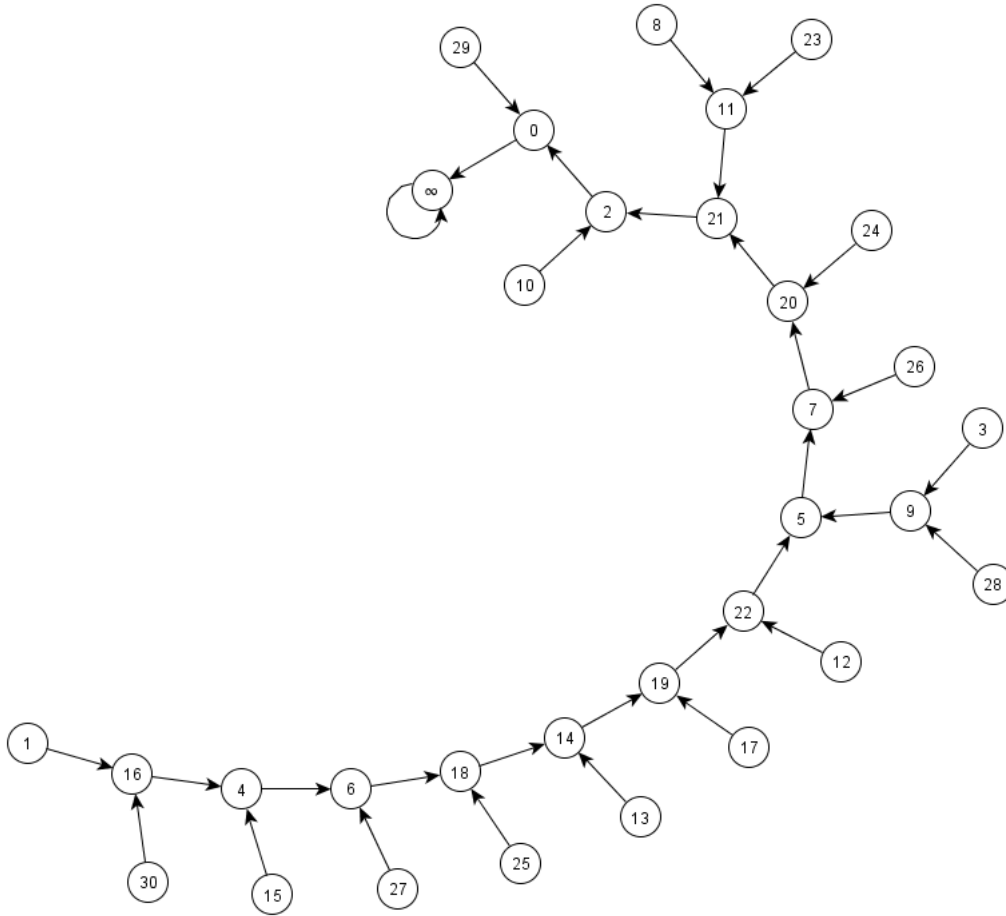
FIGURE 2. The curve $y^2 = x^3 + 7x + 3 \mod 23$ with basepoint $B = (16, 18)$ and 31 points

a list of elliptic curves of order N defined over $\mathbb{F}_p$ for $p < N$ was ported to C++ and threaded over multiple processing cores. The remaining files had their output routines tweaked, but remained functionally identical to Blumenfeld's original code. The code was optimized by threading over multiple processing cores and simplifying the format of output files to eliminate unnecessary overhead in writing to files.

3.1. **Methods.** We only need to concern ourselves with half the basepoints, since the negation of a basepoint generates the same functional graph. Thus we will only consider the *necessary* basepoints, that is, any set S of $\frac{N+1}{2}$ basepoints such that for $B \in S, -B \notin S$.

As a sketch of the overall process, one program generated a list of all elliptic curves of order N defined over some finite field of prime order $p < N$, along with

the necessary basepoints. This output was run through a second program that computed various characteristics of the graph (listed below) for each combination of elliptic curve and basepoint.

The most computationally intensive portion was the enumeration of all elliptic curves satisfying the given conditions. To calculate the number of points on the curve, the author simply multiplied a basepoint B until $kB = B$. Since we only consider elliptic curves with a prime number of points, the underlying group will be cyclic. This implies that every point on the curve (except ∞) will generate the entire curve, and so the order of any given point determines the number of points on the curve to be k.

The graph-generating code used is functionally identical to Blumenfeld's code, with slightly modified output to suit the needs of the author. Unlike the code for elliptic curves, no optimization was performed on this code. Complete reproductions of all code used are to be found in the appendix of the paper, with the exception of a few Perl scripts used to organize the output.

This paper will be concerned with the collection of the following graph characteristics for all ECFGs on N nodes:

(1) Average number of components: the total number of components over all graphs on N nodes normalized by the total number of graphs on N nodes
(2) Average number of cyclic nodes: the average number of nodes that belong to a cycle
(3) Average number of tail nodes: the average number of nodes that are not in a cycle
(4) Maximum cycle length: the number of nodes in the largest cycle in any graph on N nodes
(5) Maximum tail length: the number of nodes in the longest tail in any graph on N nodes

The expected values for various graph statistics are computed using the method of generating functions. The values given are derived in the paper by Cloutier and Holden [2]. The expected values of graph statistics for a random binary functional graph with n nodes are tabulated below.

Statistic	Expected Asymptotic Value
Number of Components	$\frac{1}{2}\left(\log\left(2n\right) + \gamma\right)$
Number of Cyclic Nodes	$\sqrt{\pi n/2} - 1$
Number of Tail Nodes	$n - \sqrt{\pi n/2} + 1$
Maximum Cycle Length	$\approx 0.78248\sqrt{n}$
Maximum Tail Length	$\approx 1.73746\sqrt{n} + 1.61371$

4. RESULTS

The results concerning the number of binary functional graphs and the related asymptotic results are sufficient for a rough analysis of the ECDLP, but fail to account for a significant amount of structure in ECFGs, primarily the fixed point at ∞ that is present in any ECFG. Failure to account for this structure may easily skew any statistical analysis, and therefore more refined enumerative results are needed for a proper normalization of the statistics.

The use of generating functions gives an exact count, and allows the computation of asymptotic values, along with mean values for various characteristic of the graph. Adapting the work done by Cloutier and Holden in [2], we arrive at an exact value for the number of binary functional graphs with one fixed point. First, however, we summarize the results for binary functional graphs given by Cloutier and Holden.

As a first observation, we notice that a binary functional graph consists of a set of weakly-connected components. Each component is a cycle, with a binary tree attached to each node. Using the theory of generating functions, we arrive at the following series of counting functions, beginning with the number of binary trees on N nodes:

$$(4) \qquad\qquad t(z) = z + \frac{z\, t(z)^2}{2}$$

This equation is used to count the number of components:

$$(5) \qquad\qquad c(z) = \ln \frac{1}{1 - z\, t(z)}$$

Combining (4) and (5), we arrive at the number of binary functional graphs on N nodes:

$$(6) \qquad\qquad f(z) = e^{c(z)} = \frac{1}{1 - z\, t(z)}$$

Now, adapting this to require that at least one component has a fixed point, we arrive at the following:

$$(7) \qquad\qquad g(z) = z\, t(z)\, f(z)$$

This follows because the $zt(z)$ term accounts for a component with a fixed point. This alone only accounts for one possible component. There are no restrictions on the other possible components other than that they are binary functional graphs. These extra components are accounted for by the $f(z)$ term.

We then adapt our component counting function to only count graphs with one fixed point. So, proceeding as in the previous example we get:

$$(8) \qquad\qquad c^*(z) = zt(z) \exp\left(\ln \frac{1}{1 - zt(z)} \right)$$

as an intermediate step. To actually calculate the number of components, we "mark" the function with the variable u to count the number of components,

and then differentiate with respect to u. Lastly, we set $u = 1$. The appropriate marking is given below, followed by the final form of the generating function, after differentiation and substitution of $u = 1$.

$$(9) \qquad c_{marked}(z) = uzt(z) \exp\left(u \ln \frac{1}{1 - zt(z)}\right)$$

$$(10) \qquad \left(1 - \sqrt{1 - 2z^2}\right) e^{\frac{-\ln\left(1 - 2z^2\right)}{2}} - \frac{1}{2}\left(1 - \sqrt{1 - 2z^2}\right) \ln\left(1 - 2z^2\right) e^{\frac{-\ln\left(1 - 2z^2\right)}{2}}$$

The use of singularity analysis, as in [2], gives the following asymptotic result for the total number of components:

$$(11) \qquad 0.3988 \ln(N) \sqrt{\frac{1}{N}} 2^{\frac{1}{2N}} + 1.304 \sqrt{\frac{1}{N}} 2^{\frac{1}{2N}}$$

Below is a summary of the collected statistics for a several prime values of N.

Connected Components for N<100		
N	Observed Average	Expected Average
13	2.7586	3.0065
17	2.5172	3.1220
19	2.6224	3.1710
23	2.9194	3.2562
29	2.7647	3.3622
31	2.9692	3.3920
37	3.2026	3.4753
41	3.2023	3.5234
43	3.3217	3.5458
47	3.4855	3.5879
53	3.3341	3.6449
59	3.3859	3.6961
61	3.4938	3.7121
67	3.5690	3.7571
71	3.5766	3.7850
73	3.5811	3.7984
79	3.6747	3.8365
83	3.7427	3.8604
89	3.7524	3.8942
97	3.8107	3.9360

5. Conclusion

Although plenty of data has been collected, namely the elliptic curves for all prime values of N less than 900, and corresponding graph statistics for all prime N less than 200, the need for a thorough statistical analysis remains. Preliminary to a thorough statistical analysis is further work on generating functions to find asymptotic values for the various graph characteristics. The option also remains open to analyze certain classes of primes, either as the field of definition for the elliptic curve, or the order of the curve in an attempt to find classes of primes that demonstrate particular structure, or lack thereof.

As for the limited results obtained, it appears from simple observation that the the observed average number of components is consistently just below the expected average. This seems to suggest that the average number of components might very well be a random property of ECFGs, though the data set is obviously too small and limited for any definitive conclusions to be made.

As a final note, in gathering these statistics, the author noted that the creation of functional graphs and the calculation of properties for each curve consumes an enormous amount of computational time, and therefore needs significant optimizations to allow the feasible study of larger sets of primes.

References

[1] Aaron Blumenfeld. Discrete logarithms on elliptic curves. Technical Report MSTR 10-04, Rose-Hulman Institute of Technology, 2010.
[2] Daniel R. Cloutier and Joshua Holden. Mapping the discrete logarithm. *Involve*, 3(2), 2010.
[3] European Network of Excellence in Cryptology II. ECRYPT II yearly report on algorithms and keysizes (2009-2010). Technical report, European Network of Excellence in Cryptology II, 2009.
[4] Joseph H. Silverman. *The Arithmetic of Elliptic Curves*. Graduate Texts in Mathematics; 106. Springer-Verlag, 1986.
[5] Lawrence C. Washington. *Elliptic Curves: Number Theory and Cryptography*. Chapman & Hall/CRC, 2003.