Conway Polynomials and other Compatible Polynomials

Rohit Gurjar
Dept. Of Computer Science and Engineering, IIT Kanpur

August 20, 2008

Abstract

Conway Polynomials, a particular class of irreducible polynomials, provides a means for efficiently representing finite fields in computational algebra systems. In 1999, Heath and Loehr gave two algorithms for generating Conway Polynomials, which perform better than the brute force algorithm in most cases. The first algorithm, based on roots of lower Conway Polynomials, was erroneous, in particular it worked only for those $GF(p^n)$ s.t. $\mu(n) \neq 0$, where μ is the möbius function. We corrected the algorithm while maintaining the time complexity and implemented it in GAP. We also discuss an approach for generating another family of polynomials which has the property of compatibility like the Conway Polynomials and is much easier to compute.

Introduction

Every finite field $\mathbb{F}$ is characterized by two parameters- its prime characteristic p and its dimension n over $\mathbb{Z}_p$, the integers modulo p. The field $\mathbb{F}$ has p^n elements and is isomorphic to any other field having p^n elements. The field $\mathbb{F}$ is often denoted as $GF(p^n)$, where GF is for *Galois field*. The multiplicative group of $\mathbb{F}$ is denoted as $\mathbb{F}^*$ and is always cyclic. A primitive element of $\mathbb{F}^*$ is any element that generates the multiplicative group. In particular, if $\alpha \in \mathbb{F}^*$ is a primitive element, then

$$1 = \alpha^0, \alpha, \alpha^2, \ldots, \alpha^{p^n-2}$$

are the elements of $\mathbb{F}^*$. As we will frequently need the cardinality $p^n - 1$ of the multiplicative group, let $M_{p,n}$ denote $p^n - 1$.

Let $\mathbb{Z}_p[x]$ be the one-variable polynomial ring over $\mathbb{Z}_p$. A polynomial $f \in \mathbb{Z}_p[x]$ is irreducible if $f = gh$ implies that either g or h is a constant. An irreducible polynomial f of degree n is primitive if some root (and hence every root) of f is a primitive element of $GF(p^n)^*$. Typically, the finite field $GF(p^n)$ is represented as the quotient ring $\mathbb{Z}_p[x]/(f)$, where f is an irreducible polynomial of degree n. Moreover, if f is primitive, then a representation of the elements of $GF(p^n)^*$ can be based on a root α of f. For an element $\gamma \in GF(p^n)^*$, define the index of γ to be the smallest integer $i \geq 0$ such that $\alpha^i = \gamma$. Alternatively, we can uniquely represent each element $\beta \in GF(p^n)$ as a polynomial in α of degree at most $n - 1$.

The index representation turns multiplication in $GF(p^n)^*$ into addition modulo $M_{p,n}$, while the polynomial representation makes for straightforward addition in $GF(p^n)$. Challenge arise when representing more than just the two fields $\mathbb{Z}_p$ and $GF(p^n)$. Consider the case of a chain of fields $\mathbb{Z}_p \subset GF(p^{n_1}) \subset GF(p^{n_2})$, where $1 < n_1 < n_2$. In this case, n_1 divides n_2. Suppose that α_1 and α_2 are primitive elements of $GF(p^{n_1})$ and $GF(p^{n_2})$, respectively. The cyclic group $GF(p^{n_1})^*$ is a subgroup of the cyclic group $GF(p^{n_2})^*$, and the smallest power of α_2 that gives a generator γ of $GF(p^{n_1})$ is

$$\gamma = \alpha_2^{M_{p,n_2}/M_{p,n_1}}$$

Arithmetic in this cahin of fields, especially multiplication, will be most convenient if $\gamma = \alpha_1$. If f_1 and f_2 are the minimal polynomials of α_1 and α_2, respectively, then it is easy to see that $\alpha_1 = \alpha_2^{M_{p,n_2}/M_{p,n_1}}$ implies

$$f_2(x) | f_1(x^{M_{p,n_2}/M_{p,n_1}}).$$

We generalize these observations as follows. Suppose that for each of the subfields $GF(p^{n'})$ of a finite field $GF(p^n)$ we have chosen a primitive, irreducible polynomial $f_{p,n'} \in \mathbb{Z}_p[x]$ of degree n'. If whenever $n_1|n_2$ and $n_2|n$, we have

$$f_{p,n_2}(x)|f_{p,n_1}(x^{M_{p,n_2}/M_{p,n_1}}),$$

then we say that the polynomial chosen are compbatible.

As defined by Jansen et al.(1995) and Parker (1990), Conway Polynomials is a particular collection of particular of compatible polynomials. The Conway polynomial for representing $GF(p^n)$ is denoted as $C_{p,n}$. To define the Conway polynomials, we first introduce a lexicographic order $<_{lex}$ on polynomials of degree d in $\mathbb{Z}_p[x]$.

$$a_d x^d + a_{d-1}x^{d-1} + \cdots + a_1 x + a_0 <_{lex} b_d x^d + b_{d-1}x^{d-1} + \cdots + b_1 x + b_0$$

iff, for some i with $d \geq i > 0$, we have

$$a_j = b_j$$

for all $j > i$ and

$$(-1)^{d-i}a_i < (-1)^{d-i}b_i,$$

where the element order $<$ in $\mathbb{Z}_p$ is given by

$$0 < 1 < \cdots < p-1$$

. The base case of the definition of Conway polynomials is $C_{p,1}(x) = x - \gamma$, where γ is the smallest primitive element of $\mathbb{Z}_p$ with respect to the element order. For the general case, choose $C_{p,n}$ to be the lexicographically smallest monic, irreducible, primitive polynomial of degree n such that, for every $n' < n$ satisfying $n'|n$, we have

$$C_{p,n}(x)|C_{p,n'}(x^{M_{p,n_2}/M_{p,n_1}}).$$

This definition yields the Conway polynomials.

Error

In the paper "New algorithms for generating conway polynoimals over finite fields" [1] Heath and Loehr gave two new algorithms to generate Conway polynomials. First algorithm based on the roots of lower polynomials had some error. The routine GENERATECONWAY(p,n) returns correct polynomials only for particular values of n, when all maximal divisors of n are pairwise relatively prime. when each prime factor of n divides it only once, i.e. if prime factorization of n is $n = q_1^{e_1} \dots q_s^{e_s}$, then $e_1 = e_2 \cdots = e_s = 1$.

Let us see why the algorithm fails to work for all the cases. Suppose n has two prime factors (for example n=12 with prime factors 2 and 3). Now maximal divisors of n are $d_1 = 4$ and $d_2 = 6$. Let us assume $p = 2$. In the algorithm we take $cp_1 = CP(p, d_1)$ and $cp_2 = CP(p, d_2)$. Now x_1 and x_2 are any roots of cp_1 and cp_2 respectively. In the algorithm there is one assumption that there exist some z with order r s.t.

$$z^{r/f_1} = x_1$$
$$z^{r/f_2} = x_2$$

where $r = p^n - 1$, $f_1 = p^{d_1} - 1$, $f_2 = p^{d_2} - 1$
Now, this assumption should hold only when x_1 and x_2 are compatible with same set of lower conway polynomial roots. Because cp_1 and cp_2 are conway polynomials, it ensures that for a root x_1 of cp_1, there exist some root(s) x_2 of cp_2 s.t. x_1 and x_2 are compatible with same set of lower conway polynomial roots. But it does not ensure that if we choose any roots x_1 and x_2, they will be compatible with same set of lower conway polynomial roots. In the algorithm we choose any roots, so the resulting $CP(p, n)$ is not compatible with cp_1 and cp_2 in some cases. When d_1 and d_2 are relatively prime, the algorithm works fine, because there is no common subfield of the two fields $GF(p^{d_1})$ and $GF(p^{d_2})$, except the base field $GF(p)$. In this particular example $f_1 = 15$, $f_2 = 63$; $\gcd\{f_1, f_2\} = 3$, now we should choose x_1 and x_2 s.t. there exists an element x_0 with order 3, s.t.

$$x_1^{f1/3} = x_0$$

$$x_2^{f2/3} = x_0$$

Then, the algorithm should work fine. The corrected algorithm is given below.

Corrected Algorithm

GENERATECONWAY(p, n)

let $n = q_1^{e_1} \dots q_s^{e_s}$ be the prime factorization of n
$r \leftarrow M_{p,n}$
for $i \leftarrow 1$ to s **do**
    $d_i \leftarrow n/q_i$
    $f_i \leftarrow M_{p,d_i}$
    $m_i \leftarrow r/f_i$
    $x_i \leftarrow$ a root of C_{p,d_i}
end for
$y \leftarrow x_1$
$v \leftarrow f_1$
for $i \leftarrow 2$ to s **do**
    find α, β such that $\alpha v + \beta f_i = \gcd\{v, f_i\}$
    $f \leftarrow \gcd\{v, f_i\}$
    for $k = 0$ to $d[i] - 1$ **do**
        if ${x_i}^{f_i/f} = y^{v/f}$ **then**
            break
        end if
        ${x_i} \leftarrow {x_i}^p$
    end for
    $y \leftarrow y^{\beta} x_i^{\alpha}$
    $v \leftarrow \mathrm{lcm}\{v, f_i\}$
end for
$g \leftarrow r/v$
$min\_poly \leftarrow \infty$
for z a g^{th} root of x in $GF(p^n)$ **do**
    poly $\leftarrow$ minimum polynomial of z

27: **if** poly has degree n and poly is primitive and poly $<$ min_poly **then**
28: min_poly $\leftarrow$ poly
29: **end if**
30: **end for**
31: **return min_poly**

Suppose at the starting of the iteration of the for loop(line 11), when $i = j$, $v = v_j$, $y = y_j$, $\alpha = \alpha_j$, $\beta = \beta_j$. And at the end of all iterations their values are v_{s+1}, y_{s+1}, α_{s+1}, β_{s+1} respectively. It is clear to see that $v_{s+1} = \text{lcm}\{f_1, f_2, \dots f_s\}$

The only difference from the original algorithm given in the paper [1] is the code added from line 13 to line 19. Instead of choosing any roots of conway polynomials, appropriate roots are chosen such that they are compatible with same set of lower conway polynomial roots.
After this modification, this part of algorithm i.e. to find a particular combination of roots will take $O(sum\ of\ all\ the\ maximal\ divisors\ of\ n)$ in the worst case. Which is still not comparable to the time taken in the last part of algorithm which is to find lexicoagraphically minimum polynomial amongst all possiblities. So, this modification does not affect the time complexity significantly. The proof of correctness is given below.
Let x_j stands for the appropriate root of CP_j, which we get after the loop ending at line 19.

Theorem 1. ${y_{s+1}}^{v_{s+1}/f_j} = x_j\ \forall j \in \{1, 2 \dots s\}$

Proof. We know that, $y_{j+1} = {y_j}^{\beta_j} {x_j}^{\alpha_j}$ and $v_{j+1} = \text{lcm}\{v_j, f_j\}\ \forall j \in \{1, 2 \dots s\}$ So for any $k \leq s+1$ and $j < k$,

$$\begin{aligned} {y_k}^{v_k/f_j} &= \left({y_{k-1}}^{\beta_{k-1}} {x_{k-1}}^{\alpha_{k-1}}\right)^{\frac{\text{lcm}\{v_{k-1}, f_{k-1}\}}{f_j}} \\ &= \left({y_{k-1}}^{\beta_{k-1}} {x_{k-1}}^{\alpha_{k-1}}\right)^{\frac{f_{k-1}}{\gcd\{v_{k-1}, f_{k-1}\}} \frac{v_{k-1}}{f_j}} \\ &= \left(y_{k-1}^{\frac{\beta_{k-1} f_{k-1}}{\gcd\{v_{k-1}, f_{k-1}\}}} x_{k-1}^{\frac{\alpha_{k-1} f_{k-1}}{\gcd\{v_{k-1}, f_{k-1}\}}}\right)^{\frac{v_{k-1}}{f_j}} \end{aligned}$$

Now, the for loop starting at line 14 ensures that ${x_j}^{\frac{f_j}{\gcd\{v_j, f_j\}}} = {y_j}^{\frac{v_j}{\gcd\{v_j, f_j\}}}\ \forall j \in \{1, 2 \dots s\}$ So,

$$\begin{aligned} {y_k}^{v_k/f_j} &= \left(y_{k-1}^{\frac{\beta_{k-1} f_{k-1}}{\gcd\{v_{k-1}, f_{k-1}\}}} y_{k-1}^{\frac{\alpha_{k-1} v_{k-1}}{\gcd\{v_{k-1}, f_{k-1}\}}}\right)^{\frac{v_{k-1}}{f_j}} \\ &= \left(y_{k-1}^{\frac{\beta_{k-1} f_{k-1} + \alpha_{k-1} v_{k-1}}{\gcd\{v_{k-1}, f_{k-1}\}}}\right)^{\frac{v_{k-1}}{f_j}} \end{aligned}$$

We know that $\beta_j f_j + \alpha_j v_j = \gcd\{v_j, f_j\}$, $\forall j \in \{1, 2 \dots s\}$, so,

$$
\begin{aligned}
{y_k}^{v_k/f_j} &= {y_{k-1}}^{\frac{v_{k-1}}{f_j}} \\
&= {y_{k-2}}^{\frac{v_{k-2}}{f_j}} \\
&\vdots \\
&= {y_{j+1}}^{\frac{v_{j+1}}{f_j}} \\
&= \left({y_j}^{\beta_j} {x_j}^{\alpha_j}\right)^{\frac{v_{j+1}}{f_j}} \\
&= \left({y_j}^{\beta_j} {x_j}^{\alpha_j}\right)^{\frac{\mathrm{lcm}\{v_j,f_j\}}{f_j}} \\
&= \left({y_j}^{\beta_j} {x_j}^{\alpha_j}\right)^{\frac{v_j}{\gcd\{v_j,f_j\}}} \\
&= {y_j}^{\frac{\beta_j v_j}{\gcd\{v_j,f_j\}}} {x_j}^{\frac{\alpha_j v_j}{\gcd\{v_j,f_j\}}} \\
&= {x_j}^{\frac{\beta_j f_j}{\gcd\{v_j,f_j\}}} {x_j}^{\frac{\alpha_j v_j}{\gcd\{v_j,f_j\}}} \\
&= {x_j}^{\frac{\beta_j f_j + \alpha_j v_j}{\gcd\{v_j,f_j\}}} \\
&= x_j
\end{aligned}
$$

We showed that

$$
{y_k}^{v_k/f_j} = x_j \ \forall j < k \tag{1}
$$

So, ${y_{s+1}}^{v_{s+1}/f_j} = x_j \ \forall j \in \{1, 2 \dots s\}$. □

We can see that CP_j is a conway polynomial $\forall j \in \{1, 2 \dots s\}$, it ensures that for any root x_j of CP_j, there exist a root x_k of CP_k s.t.

$$
{x_j}^{\frac{f_j}{\gcd\{f_j,f_k\}}} = {x_k}^{\frac{f_k}{\gcd\{f_j,f_k\}}} \tag{2}
$$

because $\gcd\{f_j, f_k\}$ is the order of the multiplicative group of the largest common subfield of the two fields $GF(p^j)$ and $GF(p^k)$.

We know that the roots of CP_i can be given by $x_i, x_i^p, x_i^{p^2}, \dots x_i^{p^{d_i-1}}$. So, it is clear to see that in the for loop starting from line 14, it goes over all the roots of CP_i, and checks that if there exist a root x_i of CP_i s.t.

$$
{x_i}^{\frac{f_i}{\gcd\{v_i,f_i\}}} = {y_i}^{\frac{v_i}{\gcd\{v_i,f_i\}}}
$$

Now, it is remained to show that there always exists such a root. Consider the following thoerem.

Theorem 2. *There exist a root x_i of CP_i s.t. ${x_i}^{\frac{f_i}{\gcd\{v_i,f_i\}}} = {y_i}^{\frac{v_i}{\gcd\{v_i,f_i\}}}$, $\forall i \leq s$*

Proof. We prove the above theorem by induction in i. We know that $y_2 = x_1$ and $v_2 = f_1$, so, it is clear from equation(2) that the theorem is true for $i = 2$.
Now, assume the theorem to be true for all $i < j$, i.e., there exist a root x_i of CP_i s.t.

$$x_i^{\frac{f_i}{\gcd\{v_i,f_i\}}} = y_i^{\frac{v_i}{\gcd\{v_i,f_i\}}} \ \forall i < j$$

From equation (1),

$$y_i^{v_i/f_k} = x_k \ \forall k < i$$

So, for all $k < i$

$$\begin{aligned} x_k^{\frac{f_k}{\gcd\{f_i,f_k\}}} &= y_i^{\frac{v_i}{\gcd\{f_i,f_k\}}} \\ &\overset{(a)}{=} \left(y_i^{\frac{v_i}{\gcd\{v_i,f_i\}}}\right)^{\frac{\gcd\{v_i,f_i\}}{\gcd\{f_i,f_k\}}} \\ &= \left(x_i^{\frac{f_i}{\gcd\{v_i,f_i\}}}\right)^{\frac{\gcd\{v_i,f_i\}}{\gcd\{f_i,f_k\}}} \\ &= x_i^{\frac{f_i}{\gcd\{f_i,f_k\}}} \end{aligned}$$

which is true for all $i < j$. (a) is true because $\frac{\gcd\{v_i,f_i\}}{\gcd\{f_i,f_k\}}$ is an integer.
In other words we can say

$$x_{k_1}^{\frac{f_{k_1}}{\gcd\{f_{k_1},f_{k_2}\}}} = x_{k_2}^{\frac{f_{k_2}}{\gcd\{f_{k_1},f_{k_2}\}}} \ \forall k_1, k_2 < j$$

So, we can say that there exists a root x_j of CP_j compatible to $x_k \ \forall k < j$, i.e.,

$$x_j^{\frac{f_j}{\gcd\{f_j,f_k\}}} = x_k^{\frac{f_k}{\gcd\{f_j,f_k\}}} \ \forall k < j$$

because they are compatible roots of conway polynomials.

Claim 1. *For that x_j,*

$$x_j^{\frac{f_j}{\gcd\{f_j,v_k\}}} = y_k^{\frac{v_k}{\gcd\{f_j,v_k\}}} \ \forall k \leq j$$

Proof. We prove claim(1) by induction in k. We know that $y_2 = x_1$ and $v_2 = f_1$, so claim(1) is true for $k = 2$. Suppose claim(1) is true for some $k \leq j - 1$. Then consider,

$$\begin{aligned} y_{k+1}^{\frac{v_{k+1}}{\gcd\{f_j,v_{k+1}\}}} &= \left(y_k^{\beta_k} x_k^{\alpha_k}\right)^{\frac{v_{k+1}}{\gcd\{f_j,v_{k+1}\}}} \\ &= y_k^{\frac{\beta_k v_{k+1}}{\gcd\{f_j,v_{k+1}\}}} x_k^{\frac{\alpha_k v_{k+1}}{\gcd\{f_j,v_{k+1}\}}} \\ &= y_k^{\left(\frac{v_k}{\gcd\{f_j,v_k\}} \frac{\beta_k v_{k+1} \gcd\{f_j,v_k\}}{v_k \gcd\{f_j,v_{k+1}\}}\right)} x_k^{\left(\frac{f_k}{\gcd\{f_j,f_k\}} \frac{\alpha_k v_{k+1} \gcd\{f_j,f_k\}}{f_k \gcd\{f_j,v_{k+1}\}}\right)} \end{aligned}$$

It can be easily verified that $\frac{\beta_k v_{k+1} \gcd\{f_j,v_k\}}{v_k \gcd\{f_j,v_{k+1}\}}$ and $\frac{\alpha_k v_{k+1} \gcd\{f_j,f_k\}}{f_k \gcd\{f_j,v_{k+1}\}}$ are integers. Now, we assumed claim(1) to be true for k. So,

$$\begin{aligned} y_{k+1}^{\frac{v_{k+1}}{\gcd\{f_j,v_{k+1}\}}} &= x_j^{\left(\frac{f_j}{\gcd\{f_j,v_k\}} \frac{\beta_k v_{k+1} \gcd\{f_j,v_k\}}{v_k \gcd\{f_j,v_{k+1}\}}\right)} x_j^{\left(\frac{f_j}{\gcd\{f_j,f_k\}} \frac{\alpha_k v_{k+1} \gcd\{f_j,f_k\}}{f_k \gcd\{f_j,v_{k+1}\}}\right)} \\ &= x_j^{\frac{f_j}{\gcd\{f_j,v_{k+1}\}}\left(\frac{\beta_k v_{k+1}}{v_k}+\frac{\alpha_k v_{k+1}}{f_k}\right)} \\ &= x_j^{\frac{f_j}{\gcd\{f_j,v_{k+1}\}}} \end{aligned}$$

So, claim(1) is also true for $k+1$. Thus, claim(1) is true for all $k \leq j$.

□

Hence, we showed that there exist a root x_j of CP_j s.t.,

$$x_j^{\frac{f_j}{\gcd\{v_j,f_j\}}} = y_j^{\frac{v_j}{\gcd\{v_j,f_j\}}}$$

i.e., the theorem is true for $i = j$. Hence the theorem is true for all $i \leq s$. □

Now, we know that $v_j = \text{lcm}\{v_{j-1}, f_{j-1}\}$. So, we can see that y_j is an element of order $\text{lcm}\{f_1, f_2, \ldots f_{j-1}\}$, and is compatible with x_k $\forall k < j$ i.e.,

$$y_j^{v_j/f_k} = x_k$$

Let us say y_{s+1} to be y. It is clear to see that, y is an element of order $v_{s+1} = \text{lcm}\{f_1, f_2, \ldots f_s\}$. And y is compatible with roots of lower conway polynomials. Let us say $g = Mp, n/v_{s+1}$, then all g^{th} roots of y that are primitive elements of $GF(p^n)$ are candidates of being the roots of the Conway Polynomial $C_{p,n}$. In the last part of the algorithm lexicographically minimum polynomial is being found among all g possiblities.

Implementation in GAP and Results

The important fuctions in the implemetation are as follows:

- irreducible(p, n)
  Takes a prime p and an integer n as arguments, and returns an irreducible polynomial over $\mathbb{F}_p$ with degree n. It goes over polynomials over $\mathbb{F}_p$ with degree n, and checks if it is irreducible, and returns the first irreducible polynomial it finds.

- gthRoot$(element, g, ff, primeField)$
  Takes a field ff, an integer g, an element of the field *element* and the prime subfield of ff, *primeField*. It returns g^{th} root of *element* in the field ff and g^{th} primitive root of unity in the field ff, so that all the g^{th} roots of the *element* can be generated. For computing g^{th} root, the algorithm given in the section 7.3 of the book [2] is used.

- comparePoly($poly1, poly2$)
  Takes two polynomials $poly1$ and $poly2$, and returns true if $poly1$ lexicographically smaller than $poly2$, false otherwise.

- conwayPolynomial(p, n)
  Takes a prime p and an integer n as arguments, and returns $C_{p,n}$. In the case when $n = 1$, it uses the algorithm $ConwayPolynomial(p, 1)$ inbuilt in GAP. In other case the algorithm given in this report is implemented. First it generates an irreducible polynomial over $\mathbb{F}_p$ with degree n, to represent the field $GF(p^n)$, because the algorithm involves some field operations in $GF(p^n)$. For computing lower conway polynomials recursively the same function is used. And for computing the roots of the lower conway polynomials an inbuilt fucntion $RootsOfUPol(polynomial)$ is used. Rest is done as in the algorithm given in this report.

Other Compatible Polynomials

We can see that major time expense incurred by the algorithm occurs when it checks all g of the g^{th} roots of y to find the primitive root whose minimal polynoimal is lexicographically smallest. This algorithm is much better than the brute force algorithm except some cases. Still we are able generate only that many polynomials which are already present in the inbuilt list of Conway Polynomials in GAP. It looks that GAP uses a similar algorithm which takes $O(g)$ time.

As given in the section Alternative directions of [1], because so many of the g^{th} roots of y are primitive, we can find a primitive root with a compatible minimal polynomial very quickly, by stopping at the first primitive root we find. The polynomimal so obtained is not, in general, the Conway polynomial. However, it does have all the desirable algebraic properties of the Conway polynomial, namely primitivity and compatibility with previously chosen polynomials. Hence, for each p, one can quickly generate a large set of compatible polynomials to represent fields of characterstic p.

Indeed one can define a new set of polynoimals via the modified version of the algorithm. The only difficulty in postulating such a definition is that certain portions of the algorithm-speicifically, finding an irreducible polynomial and taking roots in finite fields-involve randomized algorithms; hence the algorithm would produce different polynoimals each time it is executed. To obtain one statndard set of polynomials, it is necessary to remove all the randomness from the algorithm used to define these polynomials. In the function irreducible(p, n) polynomials are picked in an order rather than randomly. Also in the function gthRoot($element, g, ff, primeField$) randomness is removed. But as soon as we remove the randomness we cannot claim that the time taken by these functions is less. Practically it works but there might be cases where they take long time.

This modified algorithm is also implemented and tested. The implementation code is in the file named "compatiblePolynomial". The code is same except that the conwayPolynomial function is replaced with compatiblePolynomial. Which does not go over all the g^{th} roots, instead stops as soon as it finds first primitive root. The list generated of these new set of polynomials, is much larger than the inbuilt list of conway polynomials. Which are equally good for representing finite

Table 1: values of n upto which compatible polynomials were computed corresponding to p

p	n
2	159
3	100
5	100
7	100
11	100
13	100
17	100
23	100
29	100

fields. The table below gives the values of n upto which compatible polynomials were computed corresponding to each $p \leq 29$.

Future Work

As written above, after removal of randomness we cannot say that those routines will take less time. So, one of the work can be to prove that time taken by those routines is still less even after removal of randomness. Or to give other algorithms for finding an irreducible polynomial and for finding roots in finite field which is not randomized and also do not take much time.

In the modified algorithm, major time taking part is to compute roots of the lower compatible polynomials. Which currently uses the inbuilt function $RootsOfUPol(polynomial)$. I believe this time can be reduced, because we know that these polynomials are irreducible over $GF(p)$, which can give some advantage, which is not used in the inbuilt function.

References

[1] Heath, L.S., Loehr, N.A., 2004 "New algorithms for generating Conway polynomials over finite fields"

[2] Bach, E., Shallit, J., 1996. "Algorithimic Number Theory"