Deterministic equation solving over finite fields

Christiaan van de Woestijne, RICAM Linz

TU Graz
Zahlentheoretisches Kolloquium
25 November 2005

The surrounding landscape (1)

We consider polynomial equations in many variables over finite fields. These may arise as reductions of Diophantine equations modulo a prime, or studied for their own sake.

One may be interested in:

- solvability
- number of solutions
- obtaining one, several or all solutions

We will consider algorithms for finding solutions. (Using Hensel lifting, these are easily extended to algorithms for solving equations over local fields.)

The surrounding landscape (2)

Currently known algorithms for solving equations over finite fields include:

- brute force search
- algorithms for factoring polynomials
- Shanks' algorithm for taking square (and higher) roots
- Schoof's algorithm for taking square roots in prime fields
- methods for multivariate equations based on the above

However, all of these are either probabilistic (barring a proof of GRH for some) or take more than polynomial time.

Part I

Probabilistic methods

The Tonelli-Shanks algorithm

Best-known formulation: given a nonzero $a \in \mathbb{F}$,

1. find a nonsquare s in $\mathbb{F}$ by guessing.
2. use this s to compute a square root of a, essentially computing a discrete logarithm in the 2-Sylow subgroup of $\mathbb{F}^*$.

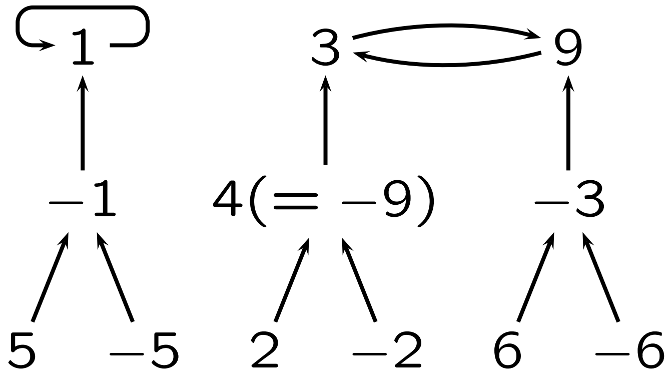
NB 1. The only probabilistic part is in Step 1.

NB 2. The algorithm works equally well with ℓth roots for any prime number ℓ (we have to guess a non-ℓth-power).

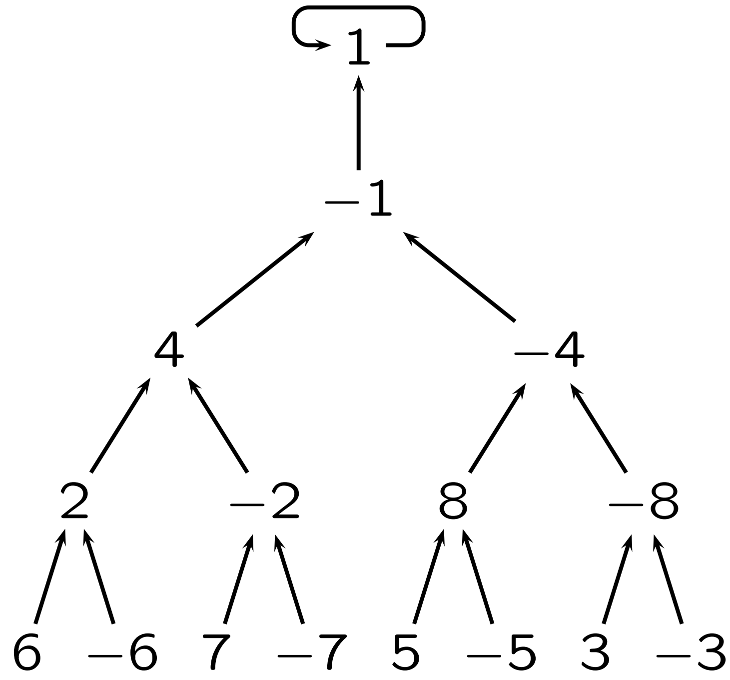
NB 3. This algorithm uses only group-theoretic properties of the group $\mathbb{F}^*$, so it works equally well in arbitrary finite cyclic groups.

Squaring in $\mathbb{F}_{13}^*$ and $\mathbb{F}_{17}^*$

$13 - 1 = 2^2 \cdot 3$:



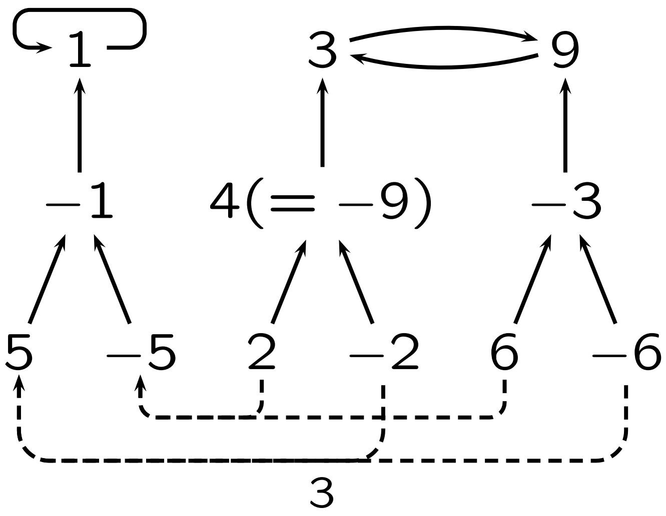
$17 - 1 = 2^4$:



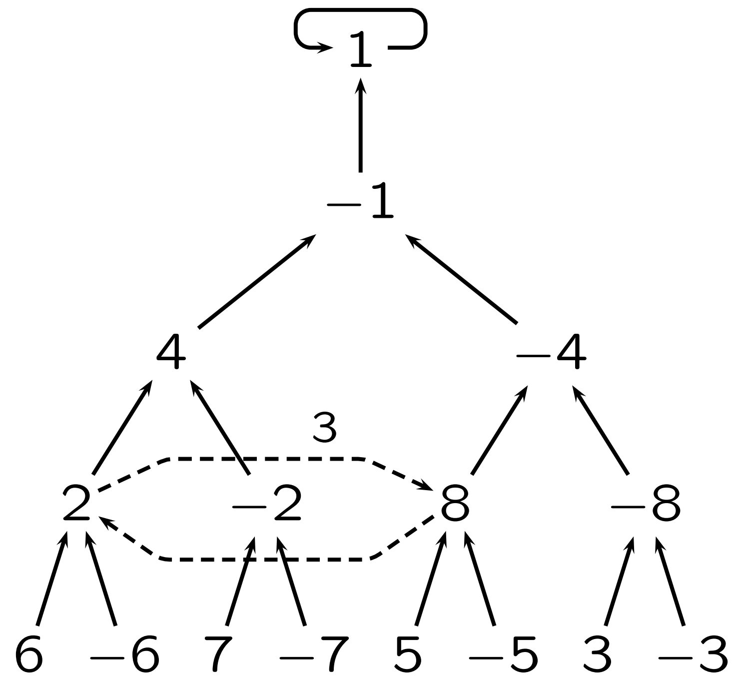
The level of an element in the tree (where the root has level 0) is equal to the number of factors 2 in its order!

Cubing in $\mathbb{F}_{13}^*$ and $\mathbb{F}_{17}^*$

$13-1=2^2\cdot 3$:



$17-1=2^4$:



Cubing an element preserves the level, but takes you to another tree (if there are more) or another node of the tree with root 1.

Where are these non-squares?!

In a field of q elements, where q is an odd prime power, there are $(q-1)/2$ squares and as many non-squares.

- The (non-)squares are almost uniformly distributed (but not quite)

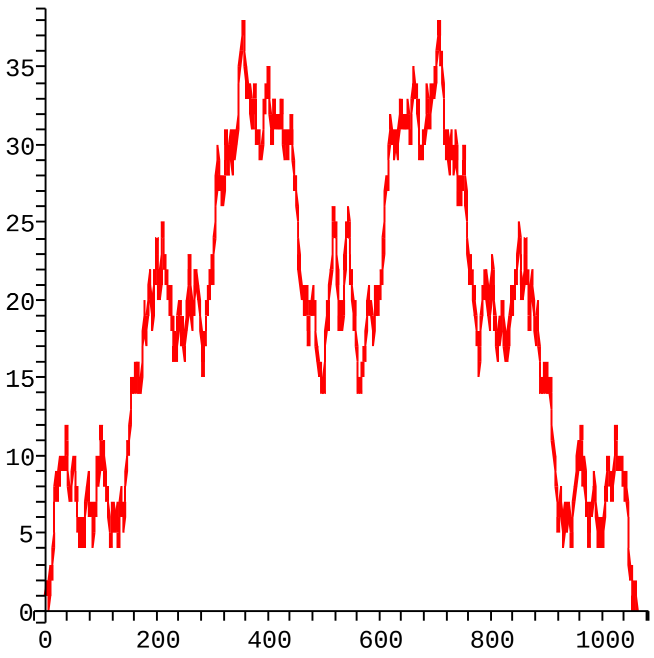
- The smallest non-square is $O(q^{\frac{1}{4\mathrm{e}}})$ (Burgess 1957)

- (Assuming GRH:) the smallest non-square is $\leq 2(\log q)^2$ (Ankeny 1952, Bach 1990)

Similar results hold for all nth powers where n is not too large compared to q. So: no guaranteed efficient deterministic algorithm to find a non-square!

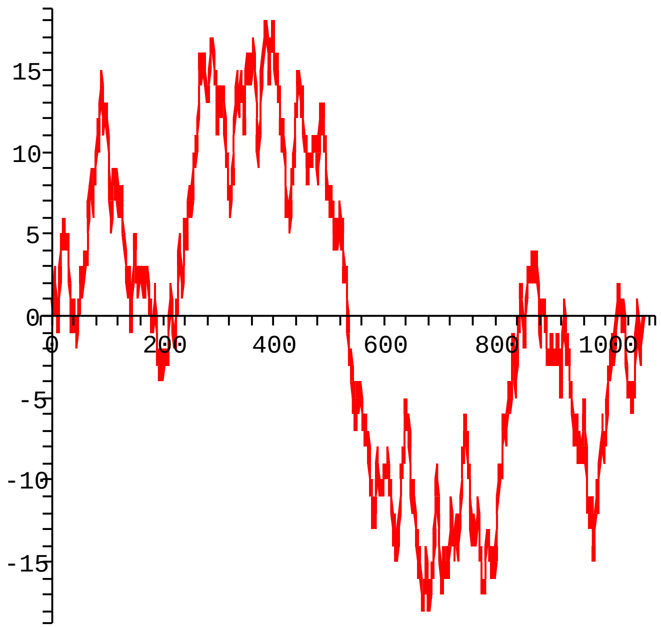
The distribution of the squares modulo

1063:



primes 3 modulo 4

1069:



primes 1 modulo 4

Briefly, the Cantor-Zassenhaus algorithm

Let f be a squarefree polynomial with coefficients in $\mathbb{F}$. We have

$$\mathbb{F}[X]/(f) \cong \mathbb{F}[X]/(f_1) \times \ldots \times \mathbb{F}[X]/(f_r)$$

if $f_1, \ldots, f_r$ are the irreducible factors of f, all of degree 1.

For any polynomial g in $\mathbb{F}[X]$, coprime to f, we have

$$g^{(q-1)/2} \equiv \{1, -1\} \pmod{f_i} \quad \text{for } i = 1, \ldots, r.$$

Now hope that the values are not the same modulo all f_i; then

$$g^{(q-1)/2} - 1$$

is divisible by some of the f_i but not by all.

Still, the Cantor-Zassenhaus algorithm

So, what we want is a polynomial g that is a square modulo some of the f_i, and a nonsquare modulo some others.

If we are unlucky, we try another g, or we redo the computation with f replaced by $f(x+c)$ for some $c \in \mathbb{F}^*$.

Several other variants, but no way to construct a g or a c that is guaranteed to work! Not even on assumption of GRH...

Only if q is a power of a small prime p does there exist an efficient deterministic method (Berlekamp's method)...

Multivariate polynomials

In other words: find a rational point on a hypersurface.

Idea: given $f \in \mathbb{F}[X_1, \ldots, X_n]$, substitute random values $x_1, \ldots, x_{n-1}$ for $X_1, \ldots, X_{n-1}$, and examine if the univariate polynomial

$$f(x_1, \ldots, x_{n-1}, X_n)$$

has a zero in X_n.

Again: no guarantee that the resulting univariate polynomial has a zero! We might have to try several (or many) tuples $(x_1, \ldots, x_{n-1})$.

Part II:

Deterministic methods

Some conventions

From now on in this talk, the phrase "we can compute X" means:

"we know explicitly a deterministic polynomial time algorithm to compute X".

The same goes for "we can decide Y".

We assume that a finite field $\mathbb{F}$ of q elements and characteristic p is given by a polynomial f that is irreducible over the prime field $\mathbb{F}_p$.

Our algorithms take $\mathbb{F}$ as input; thus the input size is about $\log q$, and our algorithms must finish in time polynomial in $\log q$.

Group theory

An important building block of my deterministic algorithms is the following adaptation of the Tonelli-Shanks root taking algorithm.

Theorem. If $a_0, a_1, \ldots, a_n$ are in $\mathbb{F}^*$, then we can compute some $\beta \in \mathbb{F}^*$ such that, for some i,j with $0 \le i < j \le n$, we have

$$a_i/a_j = \beta^n.$$

Proof. Let $H = \langle a_0, \ldots, a_n \rangle$. The a_i cover the cosets of H modulo H^n, so there exist i and j such that $a_i/a_j \in H^n$.

We can factor n into primes ℓ and use this to compute generators γ_ℓ for the ℓ-parts of H. Now, we compute an nth root β of a_i/a_j using these generators γ_ℓ, by means of the Tonelli-Shanks algorithm. $\square$

Main theorem

(This is part of my PhD project with H. W. Lenstra, Jr.)

My main theorem:

Given a finite field $\mathbb{F}$, a positive integer n and nonzero $a_0, \ldots, a_n \in \mathbb{F}$, we can compute a nontrivial solution to the equation

$$a_0 x_0^n + a_1 x_1^n + \ldots + a_n x_n^n = 0.$$

Furthermore, if possible, my algorithm will return a solution with $x_0 \neq 0$.

In other words, whenever the equation

$$a_1 x_1^n + \ldots + a_n x_n^n = b$$

has solutions for a given nonzero b, we can compute one.

Applications (for $n = 2$)

If $n = 2$ and the characteristic of $\mathbb{F}$ is odd, then every form is diagonal. Furthermore, in characteristic 2, zeros of quadratic forms can be found by means of linear algebra.

Corollary. *Given a quadric hypersurface over a finite field $\mathbb{F}$, we can compute a rational point on it.*

Corollary. *Given two regular quadratic spaces V and W over a finite field $\mathbb{F}$ (char. $\neq 2$), such that $\dim V \geq \dim W + 1$, we can compute an isometric embedding of W into V.*
On the other hand, if $\dim V = \dim W$, we can reduce the problem of finding an isometry from V to W to the computation of just one square root in $\mathbb{F}$.

More applications (for $n = 2$)

Corollary. *(Bumby) Given a prime p, we can compute integers x, y, z, w such that $p = x^2 + y^2 + z^2 + w^2$.*

This works also for any other quaternion orders of class number 1.

Corollary. *Given a central simple algebra A of degree 2 over a finite field $\mathbb{F}$, we can compute an explicit isomorphism from A to a 2×2-matrix algebra over $\mathbb{F}$.*

and one I found recently (using an identity of M. Skałba):

Corollary. *Given an elliptic curve E by a nonsingular Weierstraß equation over a finite field $\mathbb{F}$, we can compute as many rational points on E as we want.*

The main steps

I. Generating $\mathbb{F}$ over its prime field by an nth power:
find $\alpha \in \mathbb{F}$ **such that** $\mathbb{F} = \mathbb{F}_p(\alpha^n)$**.**

II. Writing field elements as sums of like powers:
given $b \in \mathbb{F}^*$**, find** $x_1, \ldots, x_n \in \mathbb{F}$ **such that** $b = \sum_{i=1}^{n} x_i^n$**.**

III. Finding the desired representation

$$a_1 x_1^n + \ldots + a_n x_n^n = b$$

by an algorithmic adaptation of ideas of Dem′yanov and Kneser.

It can be shown that...

- the set of sums of nth powers of elements, S_n, in $\mathbb{F}$ is a subfield of $\mathbb{F}$.
- $S_n = \mathbb{F}$ iff $\mathbb{F}$ can be generated over $\mathbb{F}_p$ by an nth power in $\mathbb{F}$.
- if $S_n \neq \mathbb{F}$, we have $n^2 > q$.
- if $S_n = \mathbb{F}$, then every equation of the form

  $$\sum_{i=1}^{n} a_i x_i^n = b$$

  for $a_1, \ldots, a_n$ and b in $\mathbb{F}^*$ is solvable.

The homogeneous variant $\sum_{i=0}^{n} a_i x_i^n = 0$ is always solvable by the Chevalley-Warning theorem.

By comparison...

- the results from the last slide can be much improved if q is much larger than n^2. For example, if $q > n^4$, then every equation of the form

$$ax^n + by^n = c$$

  is solvable (Weil 1948).

- the algorithms I will present are not unpractical but probabilistic algorithms will probably do better if q is much larger than n.

Overview: building blocks

I. A multiplicative version of the primitive element theorem (really elementary linear algebra)

II. Reducing the number of terms in a sum of like powers (a bisection-like idea)

III. Selective root extraction (a generalisation of the Tonelli-Shanks algorithm)

IV. Dealing with coefficients other than 1 by means of the "trapezium algorithm" (an algorithmic version of an idea of Dem′yanov and Kneser)

Algorithm I: a generator in a given subgroup (1)

Theorem. Let $G \subseteq \mathbb{F}^*$ be a multiplicative subgroup; we can compute $\beta \in G$ such that β generates $\mathbb{F}$ over its prime field, or decide that no such α exists.

Main (in fact only) example: $G = \mathbb{F}^{*n}$ for some positive integer n.

Proof. Let $n = [\mathbb{F}^* : G]$ and let α be the given generator of $\mathbb{F}$.

If $K_1 = \mathbb{F}_p(\gamma_1^n)$ and $K_2 = \mathbb{F}_p(\gamma_2^n)$ are subfields of $\mathbb{F}$, we can compute $\gamma \in \langle \gamma_1, \gamma_2 \rangle$ such that

$$\gamma^n \textbf{ generates } \mathbb{F}_p(\gamma_1^n, \gamma_2^n) \textbf{ over } \mathbb{F}_p$$

by means of elementary linear algebra.

Building block I: A "multiplicative" primitive element theorem

Lemma. Let L/K be a cyclic extension of fields of degree d, and let $b_1, \ldots, b_d$ be a K-basis for L. Then at least $\varphi(d)$ of the b_i generate L as a field over K.

Now suppose $\alpha \in L$ has degree e over K and β has degree f. The degree of β over $K(\alpha)$ is given by $g = \mathrm{lcm}(e, f)/e = f/\gcd(e, f)$, so a basis of $K(\alpha, \beta)$ is given by

$$(\alpha^i \beta^j \mid i = 0, \ldots, e-1, j = 0, \ldots, g-1).$$

By the Lemma, one of these elements generates $K(\alpha, \beta)$ over K!

Obviously, by induction we may extend this result to systems of more than two generators.

Algorithm I: a generator in a given subgroup (2)

Proof (ctd.) We start induction with $K = \mathbb{F}_p = \mathbb{F}_p(1^n)$. Assume now we have $K = \mathbb{F}_p(\gamma_1^n)$. If $|K| \leq n$, we find $\gamma_2 \in \mathbb{F}^*$ with $\gamma_2^n \notin K$.

If no such γ_2 exists, the algorithm fails (and rightly so)!

If $|K| > n$, then at least one of $(\alpha + c_i)^n$, where $c_0, \ldots, c_n$ are distinct elements of K, is not in K; now put $\gamma_2 = \alpha + c_i$. (Recall that $\mathbb{F} = \mathbb{F}_p(\alpha)$.)

Now in either case, adjoin γ_2^n to K and compute γ with $K = \mathbb{F}_p(\gamma^n)$, using Building block I. $\square$

Algorithm II: sums of like powers

Theorem. Let b be in $\mathbb{F}^*$ and n a positive integer. We can decide if b is in S_n and if so, we can compute $x_1, \ldots, x_n$ such that $b = \sum_{i=1}^{n} x_i^n$.

Proof. If $n^2 \geq q$, we have enough time to enumerate all possibilities.

If $n^2 < q$, then $S_n = \mathbb{F}$, so the answer is yes. We use Algorithm I to compute $\gamma \in \mathbb{F}$ such that γ^n generates $\mathbb{F}$ over $\mathbb{F}_p$; this gives us

$$b = \sum_{i=0}^{[\mathbb{F}:\mathbb{F}_p]-1} b_i \gamma^{ni}.$$

This is a sum of nth powers with at most $(p-1) \cdot [\mathbb{F} : \mathbb{F}_p]$ terms!

Now use Building blocks II and III to come down to just n terms. $\square$

Building block II: reducing sums of like powers

Theorem. Given $y_1, \ldots, y_N$ and $b \in \mathbb{F}^*$ with $\sum y_i^n = b$, we can compute $x_1, \ldots, x_n \in \mathbb{F}^*$ such that $\sum_{i=1}^n x_i^n = b$.

Proof. Divide $y_1, \ldots, y_N$ into $n+1$ roughly equal groups $G_0, \ldots, G_n$. Let S_i denote the sum of all terms in the first $i+1$ groups.

If one of the S_i is zero, we discard all terms in the first $i+1$ groups. Otherwise, we use selective root extraction to compute $\beta \in \mathbb{F}^*$ with

$$S_i/S_j = \beta^n.$$

(assume $i > j$). This means we can discard the groups G_{j+1} up to G_i, provided we multiply all terms in the first $i+1$ groups by β. This trick is applicable as long as we have at least $n+1$ terms. $\square$

Building block III: selective root extraction

Theorem. If $a_0, a_1, \ldots, a_n$ are in $\mathbb{F}^*$, then we can compute some $\beta \in \mathbb{F}^*$ such that, for some i,j with $0 \leq i < j \leq n$, we have

$$a_i/a_j = \beta^n.$$

Proof. Let $H = \langle a_0, \ldots, a_n \rangle$. The a_i cover the cosets of H modulo H^n, so there exist i and j such that $a_i/a_j \in H^n$.

We can factor n into primes ℓ and use this to compute generators γ_ℓ for the ℓ-parts of H. Now, we compute an nth root β of a_i/a_j using these generators γ_ℓ, by means of the Tonelli-Shanks algorithm.

$\square$

Algorithm III: representations by diagonal forms

Theorem. Let b be in $\mathbb{F}^*$ and n a positive integer. For any $a_1, \ldots, a_n \in \mathbb{F}^*$ we can decide if the equation

$$b = \sum_{i=1}^{n} a_i x_i^n$$

is solvable, and if so, we can compute a solution.

Proof. Again, if $n^2 \geq q$, we can just enumerate all possibilities.

If $n^2 < q$, there is a solution. Write $a_0 = -b$. We use now Algorithm II to write the elements b/a_i (for $i = 1, \ldots, n$) as sums of nth powers, so we get

$$-a_i \sum_j y_{ij}^n = -b = a_0 \cdot 1^n.$$

Building block IV: the trapezium algorithm (1)

We now have a system of the form

$$\begin{cases} -a_0(y^n_{0,1} + \ldots + y^n_{0,h_0}) = 0 \\ -a_1(y^n_{1,1} + \ldots + y^n_{1,h_1}) = a_0 x^n_{1,0} \\ \qquad \vdots \qquad\qquad\qquad\qquad \vdots \\ -a_n(y^n_{n,1} + \ldots + y^n_{n,h_n}) = a_0 x^n_{n,0} + \ldots + a_{n-1} x^n_{n,n-1} \end{cases}$$

Recall that we wrote $a_0 = -b$. If $h_i = 1$ for some $i \geq 1$, we are done!

We try to lower the h_i by bringing the last term $a_i y^n_{i,h_i}$ to the other side. We get the sequence

$$\left(a_0 y^n_{0,h_0}\,,\; a_0 x^n_{1,0} + a_1 y^n_{1,h_1}\,,\; \ldots, a_0 x^n_{n,0} + \ldots + a_{n-1} x^n_{n,n-1} + a_n y^n_{n,h_n} \right).$$

Building block IV: the trapezium algorithm (2)

The sequence

$$\left(a_0 y_{0,h_0}, a_0 x_{1,0}^n + a_1 y_{1,h_1}^n, \ldots, a_0 x_{n,0}^n + \ldots + a_{n-1} x_{n,n-1}^n + a_n y_{n,h_n}^n\right).$$

has $n+1$ elements, say $c_0, \ldots, c_n$. If one is zero, we are done!

Otherwise, use selective root extraction to compute $\beta \in \mathbb{F}^*$ with

$$\beta^n = c_i / c_j, \quad \text{i.e.} \quad c_i = \beta^n c_j$$

(assume $i > j$).

Replace now the ith term in the sequence by β^n times the jth term, and we can reduce h_i by one!

Thus, in at most n^2 steps, we will get one of the h_i down to zero. $\square$

The End

(The latest version of my thesis is available from my homepage: `http://www.math.leidenuniv.nl/~cvdwoest`.)