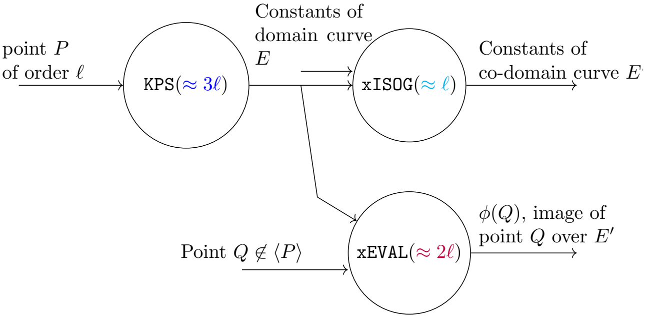
On the $\sqrt{\text{é}}$lu's formulae and its applications to CSIDH and B-SIDH constant-time implementations

Gora Adj, Jesús-Javier Chi-Domínguez and
Francisco Rodríguez-Henríquez

Universitat de Lleida
Tampere University of Applied Sciences

Ches Rump Session, September.15.2020

Computing degree-ℓ isogenies using Vélu's formulas



- For decades now, Vélu's formulae have been widely used to construct and evaluate degree-ℓ isogenies, using three main blocks,
  - KPS [Sort of a pre-computation building block. Cost: $\approx (3\ell)$**M**]
  - xISOG [Finds the image curve. Cost: $\approx (\ell)$**M**]
  - xEVAL [Evaluate a point. Cost: $\approx (2\ell)$**M**]

Computing degree-ℓ isogenies using $\sqrt{\text{é}}$lu's formulas

- Recently, Bernstein, de Feo, Leroux and Smith presented in ANTS'2020 a new approach for computing degree-ℓ isogenies at a reduced cost of just $\tilde{O}(\sqrt{\ell})$ field operations.
- This improvement was obtained by observing that the polynomial product embedded in the isogeny computations could be speedup via a baby-step giant-step method

Our implementation of $\sqrt{\text{é}}$lu

- The most demanding operations of $\sqrt{\text{é}}$lu requires computing four different resultants of the form $\texttt{Res}_Z(f(Z), g(Z))$ of two polynomials $f, g \in \mathbb{F}_q[Z]$.

Our implementation of $\sqrt{\text{élu}}$

- The most demanding operations of $\sqrt{\text{élu}}$ requires computing four different resultants of the form $\texttt{Res}_Z(f(Z), g(Z))$ of two polynomials $f, g \in \mathbb{F}_q[Z]$.
- Those four resultants are computed using a remainder tree approach supported by carefully tailored Karatsuba polynomial multiplications

Our implementation of $\sqrt{\text{élu}}$

- The most demanding operations of $\sqrt{\text{élu}}$ requires computing four different resultants of the form $\texttt{Res}_Z(f(Z), g(Z))$ of two polynomials $f, g \in \mathbb{F}_q[Z]$.
- Those four resultants are computed using a remainder tree approach supported by carefully tailored Karatsuba polynomial multiplications
- In practice, the computational cost of computing degree-ℓ isogenies using $\sqrt{\text{élu}}$, is close to $K(\sqrt{\ell})^{\log_2 3}$ field operations, with K a constant.

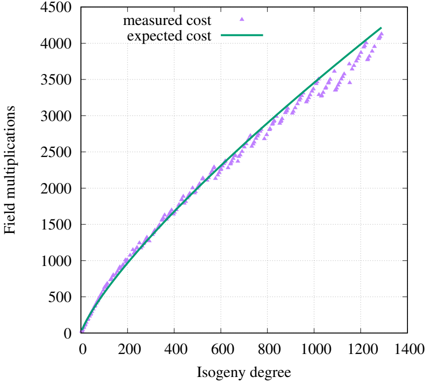
Our implementation of $\sqrt{\text{élu}}$

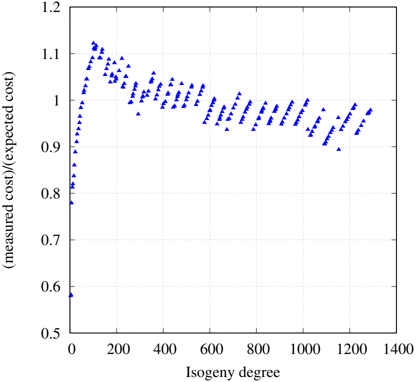
- The most demanding operations of $\sqrt{\text{élu}}$ requires computing four different resultants of the form $\texttt{Res}_Z(f(Z), g(Z))$ of two polynomials $f, g \in \mathbb{F}_q[Z]$.
- Those four resultants are computed using a remainder tree approach supported by carefully tailored Karatsuba polynomial multiplications
- In practice, the computational cost of computing degree-ℓ isogenies using $\sqrt{\text{élu}}$, is close to $K(\sqrt{\ell})^{\log_2 3}$ field operations, with K a constant.
- $\sqrt{\text{élu}}$ is easily parallelizable. A two-core implementation can compute the four resultants in parallel, yielding an expected extra saving of around 35%.

Our implementation of $\sqrt{\text{élu}}$

- The most demanding operations of $\sqrt{\text{élu}}$ requires computing four different resultants of the form $\mathtt{Res}_Z(f(Z), g(Z))$ of two polynomials $f, g \in \mathbb{F}_q[Z]$.
- Those four resultants are computed using a remainder tree approach supported by carefully tailored Karatsuba polynomial multiplications
- In practice, the computational cost of computing degree-ℓ isogenies using $\sqrt{\text{élu}}$, is close to $K(\sqrt{\ell})^{\log_2 3}$ field operations, with K a constant.
- $\sqrt{\text{élu}}$ is easily parallelizable. A two-core implementation can compute the four resultants in parallel, yielding an expected extra saving of around 35%.
- Full details are available at: https://eprint.iacr.org/2020/1109.

Cost model for computing degree-ℓ isogenies using $\sqrt{\text{é}}$lu





Computing a degree-ℓ isogeny. Let $b = \frac{\sqrt{\ell - 1}}{2}$.

$$\text{Expected Cost}(b) = 4\left(9b^{\log_2(3)}\left(1 - 2\left(\frac{2}{3}\right)^{\log_2(b)+1}\right) + 2b\log_2(b)\right)$$
$$+ 3\left(\left(1 - \frac{1}{3^{\log_2(b)+1}}\right)b^{\log_2(3)}\right) + 37b + 3\log_2(b) + 16$$
$$\approx 39 \cdot b^{\log_2(3)}$$

This approximation is a lower bound

Skylake Clock cycle timings for several key exchange isogeny-based protocols

Implementation	Protocol Instantiation	Mcycles
SIKE [NIST alternative candidate]	SIKEp434	22
Castryck *et al.* [Original CSIDH]	CSIDH-512 unprotected	4×155
Bernstein *et al.* [Original $\sqrt{}$élu]	CSIDH-512 unprotected	4×153
	CSIDH-1024 unprotected	4×760
Cervantes-Vázquez *et al.* [LC'19 CSIDH imp]	CSIDH-512	4×238
Chi-Domínguez *et al.* [CSIDH with strategies]	CSIDH-512	4×230
Hutchinson *et al.* [CSIDH with strategies]	CSIDH-512	4×229
This work (**estimated**)	CSIDH-512	4×223
	B-SIDH-p253	119

Table: Skylake Clock cycle timings for a key exchange protocol for different instantiations of the SIDH, CSIDH, and B-SIDH protocols.