Homomorphic Encryption Based Secure Genome Data Analysis

Miran Kim[⋆] and Kristin Lauter[†]

[⋆]Seoul National University

[†]Microsoft Research

iDASH Privacy&Security Workshop, March 16, 2015

Secure Outsourcing GWAS

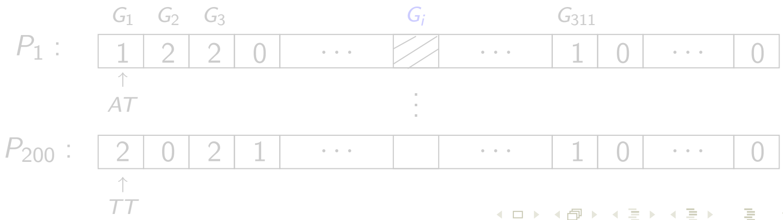
Minor Allele Frequency

- There are 200 people, and each of them has 311 genotypes.
- Each genotype has two kinds of SNPs.

- Data Encoding

  For a fixed genotype, suppose that 200 people have “AT, AT, AA, . . ., TT”. Then the encoding method is as follows:
  - If the pair consists of different SNPs (AT), then encode it into ‘1’.
  - The first pair with the same SNP (AA) is encoded into ‘0’.
  - Then the other one (TT) is encoded into ‘2’.

  (⇒ the encoded value means the number of ‘T’ in the individual SNPs.)



G_1 G_2 G_3 G_i G_{311}
P_1 : 1 2 2 0 ⋯ ▨ ⋯ 1 0 ⋯ 0
↑ AT
⋮
P_{200} : 2 0 2 1 ⋯ ⋯ 1 0 ⋯ 0
↑ TT

Minor Allele Frequency

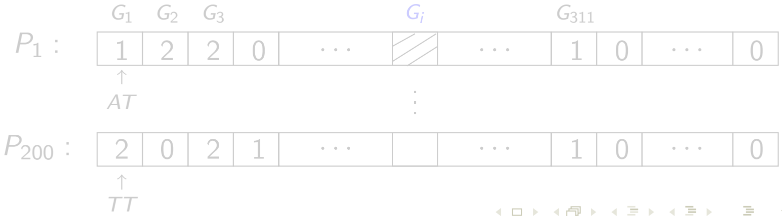
- There are 200 people, and each of them has 311 genotypes.
- Each genotype has two kinds of SNPs.

- Data Encoding

  For a fixed genotype, suppose that 200 people have "AT, AT, AA, . . ., TT". Then the encoding method is as follows:

  - If the pair consists of different SNPs (AT), then encode it into '1'.
  - The first pair with the same SNP (AA) is encoded into '0'.
  - Then the other one (TT) is encoded into '2'.

  (⇒ the encoded value means the number of 'T' in the individual SNPs.)


G_1 G_2 G_3 G_i G_{311}
P_1 : 1 2 2 0 · · · · · · 1 0 · · · 0
AT
P_{200} : 2 0 2 1 · · · · · · 1 0 · · · 0
TT

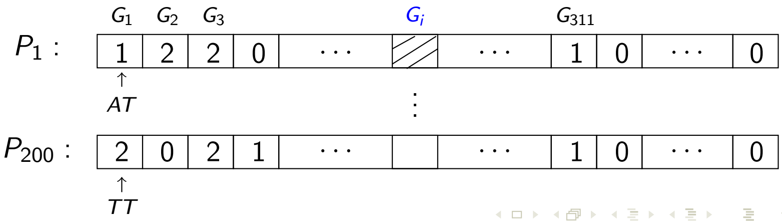
Minor Allele Frequency

- There are 200 people, and each of them has 311 genotypes.
- Each genotype has two kinds of SNPs.

- Data Encoding

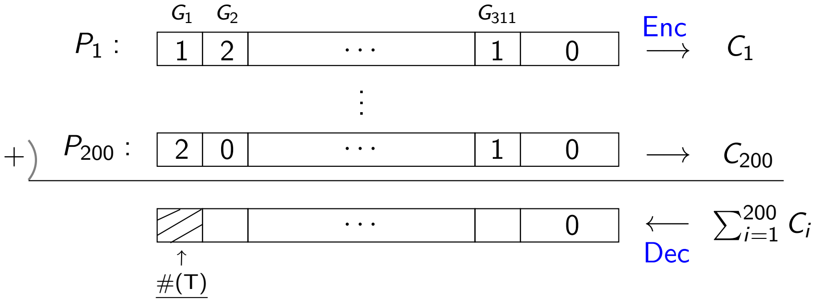
  For a fixed genotype, suppose that 200 people have "AT, AT, AA, . . ., TT". Then the encoding method is as follows:
  - If the pair consists of different SNPs (AT), then encode it into '1'.
  - The first pair with the same SNP (AA) is encoded into '0'.
  - Then the other one (TT) is encoded into '2'.

  ($\Rightarrow$ the encoded value means the number of 'T' in the individual SNPs.)



Minor Allele Frequency

- Encryption & Evaluation



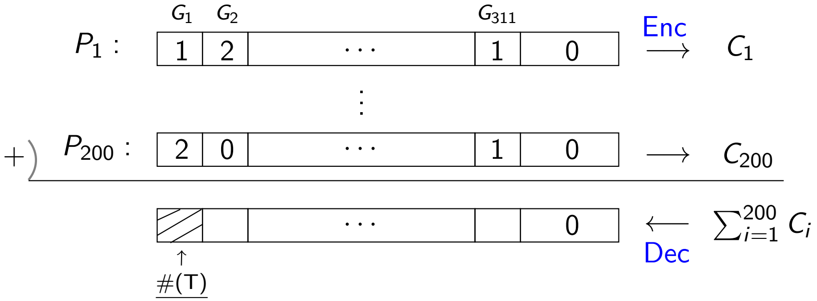
G_1 G_2 G_{311}
P_1 : 1 2 ⋯ 1 0 $\xrightarrow{\text{Enc}}$ C_1
⋮
$+)$ P_{200} : 2 0 ⋯ 1 0 $\longrightarrow$ C_{200}
⋯ 0 $\xleftarrow[\text{Dec}]{}$ $\sum_{i=1}^{200} C_i$
↑ #(T)

(We can perform the aggregate operations simultaneously for all the genotypes.)

- Decryption
  - Decrypt the ciphertext "$\sum_{i=1}^{200} C_i$" with the secret key.
  - Let ℓ_i be the value in the i'th slot.
- Decoding
  - For $1 \le i \le 311$, if $\ell_i > 200$, then $\ell_i \leftarrow (400 - \ell_i)$.
  - The minor allele frequency of the genotype G_i is $\left(\frac{\ell_i}{400}\right)$.

Minor Allele Frequency

- Encryption & Evaluation



(We can perform the aggregate operations simultaneously for all the genotypes.)

- Decryption
  - Decrypt the ciphertext "$\sum_{i=1}^{200} C_i$" with the secret key.
  - Let ℓ_i be the value in the i'th slot.
- Decoding
  - For $1 \leq i \leq 311$, if $\ell_i > 200$, then $\ell_i \leftarrow (400 - \ell_i)$.
  - The minor allele frequency of the genotype G_i is $\left(\frac{\ell_i}{400}\right)$.

Chi-squared Test

- Data Encoding
  - For each genotype, encode the given SNPs of case group and control group.

* Note that the result of chi-squared test is

$$\frac{n(ad-bc)^2}{r\cdot s\cdot g\cdot k} = \frac{800\left(a(400-c)-c(400-a)\right)^2}{400\cdot 400\cdot g\cdot k} = \frac{800\,(a-c)^2}{(a+c)(800-(a+c))}$$

where 'a' and 'c' are the allele counts of some SNP in case and control group.

Chi-squared Test

- Data Encoding
  - For each genotype, encode the given SNPs of case group and control group.

* Note that the result of chi-squared test is

$$\frac{n(ad-bc)^2}{r \cdot s \cdot g \cdot k} = \frac{800\,(a(400-c)-c(400-a))^2}{400 \cdot 400 \cdot g \cdot k}$$
$$= \frac{800\,(a-c)^{\,2}}{(a+c)(800-(a+c))}$$

where 'a' and 'c' are the allele counts of some SNP in case and control group.

Chi-squared Test

- Evaluation

  Let us denote C_i and C_i' the ciphertexts for the case&control groups.

  - Evaluate $\sum_{i=1}^{200} C_i$ ($\overset{let}{=} \mathfrak{C}_{case}$) and $\sum_{i=1}^{200} C_i'$ ($\overset{let}{=} \mathfrak{C}_{cont}$).
  - Compute "$\mathfrak{C}_{case} - \mathfrak{C}_{cont}$" and "$\mathfrak{C}_{case} + \mathfrak{C}_{cont}$"

- Decryption

  For the message space $\mathbb{Z}_t = [0, t)$,

  - den $\overset{let}{=} \mathrm{Dec}(\mathfrak{C}_{case} + \mathfrak{C}_{cont}) = a + c \ (< t)$
  - num $\overset{let}{=} \mathrm{Dec}(\mathfrak{C}_{case} - \mathfrak{C}_{cont}) = \begin{cases} a - c & \text{if } a > c, \\ (a-c) + t & \text{otherwise.} \end{cases}$

- Decoding

  - If num $> \frac{t}{2}$, then num $\leftarrow$ (num $- t$).
  - The result of chi-squared test is $\frac{800(\text{num})^2}{(\text{den})(800-\text{den})}$

Chi-squared Test

- Evaluation

  Let us denote C_i and C_i' the ciphertexts for the case&control groups.

  - Evaluate $\sum_{i=1}^{200} C_i$ ($\overset{let}{=} \mathfrak{C}_{case}$) and $\sum_{i=1}^{200} C_i'$ ($\overset{let}{=} \mathfrak{C}_{cont}$).
  - Compute "$\mathfrak{C}_{case} - \mathfrak{C}_{cont}$" and "$\mathfrak{C}_{case} + \mathfrak{C}_{cont}$"

- Decryption

  For the message space $\mathbb{Z}_t = [0, t)$,

  - den $\overset{let}{=} \mathrm{Dec}(\mathfrak{C}_{case} + \mathfrak{C}_{cont}) = a + c\ (< t)$
  - num $\overset{let}{=} \mathrm{Dec}(\mathfrak{C}_{case} - \mathfrak{C}_{cont}) = \begin{cases} a - c & \text{if } a > c, \\ (a - c) + t & \text{otherwise.} \end{cases}$

- Decoding

  - If num $> \frac{t}{2}$, then num $\leftarrow$ (num $- t$).
  - The result of chi-squared test is $\frac{800(\mathrm{num})^2}{(\mathrm{den})(800-\mathrm{den})}$

Secure Comparison between Genomic Data

Hamming Distance

- Two individuals have genotypes over many SNPs. For a fixed genotype,

$$d = \begin{cases} 1 & \text{if } (S_1 = \text{null}) \,||\, (S_2 = \text{null}) \,||\, (S_1.\text{alt} \neq S_2.\text{alt}) \\ 0 & \text{otherwise} \end{cases}$$

- $x[j] \overset{let}{=} j$-th bit of x, starting with the least significant bit of x. $\oplus$: XOR gate (= Add over $\mathbb{Z}_2$), $\wedge$: AND gate (= Mult over $\mathbb{Z}_2$).

SVTYPE	d
SV_1 or SV_2 = INS/DEL	0
SV_1 or SV_2 = null	1
SV_1 and SV_2 = SNP/SUB	$\texttt{EQU}(S_1, S_2) \oplus 1$

where $\texttt{EQU}(S_1, S_2) = \begin{cases} 1 & \text{if } S_1 = S_2 \\ 0 & \text{o.w,} \end{cases} = \wedge_{j=1}^{\mu} (S_1[j] \oplus S_2[j] \oplus 1)$

- We need the encodings to determine ‘null’ and ‘INS/DEL’.

Hamming Distance

- Two individuals have genotypes over many SNPs. For a fixed genotype,

$$d = \begin{cases} 1 & \text{if } (S_1 = \text{null}) \mid\mid (S_2 = \text{null}) \mid\mid (S_1.\text{alt} \neq S_2.\text{alt}) \\ 0 & \text{otherwise} \end{cases}$$

- $x[j] \overset{let}{=} j$-th bit of x, starting with the least significant bit of x. $\oplus$: XOR gate ($=$ Add over $\mathbb{Z}_2$), $\wedge$: AND gate ($=$ Mult over $\mathbb{Z}_2$).

SVTYPE	d
SV_1 or SV_2 $=$ INS/DEL	0
SV_1 or SV_2 $=$ null	1
SV_1 and SV_2 $=$ SNP/SUB	$\texttt{EQU}(S_1, S_2) \oplus 1$

$$\text{where } \texttt{EQU}(S_1, S_2) = \begin{cases} 1 & \text{if } S_1 = S_2 \\ 0 & \text{o.w,} \end{cases} = \wedge_{j=1}^{\mu} (S_1[j] \oplus S_2[j] \oplus 1)$$

- We need the encodings to determine 'null' and 'INS/DEL'.

Hamming Distance

- Data Encoding
  - Clean two datasets using POS, then make the merged list L.
  - For $i \in [1, \#(L)]$,

    define $m_i = \begin{cases} 1 & \text{if POS}_i \in L \\ 0 & \text{otherwise} \end{cases}$ and $h_i = \begin{cases} 0 & \text{if SV}_i = \text{INS/DEL} \\ 1 & \text{otherwise} \end{cases}$

    $$\Rightarrow (m_1 \oplus m_2) = 1 \text{ iff } (\text{SV}_{1,i} = \text{null}) \text{ or } (\text{SV}_{2,i} = \text{null})$$
    $$(h_1 \wedge h_2) = 0 \quad \text{iff } (\text{SV}_{1,i} = \text{INS/DEL}) \text{ or } (\text{SV}_{2,i} = \text{INS/DEL})$$
  - Encode the SNP string as follows:

    $$A \rightarrow 00, G \rightarrow 01, C \rightarrow 10, T \rightarrow 11,$$

    - ★ Each SNP is encoded and concatenated each other.
    - ★ Pad '1' at the end of the string, and '0' to make 21 bit string, say S_i.
    - ★ In the case of missing genotype, it is encoded as '0' string.
    - ★ For example, 'GTA' is encoded as '$01||11||00||1\underbrace{0\ldots00}_{14}$'.

Hamming Distance

- Data Encoding
  - Clean two datasets using POS, then make the merged list L.
  - For $i \in [1, \#(L)]$,

    define $m_i = \begin{cases} 1 & \text{if } \mathrm{POS}_i \in L \\ 0 & \text{otherwise} \end{cases}$ and $h_i = \begin{cases} 0 & \text{if } \mathrm{SV}_i = \mathrm{INS/DEL} \\ 1 & \text{otherwise} \end{cases}$

    $\Rightarrow (m_1 \oplus m_2) = 1$ iff $(\mathrm{SV}_{1,i} = \text{null})$ or $(\mathrm{SV}_{2,i} = \text{null})$

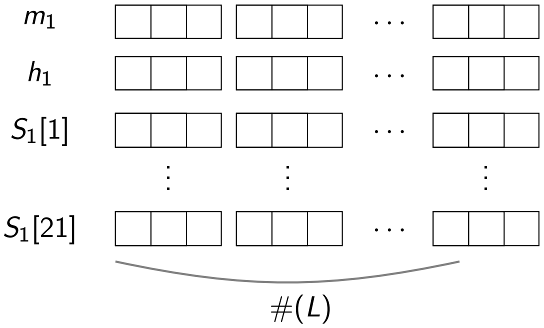
    $(h_1 \wedge h_2) = 0$ iff $(\mathrm{SV}_{1,i} = \mathrm{INS/DEL})$ or $(\mathrm{SV}_{2,i} = \mathrm{INS/DEL})$
  - Encode the SNP string as follows:

$$A \to 00, G \to 01, C \to 10, T \to 11,$$

    - ★ Each SNP is encoded and concatenated each other.
    - ★ Pad ‘1’ at the end of the string, and ‘0’ to make 21 bit string, say S_i.
    - ★ In the case of missing genotype, it is encoded as ‘0’ string.
    - ★ For example, ‘GTA’ is encoded as ‘$01||11||00||1\underbrace{0\ldots00}_{14}$’.

Hamming Distance

- Encryption
  - Embed the data of $P_1 (= m_{1,i}, h_{1,i}, S_{1,i})$ and P_2 $(= m_{2,i}, h_{2,i}, S_{2,i})$ into the plaintext slots in a bit-by-bit manner.
  - Encrypt the slots with the public key.



Hamming Distance

- Evaluation
  - Evaluate the following binary circuit over encrypted data:

    $$(h_{1,i} \wedge h_{2,i}) \wedge \Big((m_{1,i} \oplus m_{2,i}) \oplus \big(m_{1,i} \oplus m_{2,i} \oplus 1 \big) \wedge \big(\mathtt{EQU}(S_{1,i}, S_{2,i}) \oplus 1 \big) \Big)$$

  - Take $m = 8191$ so that we can embed 630 messages into one ciphertext and perform the operations simultaneously for all the messages.
- Decryption
  - Decrypt the evaluated value and let ℓ_i the value in the i'th slot.
- Decoding
  - Note that ℓ_i is the Hamming distance result of i'th genotype.
  - Compute $\sum_{i=1}^{\#(L)} \ell_i$.

Edit Distance

- For each genotype, we let
$$n = \begin{cases} \text{len}(S.\text{alt}) & \text{if SV} = \text{SNP/SUB/INS} \\ \text{len}(S.\text{ref}) & \text{if SV} = \text{DEL} \end{cases}$$
- $$d = \begin{cases} 0 & \text{if } (S_1.\text{ref} = S_2.\text{ref}) \,\&\, (S_1.\text{alt} = S_2.\text{alt}) \\ \texttt{max}(n_1, n_2) & \text{otherwise} \end{cases}$$

SVTYPE	d
$(\text{SV}_1 = \text{INS}, \text{SV}_2 \neq \text{INS}) \| \| (\text{SV}_1 \neq \text{INS}, \text{SV}_2 = \text{INS})$	$\texttt{max}(n_1, n_2)$
Otherwise	$\texttt{max}(n_1, n_2) \wedge (\texttt{EQU}(S_1.\text{alt}, S_2.\text{alt}) \oplus 1)$

- We don't need the reference comparison anymore.
- We need an encoding which determine whether the genotype is INS or not.

Edit Distance

- For each genotype, we let
$$n = \begin{cases} \mathrm{len}(S.\mathrm{alt}) & \text{if SV} = \text{SNP/SUB/INS} \\ \mathrm{len}(S.\mathrm{ref}) & \text{if SV} = \text{DEL} \end{cases}$$
- $d = \begin{cases} 0 & \text{if } (S_1.\mathrm{ref} = S_2.\mathrm{ref})\ \&\ (S_1.\mathrm{alt} = S_2.\mathrm{alt}) \\ \mathtt{max}(n_1, n_2) & \text{otherwise} \end{cases}$

SVTYPE	d
$(\mathrm{SV}_1 = \mathrm{INS}, \mathrm{SV}_2 \neq \mathrm{INS}) \| \| (\mathrm{SV}_1 \neq \mathrm{INS}, \mathrm{SV}_2 = \mathrm{INS})$	$\mathtt{max}(n_1, n_2)$
Otherwise	$\mathtt{max}(n_1, n_2) \wedge (\mathtt{EQU}(S_1.\mathrm{alt}, S_2.\mathrm{alt}) \oplus 1)$

  - We don't need the reference comparison anymore.
  - We need an encoding which determine whether the genotype is INS or not.

Edit Distance

- Data Encoding
  - Clean two datasets using POS, then make the merged list L.
  - For $i \in [1, \#(L)]$, define $e_i = \begin{cases} 1 & \text{if } \mathrm{SV}_i = \mathrm{INS}, \\ 0 & \text{o.w.} \end{cases}$

    $\Rightarrow (e_1 \oplus e_2 \oplus 1) = 1$ iff $((\mathrm{SV}_{1,i} = \mathrm{I}, \mathrm{SV}_{2,i} \neq \mathrm{I})$ or $(\mathrm{SV}_{1,i} \neq \mathrm{I}, \mathrm{SV}_{2,i} = \mathrm{I}))$
  - Encode the SNP string as S_i. (The missing genotype is encoded as '0')
  - Encode the length of SNP string, say n_i.
- Encryption
  - Embed the data of $P_1 (= e_{1,i}, S_{1,i}, n_{1,i})$ and P_2 $(= e_{2,i}, S_{2,i}, n_{2,i})$ into the plaintext slots in a bit-by-bit manner.
  - Encrypt the slots with the public key.

Edit Distance

- Evaluation

  For μ-bit integer x and y,

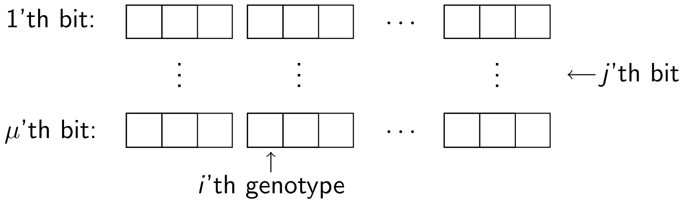
  - $\mathtt{C}(x,y) = \begin{cases} 1 & \text{if } x < y \\ 0 & \text{o.w.,} \end{cases} = c_\mu$
    - ★ $c_1 = (1 \oplus x[1]) \wedge y[1]$,
    - ★ $c_j = \big((1 \oplus x[j]) \wedge y[j]\big) \oplus \big((1 \oplus x[j] \oplus y[j]) \wedge c_{j-1}\big)$ for $2 \le j \le \mu$
  - $\mathtt{max}(x,y)[j] = \begin{cases} y[j] & \text{if } x < y, \\ x[j] & \text{o.w.,} \end{cases}$

    $$= \big(\underline{(1 \oplus \mathtt{C}(x,y)) \wedge x[j]}\big) \oplus \big(\underline{\mathtt{C}(x,y) \wedge y[j]}\big)$$
  - For $i \in [1, \#(L)]$ and $j \in [1, \mu]$, evaluate the circuits homomorphically:

    $$\Big(\big(\mathtt{EQU}(S_{1,i}, S_{2,i}) \wedge (e_{1,i} \oplus e_{2,i} \oplus 1)\big) \oplus 1\Big) \ \wedge \ \mathtt{max}(n_{1,i}, n_{2,i})[j]$$

Edit Distance

- Decryption
  - Decrypt the evaluated values, and let $\ell_{i,j}$ the i'th value in the j'th slot.



- Decoding
  - $\ell_i \overset{let}{=} \sum_{j=1}^{\mu} \ell_{i,j} \cdot 2^{j-1}$ is the Edit distance result of i'th genotype.
  - Compute $\sum_{i=1}^{\#(L)} \ell_i$.

Thank you and love you