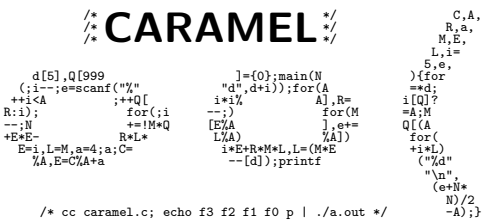
# Algorithmes de calcul de logarithme discret dans les corps finis

## Grenoble, École de printemps C2 2014

E. Thomé






Mar. 20th-21st, 2014

# Part 3

## Lecture 3/3: Recent algorithmic developments

The descent step

New algorithm setting

New tools

Complexity

Conclusion

# Plan

# Problem statement

How do we compute individual logarithms ?

- Relation collection + linear algebra have provided us the virtual logarithms of all factor base primes (and prime ideals).
- Let $h$ represent a target element in $\mathbb{F}_p$. How do we find a special pair $(a, b)$ such that:

$$a - bm = h \times \{\text{product of factor base primes}\};$$
$$a - b\alpha = \{\text{product factor base prime ideals}\};$$

  If we can do this, we complete the computation.
- Alternative: be less ambitious.
  - Do not aim at factor base bound at first. It would be too much to ask. Accept intermediary-sized $p$ and $\mathfrak{p}$.
  - Recurse.

# The descent

Assumption: we have precomputed log's of factor base elements.

The descent is the art of turning the DLP computation into a recursive problem.

## One descent step

Input: a challenge discrete logarithm to compute for an element of some given size.
Output: an identity which allows to derive the desired log from log's of smaller elements.

Historical notes:

- Coppersmith84 was the first algorithm featuring a descent.
- Many of the NFS-DL/FFS papers from 1993 to 2006 introduced new (not always correct) variations for the descent.

# Example

Start with $h =$  (some integer).

- Start with a relation involving $h$:

$$a - bm = \text{  } \times \text{  } \times \cdots \times \text{  } \times \text{  },$$

$$(a - b\alpha) = \text{  } \times \cdots \times \text{  } \times \text{  }.$$

- Then find relations killing all the outstanding terms:

# Example

Start with $h =$  (some integer).

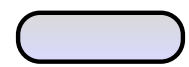
- Start with a relation involving $h$:

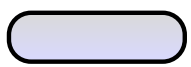
$$a - bm = \text{  } \times \text{  } \times \cdots \times \text{  } \times \text{  },$$

$$(a - b\alpha) = \text{  } \times \cdots \times \text{  } \times \text{  }.$$

- Then find relations killing all the outstanding terms:

$$a - bm = \text{  } \times \text{  } \times \cdots \times \text{  }; \quad (a - b\alpha) = \text{  } \times \cdots \times \text{};$$

$$a - bm = \text{  } \times \cdots \times \text{  }; \quad (a - b\alpha) = \text{  } \times \text{  } \times \cdots \times \text{};$$

$$a - bm = \text{  } \times \text{  } \times \cdots \times \text{  }; \quad (a - b\alpha) = \text{  } \times \cdots \times \text{};$$

$$a - bm = \text{  } \times \cdots \times \text{  }; \quad (a - b\alpha) = \text{  } \times \text{  } \times \cdots \times \text{};$$

# Tools for the descent

We may consider several "tools" for performing decent steps.

- Euclidean algorithm;
- Classical descent;
- Small characteristic: descent with Gröbner bases.
- Small characteristic: quasi-polynomial algorithm.

Present the first two in an NFS-DL context.

# Euclid algorithm (again)

There's a trivial way to rewrite $h$ as a product of smaller elements.
Use Euclid's algorithm to write

$$h \equiv u/v \mod p$$

with $u \approx v \approx \sqrt{p}$.

# Improving over the Euclid step

Improved in [JL03]

- take two successive $(u, v)$ as given by the Euclid algorithm:

$$a \equiv \frac{u_0}{v_0} \equiv \frac{u_1}{v_1} \mod p;$$

- for $\lambda, \mu$ not too large, remark that:

$$a \equiv \frac{\lambda u_0 + \mu u_1}{\lambda v_0 + \mu v_1} \mod p.$$

- Using sieving, we can force some factors in both numerator and denominator.
- Significant win in practice. No effect asymptotically. Not enough anyway.

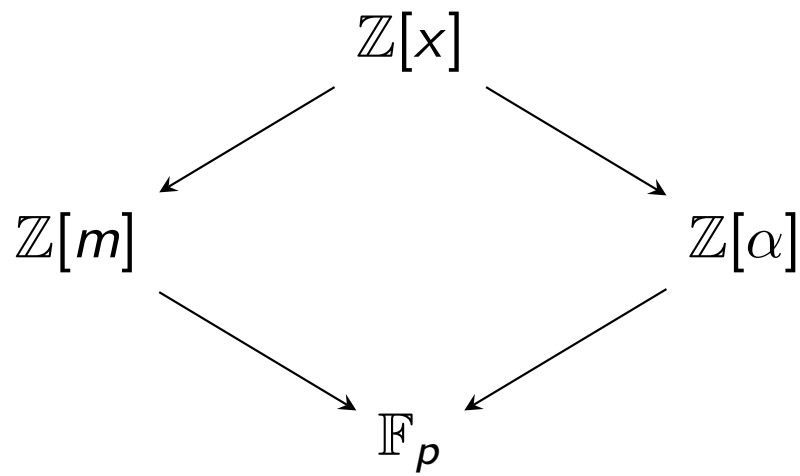
# How can we go this way ?

This method can only work for an initial step.

- Whatever we do, numerator and denominator have size $L[1]$.
- In time $L[1/3]$, all we can hope for is $L[2/3]$-smoothness.
- Checking for $L[2/3]$-smoothness is not trivial to do in complexity $L[1/3]$ (need ECM !)

# Classical descent step

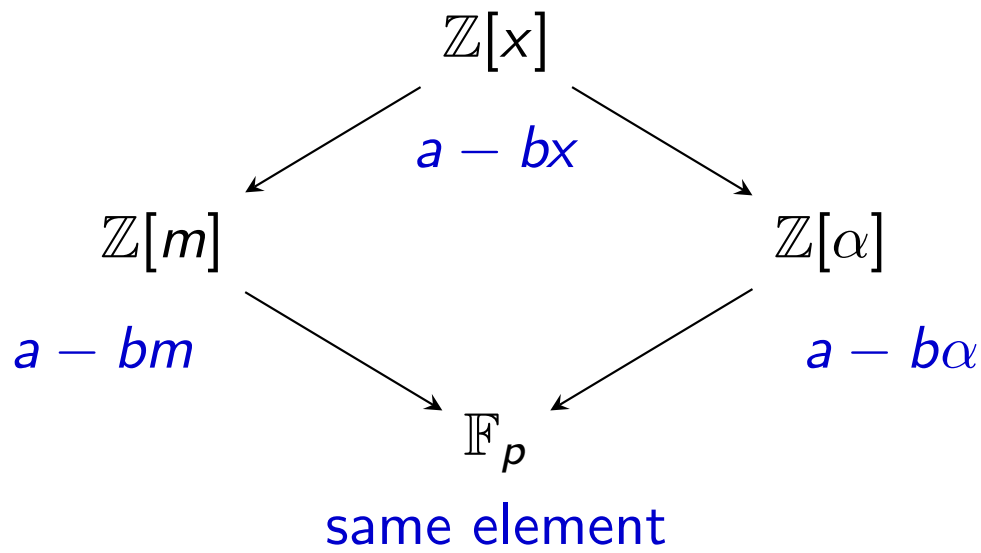
Consider $Q$ a prime appearing in factorizations obtained so far. We may suppose $Q \leq L[2/3]$.




- We want to use special-$q$'s in this context, so as to force the appearance of $Q$ in $a - bm$.
- Recursion will need to consider $Q$ on the $\alpha$ side. This works the same way.

# Classical descent step

Consider $Q$ a prime appearing in factorizations obtained so far. We may suppose $Q \leq L[2/3]$.




- We want to use special-$q$'s in this context, so as to force the appearance of $Q$ in $a - bm$.
- Recursion will need to consider $Q$ on the $\alpha$ side. This works the same way.

# Complexity of classical descent steps

Forcing a factor of size $L_p[2/3]$ in $a - bm$ or $a - b\alpha$ has a significant cost.

- The value Norm$(a - b\alpha)$ becomes large;
- It is not possible to ask for $L[1/3]$ smoothness immediately.

It is possible (but not trivial) to have the following bounds: $L[2/3]$, $L[1/2]$, $L[5/12]$, $L[3/8]$, converging to $L[1/3]$.

The bit size (or degree) is transformed by a progression: $x \to \sqrt{\beta x}$.

# How far can classical descent go ?

The classical descent is expensive.

- Converging to $L[1/3]$ costs $L[1/3]$.
- One may converge to $L[1/2]$ in time $L[1/4]$.
- One may converge to $L[c]$ in time $L[(1-c)/2]$.
- There is *no way* to go below $L[1/3]$.

Still, it's a tool which works well in practice at least down to some bound; and software is available.

# Wouldn't it be nice if ?

The classical descent is impaired by the fact that the bit-size (or degree) decreases in the form $x \to \sqrt{\beta x}$, with $\beta$ which we cannot get rid of.

What if we could transform the degree with the rule $x \to x/2$ instead ?

# Wouldn't it be nice if ?

The classical descent is impaired by the fact that the bit-size (or degree) decreases in the form $x \to \sqrt{\beta x}$, with $\beta$ which we cannot get rid of.

What if we could transform the degree with the rule $x \to x/2$ instead ?

- In polynomial time for each step ?

# Wouldn't it be nice if ?

The classical descent is impaired by the fact that the bit-size (or degree) decreases in the form $x \to \sqrt{\beta x}$, with $\beta$ which we cannot get rid of.

What if we could transform the degree with the rule $x \to x/2$ instead ?

- In polynomial time for each step ?
- With polynomial number of child nodes ?

# Wouldn't it be nice if ?

The classical descent is impaired by the fact that the bit-size (or degree) decreases in the form $x \to \sqrt{\beta x}$, with $\beta$ which we cannot get rid of.

What if we could transform the degree with the rule $x \to x/2$ instead ?

- In polynomial time for each step ?
- With polynomial number of child nodes ?
- and \$10 000 just as prize money ?

# Wouldn't it be nice if ?

The classical descent is impaired by the fact that the bit-size (or degree) decreases in the form $x \to \sqrt{\beta x}$, with $\beta$ which we cannot get rid of.

What if we could transform the degree with the rule $x \to x/2$ instead ?

- In polynomial time for each step ?
- With polynomial number of child nodes ?
- and \$10 000 just as prize money ?

This would give a quasi-polynomial algorithm.

# The new descent methods

The small characteristic case has two novel descent methods

- First method using Gröbner bases [Jou13], which almost gives the result, except for the cost for each step. This yields $L[1/4]$ complexity and works well in practice.
- Second method which does have the desired polynomial complexity at each step, yielding to a quasi-polynomial complexity in the end [BGJT14] (but practicality is less clear).

# Plan

# Christmas break improvements

Christmas break in 2012 was very productive (healthy food?).

- Stream of a dozen computational records between Dec. 25th, 2012 and Spring 2013, and we keep going.
- New algorithms.
- This work: the most effective (and simplest!) to date. Found in March 2013, preprint in June 2013; to be presented at Eurocrypt 2014 [BGJT14].

We will not cover all the recent algorithmic advances here [GGMZ13, Jou13, Jou13b, AMORH14, AMORH14b, GKZ14, GKZ14b].

# Problem sizes considered

We present a new algorithm which is well-suited to $\mathbb{F}_{q^{2k}}$, for $q \approx k$.

# Problem sizes considered

We present a new algorithm which is well-suited to $\mathbb{F}_{q^{2k}}$, for $q \approx k$.

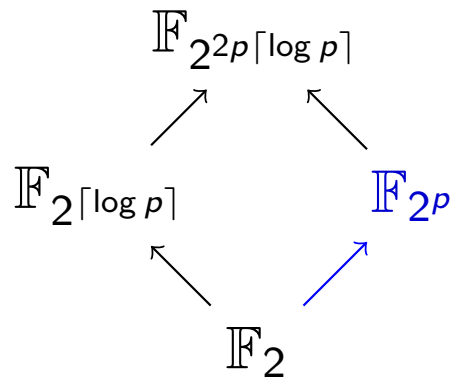
- $q \approx k$ does not pass basic type checks...
- To fit within this setting, a field must have a subfield of appropriate size.
- Not-too-adapted case is $\mathbb{F}_{2^p}$ for prime $p$.

# Problem sizes considered

We present a new algorithm which is well-suited to $\mathbb{F}_{q^{2k}}$, for $q \approx k$.

- $q \approx k$ does not pass basic type checks...
- To fit within this setting, a field must have a subfield of appropriate size.
- Not-too-adapted case is $\mathbb{F}_{2^p}$ for prime $p$.

Yet, working in $\mathbb{F}_{2^{2p\lceil \log p \rceil}}$ is viable.




- If one can solve DLP in (subgroups of) $\mathbb{F}_{2^{2p\lceil \log p \rceil}}$, then so holds for $\mathbb{F}_{2^p}$;
- It may even yield a better complexity in the end.

# Problem sizes considered

We present a new algorithm which is well-suited to $\mathbb{F}_{q^{2k}}$, for $q \approx k$.

- $q \approx k$ does not pass basic type checks...
- To fit within this setting, a field must have a subfield of appropriate size.
- Not-too-adapted case is $\mathbb{F}_{2^p}$ for prime $p$.
- A more adapted case is that of small extensions of $\mathbb{F}_{2^p}$, as provided by pairing-based crypto.
  Example: pairing over some curve defined over $\mathbb{F}_{2^{239}}$ takes values in $\mathbb{F}_{2^{239*4}}$.
  - $\mathbb{F}_{2^{239*4}}$ has $\mathbb{F}_{2^{239}}$ has a subfield, but its index is too small.
  - $\mathbb{F}_{2^{239*16}}$ is well amenable to computation by the new algorithm, The inherent "*4" made part of the way for us.

# Comparison with older stuff

NFS and FFS are hairy algorithms. In a nutshell, their workplan is:

- Collect multiplicative relations between elements of (some structures which map to) $\mathbb{F}_q^\times$.

$$\text{Example:} \quad a_{31763} \times \cdots \times a_{98231} = 1 \text{ in } \mathbb{F}_q,$$

$$\text{whence} \quad \log(a_{31763}) + \cdots + \log(a_{98231}) = 0 \mod q - 1.$$

- Solve the resulting linear system.
  $\Rightarrow$ we get log's of the elements involved.
- Relate any arbitrary element to the set of elements with known logs (descent).

NFS/FFS habit: rel. collec. $\approx$ lin. alg..

For the new algorithm: the descent is hard, the rest is trivial.

# Setting

Let $K = \mathbb{F}_{q^{2k}}$, with $k \lessapprox q$.

The field $\mathbb{F}_{q^2}$ is represented in any usual way.

Elements of $K$ are polynomials over $\mathbb{F}_{q^2}$
(understood modulo an irreducible polynomial defining $\mathbb{F}_{q^{2k}}$).

The definition polynomial is constructed as follows:

- Take $h_0, h_1 \in \mathbb{F}_{q^2}[x]$, of small degree (2 should be ok).
- Let $\Phi(x) = h_1(x)x^q - h_0(x)$.
- Until $\Phi(x)$ has an irreducible factor $f(x)$ of degree $k$.

## Heuristic 1

Taking $h_0$, $h_1$ of degree bounded by a constant always suffices to find a suitable $f$. Let $\delta$ be this constant.

# Plan

# Key tool

Statement of DLP in $\mathbb{F}_{q^{2k}}$:

- Input: $P(x) \in \mathbb{F}_{q^{2k}}$, of $D < k$.
- Ultimate goal: $\log P(x)$, modulo some $\ell \mid q-1$.
- Intermediary goal, called one descent step:
  Rewrite $P$ as a product of smaller and smaller polynomials.

# Key tool

Statement of DLP in $\mathbb{F}_{q^{2k}}$:

- Input: $P(x) \in \mathbb{F}_{q^{2k}}$, of $D < k$.
- Ultimate goal: $\log P(x)$, modulo some $\ell \mid q-1$.
- Intermediary goal, called one descent step:
  Rewrite $P$ as a product of smaller and smaller polynomials.

## Useful descent tools

- It is trivial to rewrite $P(x) = A(x)/B(x)$ with $\deg A, B < k/2$.
- All the algorithmic background from NFS/FFS can do some steps of descent, but only to a limited extent.
- We present a new descent procedure which is conceptually simpler and asymptotically faster.

# Key tool

Statement of DLP in $\mathbb{F}_{q^{2k}}$:

- Input: $P(x) \in \mathbb{F}_{q^{2k}}$, of $D < k$.
- Ultimate goal: $\log P(x)$, modulo some $\ell \mid q-1$.
- Intermediary goal, called one descent step:
  Rewrite $P$ as a product of smaller and smaller polynomials.

## Useful descent tools

- It is trivial to rewrite $P(x) = A(x)/B(x)$ with $\deg A, B < k/2$.
- All the algorithmic background from NFS/FFS can do some steps of descent, but only to a limited extent.
- We present a new descent procedure which is conceptually simpler and asymptotically faster. In practice it's a nightmare.

# A starting relation

What does $h_1(x)x^q - h_0(x) \equiv 0 \mod f(x)$ give us ?

$$x^q - x = \prod_{\alpha \in \mathbb{F}_q} (x - \alpha),$$

$$h_0(x) - xh_1(x) \equiv h_1(x) \prod_{\alpha \in \mathbb{F}_q} (x - \alpha).$$

LHS is very small, hence factors into small degree polynomials.

This is one relation. Next step: turn it into a proper descent tool.

# Plan

New tools

- Descent with Gröbner bases
- Quasi-polynomial descent

# More relations

We like to view (momentarily) the equation as bivariate:

$$X^q Y - X Y^q = \prod_{(\alpha:\beta)\in\mathbb{P}^1(\mathbb{F}_q)} (\beta X - \alpha Y)$$

where any appropriate set of representatives is chosen for $\mathbb{P}^1(\mathbb{F}_q)$.
Replace $(X : Y)$ by $(ax + b : cx + d)$, for $a, b, c, d$ all in $\mathbb{F}_{q^2}$.

$$\begin{aligned}(a^q h_0 + b^q h_1)(cx + d) - (ax + b)(c^q h_0 + d^q h_1)\\ \equiv h_1 \prod_{(\alpha:\beta)\in\mathbb{P}^1(\mathbb{F}_q)} (\beta(ax + b) - \alpha(cx + d)).\end{aligned}$$

- Degrees of LHS and RHS are exactly as before.
- Only the set of terms in the RHS varies.

# More relations

When $a, b, c, d$ run through $\mathbb{F}_{q^2}$, we get distinct relations.

- These relations link together the log's of elements of degree 1 in $K$.
- In polynomial time ($O(q^5)$), we can obtain the log's of elements of degree 1.
- We may even do the same for degree 2 in $O(q^7)$, up to degree $k$ in $O(q^{2k+1})$.
- We have sort of made trivial part of the relation collection, but will it blend for making a descent procedure ?

# A descent ?

Recall that we would like to force a factor $Q$ in the relation. Say $Q$ has degree $D$.

$$X^q Y - X Y^q = \prod_{(\alpha:\beta)\in\mathbb{P}^1(\mathbb{F}_q)} (\beta X - \alpha Y)$$

Replace $(X : Y)$ by $(au(x) + bv(x) : cu(x) + dv(x))$:

- Take $u$ and $v$ polynomials of prescribed degree.

  $$\text{LHS} = u(x)^q v(x) - u(x) v(x)^q,$$

  which can be rewritten using $x^q \equiv \frac{h_0(x)}{h_1(x)}$.
- Coefficients of the LHS are bilinear expressions in the sets of variables forming the coefficients of $u$ and $v$.
- When is the LHS divisible by $Q$ ? When some polynomial system has solutions !

# The polynomial system arising

We write the following equation as a polynomial system:

$$u(x)^q v(x) - u(x)v(x)^q \equiv 0 \mod Q$$

- Variables split into two sets:
  - Set1: coefficients of $u$ (2 coeffs over $\mathbb{F}_q$ for each).
  - Set2: coefficients of $v$ (2 coeffs over $\mathbb{F}_q$ for each).
- Equations: cancellation of the coefficients of LHS mod $Q$.
- Equations are bilinear. This is a bilinear system.

## Complexity of GB for bilinear systems

Spaenlehauer: complexity of bilinear GB computation depends on the min of the numbers of variables.

All in all, we see that we can achieve this descent step in complexity L[1/4].

# Plan

New tools

- Descent with Gröbner bases
- Quasi-polynomial descent

# Some rewriting: part 1

We like to view (momentarily) the equation as bivariate:

$$X^q Y - X Y^q = \prod_{(\alpha:\beta)\in\mathbb{P}^1(\mathbb{F}_q)} (\beta X - \alpha Y)$$

where any appropriate set of representatives is chosen for $\mathbb{P}^1(\mathbb{F}_q)$.

# Some rewriting: part 1

We like to view (momentarily) the equation as bivariate:

$$X^q Y - X Y^q = \prod_{(\alpha:\beta)\in\mathbb{P}^1(\mathbb{F}_q)} (\beta X - \alpha Y)$$

where any appropriate set of representatives is chosen for $\mathbb{P}^1(\mathbb{F}_q)$. Substitute polynomials $P$ and 1 in place of $X$ and $Y$.

- $P^q$ is something special for $P \in \mathbb{F}_{q^2}[x]$.
- For $P(x) = \sum a_i x^i$, let $\tilde{P}(x) = \sum a_i^q x^i$ so that $P(x)^q = \tilde{P}(x^q)$.

$$\tilde{P}(x^q) - P(x) = \prod_{(\alpha:\beta)\in\mathbb{P}^1(\mathbb{F}_q)} (\beta P(x) - \alpha),$$

$$h_1^D \times \left(\tilde{P}(h_0/h_1) - P\right) \equiv h_1^D \prod_{(\alpha:\beta)\in\mathbb{P}^1(\mathbb{F}_q)} (\beta P - \alpha).$$

# Examing the terms

$$h_1^D \times \left(\tilde{P}(h_0/h_1) - P\right) \equiv h_1^D \prod_{(\alpha:\beta)\in\mathbb{P}^1(\mathbb{F}_q)} (\beta P - \alpha).$$

The LHS has degree at most $(\delta + 1)D$.

- $\delta$ being a constant, there is constant chance, as $P$ varies, that LHS factors in polynomials of degree, say, $D/2$.
- The RHS has many factors of degree equal to $D$ (and $P$ itself is among them).

This is a neat relation, BUT:

- This is just one relation.
- Anyway, we have some chance to rewrite $P$ as product of polynomials of the same size, so what ?

# Some rewriting: part 2

$$X^q Y - X Y^q = \prod_{(\alpha:\beta)\in\mathbb{P}^1(\mathbb{F}_q)} (\beta X - \alpha Y).$$

# Some rewriting: part 2

$$X^q Y - X Y^q = \prod_{(\alpha:\beta)\in\mathbb{P}^1(\mathbb{F}_q)} (\beta X - \alpha Y).$$

Now we substitute $(aP + b)$ and $(cP + d)$, for $a, b, c, d$ all in $\mathbb{F}_{q^2}$.

$$h_1^D \Big( (a^q \tilde{P}(h_0/h_1) + b^q)(cP + d) - (aP + b)(c^q \tilde{P}(h_0/h_1) + d^q) \Big)$$
$$\equiv h_1^D \prod_{(\alpha:\beta)\in\mathbb{P}^1(\mathbb{F}_q)} (\beta(aP + b) - \alpha(cP + d)).$$

- Degrees of LHS and RHS are exactly as before.
- Only the set of terms in the RHS varies.
  Let $\mathcal{F}$ be the set of terms $(P - \gamma)$ encountered (normalized multiplicatively).

# Understanding $\mathcal{F}$

For our easy relation, the set of terms was simply $\{P - \alpha,\ \alpha \in \mathbb{F}_q\}$.

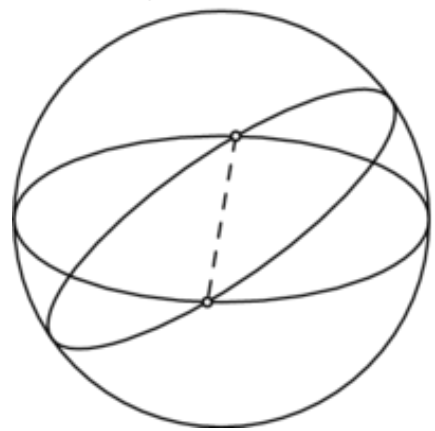
Now for the more general case:
Let $h$ be homography $z \mapsto \frac{az+b}{cz+d}$. Recall that $a, b, c, d$ are in $\mathbb{F}_{q^2}$.

The terms in $\mathrm{RHS}_{a,b,c,d}$ are:

$$\{P - \text{"abscissa" of } h^{-1} \cdot (\alpha : \beta),\ (\alpha : \beta) \in \mathbb{P}^1(\mathbb{F}_q)\}.$$

The possible RHS combinations are in bijective mapping with:

- All image sets of $\mathbb{P}^1(\mathbb{F}_q)$ within $\mathbb{P}^1(\mathbb{F}_q^2)$ by homographies.
- The cosets of $\mathrm{PGL}(2, q^2)/\mathrm{PGL}(2, q)$.



This makes exactly $q^3 + q$ relations;
$\#\mathcal{F} = q^2$ terms allowed to appear in the RHS.

# Outcome

## Our formula for getting relations:

One choice of $a, b, c, d$ (one coset in $\mathrm{PGL}(2, q^2)/\mathrm{PGL}(2, q)$)

$\longleftrightarrow$

Some subset of the possible terms $P - \gamma$ in the RHS.

We have $q^3 + q$ potential relations, and $q^2 + 1$ terms may appear in the RHS (each RHS has $q + 1$ terms).

However only some of the LHS obtained are smooth.

# Outcome

## Our formula for getting relations:

One choice of $a, b, c, d$ (one coset in $\mathrm{PGL}(2, q^2)/\mathrm{PGL}(2, q)$)

$\longleftrightarrow$

Some subset of the possible terms $P - \gamma$ in the RHS.

We have $q^3 + q$ potential relations, and $q^2 + 1$ terms may appear in the RHS (each RHS has $q + 1$ terms).

However only some of the LHS obtained are smooth.

## Heuristic 2

Assume that the LHS behave like random as $a, b, c, d$ vary.
We get $\Theta(q^3 + q)$ relations which are $D/2$-smooth.

# Letting only $P$ show up

Keep only smooth relations.

$$\prod(\text{polynomials smaller than } P) \equiv \prod(\text{some of the } P - \gamma),$$
$$\mathcal{L}_k = \mathcal{R}_k.$$

Our starting polynomial $P$ sometimes appears in $\mathcal{R}_k$.

We now do a multiplicative combination of all relations.

$$\prod_k \mathcal{L}_k^{e_k} \equiv \prod_k \mathcal{R}_k^{e_k}.$$

- We would like to have $\prod_k \mathcal{R}_k^{e_k} \equiv P$.
- This is a linear system. We need the vector $(1, \overbrace{0, \ldots, 0}^{q^2})$ be in the image (assuming some adequate indexing).

# Can we solve the linear system ?

For $\prod_k \mathcal{R}_k^{e_k} \equiv P$ to be possible, we need:

**Heuristic 3**

The linear system of size $\Theta(q^3+q) \times (q^2+1)$ has full rank.

Note: if we hadn't restricted our view to smooth relations:

- We would have a system of size $(q^3+q) \times (q^2+1)$.
- Combinatorial design theory recognizes an inversive plane there. This proves that the (big) matrix has full rank.
- Yet, saying something about our few selected rows is difficult.

# Plan

# Complexity of one descent step

Workplan:

- Traverse representatives $a, b, c, d$ of $\mathrm{PGL}(2, q^2)/\mathrm{PGL}(2, q)$.
  - Compute LHS. Write down RHS.
  - Test LHS for smoothness.
- Solve the linear system of size $\Theta(q^3 + q) \times (q^2 + 1)$, with $q + 1$ non-zero entries per row.

Cost: $O(q^5)$ using sparse linear algebra ($O(q^{2w})$ dense).

**Outcome**

$P$ as a product of $q^2 D$ smaller polynomials.

# The descent tree

Each node of the descent tree corresponds to one application of the new descent tool, hence its arity is in $q^2 D$.

| level | $\deg P_i$ | breadth of tree |
|---|---|---|
| 0 | $k$ | 1 |
| 1 | $k/2$ | $q^2 k$ |
| 2 | $k/4$ | $q^2 k \cdot q^{2\frac{k}{2}}$ |
| 3 | $k/8$ | $q^2 k \cdot q^{2\frac{k}{2}} \cdot q^{2\frac{k}{4}}$ |
| $\vdots$ | $\vdots$ | $\vdots$ |
| $\log k$ | 1 | $\leq q^{2 \log k} k^{\log k}$ |

**Total number of nodes** $= q^{O(\log k)}$.

Each node entails a cost which is polynomial in $q$.

# Bottom of the tree

At level $\log k$, we are below some constant degree (say 2).

Restating the descent tool shows that we obtain an overdetermined system of equations relating all logarithms of these small polynomials.

- Follow the descent tree so as to reduce the original DLP to many of degree 2.
- Re-do the descent tool "from 2 to 2", use the linear system to find all logs of degree 2.
- Back-substitute.

**Overall complexity**

$q^{O(\log k)}$ nodes, $O(q^5)$ per node $\rightsquigarrow$ $\boxed{\text{Complexity } q^{O(\log k)}}$.

# Main result

## Main result

Let $K$ be a finite field of the form $\mathbb{F}_{q^{2k}}$. A discrete logarithm in $K$ can be computed in heuristic time

$$\max(q,k)^{O(\log k)}.$$

- We covered the case $k \approx q$.
- For $q < k$, we compose the field with $\mathbb{F}_{q^c}$ with $c = \left\lceil \frac{\log k}{\log q} \right\rceil$ (hence the max in the complexity formula).

Example for $K = \mathbb{F}_{2^p}$:
- We take $q \approx k \approx p$.
- Cost is $p^{O(\log p)} = \exp(O((\log p)^2))$.
- This is quasi-polynomial.

# Bad days for pairings

The fields which are most endangered by this attack are the fields where it applies almost as-is.

Pairings over small characteristic fields provide a natural set-up.

- Let $\mathbb{F}_q = \mathbb{F}_{2^n}$. We have $E[r] \times E[r] \to \mu_r \subset K = \mathbb{F}_{q^k}$, where $k$ is the embedding degree.
- DLP security in $E$ and $K$ should match, e.g. $\#E \approx 2^{160}$ and $K$ large enough. Hence $k \geq 10$ desired.
- Disturbing belief that DLP security in $K$ is not affected by the fact the $[K : \mathbb{F}_2] = nk$ is composite. Wrong.
- For most practical cases, it appears that DLP security in $K = \mathbb{F}_{2^{kn}}$ here is hardly better than in $\mathbb{F}_{2^{4n}}$, which cancels the "security improvement" by the embedding degree.

Small characteristic pairings are practically dead.

# Plan

# Conclusion

New algorithm
- Quasi-polynomial complexity.
- Relies on heuristic, experimentally checked.
- The algorithm as a whole has not been used yet.

**Cross-over?** Right now, practicality is unclear.

- The arity $O(q^2 D)$ at each step is large !
- Fortunately some other descent tools are compatible, so that we do not have to pay arity $O(q^2 D)$ always.
- See e.g. other works like [AMORH14, GKZ14].
- Largest computed dlogs this in January 2014: $\mathbb{F}_{2^{4\cdot 1223}}$, $\mathbb{F}_{2^{12\cdot 367}}$ [GKZ14], and $\mathbb{F}_{2^{9234=2\cdot 3^5\cdot 19}}$ [GKZ14b].

Take home:

- RIP small characteristic pairings.
- No, this does not affect RSA, nor large characteristic DLP.

# References I


[AMORH14a] Gora Adj, Alfred J. Menezes, Thomaz Oliveira, and Francisco Rodríguez-Henríquez, *Weakness of $\mathbb{F}_{3^{6\cdot 509}}$ for Discrete Logarithm Cryptography*, Pairing-Based Cryptography – Pairing 2013, 2014, pp. 20–44. 6th International Conference, Beijing, China, November 22-24, 2013, Revised Selected Papers.

[AMORH14b] ———, *Computing Discrete Logarithms in $\mathbb{F}_{3^{6*137}}$ and $\mathbb{F}_{3^{6*163}}$ using Magma*, 2014, available at `http://eprint.iacr.org/2014/057`.

[BGJT14] Răzvan Bărbulescu, Pierrick Gaudry, Antoine Joux, and Emmanuel Thomé, *A heuristic quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic*, Advances in Cryptology – EUROCRYPT 2014, 2014, pp. ???–???, available at `http://eprint.iacr.org/2013/400`. Proc. 33th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, May 11 - 15, 2014.

# References II


[GGMZ13] Faruk Göloglu, Robert Granger, Gary McGuire, and Jens Zumbrägel, *On the Function Field Sieve and the Impact of Higher Splitting Probabilities*, Advances in Cryptology – CRYPTO 2013, 2013, pp. 109–128. Proc. 33rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2013, Part II.

[GKZ14a] Robert Granger, Thorsten Kleinjung, and Jens Zumbrägel, *Breaking '128-bit Secure' Supersingular Binary Curves (or how to solve discrete logarithms in* $\mathbb{F}_{2^{4\cdot 1223}}$ *and* $\mathbb{F}_{2^{12\cdot 367}}$*)*, 2014.

[GKZ14b] ———, *Discrete Logarithms in GF(2ŷ234)*, January 2014. Email to the `NMBRTHRY` mailing-list.

[Jou13a] Antoine Joux, *A new index calculus algorithm with complexity* $L(1/4 + o(1))$ *in very small characteristic*, 2013. Cryptology ePrint Archive report.

# References III


[Jou13b] ———, *Faster Index Calculus for the Medium Prime Case. Application to 1175-bit and 1425-bit Finite Fields*, Advances in Cryptology – EUROCRYPT 2013, 2013, pp. 177–193. Proc. 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, May 26–30, 2013.

[JL03] Antoine Joux and Reynald Lercier, *Improvements to the general number field sieve for discrete logarithms in prime fields. A comparison with the gaussian integer method*, Math. Comp. **72** (2003), no. 242, 953–967.