Improved Circuit-based PSI via Equality Preserving Compression

Zhang Xiangling

1 Circuit-based PSI

Parameters: A receiver with an input set X of size N and a sender with an input set Y of size N.
Functionality: The functionality sends to the receiver an injective indexing function $\iota : X \to [M]$ for some $M \geq N$ and a vector $s_0 \in \{0,1\}^M$, and to the sender a vector $\mathbf{s}_1 \in \{0,1\}^M$ such that $s_{0,i} \oplus s_{1,i} = \mathbf{1}\left(\iota^{-1}(i) \in X \cap Y\right)$ for $i \in \iota(X)$, and $s_{0,i} \oplus s_{1,i} = 0$ for $i \notin \iota(X)$.

Note that the indexing function ι is determined by the mapping from $x \in X$ to $h_j(x)$ (cuckoo hash mapping).

2 The OPPRF-based Circuit-PSI Framework

The main idea is to apply OPPRF in circuit-PSI protocol, and use EPC to reduce workload of ESG, which occupied the largest part of circuit-PSI. ESG has complexity linear in ℓ, the bit-length of input string. EPC can change the input bit-length of ESG from ℓ to $\ell_c = O(\log \ell)$.

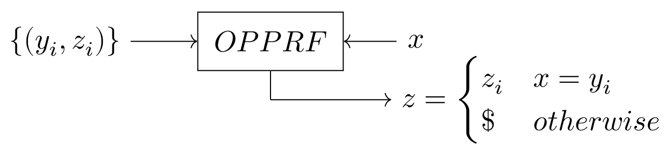
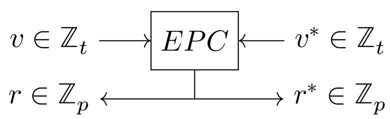
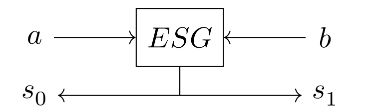


Figure 1: Functionality of oblivious programmable PRF



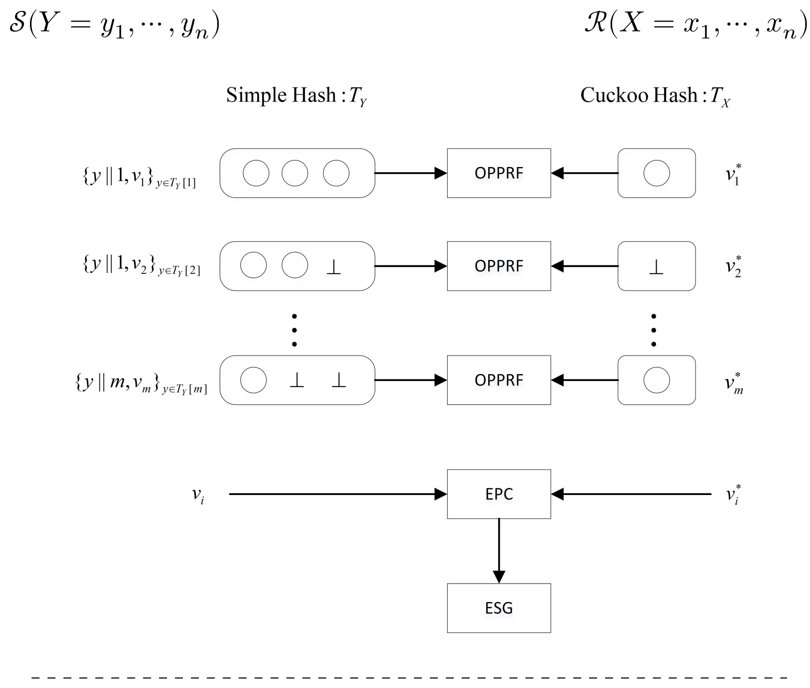
$r = r^*$ if and only if $v = v^*$

Figure 2: Functionality of equality preserving compression



$s_0 \oplus s_1 = 1$ if and only if $a = b$

Figure 3: Functionality of equality share generation



[OPPRF]The sender samples uniformly random tags $\mathbf{v} \in \mathbb{Z}_{2^\ell}^M$ and sends $L = \{(y' \| i, v_i)\}_{i \in [M], y' \in T_Y[i]}$ to $\mathcal{F}_{\text{OPPRF}}$. The receiver sends $\tilde{T}_X = \{T_X[i] \| i\}_{i \in [M]}$ to $\mathcal{F}_{\text{OPPRF}}$, and receives $\mathbf{v}^* \in \mathbb{Z}_{2^\ell}^M$ from $\mathcal{F}_{\text{OPPRF}}$.

Figure 4: OPPRF-based circuit PSI+EPC

2.1 A basic protocol for $\mathcal{F}_{\text{EPC}}$ functionalities

The w-base decomposition of $v \in \mathbb{Z}_t$ and $v^* \in \mathbb{Z}_t$. $(t = 2^\ell, p = 2^{\ell_c})$

$$v = \sum_{i=0}^{u-1} v_i \cdot w^i, \quad v^* = \sum_{i=0}^{u-1} v_i^* \cdot w^i$$

where $u := \lceil \log_w t \rceil$ and $v_i, v_i^* \in [0, w)$ satisfies

$$v = v^* \Longleftrightarrow D := \sum_{i=0}^{u-1} (v_i - v_i^*)^2 = 0 \text{ in } \mathbb{Z}_p$$

Requirement:

$$p > u \cdot (w-1)^2, \text{ avoid } D \text{ is divisible by } p$$

$$p = 1 \bmod 2n, \text{ requires by RLWE-HE batching } n \text{ times}$$

$$\ell = \sigma + 1 + \lceil \log N \rceil$$

		EPC output ℓ_c				
		16	18	20	22	28
	p	40961	188417	1032193	4169729	268369921
	$\log q$	84	88	92	96	108
$N = 2^{16}$	w	65	154	385	834	7327
$(\ell = 57)$	u	10	8	7	6	5
$N = 2^{20}$	w	62	145	360	772	7327
$(\ell = 61)$	u	11	9	8	7	5

Table 3. EPC Parameters for input set size N and EPC output length ℓ_c: w is the word-decompose base, and u is the length of decomposition. Note that HE parameters are independent to set size N.

$$D = \sum_{i=0}^{u-1} v_i^2 - 2 \cdot \sum_{i=0}^{u-1} v_i \cdot v_i^* + \sum_{i=0}^{u-1} v_i^{*2}$$

Parameters: A sender with input $v \in \mathbb{Z}_t$ and a receiver with input $v^* \in \mathbb{Z}_t$ and the target size p.
Protocol:
offline phase

Sender generates a HE secret key sk, and decomposes $v \in \mathbb{Z}_t$ to $\{v_i\}_{0 \le i < u}$. After that sender encrypts each v_i and $\sum_{i=0}^{u-1} v_i^2$ using sk, obtains ciphertext $\{ct_i\}_{0 \le i < u}$ and cts.

online phase

1. Sender sends ciphertext $\{ct_i\}_{0 \le i < u}$ and cts to receiver.
2. Receiver picks a random integer $r \in \mathbb{Z}_p$, and decomposes $v^* \in \mathbb{Z}_t$ to $\{v_i^*\}_{0 \le i < u}$ Then receiver homomorphically compute $r + (cts - 2 \cdot \sum_{i=0}^{u-1} ct_i \cdot v_i^* + \sum_{i=0}^{u-1} v_i^{*2})$, and sends the resulting ciphertext back to sender.
3. Sender decrypts the received ciphertext using sk, to obtain $r^* = r + \sum_{i=0}^{u-1} (v_i - v_i^*)^2 \in \mathbb{Z}_p$.

The HE in EPC protocol requires additive homomorphic. The encryption target message size is much less than 32-bit($\log w$ in table 3) and decryption ciphertext size is also less than 32-bit(ℓ_c in table 3).