Bulletproof: 子弹证明的简单介绍

fubiao.xia@atmatrix.org

1 背景

区块链中的隐私保护是一个比较大的话题。目前隐私币例如门罗币(XMR)和Zcash均用到了零知识证明技术。 零知识证明系统主要是能解决隐私交易里不泄露具体的金额的问题。至于如何保护交易双方的身份信息，目前的区块链项目多数采用环签名技术。 零知识证明是一种证明系统,需要满足完备性，可靠性和零知识性。简单来说，零知识证明是证明人跟验证人之间执行一套证明协议，用于证明某一个陈述(statement)是否为真(结果验证人会接受该证明)或假(结果是验证人拒绝接受该证明)，在协议执行结束后，验证人除了知道该陈述是真或假的信息外，无法获得任何其他额外的信息。 例如Alice向Bob进行零知识证明，关于Alice知道方程某个方程的解。证明成功后，Bob会相信"Alice知道某个方程的解"，但是Bob不知道具体的解是什么，并且Bob不知道任何额外的信息有助于他获知具体的解。 更严谨的形式化定义可以参考wiki的"zero knowledge proof"词条。目前区块链领域用到零知识主要是以下三种，其中第二种是第一种到改造版本（并不完全是优化）。

ZK-snarks：一个很好的零知识证明技术，用在Zcash里。具体的zk-snarks技术原理可以参考v神的博客(https://medium.com/@VitalikButerin/zk-snarks-under-the-hood-b33151a013f6)以及Zcash的官方博客的系列介绍。 优点是简洁——证明长度和验证时间都很短，非交互式的——验证人不需要与证明人进行多轮交互式通讯。最大的缺点是需要一个可信的初始化阶段(Trusted setup)，这个阶段需要预先计算一组椭圆曲线上的点，并且必须删除这些点之间的线性组合关系的系数，否则zk-snarks所依赖的KEA(knowledge of exponent assumption)安全假设就会失效。这个trusted setup就是制约zk-snarks成为完美的证明系统的一大障碍。zcash目前是通过多方计算这些点，并且各方会删除相应的系数完成初始化阶段的。

ZK-starks： 针对zk-snarks的弱点进行了进一步提升，主要包括： 不需要trusted setup，这个是最大的提升之一。 提升了性能，当然这块提升不是很明显，且在部分指标上还有所下降，参见（https://medium.com/coinmonks/zk-starks-create-verifiable-trust-even-against-quantum-computers-dd9c6a2bb13d）中的complexity 比较。提升了安全性，能抵抗量子攻击。 ZK-starks最大的问题可能就是证明长度太长，这点也制约了zk-starks在很多情况下不如zk-snarks更实用。

Bulletproof: 与zk-snarks, zk-starks一样，bulletproof也是一个可以证明任意陈述的零知识证明系统。Bulletproof也像zk-starks一样，不需要trusted setup。Bulletproof主要是可以证明一个数在一个特定的范围内。在涉及到隐私的交易里，如果一笔转账交易需要隐藏具体的金额，通常称为CT(confidential transaction)。在UTXO模型里，只要证明Input大于Output，这笔交易就可以被认为是有效的。也就是Output在0和Input之间。之前有一种证明技术叫range proof就是解决此类问题，但是range proof性能很不理想。因此，Bulletproof特别适合用来作为range proof的代替品。 BUlletproof也是非交互式的（采用Fiat-Shamir技术转换得到）。 Bulletproof的具体技术原理可以参考原论文或者这个笔记(后面简称notes, https://doc-internal.dalek.rs/bulletproofs/notes/index.html），后者可读性更好。 Bulletproof的主要特点是各项性能指标比较好并且不需要trusted setup。唯一的问题是对于单个证明的验证比较花时间(另一方面它的证明长度非常短)，但是批量验证反而能非常节省时间，这样看来，bulletproof有可能是大规模应用与隐私交易里的最有前景的技术了。我们接下来对bulletproof的数学原理进行一定程度的阐述，主要参考了notes和原论文。

2 原理

Pedersen commitments 是一种承诺方案，用来对一个数进行保证/承诺，同时又不让其他人看到这个数是多少。举个例子，赌博游戏里我想投注，但是我不想让其他人知道我投了多少，同时我也不能更改我的投注额或者进行其他的抵赖欺诈，承诺方案可以满足这样的要求.

2.1 Notation

我们用如下的符号定义系统。 小写字母a, b, c 表示$\mathbb{Z}_p$里的标量， 大写字母 G, H, P, Q 表示群 $\mathbb{G}$里的元素. 向量被记为粗体，例如 $\mathbf{a}$ 和 $\mathbf{G}$。 而两个向量的内积记为 $\langle -, - \rangle$。请注意内积 $\langle \mathbf{a}, \mathbf{b} \rangle \in \mathbb{Z}_p$ 输出的是一个标量, 而内积 $\langle \mathbf{a}, \mathbf{G} \rangle \in \mathbb{G}$是一个多标量的乘法。 全 0 和全 1 的向量记为 $\mathbf{0}, \mathbf{1}$。

对一个标量 y, 我们用

$$\mathbf{y}^n = (1, y, y^2, \ldots, y^{n-1})$$

表示相关的一个向量，且 第 i-th 元素是 y^i。对于具有偶数长度$2k$的向量v，我们定义分布为该向量的低半段和高半段:

$$\mathbf{v}_{\text{lo}} = (v_0, \ldots, v_{k-1})$$
$$\mathbf{v}_{\text{hi}} = (v_k, \ldots, v_{2k-1})$$

Pedersen commitments 被记为

$$\text{Com}(v) = \text{Com}(v, \widetilde{v}) = v \cdot B + \widetilde{v} \cdot \widetilde{B},$$

并且B 和 $\widetilde{B}$ 是这里被使用到的生成元和"盲化"因子(blinding factors). 我们将 v的盲化因子记为 $\widetilde{v}$, 使得变量和它的盲化因子之间可以清晰地关联起来。 为方便起见，我们记Com(v)符号为Commitment, 代替 Com$(v, \widetilde{v})$.

同时我们也用到了vector Pedersen commitments, 我们定义为

$$\text{Com}(\mathbf{a}_L, \mathbf{a}_R) = \text{Com}(\mathbf{a}_L, \mathbf{a}_R, \widetilde{a}) = \langle \mathbf{a}_L, \mathbf{G} \rangle + \langle \mathbf{a}_R, \mathbf{H} \rangle + \widetilde{a}\widetilde{B},$$

且 $\mathbf{G}$ 和 $\mathbf{H}$ 都是生成元组成的向量. 接下来进入我们的主题。 先介绍range proof. 顾名思义，range proof是"零知识地"去证明一个数值在某一个区间范围内。prover将要证明的值v, 先进行承诺 $V = com(v)$, 发送V给verifier。prover希望在接下来的过程里能证明v属于$[0, 2^n)$，同时不泄露v的具体的值。 假设a是v的各个bit组成的向量，例如$n = 3, v = 7$, a就是$< 1, 1, 1 >$; $n = 4, v = 10$, a就是$< 1, 0, 1, 0 >$。v可以被表示成一个内积，即:

$$\begin{aligned} v &= \langle \mathbf{a}, \mathbf{2}^n \rangle \\ &= a_0 \cdot 2^0 + \cdots + a_{n-1} \cdot 2^{n-1}. \end{aligned}$$

我们必须保证 $\mathbf{a}$ 是一个只包含 $\{0, 1\}$的向量. 这也可以用另一种形式来表示:

$$\mathbf{a} \circ (\mathbf{a} - \mathbf{1}) = \mathbf{0},$$

且 $\mathbf{x} \circ \mathbf{y}$ 记为两个向量之间的元素积，或者叫做Hadamard products.

因此二进制表示v时, $v \in [0, 2^n)$ 等价于以下两个等式

$$\langle \mathbf{a}, \mathbf{2}^n \rangle = v,$$
$$\mathbf{a} \circ (\mathbf{a} - \mathbf{1}) = \mathbf{0}.$$

更进一步，我们其实需要关注的是向量$\mathbf{a}$ 和 $\mathbf{a} - \mathbf{1}$, 因此我们记$\mathbf{a}_L = \mathbf{a}, \mathbf{a}_R = \mathbf{a} - \mathbf{1}$， 从而获得

$$\langle \mathbf{a}_L, \mathbf{2}^n \rangle = v,$$
$$\mathbf{a}_L \circ \mathbf{a}_R = \mathbf{0},$$
$$(\mathbf{a}_L - \mathbf{1}) - \mathbf{a}_R = \mathbf{0}.$$

2.2 Proving vectors of statements with a single statement

接下来我们需要对这三个等式进一步处理，使之转化为一个式子(statement)，方便我们的证明。 因为 $\mathbf{b} = \mathbf{0}$当且仅当 $\langle \mathbf{b}, \mathbf{y}^n \rangle = 0$ 对于任意的 y。 因此上述三个等式可以转化为

$$\langle \mathbf{a}_L, \mathbf{2}^n \rangle = v,$$
$$\langle \mathbf{a}_L - \mathbf{1} - \mathbf{a}_R, \mathbf{y}^n \rangle = 0,$$
$$\langle \mathbf{a}_L, \mathbf{a}_R \circ \mathbf{y}^n \rangle = 0$$

，对于verifier选择的任意的y都成立.

更进一步地，对于verifier选择地任意z,我们都可以有

$$z^2 v = z^2 \langle \mathbf{a}_L, \mathbf{2}^n \rangle + z \langle \mathbf{a}_L - \mathbf{1} - \mathbf{a}_R, \mathbf{y}^n \rangle + \langle \mathbf{a}_L, \mathbf{a}_R \circ \mathbf{y}^n \rangle$$

2.3 Combining inner products

我们需要再将上面地等式里地这些项进一步处理，转化成一个内积地形式，且转化后的内积<,>里，$\mathbf{a}_L$ 只出现在左边，$\mathbf{a}_R$ 只出现在右边，且我们将不包含秘密数的那些项合并起来记为一个新变量δ.

将这个statement拆开，再重新排列:

$$
\begin{aligned}
z^2 v &= z^2\langle \mathbf{a}_L, \mathbf{2}^n\rangle + z\langle \mathbf{a}_L, \mathbf{y}^n\rangle - z\langle \mathbf{a}_R, \mathbf{y}^n\rangle - z\langle \mathbf{1}, \mathbf{y}^n\rangle + \langle \mathbf{a}_L, \mathbf{a}_R \circ \mathbf{y}^n\rangle \\
z^2 v + z\langle \mathbf{1}, \mathbf{y}^n\rangle &= z^2\langle \mathbf{a}_L, \mathbf{2}^n\rangle + z\langle \mathbf{a}_L, \mathbf{y}^n\rangle - z\langle \mathbf{1}, \mathbf{a}_R \circ \mathbf{y}^n\rangle + \langle \mathbf{a}_L, \mathbf{a}_R \circ \mathbf{y}^n\rangle \\
z^2 v + z\langle \mathbf{1}, \mathbf{y}^n\rangle &= \langle \mathbf{a}_L, z^2\mathbf{2}^n\rangle + \langle \mathbf{a}_L, z\mathbf{y}^n\rangle + \langle -z\mathbf{1}, \mathbf{a}_R \circ \mathbf{y}^n\rangle + \langle \mathbf{a}_L, \mathbf{a}_R \circ \mathbf{y}^n\rangle \\
z^2 v + z\langle \mathbf{1}, \mathbf{y}^n\rangle &= \langle \mathbf{a}_L, z^2\mathbf{2}^n + z\mathbf{y}^n + \mathbf{a}_R \circ \mathbf{y}^n\rangle + \langle -z\mathbf{1}, \mathbf{a}_R \circ \mathbf{y}^n\rangle
\end{aligned}
$$

两边同时加上 $\langle -z\mathbf{1}, z^2\mathbf{2}^n + z\mathbf{y}^n\rangle$ 之后再约简:

$$
\begin{aligned}
z^2 v + z\langle \mathbf{1}, \mathbf{y}^n\rangle - \langle z\mathbf{1}, z^2\mathbf{2}^n + z\mathbf{y}^n\rangle &= \langle \mathbf{a}_L, z^2\mathbf{2}^n + z\mathbf{y}^n + \mathbf{a}_R \circ \mathbf{y}^n\rangle \\
&\quad + \langle -z\mathbf{1}, z^2\mathbf{2}^n + z\mathbf{y}^n + \mathbf{a}_R \circ \mathbf{y}^n\rangle \\
z^2 v + (z - z^2)\langle \mathbf{1}, \mathbf{y}^n\rangle - z^3\langle \mathbf{1}, \mathbf{2}^n\rangle &= \langle \mathbf{a}_L - z\mathbf{1}, z^2\mathbf{2}^n + z\mathbf{y}^n + \mathbf{a}_R \circ \mathbf{y}^n\rangle
\end{aligned}
$$

将所有地非秘密项整理到内积之外，记为

$$\delta(y, z) = (z - z^2)\langle \mathbf{1}, \mathbf{y}^n\rangle - z^3\langle \mathbf{1}, \mathbf{2}^n\rangle,$$

最后我们获得了等式

$$z^2 v + \delta(y, z) = \langle \mathbf{a}_L - z\mathbf{1}, \mathbf{y}^n \circ (\mathbf{a}_R + z\mathbf{1}) + z^2\mathbf{2}^n\rangle. ------------ \quad (1)$$

我们将内积地左边部分记为"unblinded" $\mathbf{l}(x)$，右边部分记为"unblinded" $\mathbf{r}(x)$, 因此有

$$
\begin{aligned}
\text{unblinded } \mathbf{l}(x) &= \mathbf{a}_L - z\mathbf{1} \\
\text{unblinded } \mathbf{r}(x) &= \mathbf{y}^n \circ (\mathbf{a}_R + z\mathbf{1}) + z^2\mathbf{2}^n \\
z^2 v + \delta(y, z) &= \langle \text{unblinded } \mathbf{l}(x), \text{unblinded } \mathbf{r}(x)\rangle
\end{aligned}
$$

2.4 Blinding the inner product

prover不能简单粗暴地将"unblinded" $\mathbf{l}(x)$和"unblinded" $\mathbf{r}(x)$直接发送给verifier,这样将会导致证明过程不是"零知识化"。因此，聪明的prover会选择这两个向量的blining factors（盲化因子？）：

$$\mathbf{s}_L, \mathbf{s}_R \xleftarrow{\$} \mathbb{Z}_p{}^n,$$

, 并且用他们来构造盲化后的多项式:

$$
\begin{aligned}
\mathbf{l}(x) &= \mathbf{l}_0 + \mathbf{l}_1 x = (\mathbf{a}_L + \mathbf{s}_L x) - z\mathbf{1} & &\in \mathbb{Z}_p[x]^n \\
\mathbf{r}(x) &= \mathbf{r}_0 + \mathbf{r}_1 x = \mathbf{y}^n \circ ((\mathbf{a}_R + \mathbf{s}_R x) + z\mathbf{1}) + z^2\mathbf{2}^n & &\in \mathbb{Z}_p[x]^n
\end{aligned}
$$

"blinded" $\mathbf{l}(x)$和"blinded" $\mathbf{r}(x)$ 将项$\mathbf{a}_L, \mathbf{a}_R$ 盲化，用项 $\mathbf{a}_L + \mathbf{s}_L x, \mathbf{a}_R + \mathbf{s}_R x$代替. $\mathbf{l}_0$ 和 $\mathbf{r}_0$ 项表示多项式里度数为0的项（关于 x）, 类似的 $\mathbf{l}_1$ 和 $\mathbf{r}_1$ 表示多项式里度数为1的项.

很显然，我们有:

$$\langle \mathbf{l}_0, \mathbf{r}_0\rangle = z^2 v + \delta(y, z)$$

,

然后我们记

$$t(x) = \langle \mathbf{l}(x), \mathbf{r}(x)\rangle = t_0 + t_1 x + t_2 x^2,$$

， 我们将系数 $t(x)$ 用 Karatsuba's 方法展开:

$$
\begin{aligned}
t_0 &= \langle \mathbf{l}_0, \mathbf{r}_0\rangle, \\
t_2 &= \langle \mathbf{l}_1, \mathbf{r}_1\rangle, \\
t_1 &= \langle \mathbf{l}_0 + \mathbf{l}_1, \mathbf{r}_0 + \mathbf{r}_1\rangle - t_0 - t_2
\end{aligned}
$$

因为

$$t_0 = \langle \mathbf{a}_L - z\mathbf{1}, \mathbf{y}^n \circ (\mathbf{a}_R + z\mathbf{1}) + z^2\mathbf{2}^n \rangle,$$

prover希望对verifier证明上面那个未盲化的内积等式(等式(1))成立，实质上等价于证明1. $t(x)$的常数项 t_0等于 $z^2v + \delta(y,z)$, 并且 2.$t(x)$ 是正确的多项式. 证明$t(x)$ 是正确的多项式，等价于证明 $\mathbf{l}(x), \mathbf{r}(x)$ 均是正确的,并且 $t(x) = \langle \mathbf{l}(x), \mathbf{r}(x) \rangle$. 接下来我们分别解释怎么证明以上两点。

2.5 Proving that t_0 is correct

prover 首先制作一个关于$t(x)$的系数的commitment, 然后将通过准确回答verifier给出的任意挑战值x,来向verifier证明"这些commitments是对$t(x)$的正确承诺"。 prover已经用$V = \mathrm{Com}(v)$来承诺了v(本质是承诺了t_0), 因此prover再计算两个承诺: $T_1 = \mathrm{Com}(t_1)$ and $T_2 = \mathrm{Com}(t_2)$，并且把这些承诺发送给了verifier。 注意到，这些承诺V, T_1, T_2 互相之间且与 $t(x)$均形成了一些关系，即我们有如下的式子：

$$\begin{array}{ccccccccc}
t(x)B & = & z^2vB & + & \delta(y,z)B & + & xt_1B & + & x^2t_2B \\
+ & & + & & + & & + & & + \\
\tilde{t}(x)\widetilde{B} & = & z^2\tilde{v}\widetilde{B} & + & 0\widetilde{B} & + & x\tilde{t}_1\widetilde{B} & + & x^2\tilde{t}_2\widetilde{B} \\
\shortparallel & & \shortparallel & & \shortparallel & & \shortparallel & & \shortparallel \\
 & = & z^2V & + & \delta(y,z)B & + & xT_1 & + & x^2T_2
\end{array}$$

请注意每个列的和是一个对该列第一行里的变量承诺，且该承诺用了该列第二行里的变量作为blinding factor。而所有列的和就是 $t(x)B + \tilde{t}(x)\widetilde{B}$, 即对$t$在取值$x$时进行承诺, 且用了正交盲化因子:

$$\tilde{t}(x) = z^2\tilde{v} + x\tilde{t}_1 + x^2\tilde{t}_2.$$

为了向verifier证明$t(x) = z^2v + \delta(y,z) + t_1x + t_2x^2$, prover将向verifier发送$t(x), \tilde{t}(x)$, 后者根据以下式子检查一致性：:

$$t(x)B + \tilde{t}(x)\widetilde{B} \stackrel{?}{=} z^2V + \delta(y,z)B + xT_1 + x^2T_2.$$

2.6 Proving that $\mathbf{l}(x), \mathbf{r}(x)$ are correct

我们希望将 $\mathbf{l}(x)$ 和$\mathbf{r}(x)$与$\mathbf{a}_L, \mathbf{a}_R, \mathbf{s}_L$, and $\mathbf{s}_R$这一组变量进行关联。因为有:

$$\mathbf{r}(x) = \mathbf{y}^n \circ ((\mathbf{a}_R + \mathbf{s}_Rx) + z\mathbf{1}) + z^2\mathbf{2}^n,$$

我们需要找到关于 $\mathbf{y}^n \circ \mathbf{a}_R$ 和 $\mathbf{y}^n \circ \mathbf{s}_R$这两个复合变量的承诺. 但是我们知道prover必须在verifier给出挑战值y之前计算出承诺, 因此prover只具备对a_R和s_R计算对应的承诺值的能力。 verifier需要将prover的承诺$\mathrm{Com}(\mathbf{a}_L, \mathbf{a}_R, \widetilde{a})$变形成 $\mathrm{Com}(\mathbf{a}_L, \mathbf{y}^n \circ \mathbf{a}_R, \widetilde{a})$ (对$\mathbf{s}_R$也要进行类似的变形). 我们注意到

$$\begin{aligned}
\mathrm{Com}(\mathbf{a}_L, \mathbf{a}_R, \widetilde{a}) &= \langle \mathbf{a}_L, \mathbf{G} \rangle + \langle \mathbf{a}_R, \mathbf{H} \rangle + \widetilde{a}\widetilde{B} \\
&= \langle \mathbf{a}_L, \mathbf{G} \rangle + \langle \mathbf{y}^n \circ \mathbf{a}_R, \mathbf{y}^{-n} \circ \mathbf{H} \rangle + \widetilde{a}\widetilde{B},
\end{aligned}$$

因此我们记 $\mathbf{H}' = \mathbf{y}^{-n} \circ \mathbf{H}$, 则关于 $(\mathbf{G}, \mathbf{H}, \widetilde{a})$的承诺$(\mathbf{a}_L, \mathbf{a}_R, \widetilde{a})$ 被变形为关于 $(\mathbf{G}, \mathbf{H}', \widetilde{a})$的承诺$(\mathbf{a}_L, \mathbf{y}^n \circ \mathbf{a}_R, \widetilde{a})$。为了将prover的承诺 $A = \mathrm{Com}(\mathbf{a}_L, \mathbf{a}_R)$ 和承诺$S = \mathrm{Com}(\mathbf{s}_L, \mathbf{s}_R)$ 关联到 $\mathbf{l}(x)$ 和 $\mathbf{r}(x)$, 我们用到如下式子:

$$\begin{array}{ccccccc}
\langle \mathbf{l}(x), \mathbf{G} \rangle & = & \langle \mathbf{a}_L, \mathbf{G} \rangle & + & x\langle \mathbf{s}_L, \mathbf{G} \rangle & + & \langle -z\mathbf{1}, \mathbf{G} \rangle \\
+ & & + & & + & & + \\
\langle \mathbf{r}(x), \mathbf{H}' \rangle & = & \langle \mathbf{a}_R, \mathbf{H} \rangle & + & x\langle \mathbf{s}_R, \mathbf{H} \rangle & + & \langle z\mathbf{y}^n + z^2\mathbf{2}^n, \mathbf{H}' \rangle \\
+ & & + & & + & & \\
\widetilde{e}\widetilde{B} & = & \widetilde{a}\widetilde{B} & + & x\widetilde{s}\widetilde{B} & & \\
\shortparallel & & \shortparallel & & \shortparallel & & \shortparallel \\
 & = & A & + & xS & + & \langle z\mathbf{y}^n + z^2\mathbf{2}^n, \mathbf{H}' \rangle - z\langle \mathbf{1}, \mathbf{G} \rangle
\end{array}$$

与前面那个和式对列和行的分析类似，不难发现，上面式子里的每一列都是一个vector Pedersen commitment，且第三行里的元素均是相应的盲化因子。所有列的和，也是一个vector Pedersen commitment，其正交盲化因子是$\widetilde{e}$. 为了向verifier证明$t(x) = \langle \mathbf{l}(x), \mathbf{r}(x) \rangle$,prover需要将$\widetilde{e}$发送给verifier,后者计算以下的式子：

$$\begin{aligned} P &= -\widetilde{e}\widetilde{B} + A + xS + \langle z\mathbf{y}^n + z^2\mathbf{2}^n, \mathbf{H}' \rangle - z\langle \mathbf{1}, \mathbf{G} \rangle \\ &= -\widetilde{e}\widetilde{B} + A + xS + \langle z\mathbf{1} + z^2\mathbf{y}^{-n} \circ \mathbf{2}^n, \mathbf{H} \rangle - z\langle \mathbf{1}, \mathbf{G} \rangle; \end{aligned}$$

如果prover是诚实的，则$P = \langle \mathbf{l}(x), \mathbf{G} \rangle + \langle \mathbf{r}(x), \mathbf{H}' \rangle$, 因此verifier用 $P, t(x)$ 作为内积协议的输入来证明 $t(x) = \langle \mathbf{l}(x), \mathbf{r}(x) \rangle$.我们接下来叙述如何执行内积协议。

2.7 内置了inner-product argument protocol (”内积论证协议”)的inner-product proof

首先，一个直接的办法是prover将向量 $\mathbf{l}(x)$ and $\mathbf{r}(x)$ 直接发送给verifier,后者可以根据以下式子直接计算内积$t(x)$和承诺P是否是正确的。

$$\begin{aligned} t(x) &= \langle \mathbf{l}(x), \mathbf{r}(x) \rangle \\ P &= \langle \mathbf{l}(x), \mathbf{G} \rangle + \langle \mathbf{r}(x), \mathbf{H}' \rangle \end{aligned}$$

尽管 这样做不会造成信息泄露，即证明过程确实是零知识化到，但是需要出在prover和verifier之间传递$2n$个标量。为了节省带宽，我们给出inner-product argument protoocl(内积论证协议),可以让我们进行间接地证明，并且通信开销减低到 $O(log(n))$.

接下来我们需要修改一下符号定义系统，使得这部分将要阐述的inner-product argument protocol的定义不会与前文的range proof定义冲突：

$$\begin{aligned} \mathbf{a}, \mathbf{b} &\in \mathbb{Z}_p^n \\ \mathbf{G}, \mathbf{H} &\in \mathbb{G}^n \\ c &= \langle \mathbf{a}, \mathbf{b} \rangle \\ P &= \langle \mathbf{a}, \mathbf{G} \rangle + \langle \mathbf{b}, \mathbf{H} \rangle \end{aligned}$$

根据这套新的定义，我们需要证明的是以下这一个”proof of knowledge”（知识证明 – 是指一种prover可以向verifier交互式地证明他知道某个知识）：

$$\begin{aligned} P = &\quad \langle \mathbf{a}, \mathbf{G} \rangle + \langle \mathbf{b}, \mathbf{H} \rangle \ \wedge \\ c = &\quad \langle \mathbf{a}, \mathbf{b} \rangle \end{aligned}$$

我们引入一个中间变量$w \in \mathbb{Z}_p^\times$，再对第二个等式两侧同时乘以一个正交生成元$B \in \mathbb{G}$，将这两个statement合并为一个等式，即：

$$\begin{aligned} P = &\quad \langle \mathbf{a}, \mathbf{G} \rangle + \langle \mathbf{b}, \mathbf{H} \rangle \\ + & \\ cwB = &\quad \langle \mathbf{a}, \mathbf{b} \rangle wB \end{aligned}$$

继续引入以下符号对上面地等式简化：

$$\begin{aligned} k &= \lg n \\ P' &= P + cwB \\ Q &= wB \end{aligned}$$

等式变成了:

$$P' = \langle \mathbf{a}, \mathbf{G} \rangle + \langle \mathbf{b}, \mathbf{H} \rangle + \langle \mathbf{a}, \mathbf{b} \rangle Q$$

上面这个合并后的等式非常关键，因为它可以让我们对等式里的各个向量进行持续地“对半压缩”并且压缩后获得的新等式仍然保持相同的结构。通过压缩$\lg n$次，我们会获得一个最终等式只有2个向量且每个向量的长度只包含有一个元素，这样最后的校验就变得非常简单。 再强调一下，这里如果我们证明了 对于所有的$w \in \mathbb{Z}_p^*$，P'都有上述等式里的组成结构，那么上面的P 和 c 也一定会符合等式($P = \langle \mathbf{a}, \mathbf{G} \rangle + \langle \mathbf{b}, \mathbf{H} \rangle \wedge c = \langle \mathbf{a}, \mathbf{b} \rangle$)。在UTXO模型里，只要证明Input大于Output，这笔交易就可以被认为是有效的。也就是Output在

0和Input之间。之前有一种证明技术叫range 接下来我们介绍一下具体的压缩过程。 我们引入一个中间变量$u_k \in \mathbb{Z}_p^\times$,并且我们对原始的$\mathfrak{a}$,b,G,H进行如下变换：

$$\begin{aligned}
\mathfrak{a}^{(k-1)} &= \mathfrak{a}_{\mathrm{lo}} \cdot u_k + u_k^{-1} \cdot \mathfrak{a}_{\mathrm{hi}} \\
\mathbf{b}^{(k-1)} &= \mathbf{b}_{\mathrm{lo}} \cdot u_k^{-1} + u_k \cdot \mathbf{b}_{\mathrm{hi}} \\
\mathbf{G}^{(k-1)} &= \mathbf{G}_{\mathrm{lo}} \cdot u_k^{-1} + u_k \cdot \mathbf{G}_{\mathrm{hi}} \\
\mathbf{H}^{(k-1)} &= \mathbf{H}_{\mathrm{lo}} \cdot u_k + u_k^{-1} \cdot \mathbf{H}_{\mathrm{hi}}
\end{aligned}$$

我们令$P_k = P'$，并且我们采用与P_k类似的结构,但是用的是压缩后的向量来定义P_{k-1}：

$$P_{k-1} = \langle \mathfrak{a}^{(k-1)}, \mathbf{G}^{(k-1)} \rangle + \langle \mathbf{b}^{(k-1)}, \mathbf{H}^{(k-1)} \rangle + \langle \mathfrak{a}^{(k-1)}, \mathbf{b}^{(k-1)} \rangle \cdot Q$$

将它展开得到:

$$\begin{aligned}
P_{k-1} = \quad & \langle \mathfrak{a}_{\mathrm{lo}} \cdot u_k + u_k^{-1} \cdot \mathfrak{a}_{\mathrm{hi}}, \mathbf{G}_{\mathrm{lo}} \cdot u_k^{-1} + u_k \cdot \mathbf{G}_{\mathrm{hi}} \rangle + \\
& \langle \mathbf{b}_{\mathrm{lo}} \cdot u_k^{-1} + u_k \cdot \mathbf{b}_{\mathrm{hi}}, \mathbf{H}_{\mathrm{lo}} \cdot u_k + u_k^{-1} \cdot \mathbf{H}_{\mathrm{hi}} \rangle + \\
& \langle \mathfrak{a}_{\mathrm{lo}} \cdot u_k + u_k^{-1} \cdot \mathfrak{a}_{\mathrm{hi}}, \mathbf{b}_{\mathrm{lo}} \cdot u_k^{-1} + u_k \cdot \mathbf{b}_{\mathrm{hi}} \rangle \cdot Q
\end{aligned}$$

进一步拆分内积，获得更细的内积项得到:

$$\begin{aligned}
P_{k-1} = \quad & \langle \mathfrak{a}_{\mathrm{lo}}, \mathbf{G}_{\mathrm{lo}} \rangle + \langle \mathfrak{a}_{\mathrm{hi}}, \mathbf{G}_{\mathrm{hi}} \rangle & + u_k^2 \langle \mathfrak{a}_{\mathrm{lo}}, \mathbf{G}_{\mathrm{hi}} \rangle + u_k^{-2} \langle \mathfrak{a}_{\mathrm{hi}}, \mathbf{G}_{\mathrm{lo}} \rangle + \\
& \langle \mathbf{b}_{\mathrm{lo}}, \mathbf{H}_{\mathrm{lo}} \rangle + \langle \mathbf{b}_{\mathrm{hi}}, \mathbf{H}_{\mathrm{hi}} \rangle & + u_k^2 \langle \mathbf{b}_{\mathrm{hi}}, \mathbf{H}_{\mathrm{lo}} \rangle + u_k^{-2} \langle \mathbf{b}_{\mathrm{lo}}, \mathbf{H}_{\mathrm{hi}} \rangle + \\
& (\langle \mathfrak{a}_{\mathrm{lo}}, \mathbf{b}_{\mathrm{lo}} \rangle + \langle \mathfrak{a}_{\mathrm{hi}}, \mathbf{b}_{\mathrm{hi}} \rangle) \cdot Q & + (u_k^2 \langle \mathfrak{a}_{\mathrm{lo}}, \mathbf{b}_{\mathrm{hi}} \rangle + u_k^{-2} \langle \mathfrak{a}_{\mathrm{hi}}, \mathbf{b}_{\mathrm{lo}} \rangle) \cdot Q
\end{aligned}$$

观察后发现，上面这个等式的左边两列其实就是P_k。然后我们将这个等式进一步表示成如下这个等式：

$$\begin{aligned}
P_{k-1} &= P_k + u_k^2 \cdot L_k + u_k^{-2} \cdot R_k \\
L_k &= \langle \mathfrak{a}_{\mathrm{lo}}, \mathbf{G}_{\mathrm{hi}} \rangle + \langle \mathbf{b}_{\mathrm{hi}}, \mathbf{H}_{\mathrm{lo}} \rangle + \langle \mathfrak{a}_{\mathrm{lo}}, \mathbf{b}_{\mathrm{hi}} \rangle \cdot Q \\
R_k &= \langle \mathfrak{a}_{\mathrm{hi}}, \mathbf{G}_{\mathrm{lo}} \rangle + \langle \mathbf{b}_{\mathrm{lo}}, \mathbf{H}_{\mathrm{hi}} \rangle + \langle \mathfrak{a}_{\mathrm{hi}}, \mathbf{b}_{\mathrm{lo}} \rangle \cdot Q
\end{aligned}$$

如果prover确实是诚实地在随机选择u_k之前对L_k 和 R_k进行了承诺计算，并且上面这个等式成立，则原始的statement（关于P的那个等式）是极大概率成立（密码学里，极大概率就是可以直接看作是成立的）。 接下来我们可以继续对P_{k-1}压缩，与上面过程类似地引入中间变量u_{k-1}，...，一直到我们到达最后的$\mathfrak{a}^{(0)}, \mathbf{b}^{(0)}, \mathbf{G}^{(0)}, \mathbf{H}^{(0)}$，我们有：

$$P_0 = a_0^{(0)} G_0^{(0)} + b_0^{(0)} H_0^{(0)} + a_0^{(0)} b_0^{(0)} Q$$

$$P_0 = P_k + \sum_{j=1}^{k} \left(L_j u_j^2 + u_j^{-2} R_j \right)$$

将上面的等式按照定义$P_k = P' = P + cwB$ 和 $Q = wB$，进行重写后我们发现：

$$P + cwB = a_0^{(0)} G_0^{(0)} + b_0^{(0)} H_0^{(0)} + a_0^{(0)} b_0^{(0)} wB - \sum_{j=1}^{k} \left(L_j u_j^2 + u_j^{-2} R_j \right)$$

到这一步，prover可以简单地发送2个标量$a_0^{(0)}$ 和$b_0^{(0)}$给verifier,这样后者可以直接校验上面这个最终步的等式是否成立。 总体的inner-product argument protocol有 $\lg n$步，并且每一步都需要prover将(L_j, R_j)发送给verifier，$j = k \ldots 1$。至此，对于证明$\mathbf{l}(x), \mathbf{r}(x)$ are correct, 一共需要发送$2 \lg n + 2$个元素。

关于如何将整个证明协议转化为非交互式地，和如何聚合range proof提高批量验证效率的部分，由于篇幅的关系我们在此不做赘述。有兴趣的朋友可以查看原文或者直接学习notes（https://doc-internal.dalek.rs/bulletproofs/notes/index.html）。