CHAPTER 5

zkSNARKs from Extended Algebraic Programs

In this section we provide a new type of NIZK argument which allows to prove knowledge of a secret key, given the corresponding public key, with only one circuit gate instead of >864 as in current state-of-the-art implementations (disregarding the additional constraints that are required in order to assure that the input is valid). It is an extension of Groth's [Gro16] QAP-based zkSNARK construction, where we roughly speaking add the word for 'point multiplication on an elliptic curve', to the words '+' and '×'. Therefore the language of the arithmetic circuit, R1CS and QAP had to be extended consequently to what we name R1CS* and extended algebraic program (EAP) instead of the commonly used QAPs.

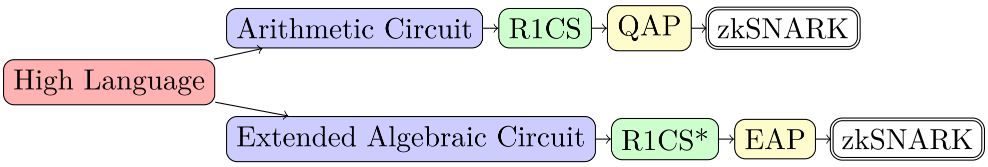


Figure 5.1: Shows the necessary steps to make a program written in a high level language (could be a subset of common C), provable with a zkSNARK. The upper path is the standard procedure. The path below our contribution.

The verifier work increases from 3 to 4 pairing function calls, 1 exponentiation in $\mathbb{G}_\mathbb{T}$, 1 in $\mathbb{G}_1$ and 2 in $\mathbb{G}_2$. Before this extension, proving knowledge of the discrete logarithm (e.g. I prove I know x s.t. $pk = g^x$ without revealing x) had to be done by expressing the point multiplication as arithmetic circuit. This required at least 864 gates for each point multiplication within a circuit, $N{\cdot}864$ for N point multiplications, which increases the common reference string (CRS) by at least $N \cdot 3 \cdot 864$ elements. Here it takes N gates and therefore adds $3N$ elements to the CRS. The prove time improves drastically as it is not necessary any more to compute the coefficients of the polynomials that were needed for proving knowledge of a valid arithmetic circuit assignment that expressed a scalar point multiplication. Consequently the prover also does not need to perform the corresponding blind evaluations, which require a point multiplication for each coefficient what therefore usually poses the biggest source of overhead. The polynomial division effort decrease as the degree is reduced by $\approx N{\cdot}864$. Before this scheme, it was almost infeasible to pack multiple point multiplications into one SNARK. Performing a shielded transaction in zCash using our extension can be performed in a few milliseconds instead of seconds. To make the proof elements indistinguishable from uniform random and therefore zero-knowledge, only one additional gate (see section 5.1.1)is required.

Our NIZK arguments for EAPs considers a circuit consisting of addition, multiplication and elliptic curve point multiplication gates. The computations are performed over a finite field $\mathbb{F}$. Figure 5.2 shows how a EAP enforced R1CS* constraint for a simple statement looks like. In the following lines we write $\langle \mathbf{A}, \mathbf{B} \rangle_{[N_i,N_j]} := \sum_{i=N_i}^{N_j} A_i B_i$, and $\langle \mathbf{A}, \mathbf{B} \rangle_{[N_j]} := \langle \mathbf{A}, \mathbf{B} \rangle_{[0,N_j]}$ for shortness. Also we use multiplicative notation for the elliptic-curve group operation.

An efficient-prover publicly verifiable non interactive argument is a quadruple of probabilistic polynomial-time algorithms

- $(\sigma_P, \sigma_V, \tau) \leftarrow$Setup$(R, 1^\lambda)$: where R is a poly-time decidable binary relation with elements $(\phi, w) \in R$ where ϕ is a statement and w the witness. λ is the security parameter.
  Setup returns two common reference strings: σ_P for the prover and σ_V for the verifier. The non-interactive argument is called publicly verifiable since σ_V can be deduced from σ_P. Otherwise it is called a designated-verifier argument, which are of minor importance for distributed ledgers and therefore not treated in this work. τ we call a simulation trapdoor. It allows the creation of 'fake'-proofs. Usually it expected to be deleted right after the setup has been performed.

- $\pi \leftarrow$ Prove (R, σ_P, ϕ, w): Takes the relation R such as 'I know w s.t. $w^2 + w = \phi$', where w is called the witness, and ϕ the statement. It outputs an argument π on the knowledge of w.

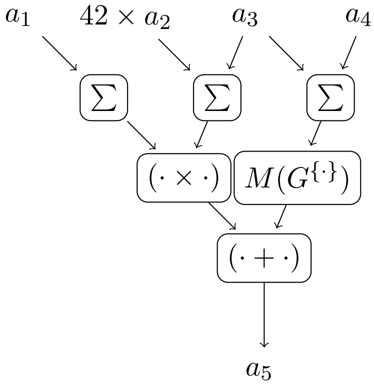
- 0,1$\leftarrow$Vfy(R, σ_V, ϕ, π): Takes a relation, statement and an argument as input and outputs '0' (reject) if the argument is not convincing or '1' (accept) otherwise.

- $\pi \leftarrow$Sim(R, τ, ϕ) creates arguments for a statement ϕ without a witness, but using the trapdoor instead.

The description of a EAP is

$$EAP := (\mathbb{F}, e(\cdot,\cdot), \mathcal{G}_M^{\mathcal{A}}(\cdot), l, \{L_i(X), R_i(X), E_i(X), O_i(X)\}_{i=0}^m, D(X)),$$

where



$$\cong \left\{ \underbrace{\begin{bmatrix}1\\0\\0\\0\\0\end{bmatrix}}_{\text{L}}, \underbrace{\begin{bmatrix}0\\42\\1\\0\\0\end{bmatrix}}_{\text{R}}, \underbrace{\begin{bmatrix}0\\0\\1\\1\\0\end{bmatrix}}_{\text{E}}, \underbrace{\begin{bmatrix}0\\0\\0\\0\\1\end{bmatrix}}_{\text{O}} \right\}_{R1C^*}, \underbrace{\begin{bmatrix}a_1\\a_2\\a_3\\a_4\\a_5\end{bmatrix}}_{\text{W}}$$

Figure 5.2: Shows how the example statement $a_1 \times (42 \times a_2 + a_3) + M\left(G^{a_3+a_4}\right) = a_5$ is realizable with one EAP gate and hence can be expressed as a single extended Rank 1 Constraint (R1C*). The prover will convince the verifier that he has knowledge of W s.t. it satisfies: $\mathcal{G}_M^{\mathcal{A}}\left(\langle E, W\rangle\right) + \langle L, W\rangle \cdot \langle R, W\rangle = \langle O, W\rangle$.

- $\mathbb{F}$ is a finite field.
- e is a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ as described in section 4.3. G as $\mathbb{G}_1$ and H as $\mathbb{G}_2$ generators.
- n is the number of multiplication gates in a circuit
- m is the number of wires of the circuit
- $\mathcal{A} \subset \mathbb{N}_{\geq 0}$ defines the set of gate indexes. We enumerate the gates from 0 to $n-1$ e.g. $\mathcal{A} = \{0, .., n-1\}$
- $L_i(X), R_i(X), E_i(X), O_i(X)$ are polynomials of degree $n-1$ that express the scale factor for the i-th wire at gate X. For example: $L_3(42) = 1$ states that the 3rd wire is the 'left' multiplication input of gate with index 42. $R_7(42) = 2$ implies, that the 7th wire is 'right' multiplication input of gate with index 42 and that this wires assigned signal will be scaled by the linear factor of 2. $E_7(42) = 7$ implies, that the 7th wire is used as an input to gate the with index 42 and whatever will be assigned to this wire, it will be amplified by a factor of 7. We write **L** for the vector of polynomials $(L_0(X), L_1(X), .., L_m(X))$ (same for **R,E,O**).
- Each EAP-gate has a uniquely assigned index $r_i \in \mathcal{A}$, which compose the domain polynomial $D(X) = \prod_{r_g \in \mathcal{A}}(X - r_g)$. The degree $deg(D(X)) = n = |\mathcal{A}|$ consequently.
- A crucial role plays the projection function $M(\cdot) : \mathbb{G}_1 \mapsto \mathbb{F}$. Its best possible design currently exceed our understanding. Its most trivial realization would be taking x coordinate of the curve point. However: This would reduce security by factor 2 since for each curve point (X,Y) (except the roots) exists (X,-Y). It reduces security even

further by a smaller factor since we have to compensate the outputs of of the point multiplication results with the assignments which end up in a linear combination with O_i. The assignments are computed in the field $\mathbb{F}$ which is preferably of the same order then the curve is defined on. In cryptographic constructions the curve order is co-prime to $|\mathbb{F}|$. Therefore we lose information as point multiplication input is modulo the curve order eg. $G^x = G^{x \bmod |\langle G \rangle|}$. It has turned out to be very beneficial if the field order $|\mathbb{F}|$ where the arithmetic operations are performed on is equivalent to the order of the field on which the curve is defined. This brings the advantage that curve operations can be performed within the circuit using far fewer gates.

- $\mathcal{G}_M^{\mathcal{A}}(P(X)) \to \widetilde{P}(X)$, where $\mathcal{A}$ is the set of gate indexes and $|\mathcal{A}| = k+1$ takes as input a polynomial of degree k and returns a polynomial of degree k s.t. $\forall x \in \mathcal{A}$: $\widetilde{P}(x) = M\left(G^{P(x)}\right)$.

A EAP is defined by the binary relation

$$R = \left\{ (\phi\ ,\mathrm{w}) \left| \begin{array}{l} \phi = (a_1, ..., a_l) \in \mathbb{F}^l \\ w = (a_{l+1}, ..., a_m) \in \mathbb{F}^{m-1} \\ \mathcal{G}_M^{\mathcal{A}}\left(\langle \mathbf{a}, \mathbf{E} \rangle\right) + \langle \mathbf{a}, \mathbf{L} \rangle \cdot \langle \mathbf{a}, \mathbf{R} \rangle \equiv \langle \mathbf{a}, \mathbf{O} \rangle \bmod D(X) \end{array} \right. \right\} \tag{5.1}$$

for some degree $n-2$ quotient polynomial $Q(X)$. This gives us the NIZK argument:

- $(\sigma, \tau) \leftarrow$Setup$(R, 1^\lambda)$: Pick $\alpha, \beta, \gamma, \delta \xleftarrow{\$} \mathbb{Z}_p^*, x \xleftarrow{\$} \mathcal{A}$. Define $\tau = (\alpha, \beta, \gamma, \delta, x)$. The common reference string is:

$$\sigma = \left(\begin{array}{c} \left\{ G^{\frac{\alpha R_i(x) + \beta(L_i(x) + E_i(x)) + O_i(x)}{\gamma}} \right\}_{i=0}^{l}, \left\{ G^{\frac{\alpha R_i(x) + \beta(L_i(x) + E_i(x)) + O_i(x)}{\delta}} \right\}_{i=l+1}^{m}, \\ G^\alpha, G^\beta, G^\delta, H^\beta, H^\gamma, H^\delta, \left\{ G^{\frac{x^i D(x)}{\delta}} \right\}_{i=0}^{n-1}, \left\{ G^{x^i} \right\}_{i=0}^{n-1}, \left\{ H^{x^i} \right\}_{i=0}^{n-1} \end{array} \right)$$

- $\pi = (A, B, C, F) \leftarrow$ Prove (EAP, σ, ϕ, w): Compute

$$\begin{array}{rcl} A &=& G^{\alpha + \langle \mathbf{a}, \mathbf{L}(x) \rangle_{[m]} + \langle \mathbf{a}, \mathbf{E}(x) \rangle_{[m]}} \\ B &=& H^{\langle \mathbf{a}, \mathbf{R}(x) \rangle_{[m]}} \\ C &=& G^{\frac{\langle \mathbf{a}, \beta \mathbf{L}(x) + \alpha \mathbf{R}(x) + \beta \mathbf{E}(x) + \mathbf{O}(x) \rangle_{[l+1,m]} + Q(x) D(x)}{\delta}} \\ F &=& G^{\langle \mathbf{a}, \mathbf{E}(x) \rangle_{[m]}} \end{array}$$

- $0/1 \leftarrow$Vfy(EAP, σ, ϕ, π) accept iff:

$$e\left(A, B \cdot H^\beta\right) \cdot e\left(G, H\right)^{M(F)} = \\ \left(e\left(G^\alpha, H^\beta\right) \cdot e\left(G^{\frac{\langle \mathbf{a}, \alpha \mathbf{R}(x) + \beta(\mathbf{L}(x) + \mathbf{E}(x)) + \mathbf{O}(x) \rangle_{[l]}}{\gamma}}, H^\gamma\right) \cdot e\left(C, H^\delta\right) \cdot e\left(F, B\right) \right)$$

Regarding efficiency we observe that the one-time setup σ runs in time linear to the circuits size $\mathcal{O}(|C|)$. The prover must perform $\mathcal{O}(|C|)$ cryptographic work and $\mathcal{O}(|C|log^2(|C|))$ to compute $Q(x)$. The polynomial representation $L_i(X), R_i(X)$ etc. is not needed for proving and the prover rather works with the vectors $l_i = (L_i(1), L_i(2), .., L_i(n)), r_i = (R_i(1), R_i(2), .., R_i(n))$ where most elements are 0. This sparsity of the evaluation vectors is then exploited. The $\mathcal{O}(|C|log^2(|C|))$ runtime is achieved by using FFT techniques and binary tree based polynomial interpolation algorithms as it is done in the Pinocchio-Protocol[PHGR13] instead of naive Lagrange interpolation and polynomial division which would take $\mathcal{O}(n^2)$.

Note that $e\left(G, H\right)$ and $e\left(G^\alpha, H^\beta\right)$ can be precomputed. Furthermore verification requires l exponentiations e.g. proofing time is quasi linear in the statement size. Constructing the pairing-friendly elliptic curves such that $\mathbb{G}_1$ representations of group elements are smaller, C is assigned to the first group for efficiency. For the same reason the verifier includes the statement ϕ into the verification process over the group $\mathbb{G}_1$.

We proof perfect completeness by direct verification:

$$
\begin{aligned}
e\left(A, B \cdot H^\beta\right) \cdot e\left(G, H\right)^{M(F)} &= \\
e\left(A, B \cdot H^\beta\right) \cdot e\left(G, H\right)^{\mathcal{G}_M^{\mathcal{A}}(\langle \mathbf{a}, \mathbf{E}\rangle)} &= \\
e\left(G, H\right)^{\left(\alpha + \langle \mathbf{a}, \mathbf{L}\rangle_{[m]} + \langle \mathbf{a}, \mathbf{E}\rangle_{[m]}\right)\left(\langle \mathbf{a}, \mathbf{R}\rangle_{[m]} + \beta\right) + M(F)} &= \\
e\left(G, H\right)^{\alpha\langle \mathbf{a}, \mathbf{R}\rangle_{[m]} + \langle \mathbf{a}, \mathbf{L}\rangle_{[m]}\langle \mathbf{a}, \mathbf{R}\rangle_{[m]} + M(F) + \langle \mathbf{a}, \mathbf{E}\rangle_{[m]}\langle \mathbf{a}, \mathbf{R}\rangle_{[m]} + \alpha\beta + \beta\langle \mathbf{a}, \mathbf{E}+\mathbf{L}\rangle_{[m]}} &= \\
e\left(G, H\right)^{\alpha\langle \mathbf{a}, \mathbf{R}\rangle_{[m]} + \langle \mathbf{a}, \mathbf{O}\rangle_{[m]} + QD + \langle \mathbf{a}, \mathbf{E}\rangle_{[m]}\langle \mathbf{a}, \mathbf{R}\rangle_{[m]} + \alpha\beta + \beta\langle \mathbf{a}, \mathbf{E}+\mathbf{L}\rangle_{[m]}} &= \\
e\left(G^\alpha, H^\beta\right) \cdot e\left(G, H\right)^{\langle \mathbf{a}, \alpha\mathbf{R} + \beta\mathbf{E} + \beta\mathbf{L} + \mathbf{O}\rangle_{[m]} + QD + \langle \mathbf{a}, \mathbf{E}\rangle_{[m]}\langle \mathbf{a}, \mathbf{R}\rangle_{[m]}} &= \\
e\left(G^\alpha, H^\beta\right) \cdot e\left(G, H\right)^{\langle \mathbf{a}, \alpha\mathbf{R} + \beta\mathbf{E} + \beta\mathbf{L} + \mathbf{O}\rangle_{[m]} + QD} \cdot e\left(F, B\right) &= \\
e\left(G^\alpha, H^\beta\right) \cdot e\left(G^{\frac{\langle \mathbf{a}, \alpha\mathbf{R} + \beta\mathbf{E} + \beta\mathbf{L} + \mathbf{O}\rangle_{[l]}}{\gamma}}, H^\gamma\right) \cdot e\left(C, H^\delta\right) \cdot e\left(F, B\right) & \quad \square
\end{aligned}
$$

Note that the first transformation $M(F(x)) = M\left(G^{\langle \mathbf{a}, \mathbf{E}(x)\rangle_{[m]}}\right) = \mathcal{G}_M^{\mathcal{A}}\left(\langle \mathbf{a}, \mathbf{E}\rangle\right)(x)$ can only be applied because $x \in \mathcal{A}$. For that reason this scheme is not secure. If the E_i polynomials are all 0 however,e.g. we don not use the point multiplication in in the entire circuit, then we can safely pick $x \xleftarrow{\$} \mathbb{Z}_p^*$ from the entire group and end up with a scheme similar to the original[Gro16] which is statistically knowledge sound again.

We try to argue why it might be sound (disregarding the issue with the small space from which x can be chosen). Lets call the exponents of the proof elements a, b, c, f where $A = G^a, B = H^b, C = G^c, F = G^f$. We also define $In = \frac{\langle \mathbf{a}, \alpha\mathbf{R}(x) + \beta\mathbf{E}(x) + \beta\mathbf{L}(x) + \mathbf{O}(x)\rangle_{[l]}}{\gamma}$ as the input that will end up in the exponent a verifier will produce given the statement $\phi = (a_1, .., a_l)$. The equation for which an adversary effectively has to find a satisfying assignment is

$$a(b + \beta) + M(F) = \alpha \cdot \beta + In \cdot \gamma + c \cdot \delta + f \cdot b. \tag{5.2}$$

We now make the following observation:

- a, b, c, f are under 100% control of the adversary. He can forge them at will.
- In is depended on $\phi = (a_1, .., a_l)$ and the identity $a_0 = 1$. An PPT adversary cannot forge In at will. He can however set it to 0 if $a_1 = a_2 = .. = a_l = 0 \wedge L_0 = R_0 = E_0 = O_0 = 0 \implies In = 0$. If the identity a_0 is used in the circuit e.g. at least one of the wire polynomials L_0, R_0, E_0, O_0 is unequal to 0 then $In \neq 0$. Notice that the unlikely possibility exists, where there is $k \in \{1, .., l\}$ s.t. $L_k = L_0, R_k = R_0, E_k = E_0, O_0 = O_k$. Then setting $a_k = |\mathbb{F}|$ pushes $In = 0$ again.
- $M(F)$ is defined as $M(G^f)$. DLP on $\mathbb{G}_1$ is assumed to be hard, therefore a PPT adversary cannot forge $M(F)$ at will.
- $\alpha, \beta, \gamma, \delta$ are unknown to the attacker, however G^α etc. are known and part of the CRS. Therefore he can set $a, c, f \in \{\alpha, \beta, \delta\}$ and $b \in \{\beta, \gamma, \delta\}$, without knowing $\{\alpha, \beta, \gamma, \delta\}$ explicitly.
- If there was an efficiently computable isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$, he can set a, c, f to γ in addition to above named choices.

We now try to reduce equation 5.2 to the seemingly easiest satisfiable form by setting $A = G^\alpha$,$b = c = 0$. Then it remains for an attacker to find an assignment for:

$$M(F) = In \cdot \gamma. \tag{5.3}$$

In case M is the trivial function that takes a curve point and returns its X-coordinate and $G^0 = (0, 1)$, an adversary can create a satisfying assignment by setting $\phi = (0, .., 0)$. Consequently a verifier needs to check the proof elements or the statement, and reject if among them is one of the trivial assignments. If the trivial cases are rejected, finding a valid assignment therefore always ends up facing the DLP, which is assumed to be hard.

We want to point out, that Groth's protocol [Gro16] upon we build this work, has the same trivial satisfying assignment. Using the same tricks and argumentation leads to:

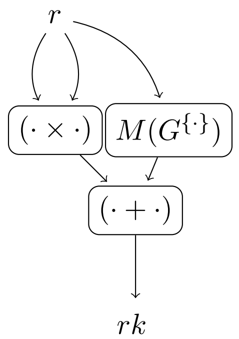
$$0 = In \cdot \gamma, \tag{5.4}$$

which is satisfiable if $\phi = (0, .., 0), A = G^\alpha, B = H^\beta, C = G^0$. In both schemes the trivial assignment is not an immanent thread however since $a_0 = 1$ and it is very unlikely, that the identity element is never used in any real world application circuit.

5.1 Special EAP Gate

5.1.1 Perfect Zero-Knowledge Gate

Since the prover did include any randomness so far, the proof elements are not zero-knowledge. Proofing the same statement twice leads to equivalent proof elements and consequently an adversary can learn more then the just the correctness of the statement. To make the proof elements indistinguishable from uniform random, we introduce a randomization gate. This gate has no connection to the rest of the circuit and is suggested to be the last gate regarding its index. The witness vector has to be extended by two more elements a_m and a_{m-1} to satisfy this gate. Its structure and constraint representation is shown in figure 5.3. The prover picks $r \overset{\$}{\leftarrow} \mathbb{Z}_p^*$ and set $a_m = r * r + M(G^r)$ and $a_{m-1} = r$. The proof elements are now uniformly random.

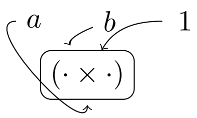


$$\rightsquigarrow \left\{ \underbrace{\begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \end{bmatrix}}_{\text{L}}, \underbrace{\begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \end{bmatrix}}_{\text{R}}, \underbrace{\begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \end{bmatrix}}_{\text{E}}, \underbrace{\begin{bmatrix} 0 \\ \vdots \\ 0 \\ 0 \\ 1 \end{bmatrix}}_{\text{O}} \right\}_{R1C^*}, \underbrace{\begin{bmatrix} a_0 = 1 \\ \vdots \\ a_{m-2} \\ a_{m-1} = r \\ a_m \end{bmatrix}}_{\mathbf{a}}$$

Figure 5.3: Shows the randomization gate and constraint. It enforces $a_m \equiv a_{m-1} \cdot a_{m-1} + M(G^{a_{m-1}}) \bmod p$. Since the r^2 might reveal some information (gut feeling) the randomization circuit could use three different random inputs for the price of increasing $|\mathbf{a}|$ by 4 instead of 2 elements.

5.1.2 Equality Gate

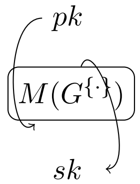
Checking weather two values are equal (or more precisely: Checking weather two values are in the same residue class) is essential for every R1CS as well as ER1CS based proof system. It is needed to verify signatures, knowledge of some DLP relation, proving that some derived Merkle-root hash is equal to some input hash etc. In general the equality gate enables recursive proving e.g. on can provide a proof that contains the verification of some other proof which in turn could be a proof of some other proof and so on. In therms of R1CS/ER1Cs description such a gate that ensures that $a = b$ is surprisingly simple to realize via:



$$\rightsquigarrow \left\{ \underbrace{\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}}_{\text{L}}, \underbrace{\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}}_{\text{R}}, \underbrace{\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}}_{\text{E}}, \underbrace{\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}}_{\text{O}} \right\}_{R1C^*}, \underbrace{\begin{bmatrix} 1 \\ a \\ b \end{bmatrix}}_{\mathbf{a}}.$$

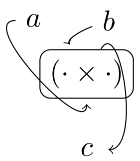
5.1.3 Inverse Gate

In case a public key is passed into the circuit as an argument and we want to proof knowledge of the secret key, we immediately observe that one gate to perform this inversion is sufficient since gates are only a guarantee that the input-output relation is satisfied and therefore they can be used in either direction.



$$\rightsquigarrow \quad \left\{ \underbrace{\begin{bmatrix}0\\0\end{bmatrix}}_{\text{L}}, \underbrace{\begin{bmatrix}0\\0\end{bmatrix}}_{\text{R}}, \underbrace{\begin{bmatrix}0\\1\end{bmatrix}}_{\text{E}}, \underbrace{\begin{bmatrix}1\\0\end{bmatrix}}_{\text{O}} \right\}_{R1C^*}, \underbrace{\begin{bmatrix}pk\\sk\end{bmatrix}}_{\mathbf{a}}.$$

Division can be understood likewise. Consider the example where we need to compute $c = a/b$ given a, b. The corresponding circuit and R1C* is



$$\rightsquigarrow \quad \left\{ \underbrace{\begin{bmatrix}0\\1\\0\end{bmatrix}}_{\text{L}}, \underbrace{\begin{bmatrix}0\\0\\1\end{bmatrix}}_{\text{R}}, \underbrace{\begin{bmatrix}0\\0\\0\end{bmatrix}}_{\text{E}}, \underbrace{\begin{bmatrix}1\\0\\0\end{bmatrix}}_{\text{O}} \right\}_{R1C^*}, \underbrace{\begin{bmatrix}a\\b\\c\end{bmatrix}}_{\mathbf{a}}.$$

5.1.4 Combined Gate

Consider a example where we want to apply multiplication and curve group multiplication at once in one gate. It can be done but it is **very important** to notice possible side effects! Lets say we have a, b, c and want to compute $d = a * b$ and $f = M(G^c)$:

$$\left\{ \underbrace{\begin{bmatrix}1\\0\\0\\0\\0\end{bmatrix}}_{\text{L}}, \underbrace{\begin{bmatrix}0\\1\\0\\0\\0\end{bmatrix}}_{\text{R}}, \underbrace{\begin{bmatrix}0\\0\\1\\0\\0\end{bmatrix}}_{\text{E}}, \underbrace{\begin{bmatrix}0\\0\\0\\1\\1\end{bmatrix}}_{\text{O}} \right\}_{R1C^*}, \underbrace{\begin{bmatrix}a\\b\\c\\d\\f\end{bmatrix}}_{\mathbf{a}},$$

would be **insufficient** since all we could derive from this assignment is that $d + f = a * b + M(G^c)$. If we continue to use d and f we loose their determinism, however they remain entangled. If another constraint assigns a value to one of them, the other one will be fixed too. This phenomena we call 'un-spooky action in a circuit'.

5.1.5 Simple Elliptic Curve Point Multiplication Circuit

Lets consider the case, where only one point multiplication defines the circuit e.g. the prover wants to convince the verifier, that he knows sk s.t $pk = M(G^{sk})$ without revealing sk. M maps a curve point onto its x coordinate for example. We therefore know that there must exist a second distinct sk that map to pk, however this is still sufficiently secure for this example. The circuit, its corresponding constraint and the assignment vector become

$$\begin{array}{c} sk \\ \downarrow \\ \boxed{M(G^{\{\cdot\}})} \\ \downarrow \\ pk \end{array} \quad \rightsquigarrow \quad \left\{ \underbrace{\begin{bmatrix} 0 \\ 0 \end{bmatrix}}_{\text{L}}, \underbrace{\begin{bmatrix} 0 \\ 0 \end{bmatrix}}_{\text{R}}, \underbrace{\begin{bmatrix} 1 \\ 0 \end{bmatrix}}_{\text{E}}, \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{\text{O}} \right\}_{R1C^*}, \underbrace{\begin{bmatrix} sk \\ pk \end{bmatrix}}_{\mathbf{a}},$$

therefore the polynomials $L_1(X) = L_2(X) = R_1(X) = R_2(X) = E_1(X) = O_2(X) = 0$, $E_2(X) = O_1(X) = 1$ are simple constant lines. The domain polynomial times the extracted polynomial therefore also $Q(X)D(X)$=0. The statement $\phi = (pk)$, the witness $w = (sk)$.

The proof elements become:

$$\begin{aligned} A &= G^{\alpha+sk} \\ B &= H^0 \\ C &= G^{\frac{\beta sk}{\delta}} \\ F &= G^{sk} \end{aligned}$$

The verifier checks if

$$\begin{aligned} e\left(A, B \cdot H^\beta\right) \cdot e\left(G, H\right)^{M(F)} = \\ e\left(G^\alpha, H^\beta\right) \cdot e\left(G^{\frac{pk}{\gamma}}, H^\gamma\right) \cdot e\left(C, H^\delta\right) \cdot e\left(F, B\right) \end{aligned}$$

what is indeed true since

$$\begin{aligned} e\left(A, B \cdot H^\beta\right) \cdot e\left(G, H\right)^{M(F)} = \\ e\left(G, H\right)^{(\alpha+sk)\beta} \cdot e\left(G, H\right)^{pk} = \\ e\left(G, H\right)^{\alpha\beta} \cdot e\left(G, H\right)^{\frac{pk}{\gamma}\cdot\gamma} \cdot e\left(G, H\right)^{\frac{\beta sk}{\delta}\cdot\delta} \cdot e\left(G, H\right)^{sk\cdot 0} = \\ e\left(G^\alpha, H^\beta\right) \cdot e\left(G^{\frac{pk}{\gamma}}, H^\gamma\right) \cdot e\left(C, H^\delta\right) \cdot e\left(F, B\right) \quad \square \end{aligned}$$

From this we derive a simple cryptographic protocol: Proving knowledge of sk, s.t. $pk = G^{sk}$: Verifier: pick random $\alpha \xleftarrow{\$} \mathbb{Z}_p^*$ send it to the prover. Prover: set the proof

$\pi = (A) = (G^{sk+\alpha})$ Verifier take $\pi = (A), \alpha, pk$ and accept iff:

$$e(pk, H) \cdot e(G, H)^{\alpha} = e(A, H). \tag{5.5}$$