Fast amortized Kate proofs

Dankrad Feist and Dmitry Khovratovich
Ethereum Foundation

March 19, 2020

1 Definitions

1.1 Setup

Let g be a group $\mathbb{G}$ element, and denote $[a] = a \cdot g$ a group element where a is an integer.

Let s be a secret, then a universal setup of degree m consists of m elements of $\mathbb{G}$:

$$[s], [s^2], \ldots, [s^m].$$

1.2 Commitment

Let $f(X) = \sum_{0 \le i \le m} f_i X^i$ be a polynomial of degree m. Then a commitment $C_f \in \mathbb{G}$ is defined as

$$C_f = \sum_{0 \le i \le m} f_i [s^i],$$

being effectively an evaluation of f at point s.

1.3 Proof

Note that for any y we have that $(X - y)$ divides $f(X) - f(y)$. Then the proof that $f(y) = z$ is defined as

$$\pi[f(y) = z] = C_T,$$

where $T_y(X) = \frac{f(X)-z}{X-y}$ is a polynomial of degree $(m-1)$.

Note that a proof can be constructed using m scalar multiplications in the group. The coefficients of T are computed with one multiplication each:

$$T_y(X) = \sum_{0 \le i \le m-1} t_i X^i; \tag{1}$$

$$t_{m-1} = f_m; \tag{2}$$

$$t_j = f_{j+1} + y \cdot t_{j+1}. \tag{3}$$

Expanding on the last equation, we get

$$T_y(X) = f_m X^{m-1} + (f_{m-1} + y f_m) X^{m-2} + (f_{m-2} + y f_{m-1} + y^2 f_m) X^{m-3} + \\ + (f_{m-3} + y f_{m-2} + y^2 f_{m-1} + y^3) X^{m-4} + \cdots + (f_1 + y f_2 + y^2 f_3 + \cdots + y^{m-1} f_m). \tag{4}$$

2 Multiple proofs

Let w be a 2^n-th root of unity. We show how to construct Kate proofs for $w, w^2, w^3, \ldots, w^{2^n} = 1$.

Note that a proof for w^k is

$$\pi[f(w^k) = z^k] = C_{T_{w^k}} = f_m[s^{m-1}] + (f_{m-1} + w^k f_m)[s^{m-2}] + (f_{m-2} + w^k f_{m-1} + w^{2k} f_m)[s^{m-3}] + \\ + (f_{m-3} + w^k f_{m-2} + w^{2k} f_{m-1} + w^{3k})[s^{m-4}] + \cdots + (f_1 + w^k f_2 + w^{2k} f_3 + \cdots + w^{(m-1)k} f_m). \tag{5}$$

Regrouping the terms, we get (for $2^n \geq m$):

$$C_{T_{w^k}} = \left(f_m[s^{m-1}] + f_{m-1}[s^{m-2}] + f_{m-2}[s^{m-3}] + \cdots + f_2[s] + f_1\right) + \tag{6}$$
$$+ \left(f_m[s^{m-2}] + f_{m-1}[s^{m-3}] + f_{m-2}[s^{m-4}] + \cdots + f_3[s] + f_2\right) w^k + \tag{7}$$
$$+ \left(f_m[s^{m-3}] + f_{m-1}[s^{m-4}] + f_{m-2}[s^{m-5}] + \cdots + f_4[s] + f_3\right) w^{2k} + \tag{8}$$
$$+ \left(f_m[s^{m-4}] + f_{m-1}[s^{m-5}] + f_{m-2}[s^{m-6}] + \cdots + f_5[s] + f_4\right) w^{3k} + \tag{9}$$
$$\cdots \tag{10}$$
$$+ (f_m[s] + f_{m-1})w^{(m-2)k} + f_m w^{(m-1)k}. \tag{11}$$

Let for $1 \leq i \leq 2^n$ denote

$$h_i = \left(f_m[s^{m-i}] + f_{m-1}[s^{m-i-1}] + f_{m-2}[s^{m-i-2}] + \cdots + f_{i+1}[s] + f_i\right).$$

with $h_i = 0$ for $i > m$. Then

$$C_{T_{w^k}} = h_1 + h_2 w^k + h_3 w^{2k} + \cdots + h_m w^{(m-1)k}. \tag{12}$$

Let us denote

$$\mathbf{C}_T = [C_{T_{w^1}}, C_{T_{w^2}}, \ldots, C_{T_{w^{2^n}}}]$$

and

$$\mathbf{h} = [h_1, h_2, \ldots, h_{2^n}]. \tag{13}$$

Recalling that the Discrete Fourier Transform of $\mathbf{a} = [a_1, a_2, \ldots, a_{2^n}]$ is

$$\widehat{\mathbf{a}} = [\widehat{a_1}, \widehat{a_2}, \ldots, \widehat{a_{2^n}}]$$

where

$$\widehat{a_k} = \sum_i a_i w^{ki}$$

Therefore, from Equation (12)

$$\mathbf{C}_T = \mathrm{DFT}(\mathbf{h}), \tag{14}$$

and can be computed in $O(n2^n)$ cost given $\mathbf{h}$ using FFT algorithms[1].

2.1 Computing h

Now we demonstrate that $\mathbf{h}$ can be also computed efficiently from $\{f_i\}$. Indeed, by definition

$$\begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ \vdots \\ h_{m-1} \\ h_m \end{bmatrix} = \begin{bmatrix} f_m & f_{m-1} & f_{m-2} & f_{m-3} & \cdots & f_1 \\ 0 & f_m & f_{m-1} & f_{m-2} & \cdots & f_2 \\ 0 & 0 & f_m & f_{m-1} & \cdots & f_3 \\ & & \ddots & & & \\ 0 & 0 & 0 & 0 & \cdots & f_{m-1} \\ 0 & 0 & 0 & 0 & \cdots & f_m \end{bmatrix} \cdot \begin{bmatrix} [s^{m-1}] \\ [s^{m-2}] \\ [s^{m-3}] \\ \vdots \\ [s] \\ 1 \end{bmatrix}$$

The matrix

$$A = \begin{bmatrix} f_m & f_{m-1} & f_{m-2} & f_{m-3} & \cdots & f_1 \\ 0 & f_m & f_{m-1} & f_{m-2} & \cdots & f_2 \\ 0 & 0 & f_m & f_{m-1} & \cdots & f_3 \\ & & \cdots & & & \\ 0 & 0 & 0 & 0 & \cdots & f_{m-1} \\ 0 & 0 & 0 & 0 & \cdots & f_m \end{bmatrix}$$

is a *Toeplitz* matrix. It is known that a multiplication of a vector by a $m \times m$ Toeplitz matrix costs $O(m \log m)$ operations using FFT `http://www.netlib.org/utk/people/JackDongarra/etemplates/node384.html`.

Therefore, we can first compute $\mathbf{h}$ from SRS and then all 2^n Kate proofs, i.e. $\mathbf{C}_T$, using $O(n2^n)$ multiplications compared to $O(2^{2n})$ for the naive approach (assuming $m = O(2^n)$).

[1]This works directly as described if $m \leq 2^n$. Otherwise, the vector $\mathbf{h}$ can be modified, "wrapping around" the extra $\neq 0$ terms, so the idea also works for $m > 2^n$.

3 Multi-reveal

Let $\psi \in \mathbb{F}_p$ be an ℓ-th root of unity ($\psi^\ell = 1$). Let's say we want to reveal the polynomial evaluations $f(y) = z_0$, $f(\psi y) = z_1$, ..., $f(\psi^{\ell-1}y) = z_{\ell-1}$.

Noting that $(x-y)\cdot(x-\psi y)\cdots(x-\psi^{\ell-1}y) = x^\ell - y^\ell$, the proof for this can be given by computing the polynomial

$$g(x) = f(x)//(x^\ell - y^\ell), \tag{15}$$

where // stands for the truncated long division, and then computing the proof

$$\pi[f(y) = z_0, \ldots, f(\psi^{\ell-1}y) = z_{\ell-1}] = [g(s)]. \tag{16}$$

This proof can be verified by computing the checking polynomial $h(x) = f(x) \mod (x^\ell - y^\ell)$ (which can be interpolated from the given values), and checking that

$$e(C_f, \cdot) = e(\pi[f(y) = z_0, \ldots, f(\psi^{\ell-1}y) = z_{\ell-1}], [s^\ell - y^\ell])e(h(s), \cdot). \tag{17}$$

3.1 Multiple multi-reveals – odd ℓ

We now want to generalise the result in section 2 to the case were each of the reveals is actually a multireveal.

Letting w a 2^n-th root of unity, we want to compute proofs

$$\begin{aligned}
\pi[f(1), \ldots, f(\psi^{\ell-1})] &= C_{T_{w^0,\ell}} && (18)\\
\pi[f(w), \ldots, f(w\psi^{\ell-1})] &= C_{T_{w^1,\ell}} && (19)\\
&\vdots \\
\pi[f(w^{2^n-1}), \ldots, f(w^{2^n-1}\psi^{\ell-1})] &= C_{T_{w^{2^n-1},\ell}}. && (20)
\end{aligned}$$

The proof for w^k is

$$\begin{aligned}
\pi[f(w^k), \ldots, f(w^k\psi^{\ell-1})] = C_{T_{w^k,\ell}} = f_m[s^{m-\ell}] + f_{m-1}[s^{m-\ell-1}] + \cdots + f_{m-\ell+1}[s^{m-2\ell+1}] + \\
+ (f_{m-\ell} + w^{k\ell}f_m)[s^{m-2\ell}] + (f_{m-\ell-1} + w^{k\ell}f_{m-1})[s^{m-2\ell-1}] + \cdots + (f_{m-2\ell+1} + w^{k\ell}f_{m-\ell+1})[s^{m-3\ell+1}] + \\
+ (f_{m-2\ell} + w^{k\ell}f_{m-\ell} + w^{2k\ell}f_m)[s^{m-3\ell}] + (f_{m-2\ell-1} + w^{k\ell}f_{m-\ell-1} + w^{2k\ell}f_{m-1})[s^{m-3\ell-1}] + \\
\cdots + (f_{m-3\ell+1} + w^{k\ell}f_{m-2\ell+1} + w^{2k\ell}f_{m-\ell+1})[s^{m-4\ell+1}] + \\
\vdots
\end{aligned} \tag{21}$$

Regrouping the terms, we get (for $2^n \geq m$):

$$\begin{aligned}
C_{T_{w^k,\ell}} = &\left(f_m[s^{m-\ell}] + f_{m-1}[s^{m-\ell-1}] + f_{m-2}[s^{m-\ell-2}] + \cdots + f_{\ell+1}[s] + f_\ell\right) + && (22)\\
&+ \left(f_m[s^{m-2\ell}] + f_{m-1}[s^{m-2\ell-1}] + f_{m-2}[s^{m-2\ell-2}] + \cdots + f_{2\ell+1}[s] + f_{2\ell}\right) w^{k\ell} + && (23)\\
&+ \left(f_m[s^{m-3\ell}] + f_{m-1}[s^{m-3\ell-1}] + f_{m-2}[s^{m-3\ell-2}] + \cdots + f_{3\ell+1}[s] + f_{3\ell}\right) w^{2k\ell} + && (24)\\
&+ \left(f_m[s^{m-4\ell}] + f_{m-1}[s^{m-4\ell-1}] + f_{m-2}[s^{m-4\ell-2}] + \cdots + f_{4\ell+1}[s] + f_{4\ell}\right) w^{3k\ell} + && (25)\\
&\vdots && (26)\\
&(\cdots)w^{\lfloor m/\ell \rfloor k\ell} && (27)
\end{aligned}$$

Let for $i \geq 1$ denote

$$h_i = \left(f_m[s^{m-i}] + f_{m-1}[s^{m-i-1}] + f_{m-2}[s^{m-i-2}] + \cdots + f_{i+1}[s] + f_i\right).$$

with $h_i = 0$ for $i > m$. Then

$$C_{T_{w^k,\ell}} = h_\ell + h_{2\ell}w^{k\ell} + h_{3\ell}w^{2k\ell} + \cdots + h_{m\ell}w^{(m-1)k\ell}. \tag{28}$$

Let us denote

$$\mathbf{C}_{T_\ell} = [C_{T_{w,\ell}}, C_{T_{w^2,\ell}}, \ldots, C_{T_{w^{2^n},\ell}}]$$

and

$$\mathbf{h}_\ell = [h_\ell, 0h_{\ell+1}, \ldots, 0h_{2\ell-1}, h_{2\ell}, 0h_{2\ell+1}, \ldots, 0h_{3\ell-1}, h_{3\ell+1}, 0h_{3\ell+1}, \ldots, 0h_{2^n+\ell-1}].$$

Then, from Equation (28)

$$\mathbf{C}_{T_\ell} = \mathrm{DFT}(\mathbf{h}_\ell), \tag{29}$$

and can be computed in $O(n2^n)$ cost given $\mathbf{h}_\ell$ using FFT algorithms.

3.1.1 Computing $\mathbf{h}_\ell$

In order to compute $\mathbf{h}_\ell$ we need to slightly modify the algorithm from section 2.1. First, define

$$\mathbf{h}_{\ell,i\ell,j} = f_{m-j}[s^{m-i\ell-j}] + f_{m-\ell-j}[s^{m-(i+1)\ell-j}] + f_{m-2\ell-j}[s^{m-(i+2)\ell-j}] + \cdots + f_{(m-j)\%\ell+(i+1)\ell}[s^{(m-j)\%\ell+\ell}] + f_{(m-j)\%\ell+i\ell}[s^{(m-j)\%\ell}]. \quad (30)$$

Using this,

$$\mathbf{h}_{i\ell} = \sum_{j=0}^{\ell-1} \mathbf{h}_{\ell,i\ell,j}. \quad (31)$$

The $\mathbf{h}_{\ell,i\ell,j}$ can be computed by the ℓ Toeplitz matrix multiplications:

$$\begin{bmatrix} h_{\ell,1\ell,j} \\ h_{\ell,2\ell,j} \\ h_{\ell,3\ell,j} \\ \vdots \\ h_{\ell,(\lfloor \frac{m-j}{\ell} \rfloor -1)\ell,j} \\ h_{\ell,\lfloor \frac{m-j}{\ell} \rfloor \ell,j} \end{bmatrix} = \begin{bmatrix} f_{m-j} & f_{m-1\ell-j} & f_{m-2\ell-j} & f_{m-3\ell-j} & \cdots & f_{(m-j)\%\ell+\ell} \\ 0 & f_{m-j} & f_{m-1\ell-j} & f_{m-2\ell-j} & \cdots & f_{(m-j)\%\ell+2\ell} \\ 0 & 0 & f_{m-j} & f_{m-1\ell-j} & \cdots & f_{(m-j)\%\ell+3\ell} \\ & & & & \ddots & \\ 0 & 0 & 0 & 0 & \cdots & f_{m-\ell-j} \\ 0 & 0 & 0 & 0 & \cdots & f_{m-j} \end{bmatrix} \cdot \begin{bmatrix} [s^{m-1\ell-j}] \\ [s^{m-2\ell-j}] \\ [s^{m-3\ell-j}] \\ \vdots \\ [s^{(m-j)\%\ell+\ell}] \\ [s^{(m-j)\%\ell}] \end{bmatrix}$$

These matrix multiplications can be done in the same way as the Toeplitz matrix multiplication in section 2.1 using FFTs. Note that the vector multiplied by the matrix is independent from the polynomial coefficients, so its Fourier transform can be precomputed, necessitating ℓ Fourier transforms of size $2m/\ell$ in the precompute. However, the output of the multiplication can be added before transforming back, so only one IFT of size $2m/\ell$ is necessary.

3.2 Multiple multi-reveals – ℓ power of two

Let $\ell = 2^r < 2^n$. Define $\psi = w^{2^{n-r}}$.

We want to compute the proofs

$$\begin{aligned} \pi[f(1), \ldots, f(\psi^{\ell-1})] &= C_{T_{w^0,\ell}} && (32) \\ \pi[f(w), \ldots, f(w\psi^{\ell-1})] &= C_{T_{w^1,\ell}} && (33) \\ &\vdots \\ \pi[f(w^{2^{n-r}-1}), \ldots, f(w^{2^{n-r}-1}\psi^{\ell-1})] &= C_{T_{w^{2^{n-r}-1},\ell}}. && (34) \end{aligned}$$

We do not need proofs for $w^{2^{n-r}}$ to w^{2^n-1}, as these would already be covered.

Things follow analogously to equation (28):

$$C_{T_{w^k,\ell}} = h_\ell + h_{2\ell} w^{k\ell} + h_{3\ell} w^{2k\ell} + \cdots + h_{m\ell} w^{(m-1)k\ell} \quad (35)$$

Define $\varphi = w^\ell$ and get

$$C_{T_{w^k,\ell}} = h_\ell + h_{2\ell} \varphi^k + h_{3\ell} \varphi^{2k} + \cdots + h_{m\ell} \varphi^{(m-1)k} \quad (36)$$

Defining

$$\mathbf{C}_{T_\ell} = [C_{T_w}, C_{T_{w^2}}, \ldots, C_{T_{w^{2^{n-r}}}}]$$

and

$$\mathbf{h}'_\ell = [h_\ell, h_{2\ell}, \ldots, h_{(2^{n-r}-1)\ell}],$$

we get

$$\mathbf{C}_{T_\ell} = \mathrm{DFT}_\varphi(\mathbf{h}_\ell), \quad (37)$$

and can be computed in $O((n-r)2^{n-r})$ cost given $\mathbf{h}'_\ell$ using FFT algorithms. The vector $\mathbf{h}'_\ell$ can be computed in a similar way as described in section 3.1.1.