Groth16 学习笔记

1. 引言

Groth 2016年论文《On the Size of Pairing-based Non-interactive Arguments》。

相关代码实现有：

- https://github.com/matter-labs/bellman
- https://github.com/zkcrypto/bellman
- https://github.com/arkworks-rs/groth16

非交互式argument 允许Prover convince Verifier that a statement is true。

近期在理论和实践上，构建具有small size和low verification complexity的高效非交互式argument均取得重大进展，具有small size和low verification complexity的非交互式argument也称为SNARGs (succinct non-interactive arguments) 或 SNARKs (succinct non-interactive arguments of knowledge)。

很多SNARGs都是基于pairing 构建的，这种构建方式：

- proof中包含多个group elements
- verification中需验证多个pairing product方程式

1.1 相关研究

Goldwasser等人在1989年论文[GMR89]中指出，零知识证明应具有如下属性：

- Completeness完备性：已知statement和相应的witness，Prover可convince Verifier。
- Soundness 可靠性：malicious Prover无法convince Verifier of a false statement。
- Zero-knowledge 零知识性：proof除了说明the truth of the statement，不会泄露任何其它信息，尤其是Prover的witness。

Blum等人在[FBM88]中将以上观点扩展至 non-interactive zero-knowledge (NIZK) proofs in the common reference string model。NIZK proofs可用于构建non-interactive cryptographic schemes，如数字签名和CCA-secure public key encryption。

零知识证明中的communication cost是一个重要的性能指标。Kilian在[Kil92]中提出了第一个sublinear communication cost的方案，其发送的bits数量比 待证明的statement size要小。Micali [Mic00]和Kilian [Kil95]中指出，Prover可利用a cryptographic function来计算Verifier

的challenges，从而实现public coin和zero-knowledge。

Groth, Ostrovsky和Sahai [GOS12,GOS06,Gro06,GS12]中介绍了pairing-based NIZK proofs，产生了第一个基于standard assumption的linear size proof。[Gro10]中将这些技术与[Gro09]中的interactive zero-knowledge argument结合，提出了第一个constant size NIZK argument。[Lip12]中基于progression-free sets减少了common reference string的size。

[Gro10]中的constant size NIZK argument中构建了一系列的polynomial方程式，使用pairings来高效验证这些方程式。Gennaro [GGPR13] 中富有见解的采用基于Lagrange interpolation polynomials来构建polynomial 方程式，从而实现了pairing-based NIZK argument，其common reference string size与statement size和witness size呈正比。在[GGPR13]中给出了2种类型的方程式：

- Quadratic span programs for proving Boolean circuit satisfiability
- Quadratic arithmetic programs for proving arithmetic circuit satisfiability

Lipmaa [Lip13] 建议使用error correcting codes来构建更高效的quadratic span programs。
Danezis等人 [DFGK14]中优化quadratic span program为square span program，使得boolean circuit satisfiability的proof中仅包含4个group elements。

当前一些研究在理论上也取得了进展，如：
[PHGR13,BCG+13,BFR+13,BCTV14b,KPP+14,BBFR15,CTV15,WSR+15,CFH+15,SVdV16]

大多数高效实现都是将[GGPR13]中改进的quadratic arithmetic program 与 可生成合适quadratic arithmetic program的compiler 结合。如libsnark [BCTV14b, BSCG+14]中也包含基于[DFGK14]中的NIZK argument。

非交互式argument可用于：

- Verifiable computation：如算力外包等。零知识SNARKs是Pinocchio coin [DFKP13] 和 Zerocash [BCG+14]的核心元素。

1.2 本文主要贡献

本文的主要贡献有：
（1）succinct NIZK
为arithmetic circuit satisfiability构建了pairing-based (preprocessing) SNARK。采用非对称pairing，proof中仅包含3个group element，verification中仅需验证一个paring product方程式，该方程式中一共只有3个pairing计算。
本文的构建方式支持任意类型的pairings，包括Type III pairings，Type III pairings为当前效率最高的pairings。
对boolean circuit satisfiability和arithmetic circuit satisfiability的性能对比如下图所示：【评估的维度有：common reference string (CRS)

的size、proof size、Prover的computation、Verifier的computation、验证proof所需的pairing product equations的数量。】

	CRS size	Proof size	Prover comp.	Verifier comp.	PPE
[DFGK14]	$2m+n-2\ell\ \mathbb{G}_1\ ,\ m+n-\ell\ \mathbb{G}_2$	$3\ \mathbb{G}_1\ ,\ 1\ \mathbb{G}_2$	$m+n-\ell\ E_1$	$\ell\ M_1\ ,\ 6\ P$	3
This work	$3m+n\ \mathbb{G}_1\ ,\ m\ \mathbb{G}_2$	$2\ \mathbb{G}_1\ ,\ 1\ \mathbb{G}_2$	$n\ E_1$	$\ell\ M_1\ ,\ 3\ P$	1

Table 1. Comparison for boolean circuit satisfiability with ℓ-bit statement, m wires and n fan-in 2 logic gates. Notation: $\mathbb{G}$ means group elements, M means multiplications, E means exponentiations and P means pairings with subscripts indicating the relevant group. It is possible to get a CRS size of $m+2n$ elements in $\mathbb{G}_1$ and n elements in $\mathbb{G}_2$ but we have chosen to include some precomputed values in the CRS to reduce the prover's computation, see Sect. 3.2.

	CRS size	Proof size	Prover comp.	Verifier comp.	PPE
[PHGR13]	$7m+n-2\ell\ \mathbb{G}$	$8\ \mathbb{G}$	$7m+n-2\ell\ E$	$\ell\ E\ ,\ 11\ P$	5
This work	$m+2n\ \mathbb{G}$	$3\ \mathbb{G}$	$m+3n-\ell\ E$	$\ell\ E\ ,\ 3\ P$	1
[BCTV14a]	$6m+n+\ell\ \mathbb{G}_1\ ,\ m\ \mathbb{G}_2$	$7\ \mathbb{G}_1\ ,\ 1\ \mathbb{G}_2$	$6m+n-\ell\ E_1\ ,\ m\ E_2$	$\ell\ E_1\ ,\ 12\ P$	5
This work	$m+2n\ \mathbb{G}_1\ ,\ n\ \mathbb{G}_2$	$2\ \mathbb{G}_1\ ,\ 1\ \mathbb{G}_2$	$m+3n-\ell\ E_1\ ,\ n\ E_2$	$\ell\ E_1\ ,\ 3\ P$	1

Table 2. Comparison for arithmetic circuit satisfiability with ℓ-element statement, m wires, n multiplication gates. Notation: $\mathbb{G}$ means group elements, E means exponentiations and P means pairings. We compare symmetric pairings in the first two rows and asymmetric pairings in the last two rows.

（2）lower bounds
回答了Bitansky等人（TCC2013）的开放问题，说明2-move linear interactive proofs cannot have a linear decision procedure。对于使用generic asymmetric bilinear group的SNARGs，prover和verifier的group operation操作中不可能仅包含一个group element。这给出了pairing-based SNARGs的下限值。至于是否能在现有3个group elements的基础上进一步优化proof size为2个group elements，目前仍是一个开放问题。

1.3 Bilinear groups

bilinear groups $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, h)$ 具有如下属性：

- $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ 为groups of prime order p
- pairing $e: \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ 为bilinear map
- g为generator for $\mathbb{G}_1$，h为generator for $\mathbb{G}_2$，$e(g,h)$为generator for $\mathbb{G}_T$。
- 存在高效的算法来计算group operations、evaluate bilinear map、decide membership of the groups、decide equality of group elements以及sample generators of the groups。将这些计算统称为generic group operations。

基于的安全假设为DLP。
做了如下约定：

- $[a]_1$表示g^a，$[b]_2$表示h^b，$[c]_T$表示$e(g,h)^c$。
- $g = [1]_1, h = [1]_2, e(g,h) = [1]_T$
- $[a]_T + [b]_T = [a+b]_T$
- $[\vec{a}]_i + [\vec{b}]_i = [\vec{a} + \vec{b}]_i$为a vector of group elements

1.4 non-interactive zero-knowledge arguments of knowledge

relation $(\phi, w) \in R$，其中ϕ为statement，w为witness。
publicly verifiable non-interactive argument包含了4个probabilistic polynomial algorithms $(Setup, Prove, Vfy, Sim)$：

- $(\sigma, \tau) \leftarrow Setup(R)$：setup算法，输入为relation R，输出为common reference string σ和相应的simulation trapdoor τ。

- $\pi \leftarrow \mathrm{Prove}(R, \sigma, \phi, w)$：Prover算法，输入为common reference string σ、$(\phi, w) \in R$，输出为argument π。
- $0/1 \leftarrow \mathrm{Vfy}(R, \sigma, \phi, \pi)$：Verification算法，输入为common reference string σ、statement ϕ、argument π，输出为0（reject）或 1（accept）。
- $\pi \leftarrow \mathrm{Sim}(R, \tau, \phi)$：Simulator算法，输入为simulation trapdoor τ、statement ϕ，输出为argument π。

可将common reference σ分为两部分：

- σ_P，给Prover用
- σ_V，给Veriifier用

若σ_V 可根据σ_P 推导获得，则可称为public verifiable non-interactive argument。

2. Quadratic arithmetic programs

考虑只有加法和乘法门的arithmetic circuit over finite field $\mathbb{F}$，存在statements和witnesses满足所有n个方程式（n个乘法gates，m个wires）：
$\sum a_i u_{i,q} \cdot \sum a_i v_{i,q} = \sum a_i w_{i,q}$
其中，$a_0 = 1$，$a_1, \cdots, a_m \in \mathbb{F}$，$u_{i,q}, v_{i,q}, w_{i,q}$为constants in $\mathbb{F}$ specifying the qth equation。

对于某乘法门 $a_i \cdot a_j = a_k$，设置相应行的$u_i = 1, v_j = 1, w_k = 1$，设置该行其他元素为0。
对于加法门，不计入方程式数量中。即，若$a_i + a_j = a_k$且$a_k \cdot a_l$，则表示为$(a_i + a_j) \cdot a_l$，跳过a_k的计算。

根据[GGPR13]中的规则，假设$\mathbb{F}$足够大，将arithmetic constraints 表示为quadratic arithmetic program。
取任意不同的值$r_1, \cdots, r_n \in \mathbb{F}$，定义$t(x) = \prod_{q=1}^{n}(x - r_q)$。
另$u_i(x), v_i(x), w_i(x)$为degree $n-1$多项式，满足：
$u_i(r_q) = u_{i,q}, v_i(r_q) = v_{i,q}, w_i(r_q) = w_{i,q}$ for $i = 0, \cdots, m, q = 1, \cdots, n$

于是有，对于$a_0 = 1, a_1, \cdots, a_m \in \mathbb{F}$ 满足n个方程式，当且仅当，对于每一个$r_1, \cdots, r_q$，以下等式成立：
$\sum_{i=0}^{m} a_i u_i(r_q) \cdot \sum_{i=0}^{m} a_i v_i(r_q) = \sum_{i=0}^{m} a_i w_i(r_q)$

由于$t(X)$为the lowest degree monomial with $t(r_q) = 0$ in each point，可表示为：
$\sum_{i=0}^{m} a_i u_i(X) \cdot \sum_{i=0}^{m} a_i v_i(X) = \sum_{i=0}^{m} a_i w_i(X) \mod t(X)$

最终，整个quadratic arithmetic programs R 可表示为：
$R = (\mathbb{F}, aux, l, \{u_i(X), v_i(X), w_i(X)\}_{i=0}^{m}, t(X))$
其中，$\mathbb{F}$为finite field，aux为some auxiliary information，$1 \leq l \leq m$，$u_i(X), v_i(X), w_i(X), t(X) \in \mathbb{F}[X]$，且$u_i(X), v_i(X), w_i(X)$的degree严格低于$t(X)$的degree，$t(X)$的degree为n。

定义$a_0 = 1$，以上description可表示为如下binary relation：

$$R = \left\{ (\phi, w) \middle| \begin{array}{l} \phi = (a_1, \ldots, a_\ell) \in \mathbb{F}^\ell \\ w = (a_{\ell+1}, \ldots, a_m) \in \mathbb{F}^{m-\ell} \\ \\ \sum_{i=0}^{m} a_i u_i(X) \cdot \sum_{i=0}^{m} a_i v_i(X) \equiv \sum_{i=0}^{m} a_i w_i(X) \mod t(X) \end{array} \right\}$$

3. non-interactive linear proofs （NILP）

Bitansky [BCI+13] 中提出了基于2-move algebraic input-oblivious linear interactive proofs构建SNARK的方法，为了便于区分，本文将该方法称为non-interactive linear proof (NILP)。

NILP的运行流程为：

- $(\vec{\sigma},\vec{\tau}) \leftarrow \mathrm{Setup}(\mathrm{R})$：setup算法，为probabilistic polynomial time算法，输入为relation R，输出为向量$\vec{\sigma} \in \mathbb{F}^m, \vec{\tau} \in \mathbb{F}^n$。为了简化描述，$\vec{\sigma}$中总是包含1 as an entry，使得affine和linear functions of σ是无区别的。
- $\vec{\pi} \leftarrow \mathrm{Prove}(\mathrm{R},\vec{\sigma},\phi,\mathrm{w})$：Prover算法，主要分为2个阶段：
  1）运行$\mathbf{\Pi} \leftarrow \mathrm{ProofMatrix}(\mathrm{R},\phi,\mathrm{w})$，其中$\mathrm{ProofMatrix}$为probabilistic polynomial time 算法，输出为矩阵$\mathbf{\Pi} \in \mathbb{F}^{k\times m}$。
  2）计算proof $\vec{\pi} = \mathbf{\Pi}\vec{\sigma}$。
- $0/1 \leftarrow \mathrm{Vfy}(\mathrm{R},\vec{\sigma},\phi,\pi)$：为Verifier算法，主要分为以下2个阶段：
  1）运行deterministic polynomial time algorithm $\vec{t} \leftarrow \mathrm{Test}(\mathrm{R},\phi)$，以获得arithmetic circuit $\vec{t}:\mathbb{F}^{m+k} \rightarrow \mathbb{F}^{\eta}$，对应为the evaluation of a vector of multi-variate polynomials of total degree d。
  2）当且仅当$\vec{t}(\vec{\sigma},\vec{\pi}) = \vec{0}$时，accept the proof。

NILP的价值：

- 借助pairings，可将NILP转换为publicly verifiable non-interactive arguments
- 借助a variant of Paillier encryption [BCI+13]，可将NILP转换为designated verifier non-interactive arguments。

基于Type III pairing，基于DLP假设构建NILP，需对其进行split切分：

- common reference string切分：$\vec{\sigma} = (\vec{\sigma}_1,\vec{\sigma}_2)$
- proof切分：$\vec{\pi} = (\vec{\pi}_1,\vec{\pi}_2)$，切分的proof与切分的crs存在对应关系。
- 对proof进行verify时，对应的quadratic equation中每个变量的degree应为1。

3.1 split NILP

split NILP的运行流程为：

- $(\vec{\sigma},\vec{\tau}) \leftarrow \mathrm{Setup}(\mathrm{R})$：setup算法，为probabilistic polynomial time算法，输入为relation R，输出为向量$\vec{\sigma} = (\sigma_1,\sigma_2) \in \mathbb{F}^{m_1} \times \mathbb{F}^{m_2}, \vec{\tau} \in \mathbb{F}^n$。为了简化描述，$\vec{\sigma}_1$和$\vec{\sigma}_2$中总是包含1 as an entry，使得affine和linear functions of σ是无区别的。
- $\vec{\pi} \leftarrow \mathrm{Prove}(\mathrm{R},\vec{\sigma},\phi,\mathrm{w})$：Prover算法，主要分为2个阶段：
  1）运行$\mathbf{\Pi} \leftarrow \mathrm{ProofMatrix}(\mathrm{R},\phi,\mathrm{w})$，其中$\mathrm{ProofMatrix}$为probabilistic polynomial time 算法，输出为矩阵$\mathbf{\Pi} = \begin{pmatrix} \mathbf{\Pi}_1 & 0 \\ 0 & \mathbf{\Pi}_2 \end{pmatrix}$，其中$\mathbf{\Pi}_1 \in \mathbb{F}^{k_1\times m_1}, \mathbf{\Pi}_2 \in \mathbb{F}^{k_2\times m_2}$。
  2）计算proof $\vec{\pi}_1 = \mathbf{\Pi}_1\vec{\sigma}_1$，$\vec{\pi}_2 = \mathbf{\Pi}_2\vec{\sigma}_2$，返回$\vec{\pi} = (\vec{\pi}_1,\vec{\pi}_2)$。
- $0/1 \leftarrow \mathrm{Vfy}(\mathrm{R},\vec{\sigma},\phi,\pi)$：为Verifier算法，主要分为以下2个阶段：
  1）运行deterministic polynomial time algorithm $\vec{t} \leftarrow \mathrm{Test}(\mathrm{R},\phi)$，以获得arithmetic circuit $\vec{t}:\mathbb{F}^{m_1+k_1+m_2+k_2} \rightarrow \mathbb{F}^{\eta}$，对应矩阵$T_1,\cdots,T_\eta \in \mathbb{F}^{(m_1+k_1)\times(m_2+k_2)}$。
  2）当且仅当对所有的矩阵$T_1,\cdots,T_\eta$，都有$\begin{pmatrix} \vec{\sigma}_1 \\ \vec{\pi}_1 \end{pmatrix} \cdot T_i \begin{pmatrix} \vec{\sigma}_2 \\ \vec{\pi}_2 \end{pmatrix} = 0$时，accept the proof。

3.2 基于pairing 由split NILP构建的non-interactive argument

基于pairing 由split NILP构建的non-interactive argument $(\mathrm{Setup}', \mathrm{Prove}', \mathrm{Vfy}', \mathrm{Sim}')$为：

- $(\vec{\sigma}, \vec{\tau}) \leftarrow \mathrm{Setup}'(R)$：运行 $(\vec{\sigma}_1, \vec{\sigma}_2, \vec{\tau}) \leftarrow \mathrm{Setup}(R)$，输入为relation R，输出为向量$\vec{\sigma} = ([\sigma_1]_1, [\sigma_2]_2) \in \mathbb{G}_1^{m1} \times \mathbb{G}_2^{m2}, \vec{\tau} \in \mathbb{F}^n$。
- $\vec{\pi} \leftarrow \mathrm{Prove}'(R, \vec{\sigma}, \phi, w)$：Prover算法，主要分为2个阶段：
  1）运行$\mathbf{\Pi} \leftarrow \mathrm{ProofMatrix}(R, \phi, w)$，其中$\mathrm{ProofMatrix}$为probabilistic polynomial time 算法，输出为矩阵$\mathbf{\Pi} = \begin{pmatrix} \mathbf{\Pi}_1 & 0 \\ 0 & \mathbf{\Pi}_2 \end{pmatrix}$，其中$\mathbf{\Pi}_1 \in \mathbb{F}^{k1 \times m1}, \mathbf{\Pi}_2 \in \mathbb{F}^{k2 \times m2}$。
  2）计算proof $\vec{\pi}_1 = \mathbf{\Pi}_1 \vec{\sigma}_1$，$\vec{\pi}_2 = \mathbf{\Pi}_2 \vec{\sigma}_2$，返回$\vec{\pi} = ([\vec{\pi}_1]_1, [\vec{\pi}_2]_2) \in \mathbb{G}_1^{k1} \times \mathbb{G}_2^{k2}$。
- $0/1 \leftarrow \mathrm{Vfy}'(R, \vec{\sigma}, \phi, \pi)$：为Verifier算法，主要分为以下2个阶段：
  1）运行deterministic polynomial time algorithm $\vec{t} \leftarrow \mathrm{Test}(R, \phi)$，以获得arithmetic circuit $\vec{t}: \mathbb{F}^{m1+k1+m2+k2} \rightarrow \mathbb{F}^{\eta}$，对应矩阵$T_1, \cdots, T_\eta \in \mathbb{F}^{(m1+k1)\times(m2+k2)}$。
  2）当且仅当对所有的矩阵$T_1, \cdots, T_\eta$，都有$\begin{bmatrix} \vec{\sigma}_1 \\ \vec{\pi}_1 \end{bmatrix}_1 \cdot T_i \begin{bmatrix} \vec{\sigma}_2 \\ \vec{\pi}_2 \end{bmatrix}_2 = [0]_T$时，accept the proof。
- $\pi \leftarrow \mathrm{Sim}'(R, \vec{\tau}, \phi)$：Simulate $(\vec{\pi}_1, \vec{\pi}_2) \leftarrow \mathrm{Sim}(R, \vec{\tau}, \phi)$，返回$\vec{\pi} = ([\vec{\pi}_1]_1, [\vec{\pi}_2]_2)$

4. 构建non-interactive arguments

接下来，将为quadratic arithmetic programs构建pairing-based NIZK argument，其proof中仅包含3个group elements。
具体的构建步骤分为2步：

- 1）为quadratic arithmetic programs构建NILP
- 2）该NILP为split NILP，可利用之前提到的compilation技术将其转换为pairing-based NIZK argument。

4.1 NILP for quadratic arithmetic programs

quadratic arithmetic program对应的Relation表示为：
$R = (\mathbb{F}, aux, l, \{u_i(X), v_i(X), w_i(X)\}_{i=0}^{m}, t(X))$

其中$(a_1, \cdots, a_l) \in \mathbb{F}^l$为statement，$(a_{l+1}, \cdots, a_m) \in \mathbb{F}^{m-l}$ 为witness，$a_0 = 1$。

满足：
$\sum_{i=0}^{m} a_i u_i(X) \cdot \sum_{i=0}^{m} a_i v_i(X) = \sum_{i=0}^{m} a_i w_i(X) + h(X)t(X)$

其中，$t(X)$的degree为n，quotient polynomial $h(X)$的degree 为$n-2$。

相应的$(\mathrm{Setup}, \mathrm{Prove}, \mathrm{Vfy}, \mathrm{Sim})$ 算法为：

- $(\vec{\sigma}, \vec{\tau}) \leftarrow \mathrm{Setup}(R)$：选择$\alpha, \beta, \gamma, \delta, x \leftarrow \mathbb{F}^*$，设置$\vec{\tau} = (\alpha, \beta, \gamma, \delta, x)$，$\vec{\sigma} = (\alpha, \beta, \gamma, \delta, \{x^i\}_{i=0}^{n-1}, \{\frac{\beta u_i(x) + \alpha v_i(x) + w_i(x)}{\gamma}\}_{i=0}^{l}), \{\frac{\beta u_i(x) + \alpha v_i(x) + w_i(x)}{\delta}\}_{i=l+1}^{m}, \{\frac{x^i t(x)}{\delta}\}_{i=0}^{n-2}$

- $\pi \leftarrow Prove(R, \vec{\sigma}, a_1, \cdots, a_m)$：选择$r, s \leftarrow \mathbb{F}$，计算$3 \times (m + 2n + 4)$矩阵$\mathbf{\Pi}$，使得$\vec{\pi} = \mathbf{\Pi}\vec{\sigma} = (A, B, C)$，其中：
  $A = \alpha + \sum_{i=0}^{m} a_i u_i(x) + r\delta$
  $B = \beta + \sum_{i=0}^{m} a_i v_i(x) + s\delta$
  $C = \frac{\sum_{i=l+1}^{m} a_i(\beta u_i(x) + \alpha v_i(x) + w_i(x)) + h(x)t(x)}{\delta} + As + rB - rs\delta$
- $0/1 \leftarrow Vfy(R, \vec{\sigma}, a_1, \cdots, a_l, \vec{\pi})$：计算a quadratic multi-variate polynomial t，使得$t(\vec{\sigma}, \vec{\pi}) = 0$，对应的test为：
  $A \cdot B = \alpha \cdot \beta + \frac{\sum_{i=0}^{l} a_i(\beta u_i(x) + \alpha v_i(x) + w_i(x))}{\gamma} \cdot \gamma + C \cdot \delta$
  若以上test成立，则accept the proof。
- $\vec{\pi} \leftarrow Sim(R, \vec{\tau}, a_1, \cdots, a_l)$：选择$A, B \leftarrow \mathbb{F}$，计算$C = \frac{AB - \alpha\beta - \sum_{i=0}^{l} a_i(\beta u_i(x) + \alpha v_i(x) + w_i(x))}{\delta}$，返回$\vec{\pi} = (A, B, C)$。

以上：

- α, β：用于保证A, B, C are consistent with each other in the choice of $a_0, \cdots, a_m$。
  verification equation中的$\alpha \cdot \beta$ product用于保证A and B involve non-trivial α and β components。即意味着$A \cdot B$ product中包含了a linear dependence on α and β，稍后将证明该linear dependence can only be balanced out by C with a consistent choice of $a_0, \cdots, a_m$ in all three of A, B and C。
- γ, δ：用于使verification equation中的后2个product 与 第一个product 无关，通过分别相应除以γ, δ。
  这可以避免mixing and mathching of elements intended for different products in the verification equation。
- r, s：用于randomize the proof来实现zero-knowledge。

以上NILP构建的proof具有3个field element，具有如下特性：

- perfect completeness
- perfect zero-knowledge
- statistical knowledge soundness against affine prover strategies

4.1.1 是否可进一步将proof reduce为2个Field element?

以上NILP构建的proof具有3个field element，是否可进一步将其reduce为2个field element呢？

Danezis等人[DFGK14] 中实现了2 field element NILP for boolean circuit satisfiability。

同时，通过将circuit改造为只有squaring gates，也可能实现2-element NILP for arithmetic circuit satisfiability。因为，对于每个multiplication gate $a \cdot b = c$，可将其改造为$(a + b)^2 - (a - b)^2 = 4c$。

当arithmetic circuit中仅有squaring gate时，对于所有的i，具有$u_i(x) = v_i(x)$。
选择NILP中$r = s$，则有$B = A + \beta - \alpha$，Prover仅需发送2个elements A和C 来make a convincing proof。

将arithmetic circuit 改造为仅有squaring gate 可能会使gate数量翻倍，同时，需要引入额外的wires来表达the subtraction of the squares。

因此，这种reduction是以牺牲significant computational cost为代价的。

4.2 NIZK arguments for quadratic arithmetic programs

本节将为arithmetic program relation：
$R=(p,\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_T,e,g,h,l,\{u_i(X),v_i(X),w_i(X)\}_{i=0}^m,t(X))$

构建pairing-based NIZK argument。

其中：

- $|p|=\lambda$，对应的field为$\mathbb{Z}_p$。
- $(a_1,\cdots,a_l)\in\mathbb{F}^l$为statement，即public info。
- $(a_{l+1},\cdots,a_m)\in\mathbb{F}^{m-l}$为witness，即private info。
- $a_0=1$

满足：
$\sum_{i=0}^m a_iu_i(X)\cdot\sum_{i=0}^m a_iv_i(X)=\sum_{i=0}^m a_iw_i(X)+h(X)t(X)$
其中quotient polynomial $h(X)$的degree为$n-2$。

在以上3.2节中指出了：
NILP的一个重要的设计特征是——其很容易就可实现a split NILP。

proof elements A,B,C 在verification equation中仅使用一次，因此很容易assign them to different sides of the bilinear test。

通过将common reference string split为2部分，可enable the computation of each side of the proof，从而实现split NILP。该split NILP同时是disclosure-free的，因此可compiled into a NIZK argument in the generic group model（具体参见3.2节）。

通常的pairing-friendly elliptic curve，其$\mathbb{G}_1$的group element representation要小于$\mathbb{G}_2$的 [GPS08]。因此，取$A,C\in\mathbb{G}_1,B\in\mathbb{G}_2$来使效率最优。
基于Pairing的详细实现为：

- $(\vec{\sigma},\vec{\tau})\leftarrow Setup(R)$：选择$\alpha,\beta,\gamma,\delta,x\leftarrow\mathbb{F}^*$，设置$\vec{\tau}=(\alpha,\beta,\gamma,\delta,x)$，$\vec{\sigma}=([\vec{\sigma}_1]_1,[\vec{\sigma}_2]_2)$，其中：
  $\vec{\sigma}_1=(\alpha,\beta,\gamma,\delta,\{x^i\}_{i=0}^{n-1},\{\frac{\beta u_i(x)+\alpha v_i(x)+w_i(x)}{\gamma}\}_{i=0}^l,\{\frac{\beta u_i(x)+\alpha v_i(x)+w_i(x)}{\delta}\}_{i=l+1}^m,\{\frac{x^it(x)}{\delta}\}_{i=0}^{n-2}),\vec{\sigma}_2=(\beta,\gamma,\delta,\{x^i\}_{i=0}^{n-1})$
- $\pi\leftarrow Prove(R,\vec{\sigma},a_1,\cdots,a_m)$：选择$r,s\leftarrow\mathbb{F}$，计算$\vec{\pi}=([A]_1,[C]_1,[B]_2)$，其中：
  $A=\alpha+\sum_{i=0}^m a_iu_i(x)+r\delta$
  $B=\beta+\sum_{i=0}^m a_iv_i(x)+s\delta$
  $C=\frac{\sum_{i=l+1}^m a_i(\beta u_i(x)+\alpha v_i(x)+w_i(x))+h(x)t(x)}{\delta}+As+rB-rs\delta$
- $0/1\leftarrow Vfy(R,\vec{\sigma},a_1,\cdots,a_l,\vec{\pi})$：解析$\vec{\pi}=([A]_1,[C]_1,[B]_2)\in\mathbb{G}_1^2\times\mathbb{G}_2$，对应的test为：
  $[A]_1\cdot[B]_2=[\alpha]_1\cdot[\beta]_2+\sum_{i=0}^l a_i[\frac{(\beta u_i(x)+\alpha v_i(x)+w_i(x))}{\gamma}]_1\cdot[\gamma]_2+[C]_1\cdot[\delta]_2$
  若以上test成立，则accept the proof。
- $\vec{\pi}\leftarrow Sim(R,\vec{\tau},a_1,\cdots,a_l)$：选择$A,B\leftarrow\mathbb{F}$，计算simulated proof $\vec{\pi}=([A]_1,[C]_1,[B]_2)$，其中$C=\frac{AB-\alpha\beta-\sum_{i=0}^l a_i(\beta u_i(x)+\alpha v_i(x)+w_i(x))}{\delta}$。

以上算法构建的proof具有3个Group elements （2个$\mathbb{G}_1$和1个$\mathbb{G}_2$），具有如下特性：

- perfect completeness
- perfect zero-knowledge
- statistical knowledge soundness against adversaries that only use a polynomial number of generic bilinear group operations

整个算法的效率分析为：

- proof $\vec{\pi}$中包含3个Group elements （2个$\mathbb{G}_1$和1个$\mathbb{G}_2$）
- common reference string中包含：n个$\mathbb{Z}_p$，$m+2n+3$个$\mathbb{G}_1$和$n+3$个$\mathbb{G}_2$
- 根据以上$V\ fy$算法中的公式可知，Verifier无需知悉整个common reference string，仅需知悉以下内容就足够：
  $\vec{\sigma}_V = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, [1]_1, \{[\frac{(\beta u_i(x)+\alpha v_i(x)+w_i(x))}{\gamma}]_1\}_{i=0}^{l}, [1]_2, [\gamma]_2, [\delta]_2, [\alpha\beta]_T)$
  以上Verifier的reference string中仅包含了$l+2$个$\mathbb{G}_1$，3个$\mathbb{G}+2$和1个$\mathbb{G}_T$。
- $V\ fy$算法中，Verifier需验证$([A]_1, [C]_1, [B]_2)$为有效的group elements，同时验证single pairing product equation：
  $[A]_1 \cdot [B]_2 = [\alpha]_1 \cdot [\beta]_2 + \sum_{i=0}^{l} a_i [\frac{(\beta u_i(x)+\alpha v_i(x)+w_i(x))}{\gamma}]_1 \cdot [\gamma]_2 + [C]_1 \cdot [\delta]_2$
  成立。
  在验证以上方程式成立过程中，Verifier的计算量为：
  1）l次exponentiations in $\mathbb{G}_1$
  2）少量的group multiplication
  3）3次pairing计算。假设$[\alpha\beta]_T = [\alpha]_1 \cdot [\beta]_2$以根据Verifier的reference string预计算了。
- $P\ rove$算法中，Prover需计算polynomial $h(X)$。
  Prover可计算以下polynomial evaluations：
  $\sum_{i=0}^{m} a_i u_i(r_q) = \sum_{i=0}^{m} a_i u_{i,q}$
  $\sum_{i=0}^{m} a_i v_i(r_q) = \sum_{i=0}^{m} a_i v_{i,q}$
  $\sum_{i=0}^{m} a_i w_i(r_q) = \sum_{i=0}^{m} a_i w_{i,q}$
  for $q = 1, \cdots, n$。
  具体计算量取决于relation，若arithmetic circuit中的每个乘法门连接了a constant number of wires，则该relation为sparse的，相应的计算为linear in n。
  由于这些polynomial的degree为$n-1$，以上n个evaluations可完全确定该多项式。若$r_1, \cdots, r_n$为root of unity for a suitable prime p，则Prover可使用标准的FFT技术来计算$h(X)$，仅需要$O(n \log n)$ operations in $\mathbb{Z}_p$。同时，Prover还可使用FFT技术来计算$\sum_{i=0}^{m} a_i u_i(X)$和$\sum_{i=0}^{m} a_i v_i(X)$的系数。为了获得所有的系数，Prover需$m+3n-l+3$次exponentiation in $\mathbb{G}_1$和$n+1$次exponentiation in $\mathbb{G}_2$。
  随着security parameter增长，这些exponentiation将为主要开销。但是，对于中等security parameter和large statement，通过FFT来计算multiplication的开销将更大。此时，通过用更大的common reference string，其中包含了precomputed $[u_i(x)]_1, [v_i(x)]_1, [v_i(x)]_2$ elements for $i = 0, \cdots, m$，使得A, B可直接构建，而Prover不再需要计算$\sum_{i=0}^{m} a_i u_i(X)$和$\sum_{i=0}^{m} a_i v_i(X)$，也就不需要做相应exponentiation运算。
  对于boolean circuit，有$a_i \in \{0, 1\}$，通过这些precomputed elements，Prover仅需分别做m个group multiplication来计算A和B。
  因此，让CRS更长可让Prover具有lower computational cost。

5. Lower bounds for non-interactive arguments

有一个有趣的问题是，non-interactive argument的efficiency极限是多少？
本文证明了，对于pairing-based non-interactive arguments，proof中至少应包含2个group elements。