A Memory Efficient Version of Satoh's Algorithm

Frederik Vercauteren *, Bart Preneel and Joos Vandewalle

K.U. Leuven, Dept. Elektrotechniek-ESAT/COSIC, Kasteelpark Arenberg 10,
B-3001 Leuven-Heverlee, Belgium.
{Frederik.Vercauteren, Bart.Preneel, Joos.Vandewalle}@esat.kuleuven.ac.be

Abstract. In this paper we present an algorithm for counting points on elliptic curves over a finite field $\mathbb{F}_{p^n}$ of small characteristic, based on Satoh's algorithm. The memory requirement of our algorithm is $O(n^2)$, where Satoh's original algorithm needs $O(n^3)$ memory. Furthermore, our version has the same run time complexity of $O(n^{3+\varepsilon})$ bit operations, but is faster by a constant factor. We give a detailed description of the algorithm in characteristic 2 and show that the amount of memory needed for the generation of a secure 200-bit elliptic curve is within the range of current smart card technology.

Keywords: elliptic curve, finite field, order counting, Satoh's algorithm

1 Introduction

In 1985 Schoof [12] described a polynomial time algorithm for counting the number of points on an elliptic curve E defined over a finite field $\mathbb{F}_q$, with $q = p^n$. The run time of the algorithm is $O(\log^{5+\epsilon} q)$ bit operations using fast arithmetic and the memory requirements are $O(\log^2 q)$. Improvements by Elkies [6] and Atkin [1] led to the so called Schoof-Elkies-Atkin algorithm with a run time of $O(\log^{4+\epsilon} q)$ bit operations and further work by Couveignes [2, 3] and Lercier [9] extended this SEA-algorithm to work in small characteristic. Csirik [4] implemented a reduced memory version of the algorithm. Recently Satoh [11] described a new algorithm for small characteristic $p \geq 5$ with run time $O(n^{3+\varepsilon})$ and memory complexity $O(n^3)$. Skjernaa [14] and Fouquet, Gaudry and Harley [7] independently extended Satoh's algorithm to characteristic 2.

In this paper we present a new version of Satoh's algorithm which still runs in $O(n^{3+\varepsilon})$ bit operations, but only needs $O(n^2)$ memory. The algorithm works for all small characteristics and is even faster than the original algorithm by a constant factor of about 1.5. Furthermore, the algorithm can be easily parallelized. We give a detailed description in the characteristic 2 case and present run times and memory usages of our implementation for elliptic curves in the range of interest to cryptography. The given data show that it now becomes feasible to compute the group order of a 200-bit elliptic curve on a smart card.

* F.W.O. research assistant, sponsored by the Fund for Scientific Research - Flanders (Belgium).

The remainder of the paper is organized as follows: after a brief review of Satoh's original algorithm in section 2, we outline our $O(n^2)$ memory version in its most general form in section 3. In section 4 we specialize this algorithm to the characteristic 2 case and give ready to implement pseudo-code. Section 5 discusses details of our implementation and contains run times and memory usages for field sizes relevant to cryptographical applications.

2 Satoh's Algorithm

Let E be an elliptic curve over $\mathbb{F}_q$, with $q = p^n$. The number of points $\#E(\mathbb{F}_q)$ satisfies the well known relation $\#E(\mathbb{F}_q) = q + 1 - t$, where t is the trace of the Frobenius endomorphism $F : E \longrightarrow E : (x, y) \mapsto (x^q, y^q)$. By Hasse's theorem [8] we have $|t| \leq 2\sqrt{q}$.

The basic idea of Satoh's algorithm is to lift both the curve E and the Frobenius endomorphism F to the valuation ring $\mathcal{R}$ of a degree n unramified extension $\mathcal{K}$ of the p-adic field $\mathbb{Q}_p$. Since this lifting is done in a canonical way, the trace of the lifted Frobenius $\mathcal{F}$ equals the trace of Frobenius t. However, the Frobenius endomorphism F itself is difficult to lift because it is inseparable. Therefore one actually works with the dual of the Frobenius endomorphism F, called the Verschiebung $\widehat{F}$. This Verschiebung is separable if and only if E is non-supersingular and can be lifted explicitly by lifting its kernel. Analyzing the action of the lift $\widehat{\mathcal{F}}$ of $\widehat{F}$ on the formal group of the canonical lift $\mathcal{E}$, we obtain an expression for the trace of $\widehat{\mathcal{F}}$ which equals the trace of Frobenius t.

2.1 The Canonical Lift of an Elliptic Curve

The main step in Satoh's algorithm is lifting the curve E and the Verschiebung $\widehat{F}$ to the valuation ring $\mathcal{R}$ of a degree n unramified extension $\mathcal{K}$ of $\mathbb{Q}_p$. Among the many possible lifts of E from $\mathbb{F}_q$ to $\mathcal{R}$ there is one which has particularly nice properties, called the canonical lift. The canonical lift $\mathcal{E}$ of a non-supersingular elliptic curve E over $\mathbb{F}_q$ is an elliptic curve over $\mathcal{K}$ which satisfies the following two properties: the reduction modulo p of $\mathcal{E}$ equals E and $\mathrm{End}(E) \cong \mathrm{End}(\mathcal{E})$ as a ring. Deuring [5] has shown that the canonical lift $\mathcal{E}$ always exists and is unique up to isomorphism. Furthermore, a theorem by Lubin, Serre and Tate [10] provides an effective, but slow algorithm to compute the j-invariant of $\mathcal{E}$ given the j-invariant of E.

Theorem 1 (Lubin-Serre-Tate) *Let E be a non-supersingular elliptic curve over $\mathbb{F}_q$ with j-invariant $j(E) \in \mathbb{F}_q \setminus \mathbb{F}_{p^2}$. Denote with Σ the Frobenius substitution on $\mathcal{R}$ and with $\Phi_p(X, Y)$ the p-th modular polynomial. Then the system of equations*

$$\Phi_p(X, \Sigma(X)) = 0 \quad \text{and} \quad X \equiv j(E) \bmod p, \tag{1}$$

has a unique solution $J \in \mathcal{R}$, which is the j-invariant of the canonical lift $\mathcal{E}$ of E.

Note that it is possible to solve the system of equations (1) directly, but this would lead to a slow algorithm because of the explicit computation of Σ. A detailed description of the Frobenius substitution Σ and its computation can be found in [13].

The hypothesis $j(E) \notin \mathbb{F}_{p^2}$ in Theorem 1 is necessary to ensure that a certain partial derivative of Φ_p does not vanish modulo p. This condition is necessary to guarantee the uniqueness of the solution of equation (1). The case $j(E) \in \mathbb{F}_{p^2}$ can be handled very easily using Weil's theorem: since $j(E) \in \mathbb{F}_{p^2}$ there exists an elliptic curve E' defined over $\mathbb{F}_{p^m}$ with $m = 1$ or $m = 2$, which is isomorphic to E over $\mathbb{F}_q$. Let $t_k = p^{mk} + 1 - \#E'(\mathbb{F}_{p^{mk}})$ then $t_{k+1} = t_1 t_k - p^m t_{k-1}$ with $t_0 = 2$ and therefore $\#E(\mathbb{F}_q) = p^n + 1 - t_{n/m}$. So in the remainder of the paper we can assume $j(E) \notin \mathbb{F}_{p^2}$ and in particular that E is non-supersingular.

Let $\sigma : E \longrightarrow E^\sigma : (x, y) \mapsto (x^p, y^p)$ be the p-th power Frobenius morphism, where E^σ is the curve obtained by raising each coefficient of E to the p-th power and let $\hat{\sigma}$ be the dual of σ. Repeatedly applying $\hat{\sigma}$ gives rise to the following cycle

$$E_0 \xrightarrow{\hat{\sigma}_0} E_1 \xrightarrow{\hat{\sigma}_1} \cdots \xrightarrow{\hat{\sigma}_{n-2}} E_{n-1} \xrightarrow{\hat{\sigma}_{n-1}} E_0 \ ,$$

with $E_{(n-i)} = E^{\sigma^i}$ and $\hat{\sigma}_i$ the dual of $\sigma_i : E_{i+1} \longrightarrow E_i : (x, y) \mapsto (x^p, y^p)$. Composing these, we see that $\widehat{F} = \hat{\sigma}_{n-1} \circ \hat{\sigma}_{n-2} \circ \ldots \circ \hat{\sigma}_0$. Instead of lifting E and $\widehat{F}$ directly, the crucial insight of Satoh was to lift the whole cycle $(E_0, E_1, \ldots, E_{n-1})$ simultaneously leading to the diagram

$$\begin{array}{ccccccccc}
\mathcal{E}_0 & \xrightarrow{\widehat{\Sigma}_0} & \mathcal{E}_1 & \xrightarrow{\widehat{\Sigma}_1} & \cdots & \xrightarrow{\widehat{\Sigma}_{n-2}} & \mathcal{E}_{n-1} & \xrightarrow{\widehat{\Sigma}_{n-1}} & \mathcal{E}_0 \\
\uparrow & & \uparrow & & & & \uparrow & & \uparrow \\
E_0 & \xrightarrow{\hat{\sigma}_0} & E_1 & \xrightarrow{\hat{\sigma}_1} & \cdots & \xrightarrow{\hat{\sigma}_{n-2}} & E_{n-1} & \xrightarrow{\hat{\sigma}_{n-1}} & E_0,
\end{array} \tag{2}$$

with $\mathcal{E}_i$ the canonical lift of E_i and $\widehat{\Sigma}_i$ the corresponding lift of $\hat{\sigma}_i$. The theorem of Lubin, Serre and Tate implies that the j-invariants of $\mathcal{E}_i$ satisfy

$$\Phi_p(j(\mathcal{E}_i), j(\mathcal{E}_{i+1})) = 0 \text{ and } j(\mathcal{E}_i) \equiv j(E_i) \bmod p, \tag{3}$$

for $i = 0, \ldots, n-1$. Define $\Theta : \mathcal{R}^n \longrightarrow \mathcal{R}^n$ by

$$\Theta(x_0, x_1, \ldots, x_{n-1}) = (\Phi_p(x_0, x_1), \Phi_p(x_1, x_2), \ldots, \Phi_p(x_{n-1}, x_0)), \tag{4}$$

then clearly we have $\Theta(j(\mathcal{E}_0), j(\mathcal{E}_1), \ldots, j(\mathcal{E}_{n-1})) = (0, 0, \ldots, 0)$. Using a multivariate Newton iteration on Θ, we can lift the cycle $(j(E_0), j(E_1), \ldots, j(E_{n-1}))$ to $\mathcal{R}^n$ with arbitrary precision. The iteration step is given by

$$(J_0, J_1, \ldots, J_{n-1}) \leftarrow (J_0, J_1, \ldots, J_{n-1}) - ((D\Theta)^{-1}\Theta)(J_0, J_1, \ldots, J_{n-1}), \tag{5}$$

with $D\Theta$ the Jacobian matrix

$$(D\Theta)(J_0, J_1, \ldots, J_{n-1}) = \begin{pmatrix} \frac{\partial \Phi_p}{\partial X}(J_0, J_1) & \frac{\partial \Phi_p}{\partial Y}(J_0, J_1) \cdots & 0 \\ 0 & \frac{\partial \Phi_p}{\partial X}(J_1, J_2) \cdots & 0 \\ \vdots & \vdots & \vdots \\ 0 & 0 & \cdots \frac{\partial \Phi_p}{\partial Y}(J_{n-2}, J_{n-1}) \\ \frac{\partial \Phi_p}{\partial Y}(J_{n-1}, J_0) & 0 & \cdots \frac{\partial \Phi_p}{\partial X}(J_{n-1}, J_0) \end{pmatrix}. \tag{6}$$

The p-th modular equation satisfies the Kronecker relation

$$\Phi_p(X, Y) \equiv (X^p - Y)(X - Y^p) \bmod p \tag{7}$$

and since $j(E_i) \notin \mathbb{F}_{p^2}$ and $j(E_i) \equiv j(E_{i+1})^p \bmod p$, this leads to the following equations

$$\begin{cases} \frac{\partial \Phi_p}{\partial X}(j(E_i), j(E_{i+1})) \equiv j(E_{i+1})^{p^2} - j(E_{i+1}) \not\equiv 0 \bmod p, \\ \frac{\partial \Phi_p}{\partial Y}(j(E_i), j(E_{i+1})) \equiv j(E_{i+1})^p - j(E_{i+1})^p \equiv 0 \bmod p. \end{cases} \tag{8}$$

The above equations imply that the Jacobian matrix $(D\Theta)(J_0, J_1, \ldots, J_{n-1})$ is invertible over $\mathcal{R}$ and therefore we see $((D\Theta)^{-1}\Theta)(J_0, J_1, \ldots, J_{n-1}) \in \mathcal{R}^n$. Since Newton iteration has quadratic convergence, we can compute $J_i \equiv j(\mathcal{E}_i) \bmod p^N$ with $\log N$ iterations.

2.2 The Trace of Frobenius

The canonical lift $\mathcal{E}$ of a non-supersingular elliptic curve E over $\mathbb{F}_q$ has the property that $\mathrm{End}(E) \cong \mathrm{End}(\mathcal{E})$. Therefore we have $\mathrm{Tr}(F) = \mathrm{Tr}(\mathcal{F})$, where F is the Frobenius endomorphism on E and $\mathcal{F}$ the image of F under the ring isomorphism $\mathrm{End}(E) \cong \mathrm{End}(\mathcal{E})$. Furthermore, the trace of an endomorphism equals the trace of its dual, so $\mathrm{Tr}(F) = \mathrm{Tr}(\widehat{F}) = \mathrm{Tr}(\mathcal{F}) = \mathrm{Tr}(\widehat{\mathcal{F}})$. The following proposition by Satoh [11] gives a very simple relation between the trace of $\widehat{\mathcal{F}}$ and the leading coefficient of the endomorphism induced by $\widehat{\mathcal{F}}$ on the formal group of $\mathcal{E}$.

Proposition 1 (Satoh) *Let $\mathcal{E}$ be an elliptic curve over $\mathcal{K}$ and let $f \in \mathrm{End}_{\mathcal{K}}(\mathcal{E})$ be of degree d. Denote with τ the local parameter of $\mathcal{E}$ at $\mathcal{O}$ and assume that the reduction $\pi(f)$ of f modulo p is separable and that $f(\mathrm{Ker}(\pi)) \subset \mathrm{Ker}(\pi)$. Let $\widetilde{f}(\tau) = c\tau + O(\tau^2)$ be the homomorphism induced by f on the formal group of $\mathcal{E}$, then $\mathrm{Tr}(f) = c + \frac{d}{c}$.*

Since the Frobenius endomorphism F is inseparable, we cannot apply the above proposition to $\mathcal{F}$. However, for a non-supersingular curve the Verschiebung $\widehat{F}$

is separable and we have $\mathrm{Tr}(F) = \mathrm{Tr}(\widehat{\mathcal{F}}) = c + \frac{q}{c}$ with $\widetilde{\widehat{\mathcal{F}}}(\tau) = c\tau + O(\tau^2)$. Diagram (2) shows that $\widehat{\mathcal{F}}$ can be written as $\widehat{\mathcal{F}} = \widehat{\Sigma}_{n-1} \circ \widehat{\Sigma}_{n-2} \circ \cdots \circ \widehat{\Sigma}_0$ and therefore we can compute c as the product of the leading coefficients of the morphisms induced by $\widehat{\Sigma}_i$. More precisely, let c_i be defined by $\tau_{i+1} \circ \widehat{\Sigma}_i = c_i\tau_i + O(\tau_i^2)$, with τ_i the local parameter of $\mathcal{E}_i$ at $\mathcal{O}$, then $c = \prod_{0 \le i < n} c_i$. Since $\widehat{\mathcal{F}}$ is separable, c will be non-zero modulo p and we conclude

$$\mathrm{Tr}(F) \equiv \prod_{0 \le i < n} c_i \bmod q. \tag{9}$$

The final step in Satoh's algorithm is to compute the coefficients c_i, based on the equations for $\mathcal{E}_i$ and $\mathcal{E}_{i+1}$ and the kernel of $\widehat{\Sigma}_i$, using Vélu's formulae [15]. The equations for $\mathcal{E}_i$ and $\mathcal{E}_{i+1}$ can be easily computed via a univariate Newton iteration, since we already know their j-invariants. The isogenies $\hat{\sigma}_i$ and $\widehat{\Sigma}_i$ are separable and of degree p, so $\hat{\sigma}_i$ can be explicitly lifted to $\widehat{\Sigma}_i$ by lifting its kernel. This kernel is a subgroup of the p-torsion group of E. The case $p \ge 5$ is discussed in [11] and proceeds by lifting a factor of the p-th division polynomial using a Hensel lift. The cases $p = 2, 3$ can be found in [7, 14] and are handled by lifting a single non-trivial torsion point using a Newton iteration.

2.3 Complexity

According to Hasse's theorem we have $|t| \le 2\sqrt{q}$. Therefore it suffices to lift all the data with precision $N \simeq n/2$. Since elements of $\mathcal{R} \bmod p^N$ can be represented as degree n polynomials with coefficients in $\mathbb{Z}/p^N\mathbb{Z}$ and since $N = O(n)$, every element will take $O(n^2)$ memory for fixed p. For each curve E_i with $0 \le i < n$ we need $O(1)$ such elements, so the total memory required is $O(n^3)$. To lift the cycle of j-invariants with precision N, we need $\log N$ iterations. Working with the lowest possible precision in every iteration, the lifting of the cycle of j-invariants amounts to $O(nM(n^2))$ bit operations, where $M(m)$ is the time to multiply two m-bit objects. The computation of one coefficient c_i needs $O(1)$ multiplications, so to compute all c_i we also need $O(nM(n^2))$ bit operations. Therefore the total run time of Satoh's algorithm is $O(nM(n^2))$ bit operations or $O(n^{3+\varepsilon})$ using fast multiplication techniques.

3 An $O(n^2)$ Memory Algorithm

In this section we present a new version of Satoh's algorithm, which requires only $O(n^2)$ memory and still runs in $O(n^{3+\epsilon})$ bit operations. The basic idea is very simple: the trace of Frobenius t can be computed as $t \equiv \prod_{0 \le i < n} c_i \bmod q$ and the c_i only depend on $\mathcal{E}_i$ and $\mathcal{E}_{i+1}$. So the main problem of Satoh's original algorithm is that it lifts all j-invariants simultaneously, instead of lifting one j-invariant at a time. Note however that lifting all j-invariants simultaneously is exactly what makes Satoh's algorithm efficient, because this avoids slow Frobenius computations in $\mathcal{R}$. Thus if we would like our algorithm to run in $O(n^{3+\epsilon})$

bit operations and only use $O(n^2)$ memory, we have to find a method to lift one j-invariant without using Frobenius computations.

Our strategy is as follows: the j-invariants $j(\mathcal{E}_i)$ and $j(\mathcal{E}_{i+1})$ satisfy the following relations

$$\Phi_p(j(\mathcal{E}_i), j(\mathcal{E}_{i+1})) = 0, \quad j(\mathcal{E}_i) \equiv j(E_i) \bmod p \quad \text{and} \quad j(\mathcal{E}_{i+1}) \equiv j(E_{i+1}) \bmod p. \tag{10}$$

Suppose we have $J_{i+1} \equiv j(\mathcal{E}_{i+1}) \bmod p^N$ to our disposal, then we can compute $J_i \equiv j(\mathcal{E}_i) \bmod p^N$ using a univariate Newton iteration on $\Phi_p(X, J_{i+1})$. This iteration is given by

$$J_i \leftarrow J_i - \frac{\Phi_p(J_i, J_{i+1})}{\frac{\partial \Phi_p}{\partial X}(J_i, J_{i+1})}, \tag{11}$$

and we can use $j(E_i) \equiv j(\mathcal{E}_i) \bmod p$ as an initial approximation. Since $\Phi_p(X, Y)$ satisfies the Kronecker relation, $\frac{\partial \Phi_p}{\partial X}(J_i, J_{i+1})$ will be invertible in $\mathcal{R}$. Note that we are forced to walk backwards in the cycle, since $\frac{\partial \Phi_p}{\partial Y}(J_i, J_{i+1}) \equiv 0 \bmod p$. Applying this method repeatedly, one easily sees that it suffices to compute one j-invariant with precision N, e.g. $J_0 \equiv j(\mathcal{E}_0) \bmod p^N$. To solve this last problem, we analyze in detail the properties of a bivariate polynomial, which satisfies the same relations as $\Phi_p(X, Y)$.

Proposition 2 *Let $\mathcal{K}$ be an unramified extension of $\mathbb{Q}_p$ and denote with $\mathcal{R}$ its valuation ring. Let $g \in \mathcal{R}[X, Y]$ and assume $x_0, y_0 \in \mathcal{R}$ such that*

$$g(x_0, y_0) \equiv 0 \bmod p, \quad \frac{\partial g}{\partial X}(x_0, y_0) \not\equiv 0 \bmod p \quad \text{and} \quad \frac{\partial g}{\partial Y}(x_0, y_0) \equiv 0 \bmod p. \tag{12}$$

Then the following properties hold:

1. *For every $y \in \mathcal{R}$ with $y \equiv y_0 \bmod p$ there exists a unique $x \in \mathcal{R}$ such that $x \equiv x_0 \bmod p$ and $g(x, y) = 0$.*
2. *Let $y' \in \mathcal{R}$ with $y \equiv y' \bmod p^M$, $M \geq 1$ and let $x' \in \mathcal{R}$ be the unique element with $x' \equiv x_0 \bmod p$ and $g(x', y') = 0$. Then $x' \equiv x \bmod p^{M+1}$.*

`Proof:`

1. Define $h \in \mathcal{R}[X]$ by $h(X) = g(X, y)$. Then $h(x_0) \equiv 0 \bmod p$ and $h'(x_0) \equiv \frac{\partial g}{\partial X}(x_0, y_0) \bmod p$. Therefore, $h'(x_0) \not\equiv 0 \bmod p$ and Hensel's lemma guarantees the existence of a unique $x \in \mathcal{R}$ such that $h(x) = g(x, y) = 0$ and $x \equiv x_0 \bmod p$. Furthermore, given y, one can compute x with arbitrary precision using a univariate Newton iteration on $g(X, y)$ with $x_0 \bmod p$ as an initial approximation.

2. Define $\delta_x = x' - x$ and $\delta_y = y' - y$. Clearly $\delta_x \equiv \delta_y \equiv 0 \bmod p^M$. Writing out the Taylor series of $g(X,Y) = \sum_{i,j} g_{i,j} X^i Y^i$ leads to

$$\begin{aligned} 0 = g(x',y') &= g(x+\delta_x, y+\delta_y) \\ &= \sum_{i,j} g_{i,j}(x+\delta_x)^i (y+\delta_y)^j \\ &= \sum_{i,j} g_{i,j}(x^i + ix^{i-1}\delta_x + \delta_x^2 R_x(x))(y^j + jy^{j-1}\delta_y + \delta_y^2 R_y(y)), \end{aligned} \tag{13}$$

with R_x, R_y polynomials with coefficients in $\mathcal{R}$. Since $\delta_x^2 \equiv \delta_y^2 \equiv 0 \bmod p^{2M}$ and $M \geq 1$ we get

$$0 \equiv \frac{\partial g}{\partial X}(x,y)(x-x') + \frac{\partial g}{\partial Y}(x,y)(y-y') \bmod p^{M+1}. \tag{14}$$

The above equation implies $x \equiv x' \bmod p^{M+1}$, since $\delta_y \equiv 0 \bmod p^M$, $\frac{\partial g}{\partial Y}(x,y) \equiv 0 \bmod p$ and $\frac{\partial g}{\partial X}(x,y) \not\equiv 0 \bmod p$. □

Repeatedly applying Proposition 2 leads to a very simple iterative algorithm to compute $J_0 \equiv j(\mathcal{E}_0) \bmod p^N$. Starting with $J_{N-1} \equiv j(\mathcal{E}_{N-1}) \bmod p$, we compute $J_{N-2} \equiv j(\mathcal{E}_{N-2}) \bmod p^2$ using a Newton iteration on $\Phi_p(X, J_{N-1})$, similar to equation 11. More generally, given $J_{N-i+1} \equiv j(\mathcal{E}_{N-i+1}) \bmod p^{i-1}$, we determine $J_{N-i} \equiv j(\mathcal{E}_{N-i}) \bmod p^i$. After $N-1$ steps we reach $J_0 \equiv j(\mathcal{E}_0) \bmod p^N$. Combining these ideas finally leads to algorithm `Satoh_Low_Memory`.

Algorithm 1 (`Satoh_Low_Memory`)

IN: *A j-invariant $j \in \mathbb{F}_{p^n} \setminus \mathbb{F}_{p^2}$ of an elliptic curve E.*
OUT: *The trace of Frobenius $t = q + 1 - \#E(\mathbb{F}_q)$ of E.*

---

1. *Compute $J \equiv j(\mathcal{E}) \bmod p^N$ with $N > n/2 + 1$ from $J_{N-1} \equiv j(\mathcal{E}_{N-1}) \bmod p$ with $N-1$ Newton iterations 11;*
2. `Set` $c^2 = 1$*;*
3. `For` $i = 1$ `To` n `Do`
   - 3.1. *Compute $J' \equiv j(\mathcal{E}_{n-i}) \bmod p^N$ using a Newton iteration 11 on $\Phi_p(X, J)$;*
   - 3.2. *Compute the square $c_{n-i}^2 \bmod p^N$ of coefficient $c_{n-i} \bmod p^N$;*
   - 3.3. `Set` $c^2 = c^2 \times c_{n-i}^2$ *and* $J = J'$*;*
4. *Compute $c \equiv \sqrt{c^2} \bmod p^N$ with the correct sign;*
5. `Return` $t \equiv c \bmod p^N$.

The memory requirement of algorithm `Satoh_Low_Memory` is $O(n^2)$ for p fixed: every element in $\mathcal{R}$ mod p^N takes $O(n^2)$ memory, and the algorithm needs $O(1)$ such elements. Therefore, the total memory required is $O(n^2)$.

Lifting one j-invariant to precision N and computing one coefficient c_i can be done with $O(M(n^2))$ bit operations, so the loop in step 3 takes $O(nM(n^2))$ bit operations. Since the j-invariant in step 1 is computed using N Newton iterations with varying precision $i = 2, \ldots, N$, the total cost of step 1 is trivially bounded by $O(nM(n^2))$ bit operations. We therefore conclude that our version still runs in $O(nM(n^2))$ bit operations or $O(n^{3+\varepsilon})$ using fast arithmetic.

4 Algorithms in Characteristic 2

In this section we specialize the $O(n^2)$ memory algorithm of the previous section to the characteristic 2 case, which from a practical point of view is most important.

Let E be an elliptic curve over a finite field $\mathbb{F}_q$, with $q = 2^n$ and $j(E) \notin \mathbb{F}_4$. It is well known that either E or its quadratic twist is isomorphic over $\mathbb{F}_q$ with an elliptic curve given by an equation of the form $y^2 + xy = x^3 + a$, with $a \in \mathbb{F}_q^*$. Therefore, we can restrict ourselves to this case.

Let $\mathcal{K}$ be a degree n unramified extension of $\mathbb{Q}_2$ and $\mathcal{R}$ its valuation ring. Then $\mathcal{R}$ is isomorphic to $\mathbb{Z}_2[T]/(f(T))$, with $f \in \mathbb{Z}_2[T]$ a monic polynomial of degree n such that its reduction modulo 2 is irreducible in $\mathbb{F}_2[T]$. In practice all computations are carried out in the ring $\mathcal{R}$ mod 2^N, which can be represented as $(\mathbb{Z}/2^N\mathbb{Z})[T]/(f(T))$.

4.1 Lifting the j-invariants

For $1 \leq i < n$ define the elliptic curve E_i by the equation $y^2 + xy = x^3 + a^{2^{n-i}}$ and let $\mathcal{E}_i$ be the canonical lift of E_i. Using Proposition 2 we can compute $J_i \equiv j(E_i) \bmod 2^N$, starting from $J_{i+1} \equiv j(E_{i+1}) \bmod 2^{N-1}$, using a univariate Newton iteration on the polynomial $\Phi_2(X, J_{i+1})$, with

$$\begin{aligned}\Phi_2(X,Y) =& X^3 + Y^3 - X^2Y^2 + 1488(XY^2 + X^2Y) - 162000(X^2 + Y^2) \\ &+ 40773375XY + 8748000000(X + Y) - 157464000000000.\end{aligned} \tag{15}$$

Algorithm `Lift_Previous_J_Invariant` computes coefficients $A, B, C \in \mathcal{R}$ mod 2^N, such that

$$\Phi_2(X, J_{i+1}) \equiv X^3 + AX^2 + BX + C \bmod 2^N, \tag{16}$$

and then calls the recursive algorithm `Lift_Previous_J_Invariant_Rec` which performs the Newton iteration on the cubic polynomial $X^3 + AX^2 + BX + C$.

With every call of algorithm `Lift_Previous_J_Invariant` we gain 1 bit of precision, so if we would like to compute $J_0 \equiv j(\mathcal{E}_0) \bmod 2^N$ then it suffices to start with $j(E_{N-1}) \equiv j(\mathcal{E}_{N-1}) \bmod 2$ and iterate this algorithm $N - 1$ times, which immediately leads to algorithm `Lift_First_J_Invariant`.

Algorithm 2 (`Lift_Previous_J_Invariant`)

IN: *$J_{i+1} \in \mathcal{R} \bmod 2^N$ with $J_{i+1} \equiv j(\mathcal{E}_{i+1}) \bmod 2^{N-1}$ and a precision N.*
OUT: *$J_i \in \mathcal{R} \bmod 2^N$ with $J_i \equiv j(\mathcal{E}_i) \bmod 2^N$.*

---

1. $A \equiv -J_{i+1}^2 + 1488J_{i+1} - 162000 \bmod 2^N$;
2. $B \equiv 1488J_{i+1}^2 + 40773375J_{i+1} + 8748000000 \bmod 2^N$;
3. $C \equiv J_{i+1}^3 - 162000J_{i+1}^2 + 8748000000J_{i+1} - 157464000000000 \bmod 2^N$;
4. $J_i =$ `Lift_Previous_J_Invariant_Rec(`J_{i+1}, A, B, C, N`)`;
5. `Return` J_i.

Algorithm 3 (`Lift_Previous_J_Invariant_Rec`)

IN: *Elements $J_{i+1}, A, B, C \in \mathcal{R} \bmod 2^N$ with $J_{i+1} \equiv j(\mathcal{E}_{i+1}) \bmod 2^{N-1}$, $\Phi_2(X, J_{i+1}) \equiv X^3 + AX^2 + BX + C \bmod 2^N$ and a precision N.*
OUT: *An element $J_i \in \mathcal{R} \bmod 2^N$ with $J_i \equiv j(\mathcal{E}_i) \bmod 2^N$.*

---

1. `If` $N = 1$ `Then`
   1.1. $J_i = J_{i+1}^2 \bmod 2$;
2. `Else`
   2.1. $N' = \lceil \frac{N}{2} \rceil$;
   2.2. $J_i =$ `Lift_Previous_J_Inv_Rec(`J_{i+1}, A, B, C, N'`)`;
   2.3. $J_i \equiv J_i - \dfrac{J_i^3 + AJ_i^2 + BJ_i + C}{3J_i^2 + 2AJ_i + B} \bmod 2^N$;
3. `Return` J_i.

Algorithm 4 (`Lift_First_J_Invariant`)

IN: *A j-invariant $j_0 \in \mathbb{F}_{2^n} \setminus \mathbb{F}_4$ and a precision N.*
OUT: *$J_0 \in \mathcal{R} \bmod 2^N$ with $J_0 \equiv j_0 \bmod 2$ and $\Phi_2(J_0, \Sigma(J_0)) \equiv 0 \bmod 2^N$.*

---

1. $J_0 \equiv j_0^{2^{(n-N+1)}} \bmod 2$;
2. `For` $i = 2$ `To` N `Do`
   2.1. $J_0 =$ `Lift_Previous_J_Invariant(`J_0, i`)`;
3. `Return` J_0.

4.2 Computing the Trace

In this section we give an explicit formula for the first coefficient c_i of the formal group expression of $\widehat{\Sigma}_i$. This suffices to compute the trace of Frobenius t, since $t \equiv \prod_{i=0}^{n-1} c_i \bmod q$.

The following proposition gives an expression for c_i^2 in terms of the j-invariant of $\mathcal{E}_i$ and the x-coordinate of the non-trivial point in $\mathrm{Ker}(\widehat{\Sigma}_i)$. Since $\widehat{\Sigma}_i$ is separable and of degree 2, its kernel is a subgroup of order 2 of the 2-torsion points and therefore contains exactly one non-trivial point. The proposition is adapted from [14]: the proof is exactly the same, but the given formulae have been simplified as much as possible.

Proposition 3 *Let $\tau_i = -X/Y$ be the local parameter of $\mathcal{E}_i$ at $\mathcal{O}$ and let c_i be defined as $\tau_{i+1} \circ \widehat{\Sigma}_i = c_i\tau_i + O(\tau_i^2)$. Denote the non-trivial point in $\mathrm{Ker}(\widehat{\Sigma}_i)$ by $Q_i = (x_i, y_i)$ and let $z_i = x_i/2$ and $t_i = (12z_i^2 + z_i)(j(\mathcal{E}_i) - 1728) - 36$, then*

$$c_i^2 = \frac{j(\mathcal{E}_i) - (504 + 12096z_i)t_i}{j(\mathcal{E}_i) + 240t_i}. \tag{17}$$

Algorithm 5 (`Compute_Trace`)

IN: *A j-invariant $j \in \mathbb{F}_{2^n} \setminus \mathbb{F}_4$ of an elliptic curve E.*
OUT: *The trace of Frobenius $t = q + 1 - \#E(\mathbb{F}_q)$ of E.*

1. $N = \lceil \frac{n}{2} \rceil + 13$; $M = N - 10$;
2. $J =$ `Lift_First_J_Invariant`(j, N);
3. $CN = 1$; $CD = 1$;
4. `For` $i = 0$ `To` $n - 1$ `Do`
   - 4.1. $J' =$ `Lift_Previous_J_Invariant`(J, N);
   - 4.2. $Z = -\dfrac{(J^2 + 195120J + 4095J' + 660960000)/2^{12}}{(J^2 + J(563760 - 512J') + 372735J' + 8981280000)/2^9}$;
   - 4.3. $T = (12Z^2 + Z)(J' - 1728) - 36$;
   - 4.4. $CN = CN \times (J' - (504 + 12096Z)T)$;
   - 4.5. $CD = CD \times (240T + J')$;
   - 4.6. $J = J'$;
5. $t =$ `Sqrt`$(CN/CD, 1, M) \bmod 2^{M-1}$;
6. `If` $t > 2\sqrt{q}$ `Then` $t = t - 2^{M-1}$;
7. `Return` t.

Thus to compute c_i^2, we need an expression for half the x-coordinate of the non-trivial point $Q_i \in \mathrm{Ker}(\widehat{\Sigma}_i)$. Again we follow [14], but considerably simplify the formula for z_i.

Proposition 4 *Let $Q_i = (x_i, y_i)$ be the non-trivial point in $\mathrm{Ker}(\widehat{\Sigma}_i)$ and let $z_i = x_i/2$, then*

$$z_i = -\frac{(j(\mathcal{E}_{i+1})^2 + 195120j(\mathcal{E}_{i+1}) + 4095j(\mathcal{E}_i) + 660960000)/2^{12}}{(j(\mathcal{E}_{i+1})^2 + j(\mathcal{E}_{i+1})(563760 - 512j(\mathcal{E}_i)) + 372735j(\mathcal{E}_i) + 8981280000)/2^9}. \tag{18}$$

Combining the above propositions we can compute $c^2 = \prod_{i=0}^{n-1} c_i^2$. Since the trace of Frobenius t satisfies $t \equiv c \bmod q$ and $|t| \leq 2\sqrt{q}$, we have $t \equiv c \bmod 2^{\lceil \frac{n+4}{2} \rceil}$. The 2-adic square root can be found via a Newton iteration for the inverse square root, i.e. via a Newton iteration on $s(X) = c^2X^2 - 1$. Clearly, we have $s(1/c) = 0$ and $s'(1/c) \equiv 0 \bmod 2$. Furthermore, $c \equiv 1 \bmod 4$, since E has a point of order 4 and thus $s'(1/c) \not\equiv 0 \bmod 4$. The vanishing of $s'(1/c)$ modulo 2 means that we lose exactly one bit of precision in the computation of the square root and therefore we need to compute c^2 modulo $2^{\lceil \frac{n+6}{2} \rceil}$. Substituting the expressions for z_i and t_i in c_i^2, we see that we have to determine the j-invariants $j(\mathcal{E}_i)$ with precision $2^{\lceil \frac{n}{2} \rceil + 13}$. This finally leads to the main algorithm `Compute_Trace`. In step 5 we use the function `Sqrt`, which computes the 2-adic square root of c^2 with precision M, such that $c \equiv 1 \bmod 4$.

5 Implementation

In this section we give practical run times and memory usages of both the original Satoh lifting-algorithm combined with the simplified formulae taken from [14] and our $O(n^2)$ memory version for elliptic curves in the range of interest to cryptography. Both algorithms have been implemented in the `C` programming language on a AMD Thunderbird 1 GHz PC with 384 MB of main memory, running Linux Redhat 6.2. All programs were compiled using the gcc compiler, version 2.7.2.3. Before giving the actual results we make some comments on our implementation.

Since efficiency was our main goal, we have written the basic operations on multiple precision integers in assembly. These include: addition and subtraction, shift left/right and the multiplication of a multi-precision integer by a word. To minimize the loop overhead for small multiple precision integers, i.e. integers which fit in four words or less, we implemented unrolled versions of the above-mentioned operations.

Elements of $\mathbb{F}_{2^n}$ are represented with respect to a standard polynomial basis, i.e. as polynomials over $\mathbb{F}_2$ modulo a degree n irreducible polynomial f. By choosing f as a trinomial or a pentanomial, reduction modulo f becomes very efficient. The same polynomial f is used to construct $\mathcal{R}$ mod 2^N as $(\mathbb{Z}/2^N\mathbb{Z})[T]/(f(T))$. Multiplication of two elements in $\mathcal{R}$ mod 2^N is implemented using Karatsuba's trick in the polynomial dimension and classical multiplication for the coefficients.

In Table 1 we compare the characteristic 2 version of Satoh's original algorithm with our $O(n^2)$ memory version for finite fields $\mathbb{F}_{2^n}$ relevant to cryptographical applications. The data in this table show that our algorithm is faster by a constant factor of about 1.5 and that the memory requirements are considerably lower than for Satoh's original algorithm. We note that our current implementation is more optimized towards speed than it is towards minimizing memory usage. Therefore it would be possible to lower the memory requirements by another 30%. Since a smart card typically has 32 KB of memory (in the near future this will be 64 KB), it becomes feasible to generate secure elliptic curves on a smart card.

Table 1. Run times and memory usage of Satoh's algorithm versus the $O(n^2)$ memory version on an AMD 1 GHz

Field size n	Original Satoh		$O(n^2)$ memory version	
	Time (s)	Memory (KB)	Time (s)	Memory (KB)
160	5.43	315	3.17	30
180	9.11	534	5.64	44
200	11.8	650	7.41	48
220	15.4	790	9.83	54
240	28.1	1162	15.8	73
260	36.0	1371	20.3	80
280	44.1	1574	25.1	86
300	64.3	2180	39.2	109
340	88.7	2790	55.3	125
380	133	4052	82.7	162
420	195	5643	123	197
460	244	6756	154	224
500	400	8964	225	275

6 Conclusion

In this paper we have presented a new version of Satoh's algorithm which only needs $O(n^2)$ memory, where the original algorithm needs $O(n^3)$ memory. Furthermore, we showed that our algorithm still runs in $O(n^{3+\varepsilon})$ bit operations, which equals the run time complexity of Satoh's original algorithm. Our version relies on univariate Newton iterations where Satoh also uses multivariate Newton iterations. In our implementation, this resulted in a speed-up of a factor of about 1.5. As a result of the $O(n^2)$ memory complexity, it now becomes feasible to generate secure elliptic curves on a smart card.

Acknowledgements

The authors are very grateful to Takakazu Satoh and Jan Denef for many interesting discussions and helpful remarks on this work.

References

1. A.O.L. Atkin. The number of points on an elliptic curve modulo a prime. *Series of e-mails to the* `NMBRTHRY` *mailing list*, 1992.
2. J.M. Couveignes. *Quelques calculs en théorie des nombres*. PhD thesis, Université de Bordeaux, 1994.
3. J.M. Couveignes. Computing l-isogenies with the p-torsion. *ANTS-II, Lecture Notes in Comp. Sci.*, 1122:59–65, 1996.
4. J.A. Csirik. Counting the number of points on an elliptic curve on a low-memory device. 1998. Preprint.
5. M. Deuring. Die Typen der Multiplikatorenringe elliptischer Funktionenkörper. *Abh. Math. Sem. Univ. Hamburg*, 14:197–272, 1941.
6. N. Elkies. Elliptic and modular curves over finite fields and related computational issues. *Computational Perspectives on Number Theory*, pages 21–76, 1998.
7. M. Fouquet, P. Gaudry, and R. Harley. On Satoh's algorithm and its implementation. *J. Ramanujan Math. Soc.*, 15:281–318, 2000.
8. H. Hasse. Beweis des Analogons der Riemannschen Vermutung für die Artinschen und F. K. Smidtschen Kongruenzzetafunctionen in gewissen elliptischen Fällen. *Ges. d. Wiss. Nachrichten. Math.-Phys. Klasse*, pages 253–262, 1933.
9. R. Lercier. *Algorithmique des courbes elliptiques dans les corps finis*. PhD thesis, L'École Polytechnique, Laboratoire D'Informatique, CNRS, Paris, June 1997.
10. J. Lubin, J.P. Serre, and J. Tate. Elliptic curves and formal groups. *Lecture notes prepared in connection with the seminars held at the Summer Institute on Algebraic Geometry, Whitney Estate, Woods Hole, Massachusetts*, 1964.
11. T. Satoh. The canonical lift of an ordinary elliptic curve over a finite field and its point counting. *J. Ramanujan Math. Soc.*, 15:247–270, 2000.
12. R. Schoof. Elliptic curves over finite fields and the computation of square roots mod p. *Math. Comput.*, 44:483–494, 1985.
13. J.P. Serre. *Local Fields*, volume 67 of *Graduate Texts in Mathematics*. Springer-Verlag, 1979.
14. B. Skjernaa. Satoh's algorithm in characteristic 2. *Preprint*, 2000.
15. J. Vélu. Isogénies entre courbes elliptiques. *C.R. Acad. Sc. Paris*, 273:238–241, 1971.