

Algorithmus basierend auf ElGamal

Client

- ⇒ Erstellt Bloom Filter der Daten
- ⇒ Verschlüsselt jede Stelle des Bloom Filters mittels ElGamal

$$(R_i, S_i) = (g^{r_i}, pk^{r_i} * g^{1-BF_1[i]})$$

- ⇒ Alice entschlüsselt mit sk Ciphertext von Bob
- ⇒ Bestimmt Anzahl der Einträge an denen beide Bloom Filter null sind
- ⇒ Schätzt hieraus die Gesamtmenge an SNPs

Server

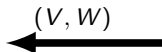
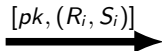
- ⇒ Erstellt Bloom Filter der Daten
- ⇒ Selektiert jene Stellen in dem BF die den Eintrag null besitzen.

⇒ Multipliziert an diesen Stellen die Werte des Ciphertextes vom Client auf

⇒ Rerandomisiert die entstandenen Ergebnisse

$$V = (g^s * \prod_{i:BF_2[i]=0} R_i)$$

$$W = (pk^s * \prod_{i:BF_2[i]=0} S_i)$$



Abschätzung der Elemente in einem Bloomfilter

$$|X| = \frac{\ln(\frac{z}{m})}{k * \ln(1 - \frac{1}{m})}$$

⇒ Basiert auf Abschätzung der Anzahl an Nullen in einem Bloomfilter in einem Bloomfilter:

$$\Rightarrow z = m * (1 - \frac{1}{m})^{k * X}$$

⇒ Wahrscheinlichkeit, dass ein Bloomfilterbit Null ist

$$z' = (1 - \frac{1}{m})^{k* X}$$

⇒ Da biomialverteilt ist der Erwartungswert: $z = m * (1 - \frac{1}{m})^{k* X}$

⇒

Paillier - Verfahren

Schlüsselerzeugung:

- ⇒ Client wählt zwei Primzahlen p, q , mit
 $\text{ggt}(pq, (p-1)(q-1)) = 1$
- ⇒ Der Generator g so gewählt, sodass $g \in (\mathbb{Z}_{n^2}^*)$ und n die
Ordnung von g teilt.
- ⇒ Secret key: $\lambda = \text{kgV}(p-1, q-1)$
- ⇒ Public Key: (n, g)

Verschlüsselung:

⇒ Client wählt Zufallszahl r wobei $0 \leq r \leq n$

⇒ Ciphertext $c = g^m * r^n \mod n^2$

Entschlüsselung:

⇒ Benötigt zunächst $L(u) = \frac{(u-1)}{n}$

⇒ Plaintext $m = \frac{L(c^\lambda \mod n^2)}{L(g^\lambda \mod n^2)} \mod n$

Homomorphie: Paillier ist homomorph gegenüber der Addition.

$$E(m_1 + m_2) = (E(m_1) + E(m_2))$$

Algorithmus basierend auf Paillier

Client

⇒ Erstellt Bloomfilter der Daten und invertiert jede Stelle des Bloomfilters.

⇒ Verschlüsselt jede Stelle des Bloomfilters mittels Paillier

$$c_{1-m} = (g^m * r^n) \bmod n^2$$

⇒ Client entschlüsselt mit sk Ciphertexte von Server

⇒ Anzahl der entschlüsselten Nullen entspricht der Anzahl der sich überschneidenden Elemente

Server

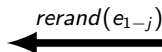
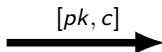
⇒ Erstellt für jedes Element des Datensatzes einen Bloomfilter seiner Daten

⇒ Selektiert in jedem Blommfilter jene Stellen die den Eintrag Eins besitzen.

⇒ Addiert an diesen Stellen die Werte des Ciphertextes des Clients auf

⇒ Rerandomisiert die entstandenen Ergebnisse mit verschlüsselter Null

$$\text{Rerand } e_j = (e_j * \text{enc}_{\text{paillier}}(0))$$



Ergebnisse - Elgamal

- ⇒ Dauer für Vergleich des gesamten Exomes bei wenigen Minuten.
- ⇒ Laufzeit Unabhängig davon wie stark die Überschneidung zwischen zwischen den Datensätzen ist.

Überschneidung	14000	7500	5000	2000
Runtime (sec)	221	247	211	222
Abw. zur Überschn.	0.01%	3.3%	8.8%	36.8%

Table 1: Hashfunktionen : 14, Anzahl Bloomfilter Bits:3029660, Größe der Datensätze: 15000 SNPs

Array	1442696	1009887	577079	144270
Runtime (sec)	108	83	47	11
Abweichung	4%	6%	13%	51%

Table 2: Datensatz 1000 SNPs, Überschneidung 100, Hashfunktionen: 10

- ⇒ Die Laufzeit ist linear abhängig zur Anzahl der Bloomfilterbits
- ⇒ Die Stärke der Abweichung ist ebenfalls linear abhängig zur Anzahl der Bloomfilter Bits

Hashf.	1	4	7	10	14
Runtime (sec)	7	27	44	62	104
Abweichung	11%	13%	10%	9%	9%

Table 3: Datensatz 1000 SNPs, Überschneidung 100, Array: 504944

⇒ Anzahl der Hashfunktionen hat deutlich weniger Einfluss, jedoch kommt es bei hoher Anzahl zu vermehrt Falsch positiven Ergebnissen.

Ergebnisse-Paillier

Array	14139	12119	10099	8080
Runtime (sec)	219	194	183	163
Abweichung	1%	4%	6%	24%

Table 4: Hashf.7, Überschneidung 100, SNPs 1000

- ⇒ Paillier deutlich langsamer als Elgamal
- ⇒ Benötigt deutlich kleinere Bloomfilter für selbe Genauigkeit, jedoch ist die Bitweise Verschlüsselung sehr langsam

Array	141385	100989	75742
Runtime (sec)	2420	2318	2007
Abweichung	1%	4%	13%

Table 5: Hashf.7, Überschneidung 7500, SNPs 15000

- ⇒ Zum Vergleich von gesamten Exomen ca. 40 min
- ⇒ Aufgrund der kleineren Bloomfiltergröße kommt es jedoch nicht so schnell zum Überlauf des Arbeitsspeichers.

Vergleich

Abweichung	0.1%	0.6%	2%	3%	4%	6%
Runtime elgamal	467	150	17	15	11	6
Runtime paillier	510	340	150	150	135	120

Table 6: Hashf.7, Überschneidung 100, SNPs 1000