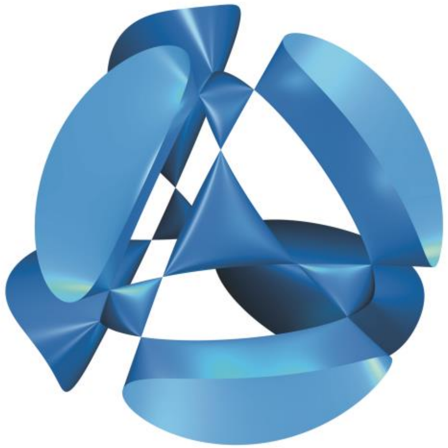
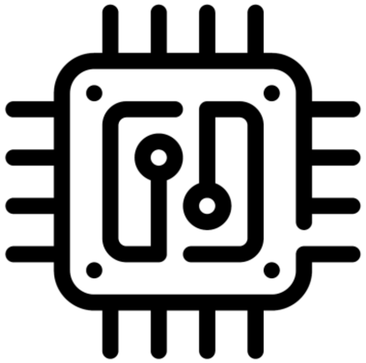
Why hyperelliptic?

Craig Costello

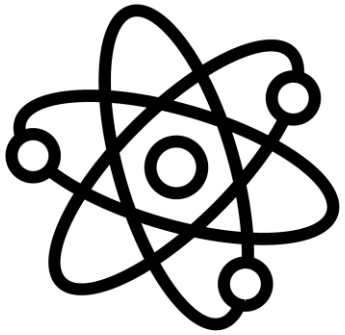
Microsoft Research



Part 1: Why use curves at all?
Part 2: Why go beyond genus 1?
Part 3: Why stop at genus 2?
Part 4: Why Kummer surfaces?



Part 5: Why are we interested in isogenies?
Part 6: Why is genus 2 (even more) promising here?
Part 7: Why stop at genus 2?
Part 8: Why not use hyperelliptic curves in genus 1?



Diffie-Hellman key exchange (circa 1976)

$$q = 1606938044258990275541962092341162602522202993782792835301301$$

$$g = 123456789$$

$$g^a \bmod q = 78467374529422653579754596319852702575499692980085777948593$$

$$560048104293218128667441021342483133802626271394299410128798 = g^b \bmod q$$

$a =$
685408003627063
761059275919665
781694368639459
527871881531452

$b =$
362059131912941
987637880257325
269696682836735
524942246807440

$$g^{ab} \bmod q = 437452857085801785219961443000845969831329749878767465041215$$

paper

Index calculus

paper

$$\text{solve} \quad g^x \equiv h \pmod{p}$$
$$\text{e.g.} \quad 3^x \equiv 37 \pmod{1217}$$

- factor base $p_i = \{2,3,5,7,11,13,17,19\}$, $\#p_i = 8$
- Find 8 values of k where 3^k splits over p_i, i.e., $3^k \equiv \pm\prod p_i \bmod p$

$\pmod{1217}$

$$\begin{aligned}
3^1 &\equiv 3\\
3^{24} &\equiv -2^2 \cdot 7 \cdot 13\\
3^{25} &\equiv 5^3\\
3^{30} &\equiv -2 \cdot 5^2\\
3^{34} &\equiv -3 \cdot 7 \cdot 19\\
3^{54} &\equiv -5 \cdot 11\\
3^{71} &\equiv -17\\
3^{87} &\equiv 13
\end{aligned}$$

$\pmod{1216}$

$$\begin{aligned}
1 &\equiv L(3)\\
24 &\equiv 608 + 2 \cdot L(2) + L(7) + L(13)\\
25 &\equiv 3 \cdot L(5)\\
30 &\equiv 608 + L(2) + 2 \cdot L(5)\\
34 &\equiv 608 + L(3) + L(7) + L(19)\\
54 &\equiv 608 + L(5) + L(11)\\
71 &\equiv 608 + L(17)\\
87 &\equiv L(13)
\end{aligned}$$

$\pmod{1216}$

$$\begin{aligned}
L(2) &\equiv 216\\
L(3) &\equiv 1\\
L(5) &\equiv 819\\
L(7) &\equiv 113\\
L(11) &\equiv 1059\\
L(13) &\equiv 87\\
L(17) &\equiv 679\\
L(19) &\equiv 528
\end{aligned}$$

Index calculus

solve $g^x \equiv h \pmod{p}$

e.g. $3^x \equiv 37 \pmod{1217}$

$L(2) \equiv 216$
$L(3) \equiv 1$
$L(5) \equiv 819$
$L(7) \equiv 113$
$L(11) \equiv 1059$
$L(13) \equiv 87$
$L(17) \equiv 679$
$L(19) \equiv 528$

Now search for j such that $g^j \cdot h = 3^j \cdot 37$ factors over p_i

$$3^{16} \cdot 37 \equiv 2^3 \cdot 7 \cdot 11 \pmod{1217}$$

$$\begin{aligned} L(37) &\equiv 3 \cdot L(2) + L(7) + L(11) - 16 \pmod{1216} \\ &\equiv 3 \cdot 216 + 113 + 1059 - 1 \\ &\equiv 588 \end{aligned}$$

Subexponential complexity $L_p[1/3, (64/9)^{1/3}] = e^{\left((64/9)^{1/3} + o(1)\right)(\ln(p))^{1/3} \cdot (\ln\ln(p))^{2/3}}$

Diffie-Hellman key exchange (circa 2016)

$q =$

58096059953699580628595025333045743706869751763628952366614861522872037309971102257373360445331184072513261577549805174439905295945400471216628856721870324010321116397
06440498844049850989051627200244765807041812394729680540024104827976584369381522292361208779044769892743225751738076979568811309579125511333093243519553784816306381580
16186020024749256844815024251530444957718760413642873858099017255157393414625583036640591500086964373205321856683254529110790372283163413859958640669032595972518744716
90595408050123102096390117507487600170953607342349457574162729948560133086169585299583046776370191815940885283450612858638982717634572948835466388795543116154464463301
99254382340016292057090751175533888161918987295591531536698701292267685465517437915790823154844634780260102891718032495396075041899485513811126977307478969074857043710
716150121315922024556759241239013152919710956468406379442914941614357107914462567329693649

$g = 123456789$

$g^a \pmod{q} =$

197496648183227193286262018614250555971909799762533760654008147994875775445667054218578105133138217497206890599554928429450667899476
854668595594034093493637562451078938296960313488696178848142491351687253054602202966247046105770771577248321682117174246128321195678
537631520278649403464797353691996736993577092687178385602298873558954121056430522899619761453727082217823475746223803790014235051396
799049446508224661850168149957401474638456716624401906701394472447015052569417746372185093302535739383791980070572381421729029651639
30423436126876497170776348430066892397286870912166558669830978657804740157916611563508569886847487772676671207386096152947607114559
706340209059103703018182635521898738094546294558035569752596676346614699327742088471255741184755866117812209895514952436160199336532
6052422101474898256696660124195726100495725510022002932814218768060112310763455404567248761396399633344901857872119208518550803791724

411604662069593306683228525653441872410777999220572079993574397237156368762038378332742471939666544968793817819321495269833613169937
986164811320795616949957400518206385310292475529284550626247132930124027703140131220968771142788394846592816111078275196955258045178
705254016469773509936925361994895894163065551105161929613139219782198757542984826465893457768888915561514505048091856159412977576049
0735632255728098809700583965017196658531101013084326474277865655251213287725871678420376241901439097879386658420056919111997396726455
110758448552553744288464337906540312125397571803103278271979007681841394534114315726120595749993896347981789310754194864577435905673
172970033596584445206671223874399576560291954856168126236657381519414592942037018351232440467191228145585909045861278091800166330876
4073238447199488070126873048860279221761629281961046255219584327714817248626243962413613075956770018017385724999495117779149416882188

$= g^b \pmod{q}$

$a =$

7147687166405;9571879053605547396582
692405186145916522354912615715297097
100679170037904924330116019497881089
087696131592831386326210951294944584
4004974889298038584931918128447572321
023987160439062006177648318875457556
2337708539125052923646318332191217321
464134655845254917228378772756695589
845219962202945089226966507426526912
7802446416400\9025927104004338958261
1419862375878988193612187945591802864
062679\864839578139273043684955597764
13009721221824915810964579376354556\6
554629883777859568089157882151127357
4220422646379170599917677567\30420698
422392494816906777896174923072071297
603455802621072109220\54662739697748
553543758990879608882627763290293452
560094576029847\3913613887675543866
22479265299978059886472414530462194
52761811989\9746477252908878060493
17954195146382922889045577804592943
73052654\10485180264002079415193983
85114342508427311982036827478946058
7100\304977477069244278989689910572
12096357725203480402449913844583448

$b =$

655456209464694;93360682685816031704
969423104727624468251177438749706128
879957701\93698826859762790479113062
308975863428283798589097017957365590
672\83571386389571224667609499300898
554802446403039544300748002507962036
386619315229886063541005322448463915
89798641210273772558373965\486539312
854838650709031919742048649235894391
90352993032676961005\088404319792729
916038927477470940948581926791161465
02863521484987\086232861934222391717
121545686125300672760188085915004248
49476686\706784051068715397706852664
532638332403983747338379697022624261
377163163204493828299206039808703403
575100467337085017748387148822224875
309641791879395483731754620034884930
540399950519191679471224\05558557093
219350747155777569598163700850920394
705281936392411084\43600686183528465
724969562186437214972625833222544865
996160464558\54629937016589470425264
445624157899586972652935647856967092
689604\42796501209877036845001246792
761563917639959736383038665362727158

$g^{ab} =$

330166919524192149323761733598426244691224199958894654036331526394350099088627302979833339501183059198113987880066739
419999231378970715307039317876258453876701124543849520979430233302777503265010724513551209279573183234934359636696506
968325769489511028943698821518689496597758218540767517885836464160289471651364552490713961456608536013301649753975875
610659655755567474438180357958360226708742348175045563437075840969230826767034061119437657466993989389348289599600338
950372251336932673571743428823026014699232071116171392219599691096846714133643382745709376112500514300983651201961186
613464267685926563624589817259637248558104903657371981684417053993082671827345252841433337325420088380059232089174946
086536664984836041334031650438692639106287627157575758383128971053401037407031731509582807639509448704617983930135028
7596589383292751993079161318839043121329118930009948197899907586986108953591420279426874779423560221038468

Diffie-Hellman key exchange (cont.)

- Individual secret keys secure under Discrete Log Problem (DLP): $g, g^x \mapsto x$
- Shared secret secure under Diffie-Hellman Problem (DHP): $g, g^a, g^b \mapsto g^{ab}$
- Fundamental operation in DH is group exponentiation: $g, x \mapsto g^x$
  ... done via "square-and-multiply", e.g., $(x)_2 = (1,0,1,1,0,0,0,1 \ldots)$
- We are working "$\mathbf{mod}\ q$", but only with <u>one operation</u>: multiplication
- Main reason for fields being so big: (sub-exponential) index calculus attacks!

DH key exchange (Koblitz-Miller style)

If all we need is a group, why not use elliptic curve groups?

paper

MATHEMATICS OF COMPUTATION
VOLUME 48, NUMBER 177
JANUARY 1987, PAGES 203–209

Elliptic Curve Cryptosystems

By Neal Koblitz

This paper is dedicated to Daniel Shanks on the occasion of his seventieth birthday

Abstract. We discuss analogs based on elliptic curves over finite fields of public key cryptosystems which use the multiplicative group of a finite field. These elliptic curve cryptosystems may be more secure, because the analog of the discrete logarithm problem on elliptic curves is likely to be harder than the classical discrete logarithm problem, especially over $GF(2^n)$. We discuss the question of primitive points on an elliptic curve modulo p, and give a theorem on nonsmoothness of the order of the cyclic subgroup generated by a global point.

1. Introduction. The earliest public key cryptosystems using number theory were based on the structure either of the multiplicative group $(\mathbf{Z}/N\mathbf{Z})^*$ or the multiplicative group of a finite field $GF(q)$, $q = p^n$. The subsequent construction of analogous systems based on other finite Abelian groups, together with H. W. Lenstra's success in using elliptic curves for integer factorization, make it natural to study the possibility of public key cryptography based on the structure of the group of points of an elliptic curve over a large finite field. We first briefly recall the facts we need about such elliptic curves (for more details, see [4] or [5]). We then describe elliptic curve analogs of the Massey-Omura and ElGamal systems. We give some concrete examples, discuss the question of primitive points, and conclude with a theorem concerning the probability that the order of a cyclic subgroup is nonsmooth.

I would like to thank A. Odlyzko for valuable discussions and correspondence, and for sending me a preprint by V. S. Miller, who independently arrived at some similar ideas about elliptic curves and cryptography.

2. Elliptic Curves. An elliptic curve E_K defined over a field K of characteristic $\neq 2$ or 3 is the set of solutions $(x, y) \in K^2$ to the equation

(1) $$y^2 = x^3 + ax + b, \qquad a, b \in K$$

(where the cubic on the right has no multiple roots). More precisely, it is the set of such solutions together with a "point at infinity" (with homogeneous coordinates $(0, 1, 0)$; if K is the real numbers, this corresponds to the vertical direction which the tangent line to E_K approaches as $x \to \infty$). One can start out with a more complicated general formula for E_K which can easily be reduced to (1) by a linear change of variables whenever char $K \neq 2, 3$. If char $K = 2$—an important case in

Received October 29, 1985; revised June 5, 1986.
1980 *Mathematics Subject Classification* (1985 *Revision*). Primary 11T71, 94A60; Secondary 68P25, 11Y11, 11Y40.

203

Use of Elliptic Curves in Cryptography

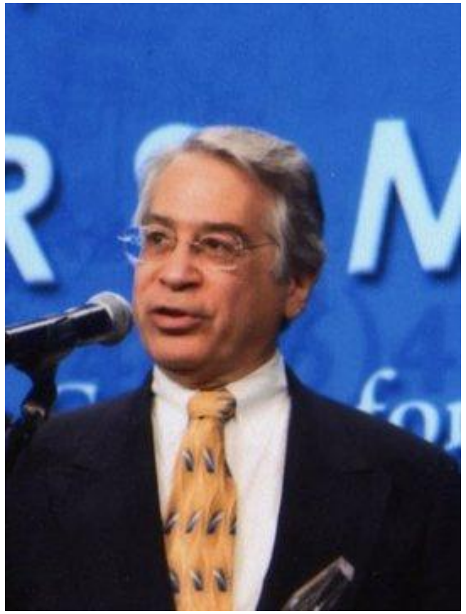
Victor S. Miller

Exploratory Computer Science, IBM Research, P.O. Box 218, Yorktown Heights, NY 10598

ABSTRACT

We discuss the use of elliptic curves in cryptography. In particular, we propose an analogue of the Diffie-Hellmann key exchange protocol which appears to be immune from attacks of the style of Western, Miller, and Adleman. With the current bounds for infeasible attack, it appears to be about 20% faster than the Diffie-Hellmann scheme over GF(p). As computational power grows, this disparity should get rapidly bigger.

H.C. Williams (Ed.): Advances in Cryptology - CRYPTO '85, LNCS 218, pp. 417-426, 1986.
© Springer-Verlag Berlin Heidelberg 1986



paper

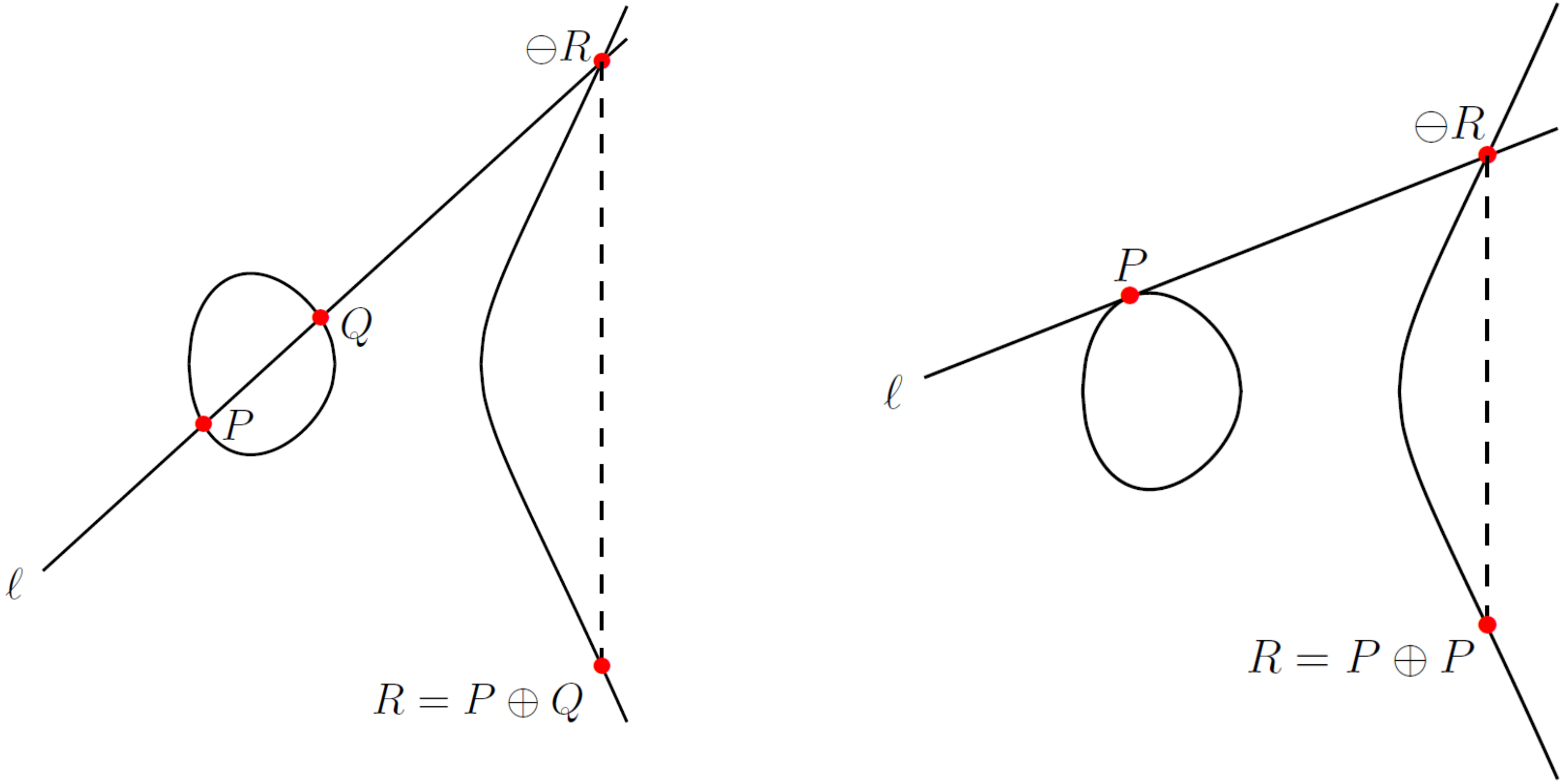
Rationale: "it is extremely unlikely that an index calculus attack on the elliptic curve method will ever be able to work" [Miller, 85]

Elliptic curve group law is easy

Fun fact: homomorphism between Jacobian of elliptic curve and elliptic curve itself

Upshot: you don't have to know what a Jacobian is to understand/do elliptic curve cryptography

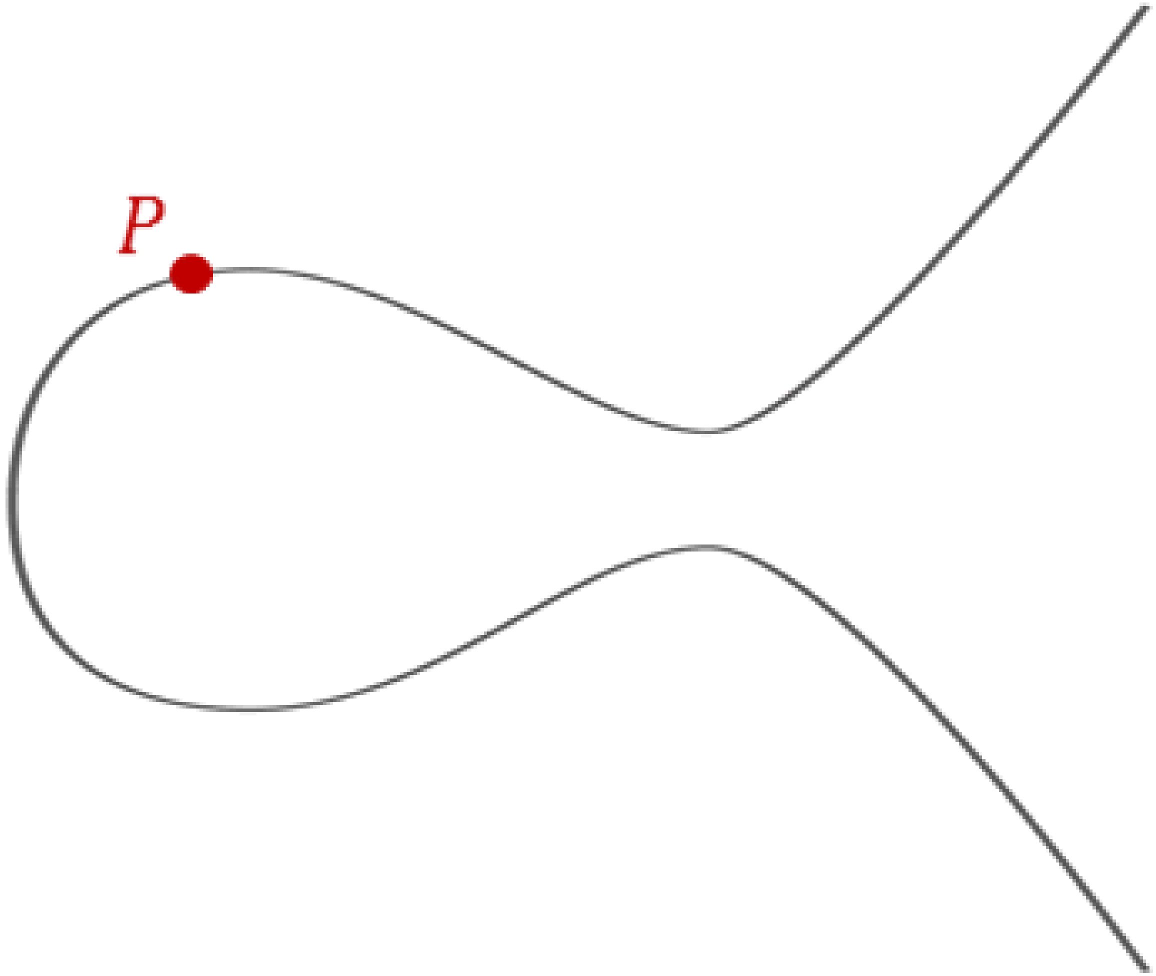
The elliptic curve group law $\oplus$



a line that intersects a cubic twice must intersect it again

The fundamental ECC operation

$$P, k \mapsto [k]P$$



GIF: Wouter Castryck

Scalar multiplications via double-and-add

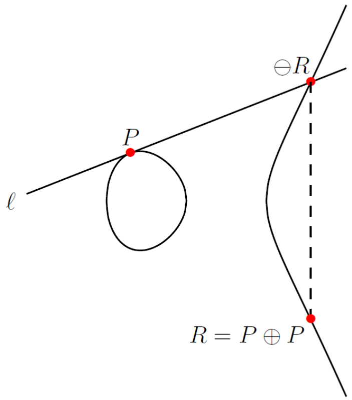
How to (naively) compute $k, Q \mapsto [k]Q$?

$$P \leftarrow Q$$

$$k = (k_n, k_{n-1}, \dots, k_0)_2$$

for i from $n-1$ downto 0 do

$$P \leftarrow [2]P$$

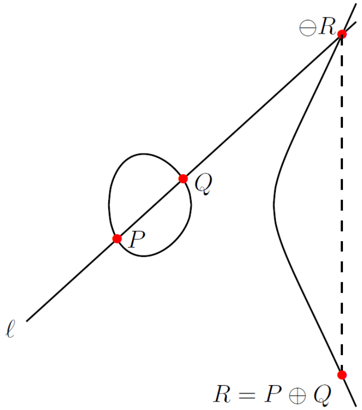


DBL

if $k_i = 1$ then

$$P \leftarrow P \oplus Q$$

end if



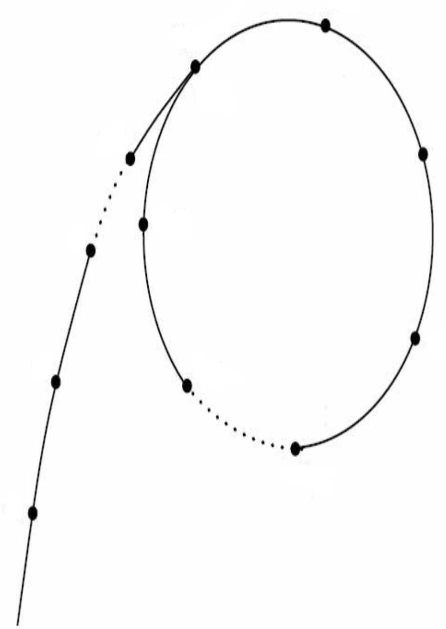
ADD

end for

return $P\ (= [k]Q)$

ECDLP security and Pollard's rho algorithm

- ECDLP: given $P, Q \in E(\mathbb{F}_p)$ of prime order N, find k such that $Q = [k]P$
- Pollard'78: compute pseudo-random $R_i = [a_i]P + [b_i]Q$ until we find a collision $R_i = R_j$ with $b_i \neq b_j$, then $k = (a_j - a_i)/(b_i - b_j)$
- Birthday paradox says we can expect collision after computing $\sqrt{\pi n/2}$ group elements R_i, i.e., after $\approx \sqrt{N}$ group operations. So 2^{128} security needs $N \approx 2^{256}$
- The best known ECDLP algorithm on (well-chosen) elliptic curves remains generic, i.e., elliptic curves are as strong as is possible



paper

Index calculus on elliptic curves?

[Miller, 85] : “it is extremely unlikely that an index calculus […] will ever be able to work”

Consider $E/\mathbb{F}_{1217}$: $\quad y^2 = x^3 - 3x + 139$

$$\#E(\mathbb{F}_{1217}) = 1277$$

$P = (3{,}401)$ and $Q = (192{,}847)$

ECDLP: find k such that $[k]P = Q$

Regardless of factor base, can’t efficiently decompose elements!

e.g., factor base $R_i = \{(3{,}401), (5{,}395), (7{,}73), (11{,}252), (13{,}104), (19{,}265)\}$

Writing $S = \sum [k_i] R_i$ involves solving discrete logarithms, compare this to integers $\mathbf{mod}\ p$ where we lift and factorise over the integers

NIST Curve P-256

NIST NISTReCur.pdf

csrc.nist.gov/groups/ST/toolkit/documents/dss/NISTReCur.pdf

RECOMMENDED ELLIPTIC CURVES FOR FEDERAL GOVERNMENT USE

July 1999

This collection of elliptic curves is recommended for Federal government use and contains choices of private key length and underlying fields.

§1. Parameter Choices

§2. Curves over Prime Fields

For each prime p, a pseudo-random curve

$$E: \quad y^2 \equiv x^3 - 3x + b \pmod{p}$$

National Institute of Standards and Technology

Curve P-256

$p =$ 115792089210356248762697446949407573530086143415290314195533631308867097853951

$r =$ 115792089210356248762697446949407573529996955224135760342422259061068512044369

$s =$ c49d3608 86e70493 6a6678e1 139d26b7 819f7e90

$c =$ 7efba166 2985be94 03cb055c 75d4f7e0 ce8d84a9 c5114abc af317768 0104fa0d

$b =$ 5ac635d8 aa3a93e7 b3ebbd55 769886bc 651d06b0 cc53b0f6 3bce3c3e 27d2604b

$G_x =$ 6b17d1f2 e12c4247 f8bce6e5 63a440f2 77037d81 2deb33a0 f4a13945 d898c296

$G_y =$ 4fe342e2 fe1a7f9b 8ee7eb4a 7c0f9e16 2bce3357 6b315ece cbb64068 37bf51f5

ECDH key exchange (1999 – nowish)

$$p = 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$$

$p = 115792089210356248762697446949407573530086143415290314195533631308867097853951$

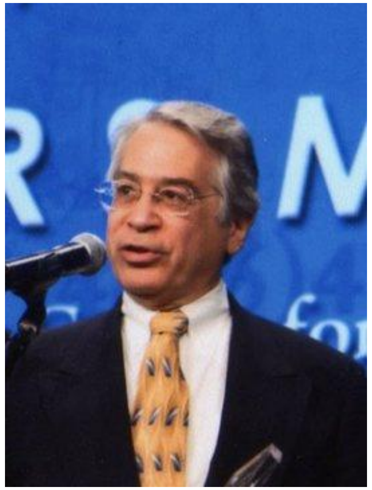
$$E/\mathbb{F}_p: y^2 = x^3 - 3x + b$$

$\#E = 115792089210356248762697446949407573529996955224135760342422259061068512044369$

$P = (48439561293906451759052585252797914202762949526041747995844080717082404635286,$
$36134250956749795798585127919587881956611106672985015071877198253568414405109)$

$[a]P = (84116208261315898167593067868200525612344221886333785331584793435449501658416,$
$102885655542185598026739250172885300109680266058548048621945393128043427650740)$

$[b]P = (101228882920057626679704131545407930245895491542090988999577542687271695288383,$
$77887418190304022994116595034556257760807185615679689372138134363978498341594)$



$a =$
89130644591246033577639
77064146285502314502849
28352556031837219223173
24614395

$[ab]P = (101228882920057626679704131545407930245895491542090988999577542687271695288383,$
$77887418190304022994116595034556257760807185615679689372138134363978498341594)$

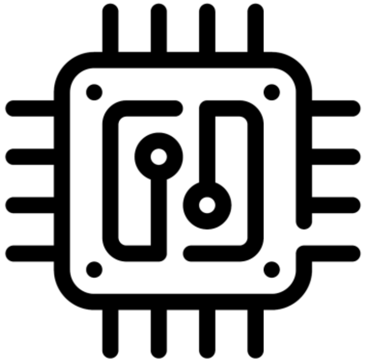
$b =$
10095557463932786418806
93831619070803277191091
90584053916797810821934
05190826

Part 1: Why use curves at all?

Part 2: Why go beyond genus 1?

Part 3: Why stop at genus 2?

Part 4: Why Kummer surfaces?

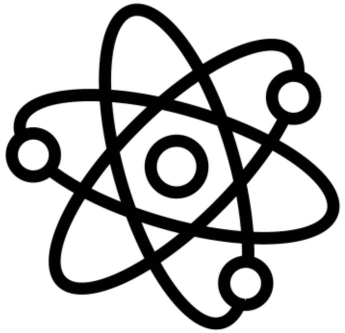


Part 5: Why are we interested in isogenies?

Part 6: Why is genus 2 (even more) promising here?

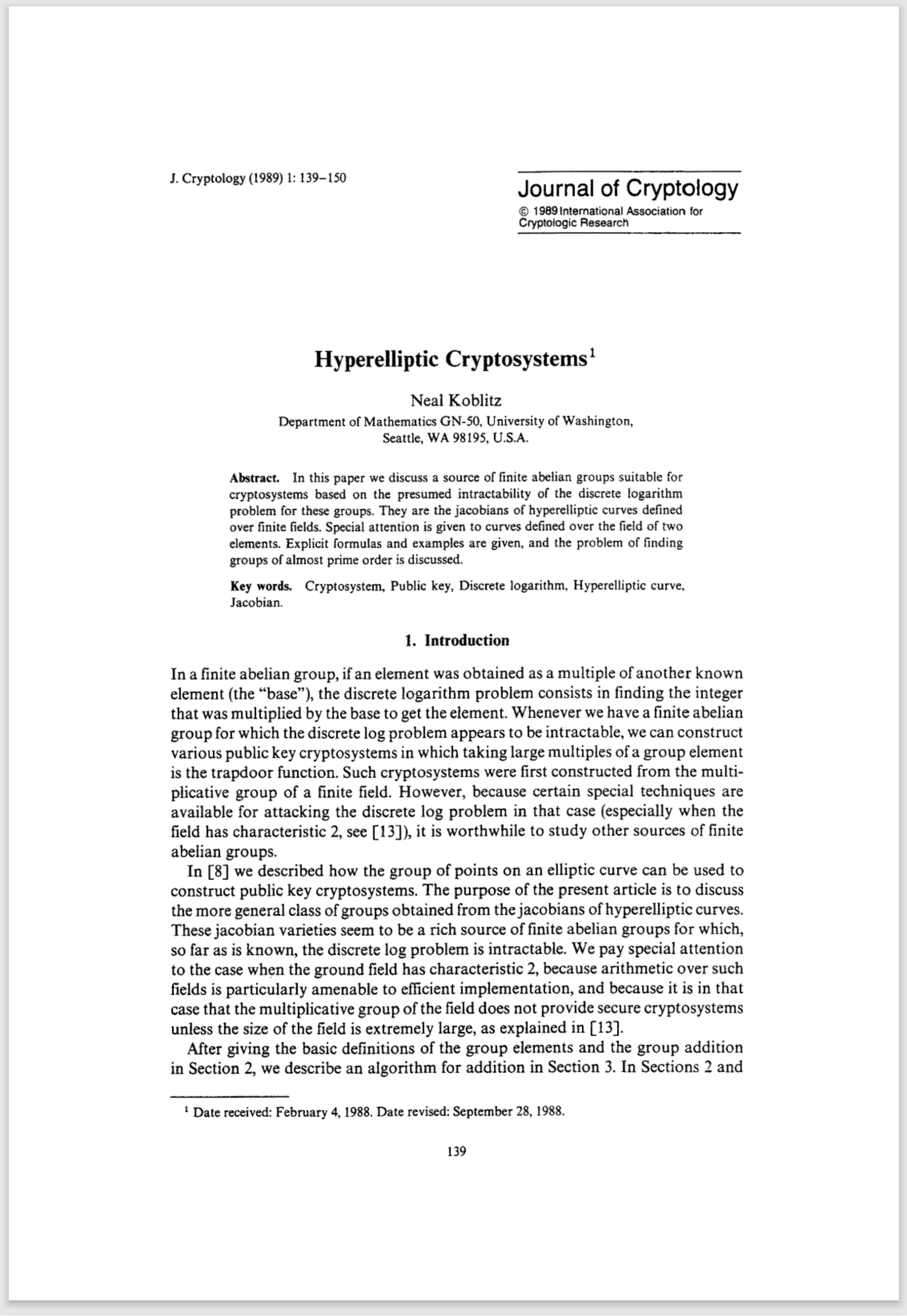
Part 7: Why stop at genus 2?

Part 8: Why not use hyperelliptic curves in genus 1?



Why hyperelliptic?

"These jacobian varieties seems to be a rich source of finite abelian groups for which, so far as is known, the discrete log problem is intractable" – [Koblitz '89]



J. Cryptology (1989) 1: 139–150

Journal of Cryptology

Hyperelliptic Cryptosystems[1]

Neal Koblitz
Department of Mathematics GN-50, University of Washington,
Seattle, WA 98195, U.S.A.

Abstract. In this paper we discuss a source of finite abelian groups suitable for cryptosystems based on the presumed intractability of the discrete logarithm problem for these groups. They are the jacobians of hyperelliptic curves defined over finite fields. Special attention is given to curves defined over the field of two elements. Explicit formulas and examples are given, and the problem of finding groups of almost prime order is discussed.

Key words. Cryptosystem, Public key, Discrete logarithm, Hyperelliptic curve, Jacobian.

1. Introduction

In a finite abelian group, if an element was obtained as a multiple of another known element (the "base"), the discrete logarithm problem consists in finding the integer that was multiplied by the base to get the element. Whenever we have a finite abelian group for which the discrete log problem appears to be intractable, we can construct various public key cryptosystems in which taking large multiples of a group element is the trapdoor function. Such cryptosystems were first constructed from the multiplicative group of a finite field. However, because certain special techniques are available for attacking the discrete log problem in that case (especially when the field has characteristic 2, see [13]), it is worthwhile to study other sources of finite abelian groups.

In [8] we described how the group of points on an elliptic curve can be used to construct public key cryptosystems. The purpose of the present article is to discuss the more general class of groups obtained from the jacobians of hyperelliptic curves. These jacobian varieties seem to be a rich source of finite abelian groups for which, so far as is known, the discrete log problem is intractable. We pay special attention to the case when the ground field has characteristic 2, because arithmetic over such fields is particularly amenable to efficient implementation, and because it is in that case that the multiplicative group of the field does not provide secure cryptosystems unless the size of the field is extremely large, as explained in [13].

After giving the basic definitions of the group elements and the group addition in Section 2, we describe an algorithm for addition in Section 3. In Sections 2 and

[1] Date received: February 4, 1988. Date revised: September 28, 1988.

Hyper is (way) harder!

- Everything is much more complicated beyond genus 1: understanding, group law, arithmetic, point counting (i.e. finding strong instantiations), implementation, etc…

- The practical incentive for HECC in genus $g > 1$ boils down to

$$\#J_C(\mathbb{F}_q) = O(q^g)$$

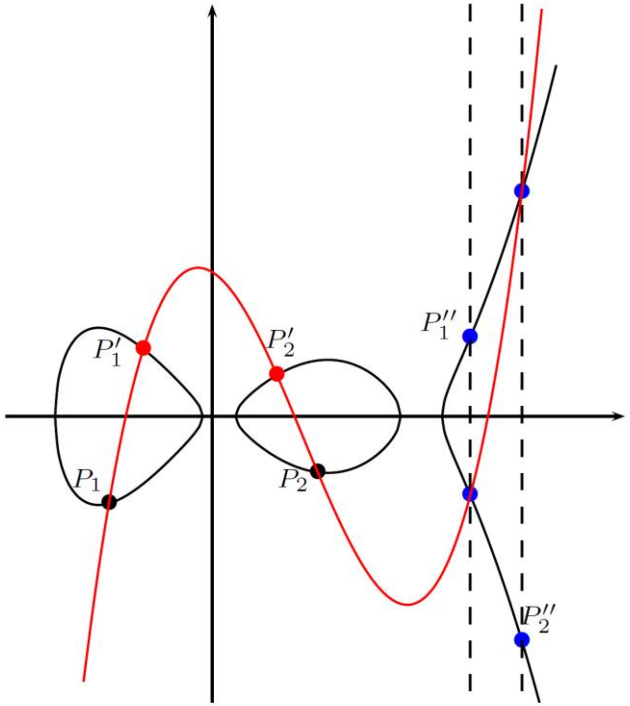
(see Ben's notes)

- E.g. $g = 2$ with $p \approx 2^n$ gets the same size cryptographic groups as $g = 1$ with $p \approx 2^{2n}$, i.e. we can use fields of half the size!

- But things no longer "easy" like it was in genus 1… must understand the language of divisors (see Ben's slides)

Genus 2 group law

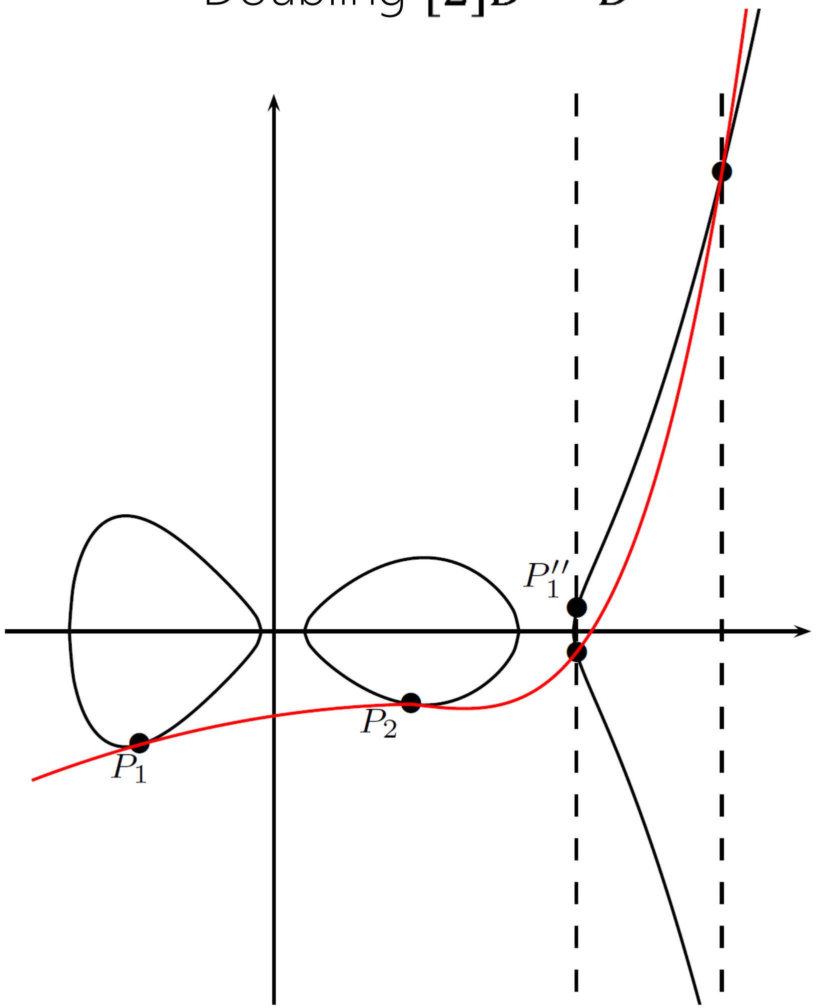
Addition $D \oplus D' = D''$



$$C/K : y^2 = x^5 + \cdots$$

$$D = (P_1) + (P_2) - 2(\infty)$$
$$D' = (P_1{}') + (P_2{}') - 2(\infty)$$
$$D'' = (P_1{}'') + (P_2{}'') - 2(\infty)$$

Doubling $[2]D = D''$

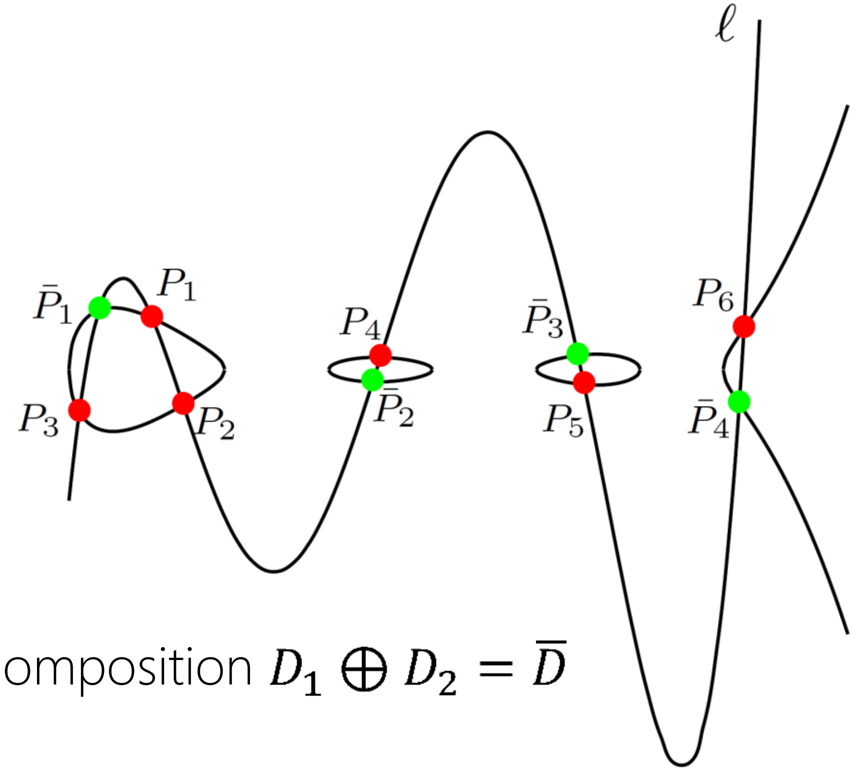


$$\mathrm{div}(\ell) = (P_1) + (P_2) + (P_1{}') + (P_2{}') + (\iota(P_1'')) + (\iota(P_2'')) - 6(\infty)$$

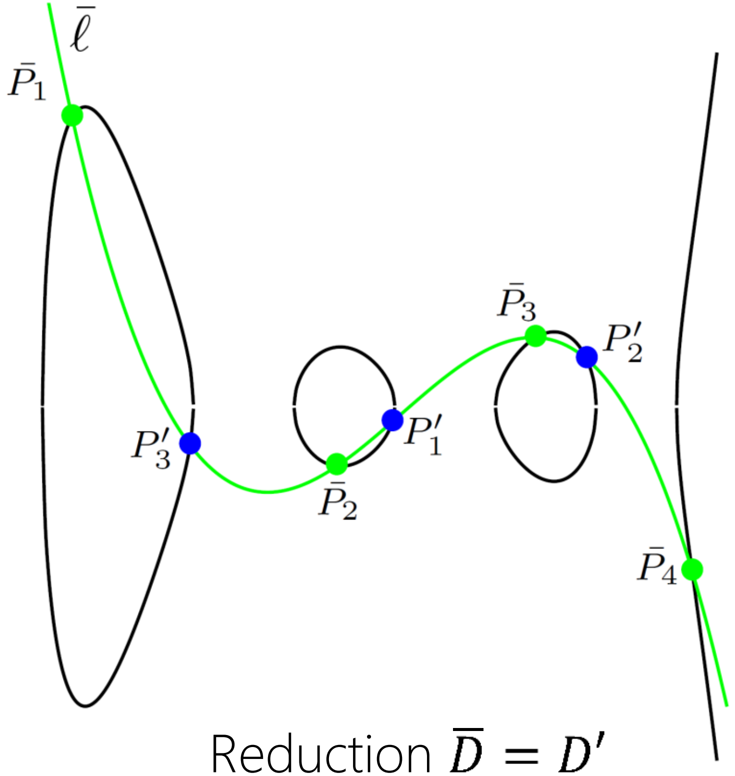
$$\mathrm{div}(\ell) = 2(P_1) + 2(P_2) + (\iota(P_1'')) + (\iota(P_2'')) - 6(\infty)$$

Genus 3 group law

$$C/K : y^2 = x^7 + \cdots$$



Composition $D_1 \oplus D_2 = \overline{D}$



Reduction $\overline{D} = D'$

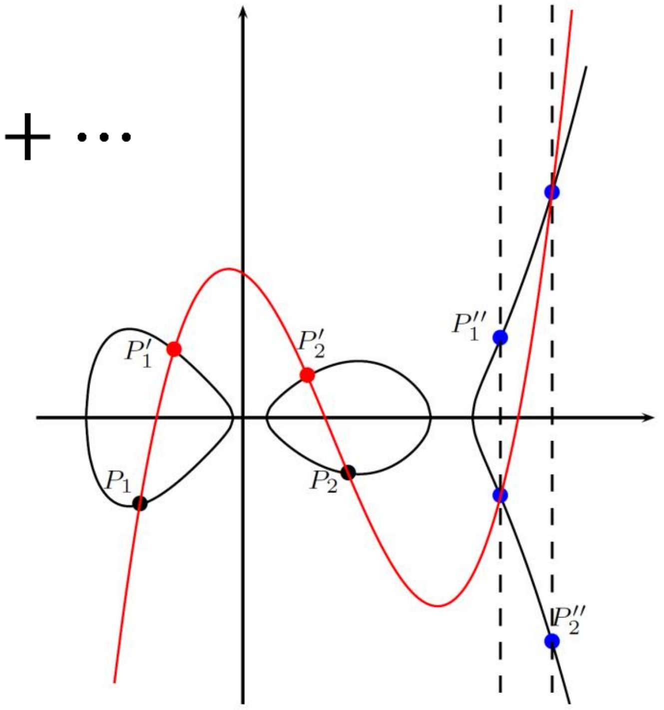
$$D_1 = (P_1) + (P_2) + (P_3) - 3(\infty)$$
$$D_2 = (P_4) + (P_5) + (P_6) - 3(\infty)$$
$$\overline{D} = (\overline{P}_1) + (\overline{P}_2) + (\overline{P}_3) + (\overline{P}_4) - 4(\infty)$$
$$D' = (P_1{}') + (P_2{}') + (P_3{}') - 3(\infty)$$

Mumford representation

$$D = \big(a(x), b(x)\big)$$
$$= (x^2 + a_1 x + a_0, b_1 x + b_0)$$

$$D' = \big(a'(x), b'(x)\big)$$
$$= (x^2 + a_1' x + a_0', b_1' x + b_0')$$

$$C/K : y^2 = x^5 + \cdots$$



$\boxed{\text{Addition } D \oplus D' = D''}$

1. Compute cubic $\ell(x) = l_3 x^3 + \cdots + l_0$ such that $l(x) \equiv b(x) \bmod a(x)$ and $l'(x) \equiv b'(x) \bmod a'(x)$
2. Solve $l(x)^2 - \big(x^5 + \cdots\big) = a(x)a'(x)a''(x)$ for $a''(x)$
3. Compute $b''(x) \equiv -l(x) \bmod a''(x)$
4. Output $D'' = (a''(x), b''(x))$

$$D = (P_1) + (P_2) - 2(\infty)$$
$$D' = (P_1{}') + (P_2{}') - 2(\infty)$$
$$D'' = (P_1{}'') + (P_2{}'') - 2(\infty)$$

Question

Why is it computationally preferable to work in Mumford coordinates rather than, say, using the coordinates of the points themselves?

Scalar multiplications via double-and-add

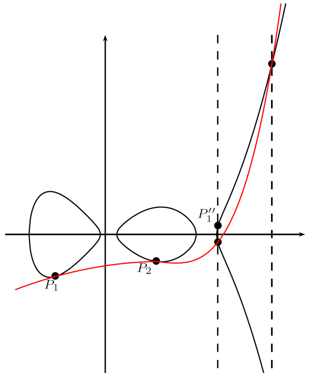
How to (naively) compute $\boldsymbol{k, Q} \mapsto [\boldsymbol{k}]\boldsymbol{Q}$?

$$P \leftarrow Q$$

$$k = (k_n, k_{n-1}, \ldots, k_0)_2$$

for i from $n-1$ downto 0 do

$$P \leftarrow [2]P$$

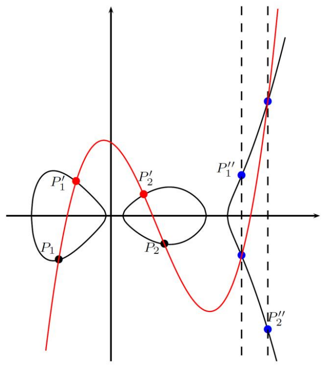


DBL

if $k_i = 1$ then

$$P \leftarrow P \oplus Q$$

end if



ADD

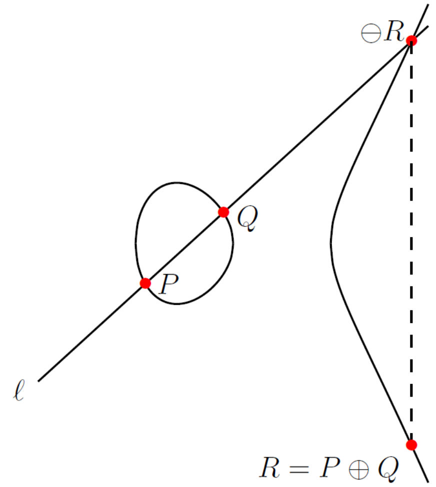
end for

return $P\ (= [k]Q)$

Trade-offs for prime order Jacobians...

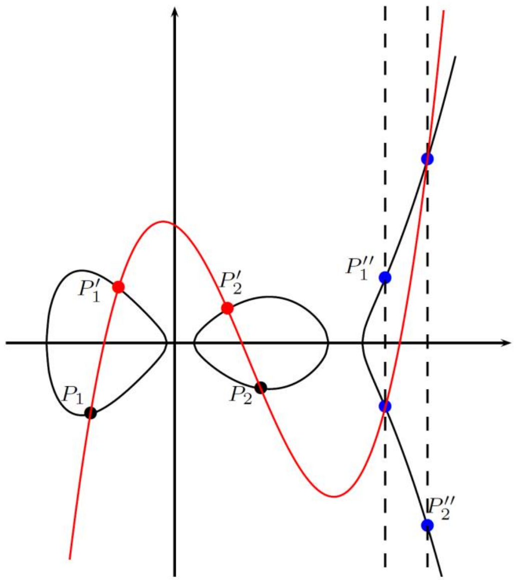
- NIST (elliptic) Curve P-256

  DBL ≈ 8M
  ADD ≈ 16M



- Hyperelliptic P-128

  DBL ≈ 35M
  ADD ≈ 63M

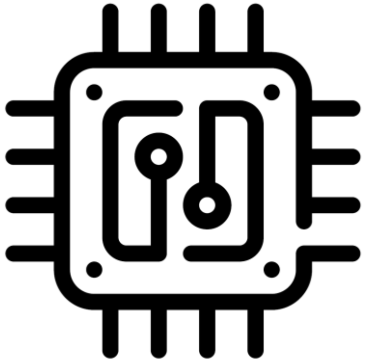


Part 1: Why use curves at all?

Part 2: Why go beyond genus 1?

Part 3: Why stop at genus 2?

Part 4: Why Kummer surfaces?

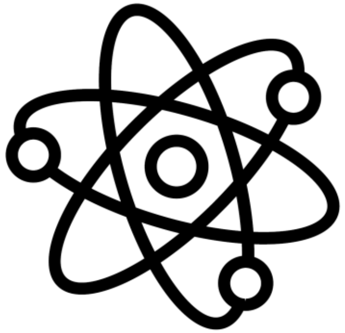


Part 5: Why are we interested in isogenies?

Part 6: Why is genus 2 (even more) promising here?

Part 7: Why stop at genus 2?

Part 8: Why not use hyperelliptic curves in genus 1?



Index calculus attacks genus $g \geq 3$

- Most reduced elements in $\mathbf{Pic}^0(C)$ look like

$$(P_1) + (P_2) + (P_3) - 3(\infty)$$

- But some "special" divisors look like $(Q_1) + (Q_2) \; - 2(\infty)$, and some look like $(R_1) - (\infty)$

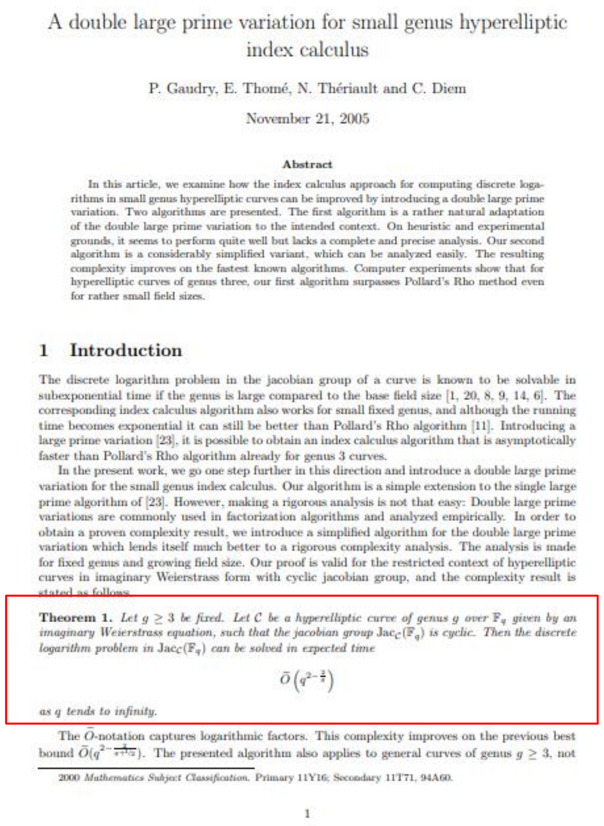
- Unlike the elliptic curve case, we now have a notion of "smallness" that allows a **factor base** for index calculus

- Compute multiples of DLP inputs until they "decompose" into special divisors and split over the factor base, i.e. $D = \big(a(x), b(x)\big) = (x^3 + a_2 x^2 + a_1 x + a_0, b_2 x^2 + b_1 x + b_0)$ where

$$a(x) = \left(x - x_{R_1}\right)\left(x - x_{R_2}\right)\left(x - x_{R_3}\right)$$

- Then $D = D_1 + D_2 + D_3$ where $D_1 = (R_1) - (\infty)$, $D_2 = (R_2) - (\infty)$, $D_3 = (R_3) - (\infty)$.

Index calculus attacks genus $g \geq 3$



A double large prime variation for small genus hyperelliptic index calculus

P. Gaudry, E. Thomé, N. Thériault and C. Diem

November 21, 2005

Abstract

In this article, we examine how the index calculus approach for computing discrete logarithms in small genus hyperelliptic curves can be improved by introducing a double large prime variation. Two algorithms are presented. The first algorithm is a rather natural adaptation of the double large prime variation to the intended context. On heuristic and experimental grounds, it seems to perform quite well but lacks a complete and precise analysis. Our second algorithm is a considerably simplified variant, which can be analyzed easily. The resulting complexity improves on the fastest known algorithms. Computer experiments show that for hyperelliptic curves of genus three, our first algorithm surpasses Pollard's Rho method even for rather small field sizes.

1 Introduction

The discrete logarithm problem in the jacobian group of a curve is known to be solvable in subexponential time if the genus is large compared to the base field size [1, 20, 8, 9, 14, 6]. The corresponding index calculus algorithm also works for small fixed genus, and although the running time becomes exponential it can still be better than Pollard's Rho algorithm [11]. Introducing a large prime variation [23], it is possible to obtain an index calculus algorithm that is asymptotically faster than Pollard's Rho algorithm already for genus 3 curves.

In the present work, we go one step further in this direction and introduce a double large prime variation for the small genus index calculus. Our algorithm is a simple extension to the single large prime algorithm of [23]. However, making a rigorous analysis is not that easy: Double large prime variations are commonly used in factorization algorithms and analyzed empirically. In order to obtain a proven complexity result, we introduce a simplified algorithm for the double large prime variation which lends itself much better to a rigorous complexity analysis. The analysis is made for fixed genus and growing field size. Our proof is valid for the restricted context of hyperelliptic curves in imaginary Weierstrass form with cyclic jacobian group, and the complexity result is stated as follows.

Theorem 1. *Let $g \geq 3$ be fixed. Let $\mathcal{C}$ be a hyperelliptic curve of genus g over $\mathbb{F}_q$ given by an imaginary Weierstrass equation, such that the jacobian group $\mathrm{Jac}_{\mathcal{C}}(\mathbb{F}_q)$ is cyclic. Then the discrete logarithm problem in $\mathrm{Jac}_{\mathcal{C}}(\mathbb{F}_q)$ can be solved in expected time*

$$\tilde{O}\left(q^{2-\frac{2}{g}}\right)$$

as q tends to infinity.

The $\tilde{O}$-notation captures logarithmic factors. This complexity improves on the previous best bound $\tilde{O}(q^{2-\frac{2}{g+1/2}})$. The presented algorithm also applies to general curves of genus $g \geq 3$, not

2000 *Mathematics Subject Classification.* Primary 11Y16; Secondary 11T71, 94A60.

1

Theorem 1. *Let $g \geq 3$ be fixed. Let $\mathcal{C}$ be a hyperelliptic curve of genus g over $\mathbb{F}_q$ given by an imaginary Weierstrass equation, such that the jacobian group $\mathrm{Jac}_{\mathcal{C}}(\mathbb{F}_q)$ is cyclic. Then the discrete logarithm problem in $\mathrm{Jac}_{\mathcal{C}}(\mathbb{F}_q)$ can be solved in expected time*

$$\tilde{O}\left(q^{2-\frac{2}{g}}\right)$$

as q tends to infinity.

- $\tilde{O}(q^{2-2/g})$ not a theoretical deal-breaker (could scale parameters up), but trade-offs become unfavorable and non-generic attacks scared people away from $g > 2$

paper

Question

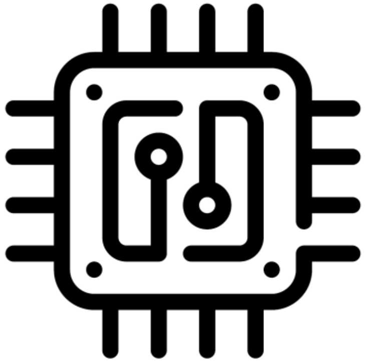
Why did the theorem on the previous page start at $\boldsymbol{g = 3}$? We handwaved that there's no special/small divisors in $\boldsymbol{g = 1}$, but there are small divisors that could be used as a factor base in genus 2! So why does index calculus not also (buzz)kill $\boldsymbol{g = 2}$?

Part 1: Why use curves at all?

Part 2: Why go beyond genus 1?

Part 3: Why stop at genus 2?

Part 4: Why Kummer surfaces?

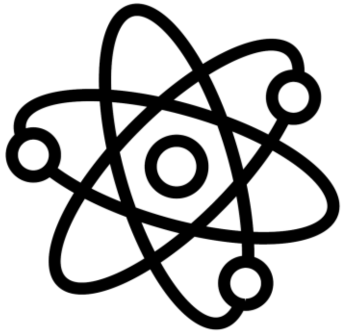


Part 5: Why are we interested in isogenies?

Part 6: Why is genus 2 (even more) promising here?

Part 7: Why stop at genus 2?

Part 8: Why not use hyperelliptic curves in genus 1?



Miller's seminal sign-off...

Use of Elliptic Curves in Cryptography

Victor S. Miller

Exploratory Computer Science, IBM Research, P.O. Box 218, Yorktown Heights, NY 10598

ABSTRACT

We discuss the use of elliptic curves in cryptography. In particular, we propose an analogue of the Diffie-Hellmann key exchange protocol which appears to be immune from attacks of the style of Western, Miller, and Adleman. With the current bounds for infeasible attack, it appears to be about 20% faster than the Diffie-Hellmann scheme over GF(p). As computational power grows, this disparity should get rapidly bigger.

H.C. Williams (Ed.): Advances in Cryptology - CRYPTO '85, LNCS 218, pp. 417-426, 1986.

In order to be secure from the Pohlig-Hellmann (or Pollard) algorithm, it is necessary that N_p, the number of points of E in $\mathbf{F}_p$ have a prime factor $> p^\alpha$, for α as close to 1 as possible. This is made possible by the algorithm of Schoof [19], which calculates N_p in time polynomial in $\log p$. In general it is not hard to find such good p. Theoretically, the best result known is one of Fouvry [20]: For any fixed non-zero integer a, a positive proportion of primes p have the property that the largest prime factor of $p + a$ is $\geq p^\delta$ where $\delta = 0.6687$.

Instead of using the Schoof algorithm, when searching for a good p, I have taken the following approach: Choose the curve to be:

$$E:\ y^2 = x^3 - ax$$

where a is not a perfect square. This curve has complex multiplication by $\sqrt{-1}$, and there is an exact formula for N_p (see [10]). In the case $p = 3 \bmod 4$ we have $N_p = p + 1$. This is the so-called "supersingular" case. In this case we know even more. It is well known (see [1]) that any field containing the coordinates of all points of order l also contains the l-th roots of unity. This shows that a necessary condition for group of point over $\mathbf{F}_p$ to contain a subgroup isomorphic to $\mathbf{Z}/l\mathbf{Z} \times \mathbf{Z}/l\mathbf{Z}$ is that $l \mid p - 1$. Because the number of points in the supersingular case is $p + 1$ we have 2 as the only possibility for l. But, in our case, this happens if and only if, a is a quadratic residue modulo p. To sum up, in the case above the group of points modulo p is of order $p + 1$, cyclic in the case $(a/p) = -1$, and a product of a cyclic group of order 2 and a cyclic group of order $(p + 1)/2$ when $(a/p) = 1$.

The above choice of curve was taken for convenience in calculation. However, it may be prudent to avoid curves with complex multiplication because the extra structure of these curves might somehow be used to give a better algorithm.

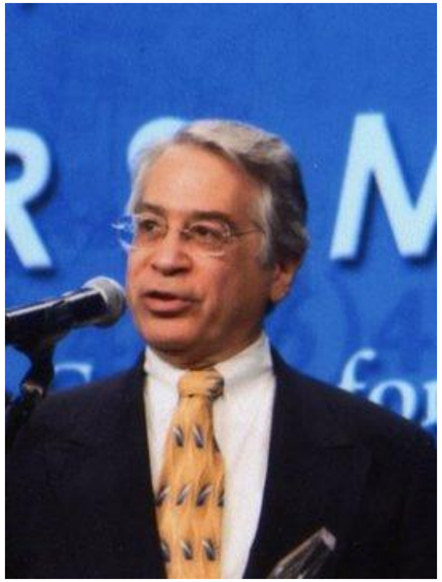
Finally, it should be remarked, that even though we have phrased everything in terms of points on an elliptic curve, that, for the key exchange protocol (and other uses as one-way functions), that only the x-coordinate needs to be transmitted. The formulas for multiples of a point cited in the first section make it clear that the x-coordinate of a multiple depends only on the x-coordinate of the original point.

BIBLIOGRAPHY [1] Lang, Serge, Elliptic Curves: Diophantine Analysis, Springer-Verlag New York, 1978.

[2] Lenstra, H. W., Letter to A. M. Odlyzko.

[3] Diffie, W. and Hellman M., New Directions in Cryptography, IEEE Trans. Inform. Theory, IT-22 (1976), 644-654.

[4] Western, A. E., and Miller, J. C. P., *Table of Indices and Primitive Roots*, Royal Society Mathematical Tables, vol. 9, Cambridge Univ. Press, 1968.

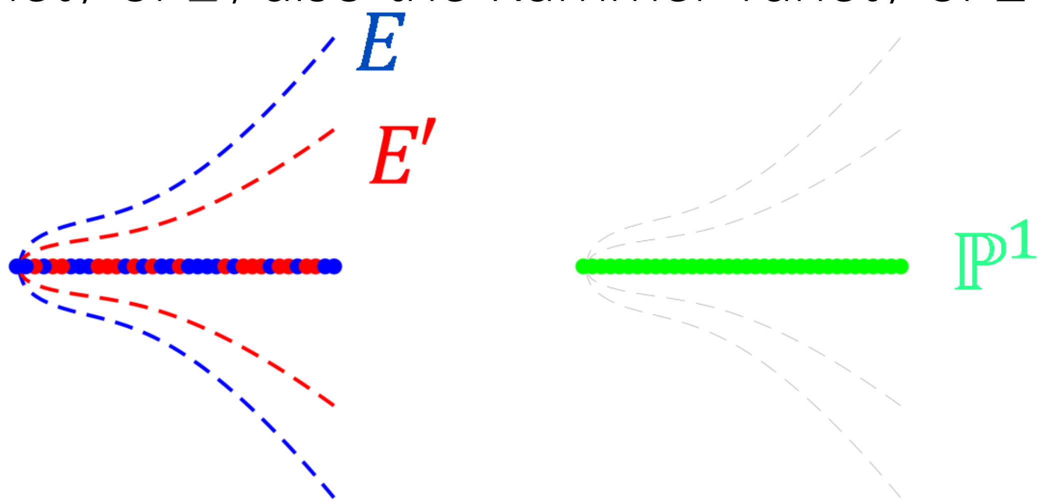


$$x([n]P) = f(x(P))$$

Finally, it should be remarked, that even though we have phrased everything in terms of points on an elliptic curve, that, for the key exchange protocol (and other uses as one-way functions), that only the x-coordinate needs to be transmitted. The formulas for multiples of a point cited in the first section make it clear that the x-coordinate of a multiple depends only on the x-coordinate of the original point.

Kummer lines in genus 1

- Recall (from Ben) that the Kummer variety of an abelian variety A is its quotient by $\ominus$
- For $E\colon y^2 = x^3 + \cdots$, we have $\ominus\,(x, y) = (x, -y)$, so $P \mapsto x(P)$ is the quotient $E/\langle\ominus\rangle$
- $\mathbb{P}^1$ is the Kummer variety of E, also the Kummer variety of E'



- E.g., every $x \in \mathbb{F}_q$ on either (or both) E or $E' = By^2 = x^3 + \cdots$, $B \notin \square$

Montgomery's fast differential arithmetic

$$E/\mathbb{F}_p : y^2 = x^3 + Ax^2 + x$$

paper

- drop the y-coordinate, and work with x-only.
- projectively, work with $(X : Z) \in \mathbb{P}^1$ instead of $(X : Y : Z) \in \mathbb{P}^2$
- But (pseudo-)addition of $\mathrm{x}(P)$ and $\mathrm{x}(Q)$ requires $x(Q \ominus P)$

Extremely fast pseudo-doubling: xDBL

$$X_{[2]P} = (X_P + Z_P)^2 (X_P - Z_P)^2$$

$$Z_{[2]P} = 4X_P Z_P ((X_P - Z_P)^2 + (A+2) X_P Z_P)$$

$2M + 2S$

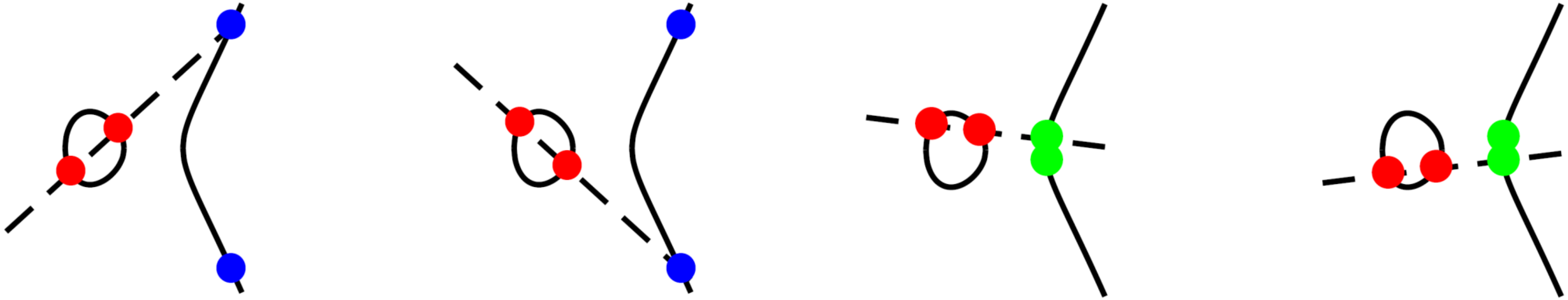
Extremely fast pseudo-addition: xADD

$$X_{P+Q} = Z_{P-Q} \left[(X_P - Z_P)(X_Q + Z_Q) + (X_P + Z_P)(X_Q - Z_Q) \right]^2$$

$$Z_{P+Q} = X_{P-Q} \left[(X_P - Z_P)(X_Q + Z_Q) - (X_P + Z_P)(X_Q - Z_Q) \right]^2$$

$4M + 2S$

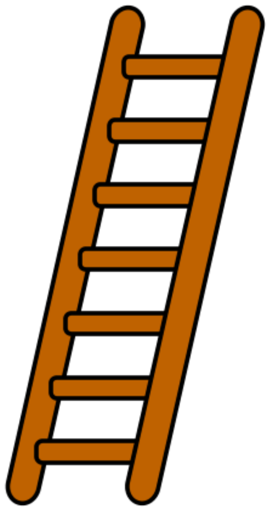
Differential additions and the Montgomery ladder



- Given only the x-coordinates of two points, the x-coordinate of their sum can be two possibilities
- Inputting the x-coordinate of the *difference* resolves ambiguity
- The (ingenious!) Montgomery ladder fixes all *differences* as the input point: in $k, x(P) \mapsto x([k]P)$, every $\mathbf{xADD}$ is of the form
$$\mathrm{xADD}(x([n+1]P), x([n]P), x(P))$$
- We carry two multiples of P "up the ladder": $x(Q)$ and $x(Q \oplus P)$
- At i^{th}step: compute $x([2]Q \oplus P) = xADD(x(Q \oplus P), x(Q), x(P))$
- At i^{th}step: pseudo-double ($\mathbf{xDBL}$) one of them depending on k_i

Fast, compact, simple, safer Diffie-Hellman

- Write $k = \sum_{i=0}^{\ell-1} k_i 2^i$ with $k_{\ell-1} = 1$ and $P = (x_P, y_P)$ in $E[n]$ (e.g., on Curve25519 or Goldilocks)



$(x_0, x_1) \leftarrow (\mathrm{xDBL}(x_P), x_P)$
for $i = \ell - 2$ downto 0 do
  $(x_0, x_1) \leftarrow \mathrm{cSWAP}\big((k_{i+1} \otimes k_i), (x_0, x_1)\big)$
  $(x_0, x_1) \leftarrow (\mathrm{xDBL}(x_0), \mathrm{xADD}(x_0, x_1, x_P))$
end for
$(x_0, x_1) \leftarrow \mathrm{cSWAP}\big(k_0, (x_0, x_1)\big)$
return x_0 $(= x_{[k]P})$

Inherently uniform, much easier to implement in constant-time

- x-only Diffie-Hellman (Miller'85): $x([ab]P) = x\big([a]([b]P)\big) = x([b]([a]P))$

see https://tools.ietf.org/html/rfc7748
(Elliptic curves for security)

Kummer surfaces

- In genus 1, we saw that working with $E/\langle \ominus \rangle$ can be much simpler/faster/easier than working with E
- In genus 2, the difference between $\mathbf{Jac}(C)$ and $\mathbf{Jac}(C)/\langle \ominus \rangle$ is way more drastic…

$$C : y^2 = f_6 x^6 + f_5 x^5 + \cdots + f_0$$

$\mathbf{Jac}(C)$ embeds into $\mathbb{P}^{15}$: 72 equations in 16 variables!!! (see here) 🤮

BUT….

$\mathbf{Jac}(C)/\langle \ominus \rangle$ embeds into $\mathbb{P}^3$: 1 equation in 4 variables!!! 🤩

Kummer surface arithmetic

paper paper paper

- In genus 1, we can use the ``general" Kummer line $E/\langle \ominus \rangle$ corresponding to $E: y^2 = x^3 + ax + b$ (a la Brier-Joye), but it's faster/simpler to work with the Montgomery x-line. The only restriction is that this forces some rational points of small order

- In genus 2, there is somewhat of an analogue. We can use the general Kummer surface (a la Flynn), which has no restrictions but is slow and bulky (see here and here), or if we insist that $\mathrm{Jac}_K(C)$ has full rational 2-torsion, we can use Kummer surfaces that arise from the theory of Theta functions

$$K: \;\; E^2 \cdot (XYZT) = \left(X^2 + Y^2 + Z^2 + T^2 - F(XT + YZ) - G(XZ + YT) - H(XY + ZT)\right)^2$$

- Points are $(X : Y : Z : T) \in \mathbb{P}^3$, and the doubling and differential addition formulae are beautiful!

Algorithm 1 Doubling on a Kummer surface

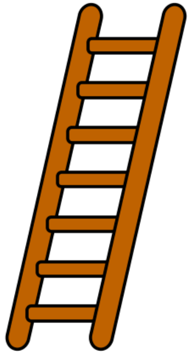
Input: a point $P = (X : Y : Z : T)$ on $\mathcal{K}_{(\alpha:\beta:\gamma:\delta)}$.
Output: the point $[2]P = (X_2 : Y_2 : Z_2 : T_2)$.

$$\begin{aligned}
X' &= (X+Y+Z+T)^2 \tfrac{1}{A}, & Y' &= (X+Y-Z-T)^2 \tfrac{1}{B}, \\
Z' &= (X-Y+Z-T)^2 \tfrac{1}{C}, & T' &= (X-Y-Z+T)^2 \tfrac{1}{D}, \\
X_2 &= (X'+Y'+Z'+T')^2 \tfrac{1}{\alpha}, & Y_2 &= (X'+Y'-Z'-T')^2 \tfrac{1}{\beta}, \\
Z_2 &= (X'-Y'+Z'-T')^2 \tfrac{1}{\gamma}, & T_2 &= (X'-Y'-Z'+T')^2 \tfrac{1}{\delta}
\end{aligned}$$

Algorithm 2 Pseudo-addition on a Kummer surface

Input: two points $P = (X : Y : Z : T)$ and $Q = (\underline{X} : \underline{Y} : \underline{Z} : \underline{T})$ on $\mathcal{K}_{(\alpha:\beta:\gamma:\delta)}$, and the point $R = (\bar{X} : \bar{Y} : \bar{Z} : \bar{T})$ equal to $P - Q$ such that $\bar{X}\bar{Y}\bar{Z}\bar{T} \neq 0$.
Output: the point $P + Q = (x : y : z : t)$.

$$\begin{aligned}
X' &= (X+Y+Z+T)(\underline{X}+\underline{Y}+\underline{Z}+\underline{T}) \tfrac{1}{A}, \\
Y' &= (X+Y-Z-T)(\underline{X}+\underline{Y}-\underline{Z}-\underline{T}) \tfrac{1}{B}, \\
Z' &= (X-Y+Z-T)(\underline{X}-\underline{Y}+\underline{Z}-\underline{T}) \tfrac{1}{C}, \\
T' &= (X-Y-Z+T)(\underline{X}-\underline{Y}-\underline{Z}+\underline{T}) \tfrac{1}{D} \\
x &= (X'+Y'+Z'+T')^2 \tfrac{1}{\bar{X}}, \\
y &= (X'+Y'-Z'-T')^2 \tfrac{1}{\bar{Y}}, \\
z &= (X'-Y'+Z'-T')^2 \tfrac{1}{\bar{Z}}, \\
t &= (X'-Y'-Z'+T')^2 \tfrac{1}{\bar{T}}
\end{aligned}$$



Kummer line vs. Kummer surface

	full group arith.		Kummer arith.
	DBL	ADD	ladder step
genus 1	8M	16M	10M
genus 2	35M	63M	25M

$\approx$

- Scalar multiplications on the Gaudry-Schost fast Kummer surface over $p = 2^{127} - 1$ solidly outperform ($\approx$ **2x**) those on Bernstein's Curve25519 (see eBACS)

- Summary: the state-of-the-art in conservative prime field Diffie-Hellman in genus 2 is significantly faster than that in genus 1

Question

The previous comparison only talked about speed, but what about key sizes? How does genus 2 compare to genus 1 in bandwidth, in both the case of uncompressed and compressed public keys?

Question

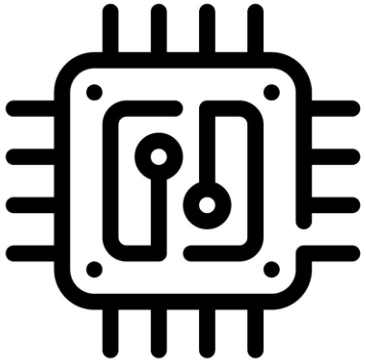
If the state-of-the-art in genus 2 prime field Diffie-Hellman performs roughly twice as fast as that of genus 1, and if index calculus fails against genus 1 and genus 2, then why isn't KummerDH a standard?

Part 1: Why use curves at all?

Part 2: Why go beyond genus 1?

Part 3: Why stop at genus 2?

Part 4: Why Kummer surfaces?

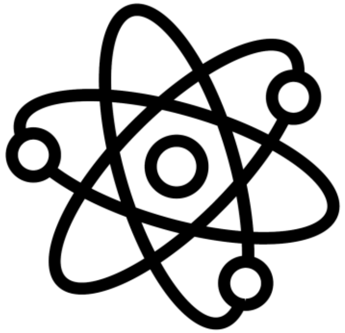


Part 5: Why are we interested in isogenies?

Part 6: Why is genus 2 (even more) promising here?

Part 7: Why stop at genus 2?

Part 8: Why not use hyperelliptic curves in genus 1?



Diffie-Hellman instantiations

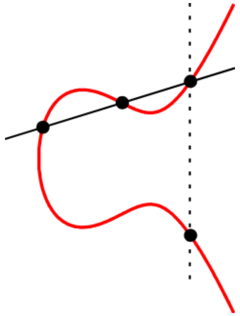
paper

$\mathbb{Z}_q^*$

$g^a \bmod q$

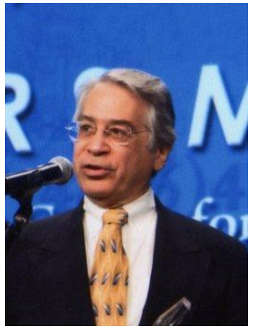
$g^b \bmod q$

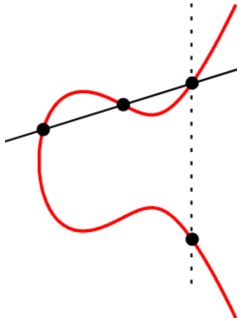
$\mathbb{Z}_q^*$

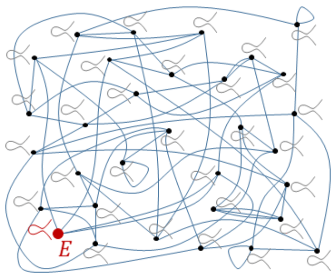


$[a]P$

$[b]P$

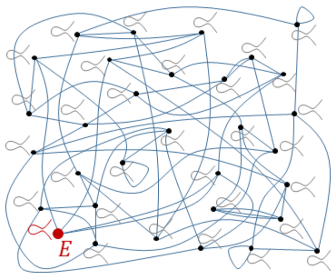






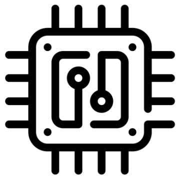
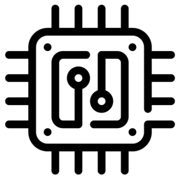
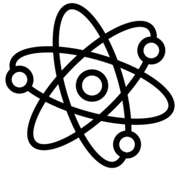
$\phi_A(E)$

$\phi_B(E)$

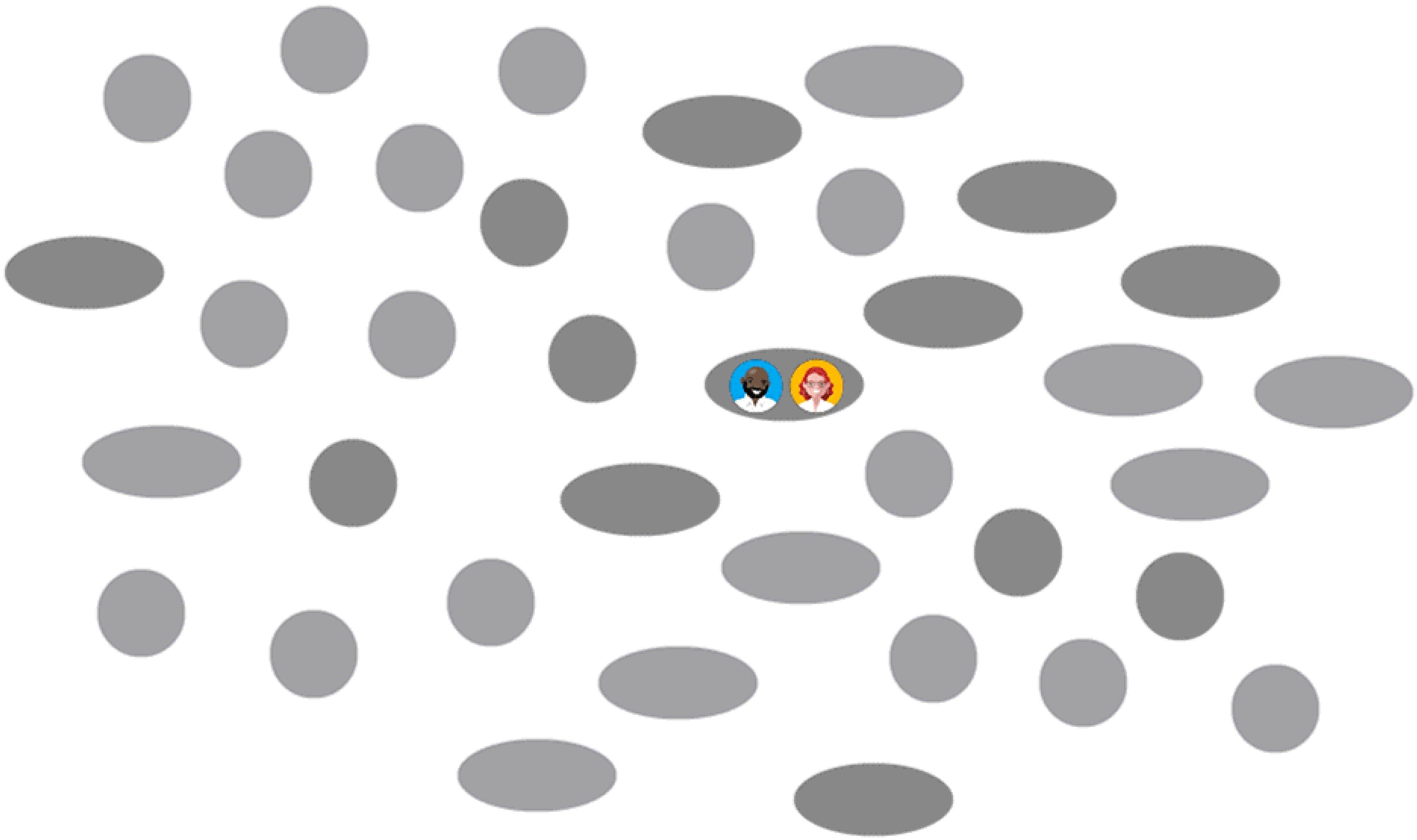


Diffie-Hellman instantiations

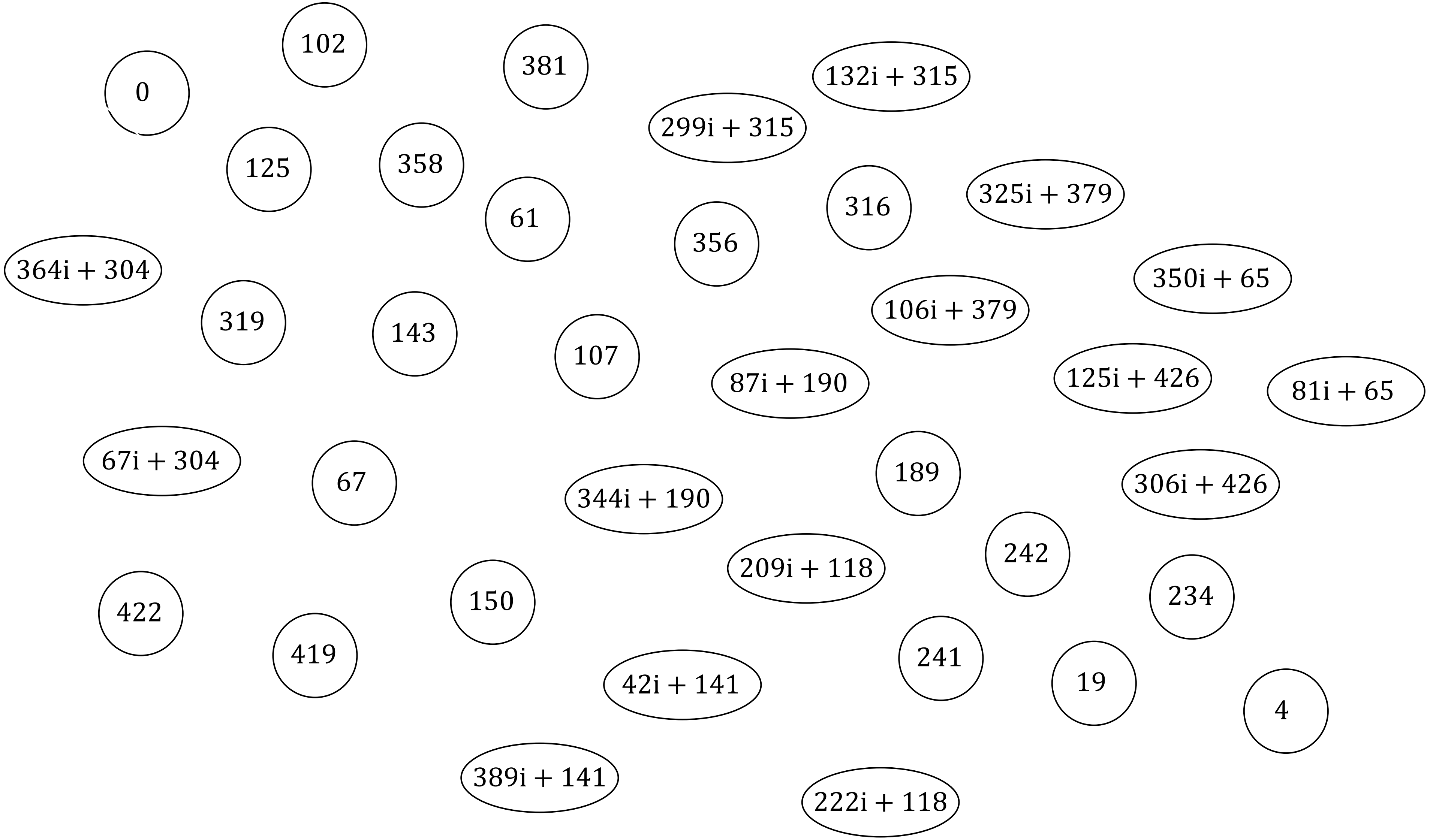
	DH	ECDH	SIDH
Elements	integers g modulo prime	points P in curve group	curves E in isogeny class
Secrets	exponents x	scalars k	isogenies ϕ
computations	$g, x \mapsto g^x$	$P, k \mapsto [k]P$	$E, \phi \mapsto \phi(E)$
hard problem	given g, g^x find x	given $P, [k]P$ find k	given $E, \phi(E)$ find ϕ

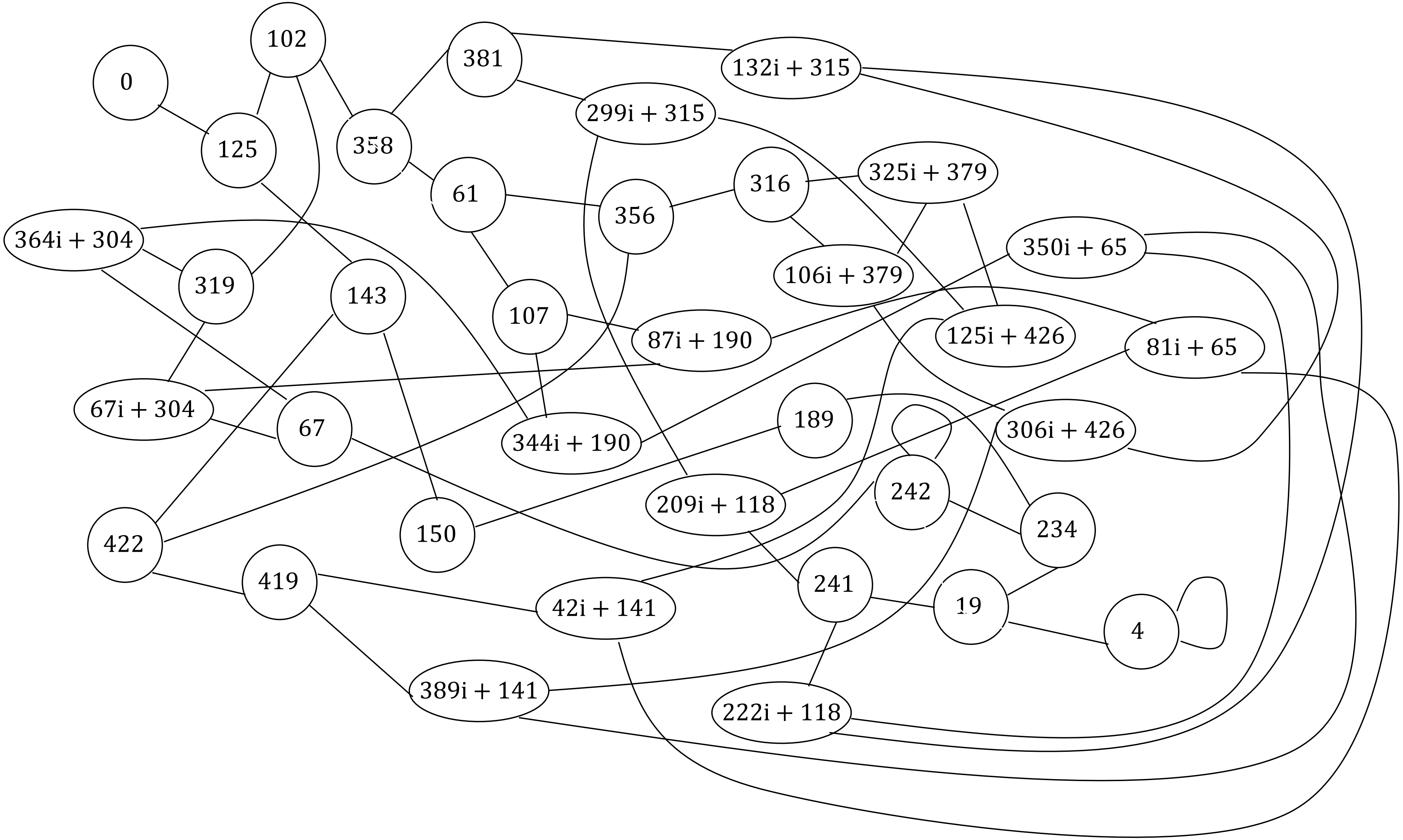
Diffie-Hellman instantiations

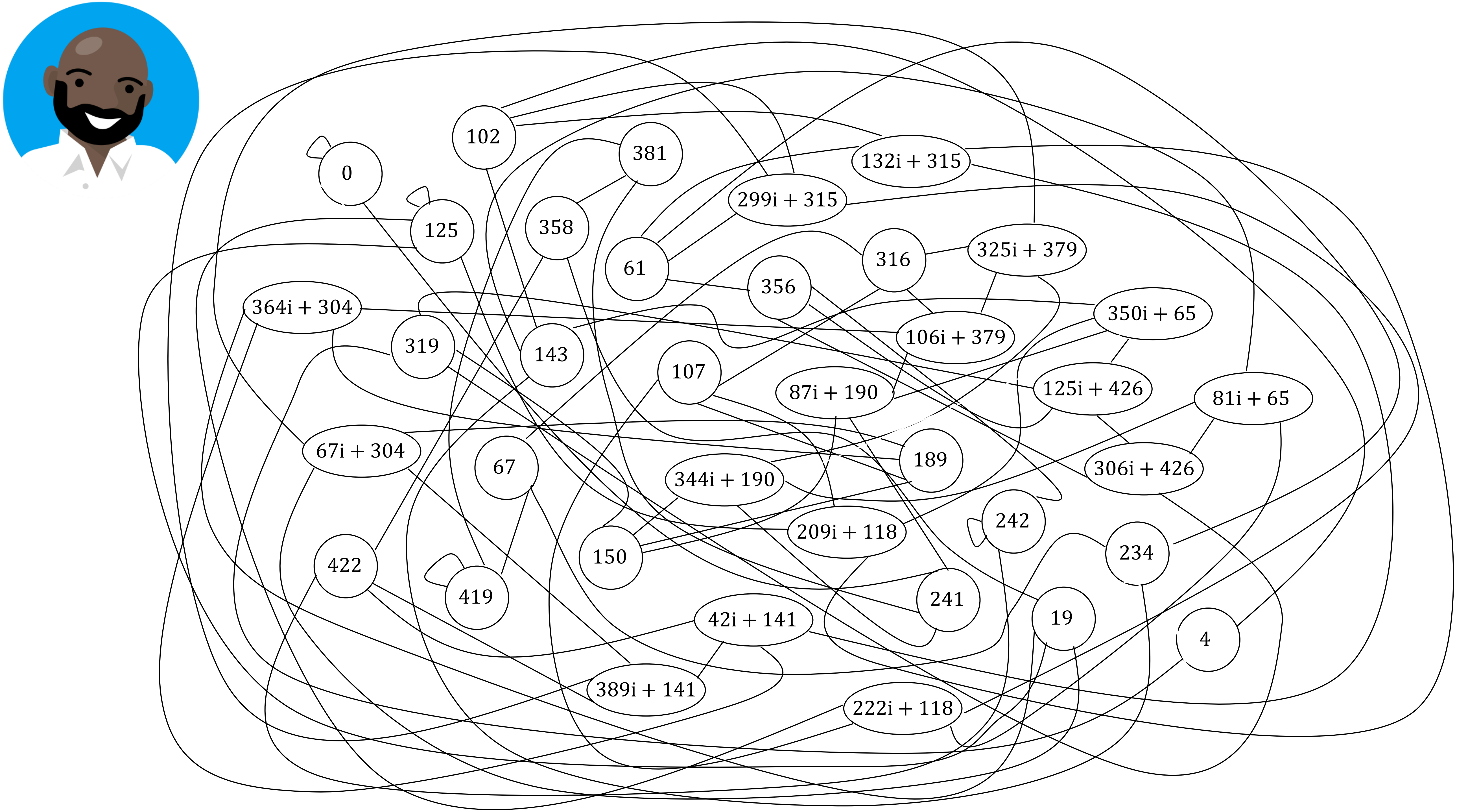


e.g. supersingular isogeny graph – the nodes



$p := 431$: there are 37 supersingular j's (all over $\mathbb{F}_{p^2} := \mathbb{F}_p(i)$, $i^2 + 1 = 0$)


0
102
381
132i + 315
299i + 315
125
358
61
356
316
325i + 379
364i + 304
319
143
107
106i + 379
350i + 65
87i + 190
125i + 426
81i + 65
67i + 304
67
344i + 190
189
306i + 426
242
209i + 118
234
422
150
419
42i + 141
241
19
4
389i + 141
222i + 118



102
381
0
132i + 315
299i + 315
125
358
61
316
325i + 379
356
364i + 304
350i + 65
319
143
106i + 379
107
87i + 190
125i + 426
81i + 65
67i + 304
67
189
344i + 190
306i + 426
242
209i + 118
234
150
422
419
241
19
42i + 141
4
389i + 141
222i + 118

Explicit formulas

$$[2] : E_a \to E_a , \qquad x \mapsto \frac{\left(x^2-1\right)^2}{4x(x-\alpha)(x-1/\alpha)}$$

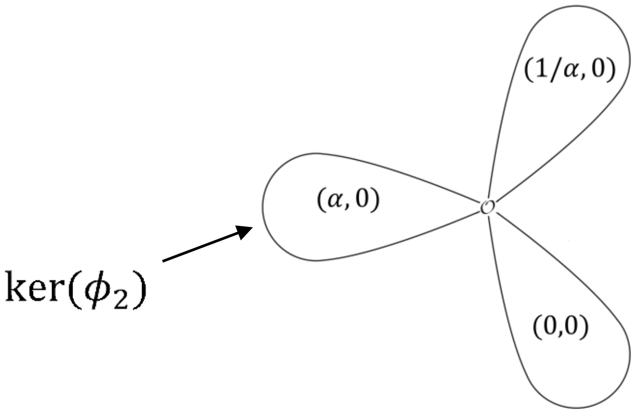
$$\phi_2 : E_a \to E_{a'} , \qquad x \mapsto x \cdot \left(\frac{\alpha x - 1}{x - \alpha}\right)$$

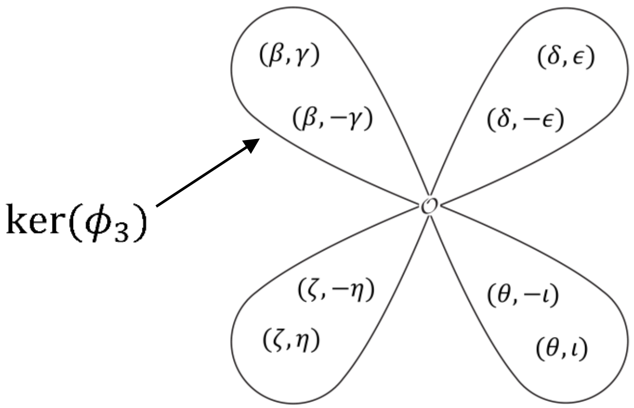
$$a' = 2\left(1 - 2\alpha^2\right)$$

$$[3] : E_a \to E_a, \qquad x \mapsto \frac{\left(x^4 - 6x^2 - 4ax - 3\right)^2 x}{(3x^4 + 4ax^3 + 6x^2 - 1)^2}$$

$$\phi_3 : E_a \to E_{a'} , \qquad x \mapsto x \cdot \left(\frac{\beta x - 1}{x - \beta}\right)^2$$

$$a' = \left(a\beta - 6\beta^2 + 6\right)\beta$$





SIKE

	prime p	PK (bytes)	Clock cycles to compute ϕ ($\times 10^6$) i7-6700 Skylake	
				
toy example	$2^4 3^3 - 1$	7	ϵ	ϵ'
SIKEp434	$2^{216} 3^{137} - 1$	330	92	98
SIKEp503	$2^{250} 3^{159} - 1$	378	142	151
SIKEp610	$2^{305} 3^{192} - 1$	462	295	297
SIKEp751	$2^{372} 3^{239} - 1$	564	468	503

https://sike.org/

https://www.microsoft.com/en-us/research/project/sike/

https://csrc.nist.gov/projects/post-quantum-cryptography

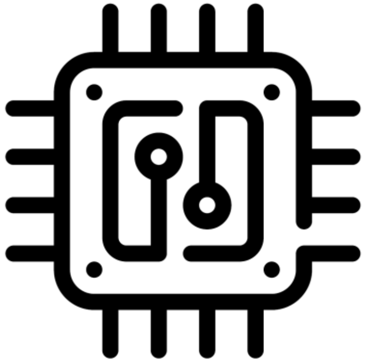
The case for SIKE…

paper

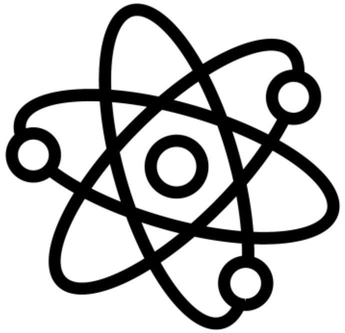
1. A decade unscathed
2. The rise and rise of classical hardness
3. Quantum computers don't help
4. Concrete cryptanalytic clarity
5. Side-channel security
6. The efficiency drawback
7. Happy hybrids
8. Other avenues of attack
9. Elegance
10. The $IKE challenges

SIKE only isogeny-based candidate…

Part 1: Why use curves at all?
Part 2: Why go beyond genus 1?
Part 3: Why stop at genus 2?
Part 4: Why Kummer surfaces?



Part 5: Why are we interested in isogenies?
Part 6: Why is genus 2 (even more) promising here?
Part 7: Why stop at genus 2?
Part 8: Why not use hyperelliptic curves in genus 1?



Genus 2 isogeny-based cryptography...

Genus Two Isogeny Cryptography

E.V. Flynn[1] and Yan Bo Ti[2]

[1] Mathematical Institute, Oxford University, UK. flynn@maths.ox.ac.uk
[2] Mathematics Department, University of Auckland, NZ. yanbo.ti@gmail.com

Abstract. We study (ℓ, ℓ)-isogeny graphs of principally polarised supersingular abelian surfaces (PPSSAS). The (ℓ, ℓ)-isogeny graph has cycles of small length that can be used to break the collision resistance assumption of the genus two isogeny hash function suggested by Takashima. Algorithms for computing $(2, 2)$-isogenies on the level of Jacobians and $(3, 3)$-isogenies on the level of Kummers are used to develop a genus two version of the supersingular isogeny Diffie–Hellman protocol of Jao and de Feo. The genus two isogeny Diffie–Hellman protocol achieves the same level of security as SIDH but uses a prime with a third of the bit length.

Keywords: Post-quantum cryptography · Isogeny-based cryptography · Cryptanalysis · Key exchange · Hash function

1 Introduction

Isogeny-based cryptography involves the study of isogenies between abelian varieties. The first proposal was an unpublished manuscript of Couveignes [6] that outlined a key-exchange algorithm set in the isogeny graph of elliptic curves. This was rediscovered by Rostovtsev and Stolbunov [18]. A hash function was developed by Charles, Goren and Lauter [4] that uses the input to the hash to generate a path in the isogeny graph and outputs the end point of the path. Next in the line of invention is the Jao–de Feo cryptosystem [12] which relies on the difficulty of finding isogenies with a given degree between supersingular elliptic curves. A key exchange protocol, called the Supersingular Isogeny Diffie–Hellman key exchange (SIDH), based on this hard problem, was proposed in the same paper. The authors proposed working with 2-isogenies and 3-isogenies for efficiency.

Elliptic curves are principally polarised abelian varieties of dimension one, hence we can turn to principally polarised abelian varieties of higher dimension when looking to generalise isogeny-based cryptosystems. As noted by Takashima elliptic curves have three 2-isogenies but abelian surfaces (abelian varieties of dimension 2) have fifteen $(2, 2)$-isogenies. Hence, this motivates the use of abelian surfaces for use in these cryptosystems.

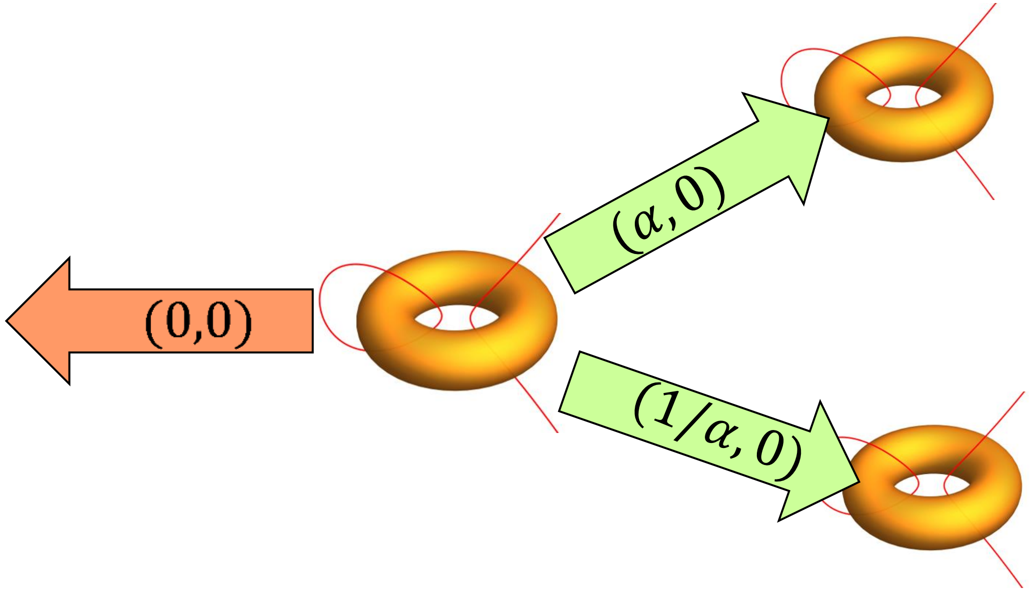
In this work, we will focus on principally polarised supersingular abelian varieties of dimension two, which we call principally polarised supersingular abelian surfaces (PPSSAS) and consider their application to cryptography. The two challenges before us are: to understand the isogeny graphs of PPSSAS, and to have

elements in the n-sphere is $\ell^{3n-3}(\ell^2+1)(\ell+1) \approx \sqrt{p^3}$, hence a naive exhaustive search on the leaves of J_H has a complexity of $O(\sqrt{p^3})$. One can improve on this by considering the meet-in-the-middle search by listing all isogenies of degree ℓ^n from J_H and J_A and finding collisions in both lists. The meet-in-the-middle search has a complexity of $O(\sqrt[4]{p^3})$. One can perform better by employing a quantum computer to reduce the complexity to $O(\sqrt[6]{p^3})$ using Claw finding algorithms [23]. This compares favourably with the genus one case which has classical security of $O(\sqrt[4]{p})$, and quantum security of $O(\sqrt[6]{p})$. An example of a prime which one can use to achieve 128-bits of security is 171-bits, whereas the genus one case requires 512-bits for the same level of security.

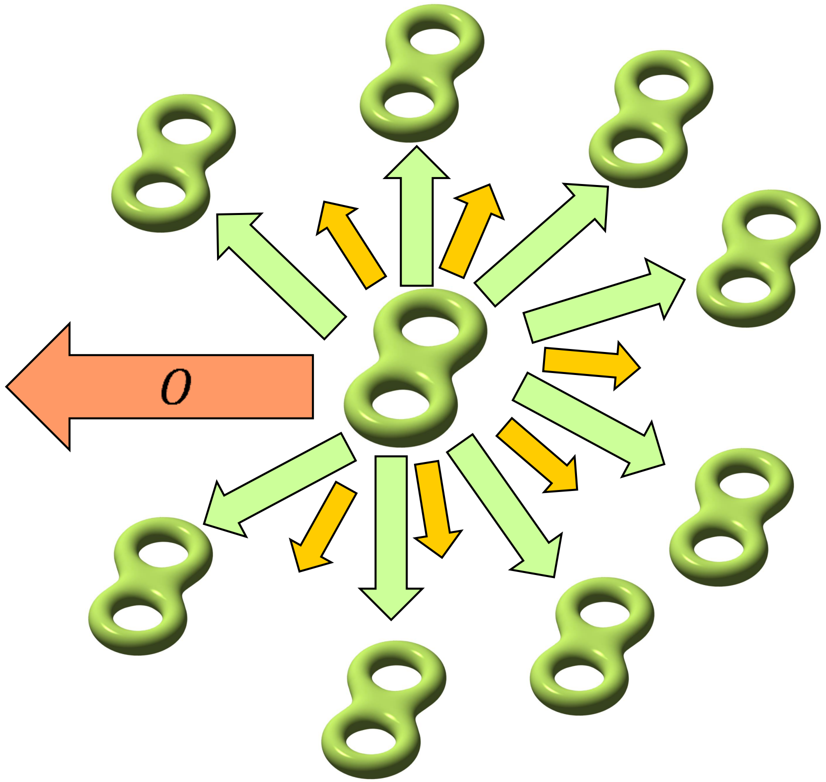
paper

2-isogenies vs. (2,2)-isogenies

$$E \cong \mathbb{Z}_{(p+1)} \times \mathbb{Z}_{(p+1)}$$
$$E[2] \cong \mathbb{Z}_2 \times \mathbb{Z}_2$$



$$J_C \cong \mathbb{Z}_{(p+1)/2} \times \mathbb{Z}_{(p+1)/2} \times \mathbb{Z}_2 \times \mathbb{Z}_2$$
$$J_C[2] \cong \mathbb{Z}_2 \times \mathbb{Z}_2 \times \mathbb{Z}_2 \times \mathbb{Z}_2$$



Castryck-Decru-Smith'19: use superspecial subgraph!

Superspecial g-dimensional PPAV's over $\mathbb{F}_{p^2}$:

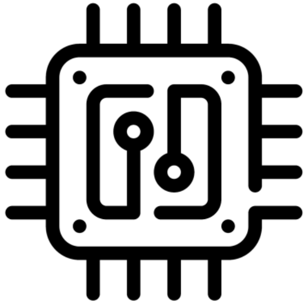
$$O(p^{g(g+1)/2}) \text{ vertices}$$

Genus 1: $O(p)$ vertices

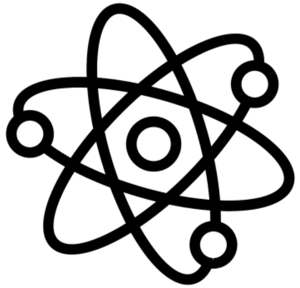
Genus 2: $O(p^3)$ vertices

Motivation for genus 2

- ECC prime field state-of-the-art (Curve25519) uses Montgomery x arith
- HECC prime field state-of-the-art uses Kummer surface arith
- **HECC wins solidly with fields of 1/2 the size**



- SIDH state-of-the-art uses Montgomery x
- **What can we expect with hyperelliptic SIDH using Kummers with fields of 1/3 the size?**



Part 1: Why use curves at all?

Part 2: Why go beyond genus 1?

Part 3: Why stop at genus 2?

Part 4: Why Kummer surfaces?

Part 5: Why are we interested in isogenies?

Part 6: Why is genus 2 (even more) promising here?

Part 7: Why stop at genus 2?

Part 8: Why not use hyperelliptic curves in genus 1?

Superspecial g-dimensional PPAV's over $\mathbb{F}_{p^2}$:

$$O(p^{g(g+1)/2}) \text{ vertices}$$

Genus 1: $O(p)$ vertices

Genus 2: $O(p^3)$ vertices

Genus 3: $O(p^6)$ vertices...

Why stop at genus 2?

- Could this mean $g = 3$ SIDH uses fields of 1/6 the size? Hyper-SIKEp72 vs. SIKEp434?
- Can $g = 4$ SIDH can use fields of 1/10 the size? Hyper-SIKEp43 vs. SIKEp434?
- What about $g > 4$?
- C-Smith'19: finds $\phi : A_g \to A_g{}'$ classically in $\tilde{O}(p^{g-1})$, quantumly in $\tilde{O}(p^{(g-1)/2})$
- Algorithm starts overtaking (asymptotically) generic algorithms for $g > 4$, but absolutely not a deal-breaker for $g \leq 4$

Why stop at genus 2?

- Upshot: no known reason to stop at $g = 2$
- My view is that $g > 1$ isogeny crypto is currently extremely promising, even more promising than $g > 1$ HECC was!
- We currently know a little bit about $g = 2$ (much more work to be done here), but we know almost* nothing for $g > 2$
- Isogeny-based crypto for $g > 1$ is clearly not for the faint-hearted, but the field is wiiiiide open and there's much work to do...

* This paper and this paper are promising starts...

Question

What are some of the issues (or open questions) that need to be resolved before we could seriously consider using $g > 1$ isogenies to compete (or maybe even replace) elliptic curve SIDH/SIKE/….?

Part 1: Why use curves at all?

Part 2: Why go beyond genus 1?

Part 3: Why stop at genus 2?

Part 4: Why Kummer surfaces?

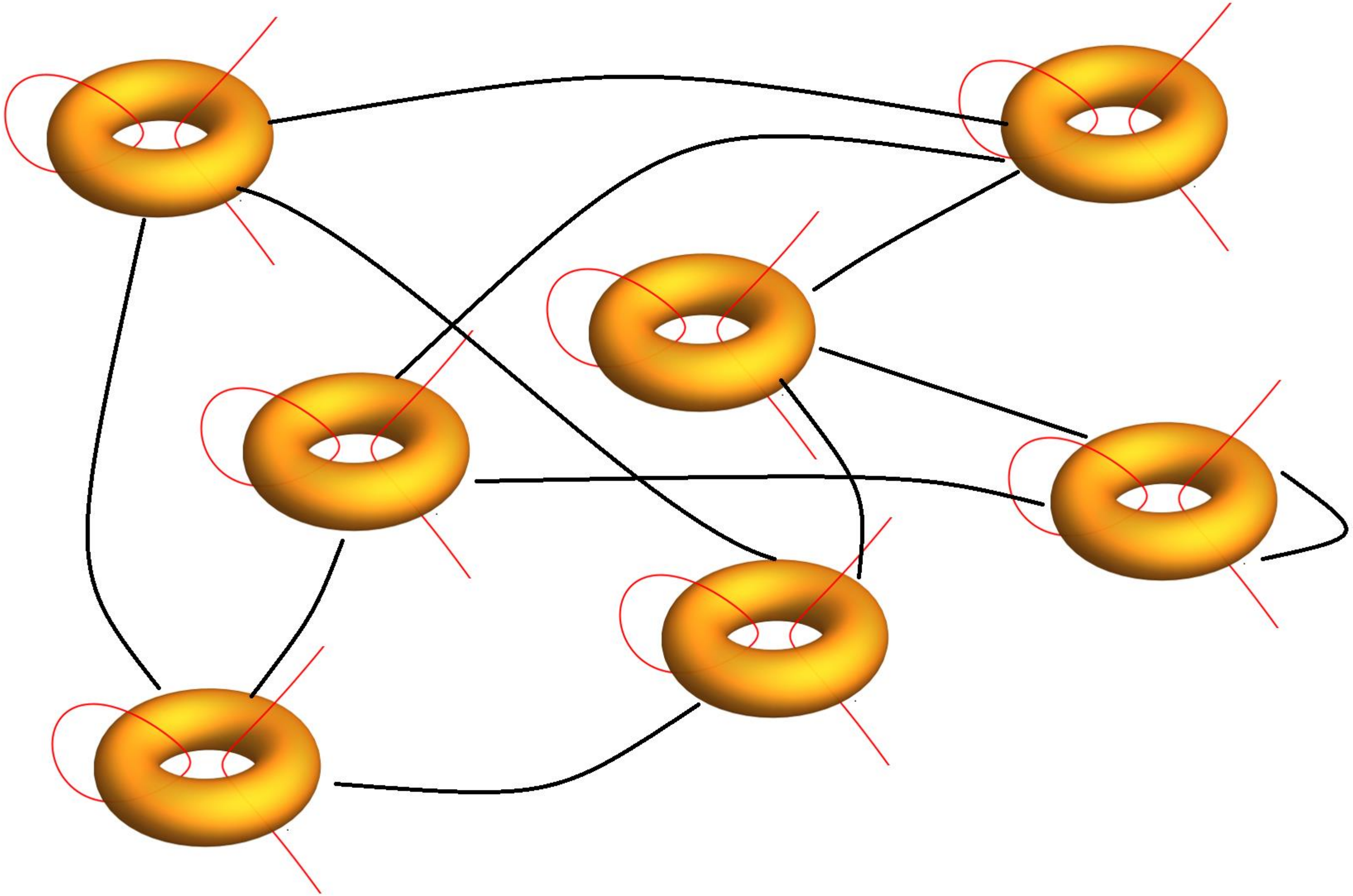
Part 5: Why are we interested in isogenies?

Part 6: Why is genus 2 (even more) promising here?

Part 7: Why stop at genus 2?

Part 8: Why not use hyperelliptic curves in genus 1?

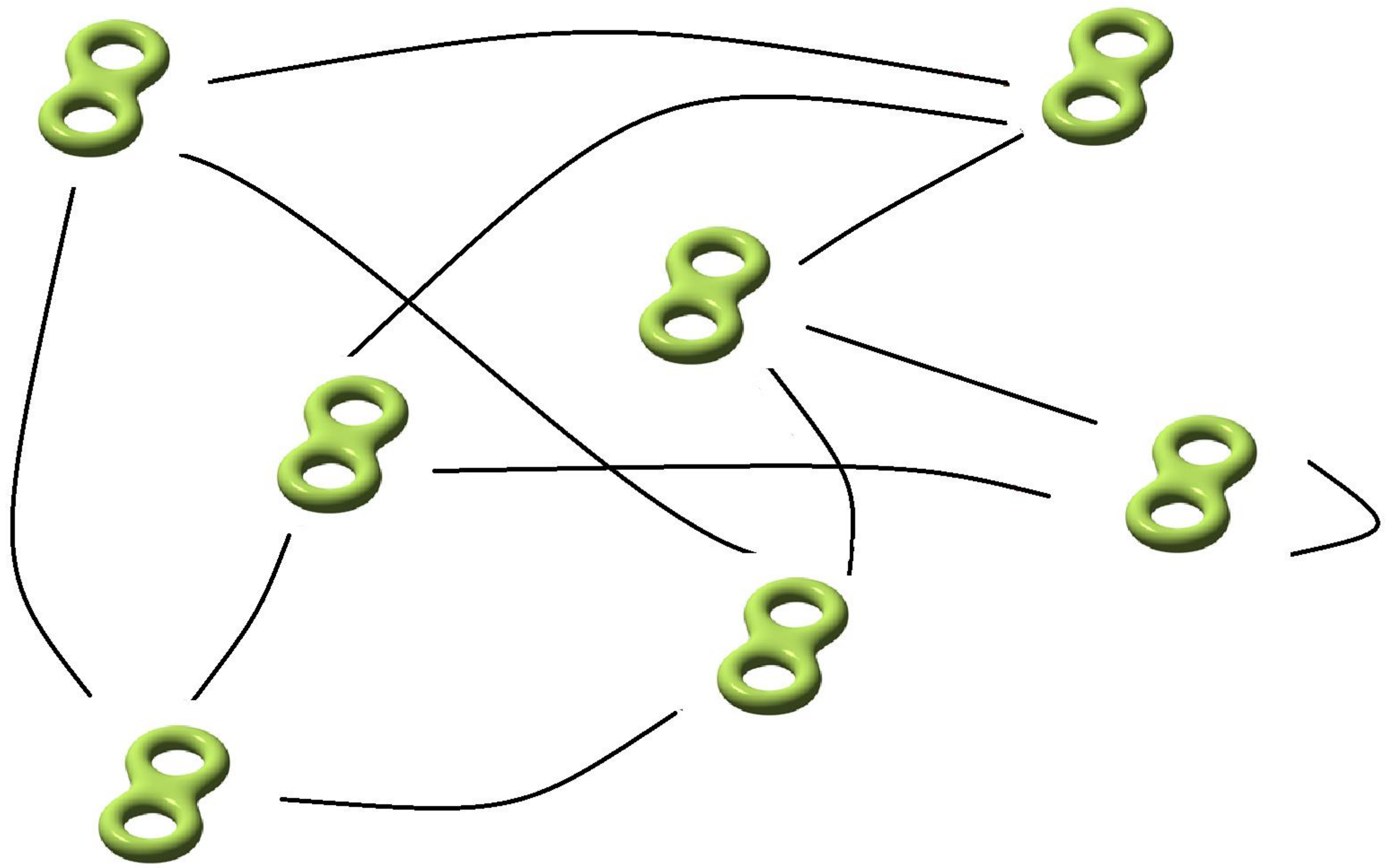
In a nutshell: $E(\mathbb{F}_{p^2})$



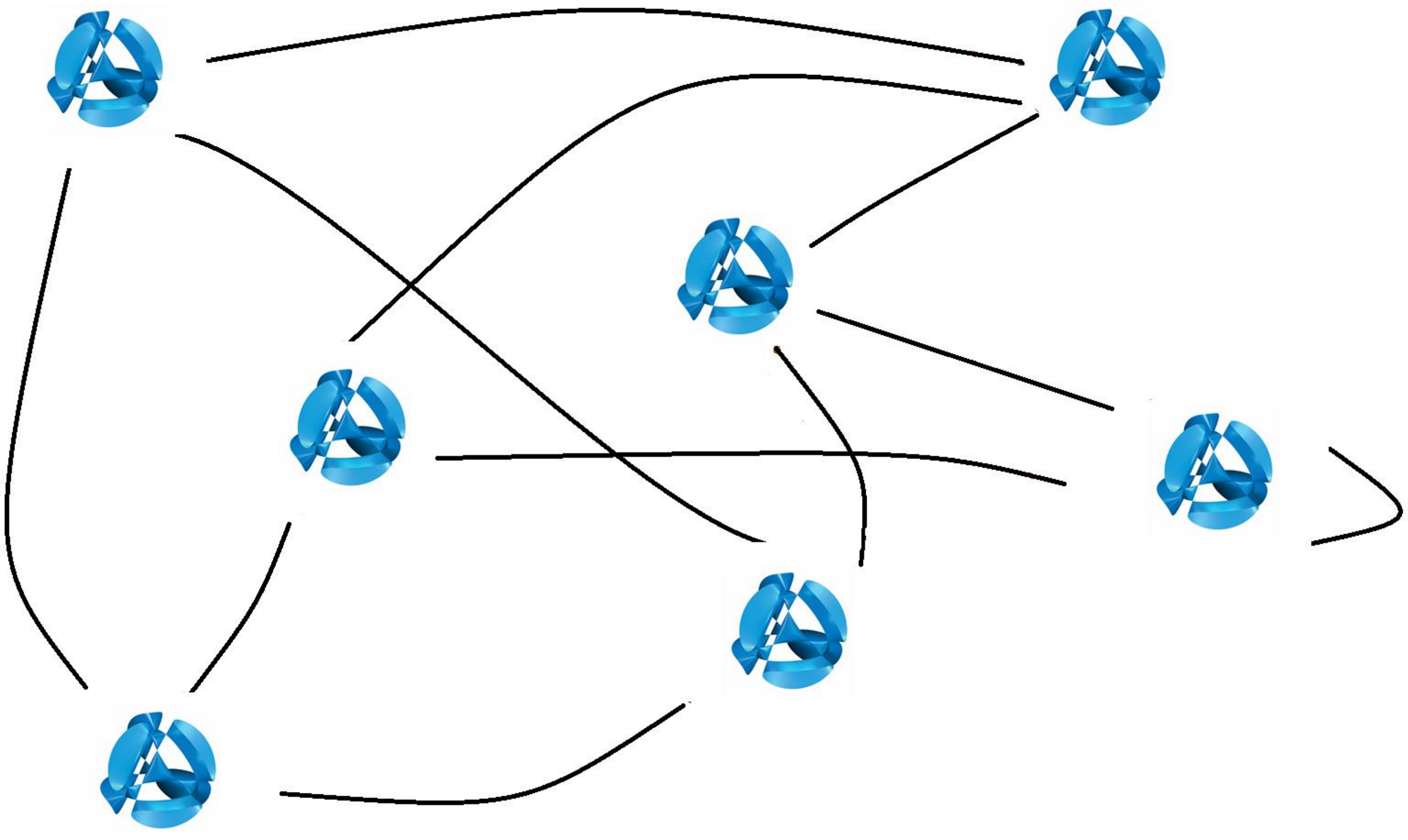
[paper]

[paper]

In a nutshell: $J_C(\mathbb{F}_p)$



In a nutshell: $K(\mathbb{F}_p)$



From elliptic to hyperelliptic

Consider

$$E/K: \quad y^2 = x^3 + 1 \qquad\qquad C/K: \quad y^2 = x^6 + 1$$

Obvious map

$$\omega: \; C(K) \to E(K)$$
$$(x, y) \mapsto (x^2, y)$$

1: But what about $\boldsymbol{\omega^{-1}}$ **:** $E(K) \to C(?)$...

2: Points on E are group elements, points on C are not...

3: Actually want map $E \to J_C$, but $\mathbf{dim}(E) = \mathbf{1}$ while $\mathbf{dim}(J_C) = \mathbf{2}$...

4: Want *general* $\boldsymbol{\omega, \omega^{-1}}$ between $y^2 = x^3 + Ax^2 + x$ to $y^2 = x^6 + Ax^4 + x^2$???

Proposition 1

$\mathbb{F}_{p^2} = \mathbb{F}_p(i)$ with $i^2 + 1 = 0$

$$E/\mathbb{F}_{p^2}: \quad y^2 = x(x-\alpha)(x - 1/\alpha)$$

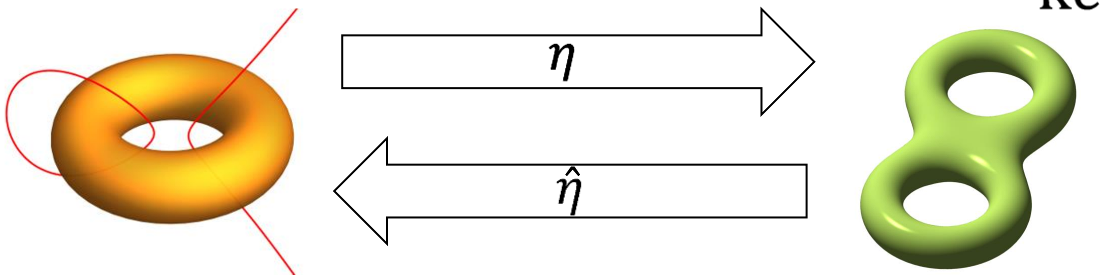
$\alpha = \alpha_0 + \alpha_1 i$ with $\alpha_0, \alpha_1 \in \mathbb{F}_p$

$$C/\mathbb{F}_p: \quad y^2 = (x^2 + mx - 1)(x^2 - mx - 1)(x^2 - mnx - 1)$$

$m = \frac{2\alpha_0}{\alpha_1}$, $n = \frac{(\alpha_0^2 + \alpha_1^2 - 1)}{(\alpha_0 + \alpha_1^2 + 1)}$ both in $\mathbb{F}_p$

Then $\mathbf{Res}_{\mathbb{F}_{p^2}/\mathbb{F}_p}(E)$ is $(2,2)$-isogenous to $J_C(\mathbb{F}_p)$

Or, pictorially,



$\ker(\eta) \cong \ker(\hat{\eta}) \cong \mathbb{Z}_2 \times \mathbb{Z}_2$

$\eta \circ \hat{\eta} = [2]$

Unpacking Proposition 1

- Weil restriction turns 1 equation over $\mathbb{F}_{p^2}$ into two equations over $\mathbb{F}_p$
- Simple linear transform of $E/\mathbb{F}_{p^2}: y^2 = f(x) = x^3 + Ax^2 + x$ to $\tilde{E}/\mathbb{F}_{p^2}:\ y^2 = g(x)$ such that $C/\mathbb{F}_{p^2}: y^2 = g(x^2)$ is non-singular $\Longleftarrow \boxed{\psi}$
- Pullback ω^* of $\omega : (x, y) \mapsto (x^2, y)$ gives 2 points in $C\big(\mathbb{F}_{p^4}\big)$, but composition with Abel-Jacobi map bring these to $J_C(\mathbb{F}_{p^2})$ $\Longleftarrow \boxed{\rho}$
- Need to go from $J_C(\mathbb{F}_{p^2})$ to $J_C(\mathbb{F}_p)$; cue good old Trace map, $\Longleftarrow \boxed{\tau}$

$$\tau:\ P \mapsto \sum_{\sigma \in \mathrm{Gal}(\mathbb{F}_{p^2}/\mathbb{F}_p)}^{n} \sigma(P)$$

$$\eta :\quad \mathrm{Res}_{\mathbb{F}_{p^2}/\mathbb{F}_p}(E) \to J_C(\mathbb{F}_p), \qquad P \mapsto (\tau \circ \rho \circ \psi)(P)$$

Question

There's a bug in Proposition 1 (pointed out to me a while ago by Castryck). Can you spot it?

Performance

Operation	chained 2-isogenies on Montgomery curves over $\mathbb{F}_{p^2}$ (**previous work**)				chained $(2,2)$-isogenies on Kummer surfaces over $\mathbb{F}_p$ (**this work**)				
	M	**S**	**A**	$\approx$ cycles	**m**	**s**	**a**	$\approx$ cycles	
								$\mathbf{s} = \mathbf{m}$	$\mathbf{s} = 0.8\mathbf{m}$
doubling	4	2	4	5862	8	8	16	6272	5714
2-isog. curve	-	2	1	2088	19	4	28	9231	8952
2-isog. point	4	0	4	4336	4	4	16	3480	3200

- Theta constants map to theta constants: point map is enough
- Comparison in Table/paper rather conservative
- Dreamt of (re-)defining SIDH entirely using Kummers over $\mathbb{F}_p$, but compression algorithms were the buzzkill …

Questions?