

Outline

- **Background**
- Additively homomorphic encryption
- Beacon search by Oblivious transfer
- Genome sequence search
 - Overview of the proposed method
 - Recursive oblivious transfer
 - Burrows Wheeler Transform
 - Results
- Conclusion

DNA sequence

- DNA is a molecule that carries genetic information.
- It consists of four nucleotides (A denine, G uanine, C ytosine, T hymine), thus it is represented as a sequence of four letters.
- Analyzing DNA sequences is one of the most important approaches in current biology.



```
GGCATGAAAGTCAGCCGAGAGCCATCTATTGC
TTACATTTGCTTCTGACAAACTGTGTTCACTA
GCAACCTCAAAGGACACCATGGTGACACCTGA
CTCCTGAGGAGAGTGTGCCGTTACTGCCCTG
TGGGGCAAGGTCCGGTGGATGAAGTTGGTG
GTGAGGCCCTGGGAGCTTGGTATCAAGGTTA
CAAGACAGGTTAAGGAGACCAATAGAACTG
GGCATGTGGAGATGAGAGACTCTTGGGTTT
CTGATAGGCACTGATCTCTGCCTATTGGT
```

Next Generation Sequencer

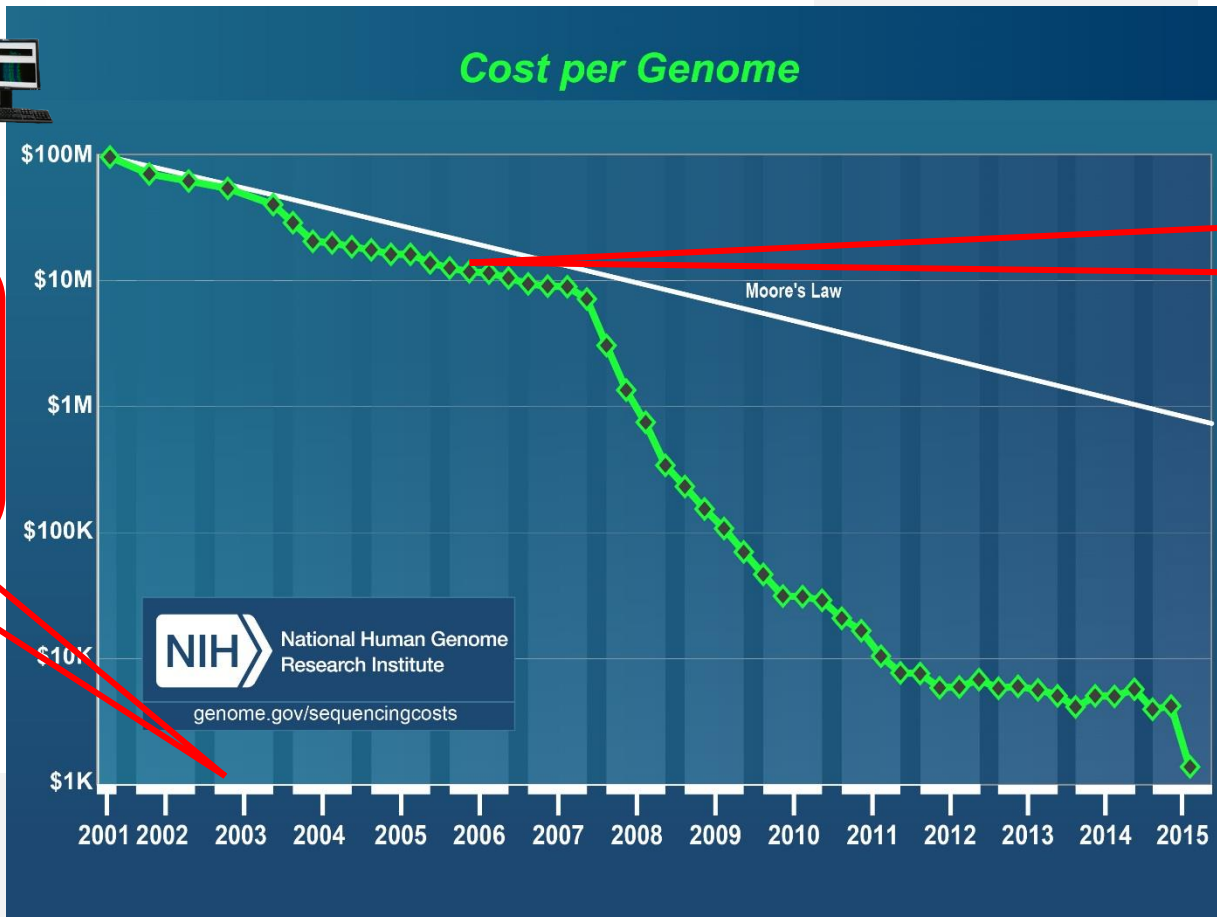
- Recently, the technology for determining DNA sequence has been dramatically improved.
- The instrument that determines DNA sequence based on the new technology is called NGS.



Genome "Big Data"



Sanger sequencer



Human Genome Project (1990~2003)

\$3B

NGS is introduced to market.

\$1K



High-throughput sequencer

Length of HG: $3 \cdot 10^9$

<http://www.genome.gov/sequencingcosts/>

Growth of personal genome data

- The huge cost down has encouraged sequencing of individual's genome.
 - Large scale cohort studies such as..
 - ToMMo will recruit 150K participants from 2013 to 2017, in Japan
 - Genomics England aims to sequence 100K individuals' genome, in UK.
 - Direct-to-consumer genetic testing
 - 23andMe has sequenced more than 1M customers' DNA.
 - openSNP: Web site of collecting DTC results
 - ≐ 2700 genotypes (June, 2016)
- It also poses **privacy risks**.

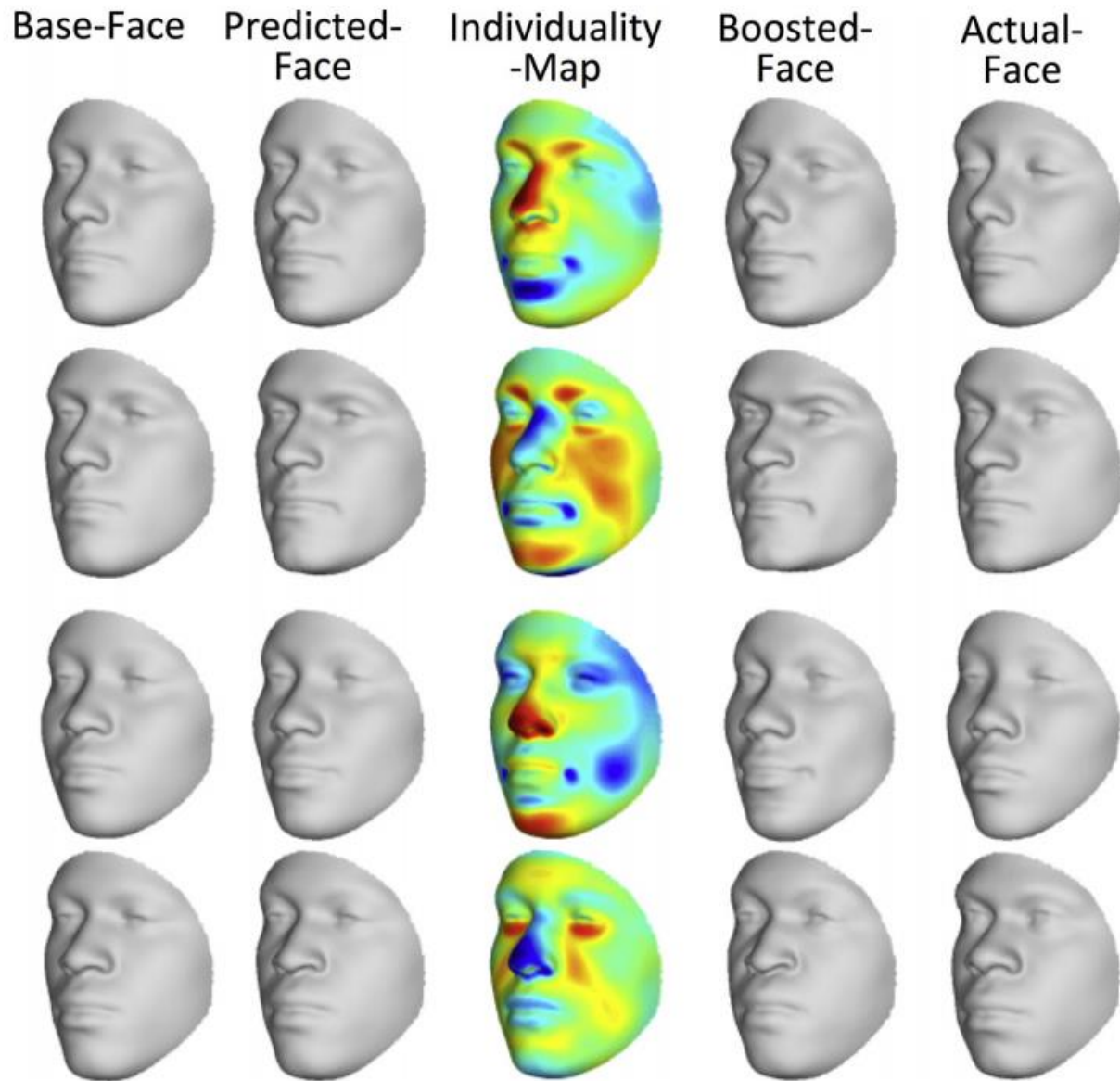
Variation of Genome

- The feature of an individual's genome is described by a difference between the genome and the reference genome.

Ref: GGCATGAAAGTCAAGGGCAGAGCCATCTATTGC
Individual: GGCATGAAAGTCTGGGCAGAGCCAT-TATTGC

- Sequence variants are considered to associate with phenotype (observable traits of the individual.)
- Num. of. Known SNP is around 3M
 - SNP: single mutation observed more than 1% of a population.
- One of the important topics of bioinformatics is to find association between phenotypes and genotypes.
- Some of such associations are already known.
 - BRCA: breast cancer, ADH4: alcohol metabolizing, etc..





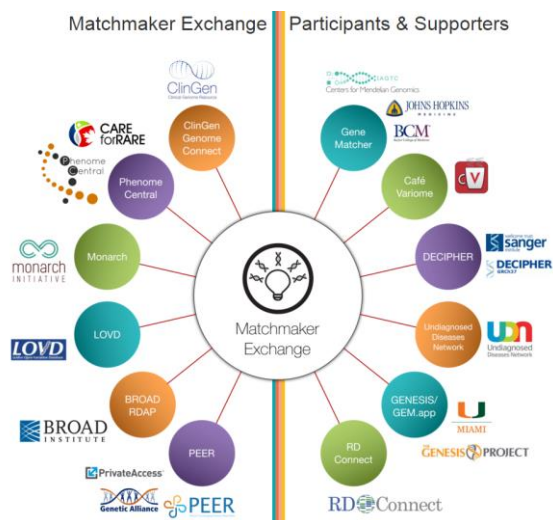
(P. Claes et al. Forensic Science International: Genetics, 2014)

The privacy problems of personal genome

- Genome can be a personal identifier, while it is strongly associated with phenotype.
- Lin et al., 2004
 - \approx 80 SNPs can identify an individual.
- Gymrek et al., Science, 2013
 - Surname can be recovered from personal genomes by profiling Y-STRs and querying genetic genealogy databases.
- Homer et al., 2008
 - Statistics of GWAS study leak whether or not a participant belongs to case/control.
- Legislation is not well prepared
 - US: Genetic information nondiscrimination act (GINA)
 - Does not apply to life insurance and the military.
 - The grand daughter of the cancer patient was rejected for the position in US army after taking genetic test (Lindor, 2012)
 - Japan: None
 - Meiji Yasuda Life Insurance Co. is deliberating using people's genetic information to provide targeted services.

The privacy problems of personal genome

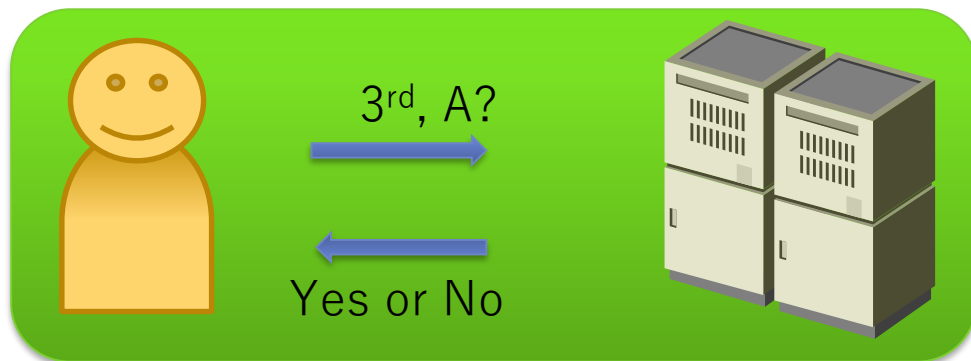
- The privacy problem hinders access to many data resources potentially useful for a variety of scientific researches.
- Global Alliance for Genomics & Health
 - Consortium aims for sharing genetic information for research purposes.
 - Established in 2013. 375 institutions has been participated so far.



Posted: May 29, 2015

Beacon Project

Being implemented on the website of the world's top genomic organizations to test the willingness of international sites to share genetic data.



Privacy Preserving Data Mining

- The term PPDM is firstly introduced by the papers (Agrawal & Srikant, 2000) and (Lindell & Pinkas, 2000)
- The goal: To release aggregate information about the data without releasing individual information.
- Example:
 - Aggregate info: Average salary of employees in the University
 - Individual info: A specific employee's salary

Two main approaches

- Perturbation approach
 - The data or the result of the database search is perturbed so that a database user is not able to obtain “true” database contents.
- Cryptographic approach
 - The data holder is called “party”. Each party uses encryption to protect his/her own data. The data are processed without decryption, and only the result of the data mining is obtained by specific parties.
- Those two approaches could be complementary.

Cryptographic approach

- Homomorphic Encryption

- Enabling add/mul operations on encrypted data.
 - Additive Homomorphic Encryption (Elgamal, 1984), (Paillier, 1999)
 - Fully Homomorphic Encryption (Gentry, 2009)

- Garbled Circuit (Yao, 1986)

- Enabling computation of any function while the input variables are encrypted.

- Secret Sharing

- A data point is divided into k shares. The data point is recovered only when θ shares are collected. Some operations can be computed on shares.

Outline

- Background
- **Additively homomorphic encryption**
- Beacon search by Oblivious transfer
- Genome sequence search
 - Overview of the proposed method
 - Recursive oblivious transfer
 - Burrows Wheeler Transform
 - Results
- Conclusion

Homomorphic Encryption

- Homomorphism: A structure-preserving map between two algebraic structures.

$$f : (G, *) \rightarrow (H, \bullet) \quad \text{s.t.} \quad f(g_1 * g_2) = f(g_1) \bullet f(g_2)$$

$$\log : (R_+, \times) \rightarrow (R, +)$$

$$\log(g_1 \times g_2) = f(g_1) + f(g_2)$$

- Additive homomorphic encryption
 - Additive op. on the plain text is equivalent to another op. on the cipher text.

$$Enc(m1 + m2) = Enc(m1) \oplus Enc(m2)$$

- Lifted ElGamal [Elgamal84], Paillier [Paillier99]

Additively Homomorphic cryptosystem

➤ Paillier [Paillier99]

Secret key : $sk = (p, q)$

Public key : $pk = (n, g), n = p \cdot q$

Cipher text of m : $Enc_{pk}(m) := g^m \cdot r^n \pmod{n^2}$

$r \in \mathbb{Z}_{n^2}^*$ is a random value $g = 1 + kn \pmod{n^2}$

$$Enc_{pk}(m1) \cdot Enc_{pk}(m2) = g^{m1+m2} \cdot (r1 \cdot r2)^n \pmod{n^2}$$

$$Dec_{sk}(Enc_{pk}(m1) \cdot Enc_{pk}(m2)) = m1 + m2$$

Secure additive operation based on additive homomorphic encryption

Computing $m1 + m2$ on the server,
without leaking $m1$ to the server.

Secret key :

For Decryption only



Public key:

For Encryption only



$m1$

$Enc(m1)$ user's data

$m2$
server's data



Secure additive operation based on additive homomorphic encryption

Secret key :
For Decryption only



Public key:
For Encryption only



m_1

$Enc(m_1)$ user's data

m_2
server's data



Secure additive operation based on additive homomorphic encryption

Secret key :
For Decryption only



Public key:
For Encryption only

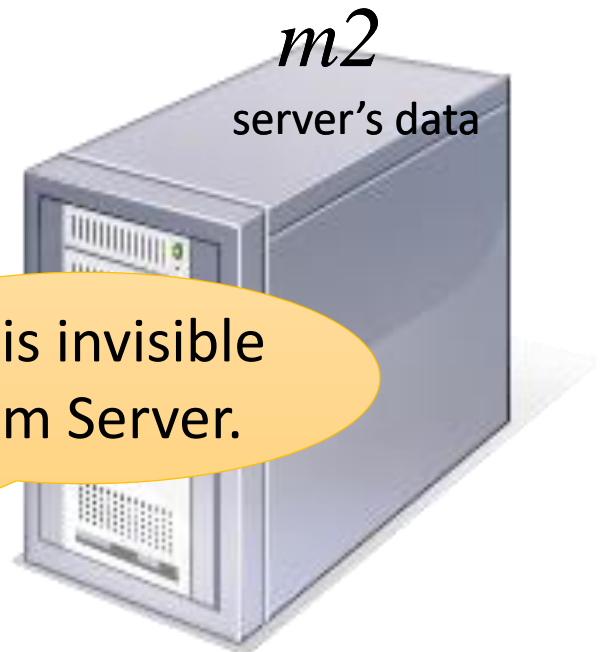


m_1
user's data



m_1 is invisible
from Server.

$Enc(m_1)$



m_2
server's data

Secure additive operation based on additive homomorphic encryption

Secret key :
For Decryption only



Public key:
For Encryption only



$m1$

user's data

$m2 \xrightarrow{\text{key}} Enc(m2)$

$Enc(m1)$

$Enc(m1) \oplus Enc(m2) \rightarrow Enc(m1 + m2)$

$m2$
server's data



Secure additive operation based on additive homomorphic encryption

Secret key :
For Decryption only



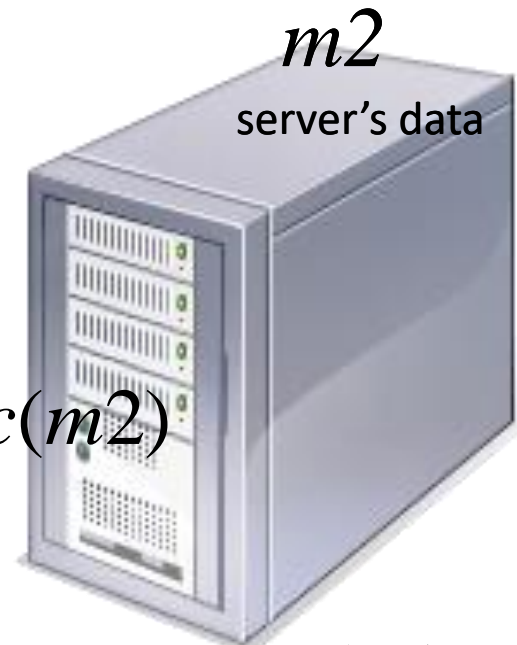
Public key:
For Encryption only



$m1$
user's data

$m2 \xrightarrow{\text{key}} Enc(m2)$

$Enc(m1)$



$m2$
server's data

$Enc(m1 + m2)$

Secure additive operation based on additive homomorphic encryption

Secret key :
For Decryption only



Public key:
For Encryption only



m_1

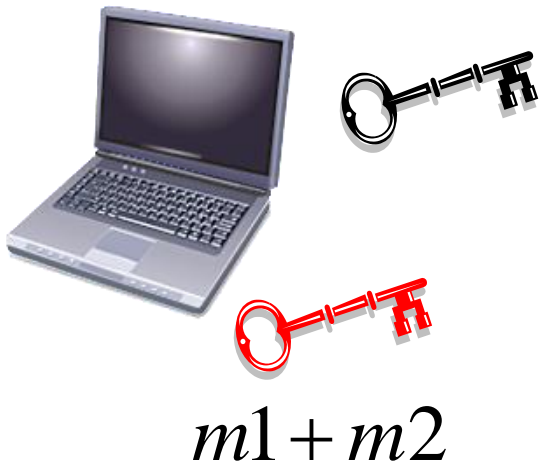
user's data

$Enc(m_1 + m_2)$

m_2
server's data

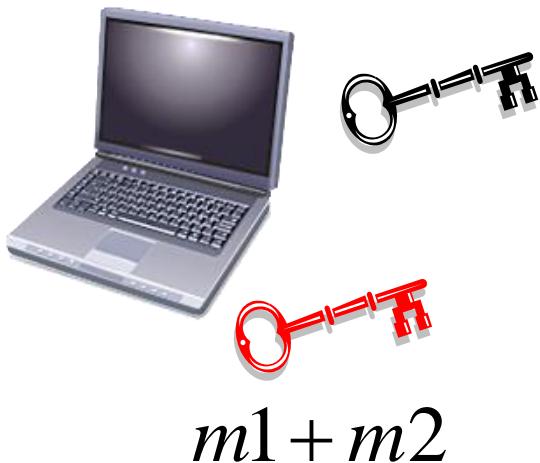


Secure additive operation based on additive homomorphic encryption



Secure additive operation based on additive homomorphic encryption

Additive operation is performed on the server without leaking client's value to the server.



Outline

- Background
- Additively homomorphic encryption
- **Beacon search by Oblivious transfer**
- Genome sequence search
 - Overview of the proposed method
 - Recursive oblivious transfer
 - Burrows Wheeler Transform
 - Results
- Conclusion

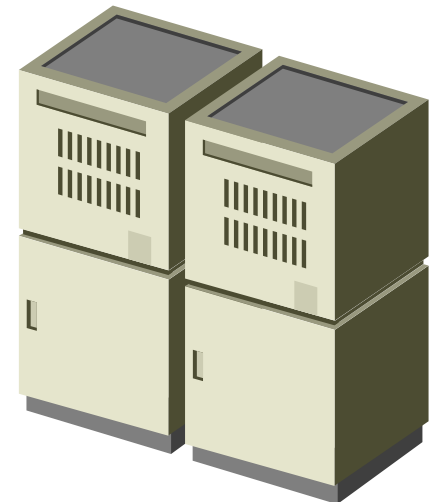
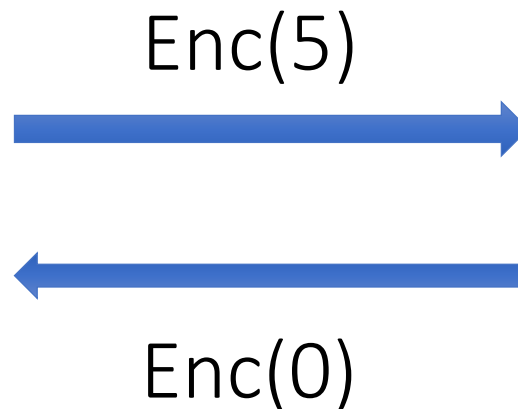
Can we make secure beacon search?

Public

Private

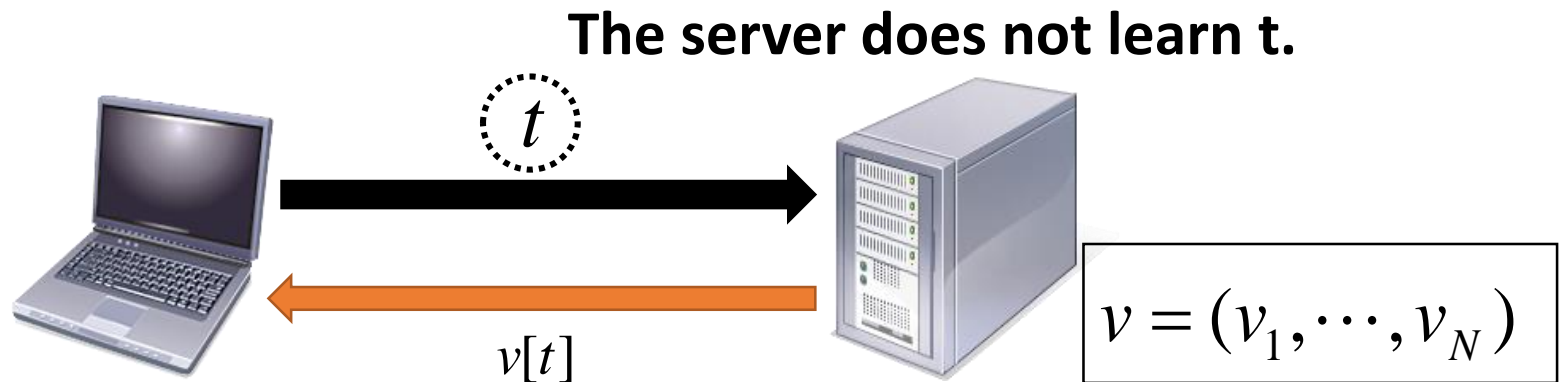
Beacon	Index	Yes: 1 No: 0
1, 'A'	1	1
1, 'T'	2	0
1, 'G'	3	0
1, 'C'	4	1
2, 'A'	5	0
...
3000000000, 'A'	11999999997	1

Query: (2, 'A')



What is necessary?

- The user needs to obtain t -th element of the server's look-up table (vector) v without leaking t to the server.
- The problem is conventionally called ***Oblivious Transfer***.



How do we implement OT?

(1 out of N) Oblivious Transfer by AHE

t-th cyphertext

[Step 1] Key setup

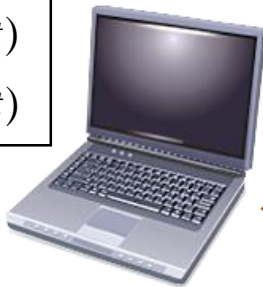
Secret key 
(for decryption)

Public key 
(for encryption)

$(\text{Enc}(0), \dots, \text{Enc}(1), \dots, \text{Enc}(0))$

[Step 2] Query entry

$$q_i = \begin{cases} 1 & (i = t) \\ 0 & (i \neq t) \end{cases}$$



$(\text{Enc}(q_1), \dots, \text{Enc}(q_N))$



$v = (v_1, \dots, v_N)$



(1 out of N) Oblivious Transfer by AHE

Repeat addition of $\text{Enc}(q[i])$ $v[i]$ times

For $i = 1, \dots, N$:

$$\text{Enc}(q[i]) \oplus \dots \oplus \text{Enc}(q[i])$$

[Step 1] Key setup

Secret key 
(for decryption)

Public key 
(for encryption)

$$q_i = \begin{cases} 1 & (i = t) \\ 0 & (i \neq t) \end{cases}$$



[Step 2] Query entry

$(\text{Enc}(q_1), \dots, \text{Enc}(q_N))$

[Step 3] Computation of an encrypted result

$$\begin{aligned} \text{Enc}(v[t]) = & (v[1] \otimes \text{Enc}(q_1)) \oplus \dots \oplus (v[N] \otimes \text{Enc}(q_N)) \\ \because i = t \Rightarrow & v[i] \otimes \text{Enc}(q_i) = \text{Enc}(v[i]) \wedge \\ i \neq t \Rightarrow & v[i] \otimes \text{Enc}(q_i) = \text{Enc}(0) \end{aligned}$$



$$v = (v_1, \dots, v_N)$$

t-th cyphertext

$$\underbrace{(v[1] \otimes \text{Enc}(0))}_{\text{Enc}(0)} \oplus \dots \oplus \underbrace{(v[t] \otimes \text{Enc}(1))}_{\text{Enc}(v[t])} \oplus \dots \oplus \underbrace{(v[N] \otimes \text{Enc}(0))}_{\text{Enc}(0)}$$

Enc(0)

Enc(v[t])

Enc(0)

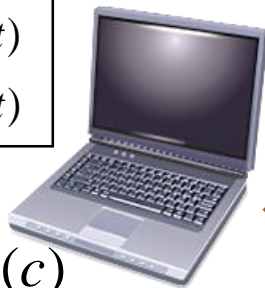
(1 out of N) Oblivious Transfer by AHE

[Step 1] Key setup

Secret key 
(for decryption)

Public key 
(for encryption)

$$q_i = \begin{cases} 1 & (i = t) \\ 0 & (i \neq t) \end{cases}$$



$$v[t] = \text{Dec}(c)$$

[Step 4] Decryption of the encrypted result

[Step 3] Computation of an encrypted result

$$\begin{aligned} \text{Enc}(v[t]) &= \\ & (v[1] \otimes \text{Enc}(q_1)) \oplus \dots \oplus (v[N] \otimes \text{Enc}(q_N)) \\ & \because i = t \Rightarrow v[i] \otimes \text{Enc}(q_i) = \text{Enc}(v[i]) \wedge \\ & \quad i \neq t \Rightarrow v[i] \otimes \text{Enc}(q_i) = \text{Enc}(0) \end{aligned}$$



[Step 2] Query entry

$$(\text{Enc}(q_1), \dots, \text{Enc}(q_N))$$



$$v = (v_1, \dots, v_N)$$



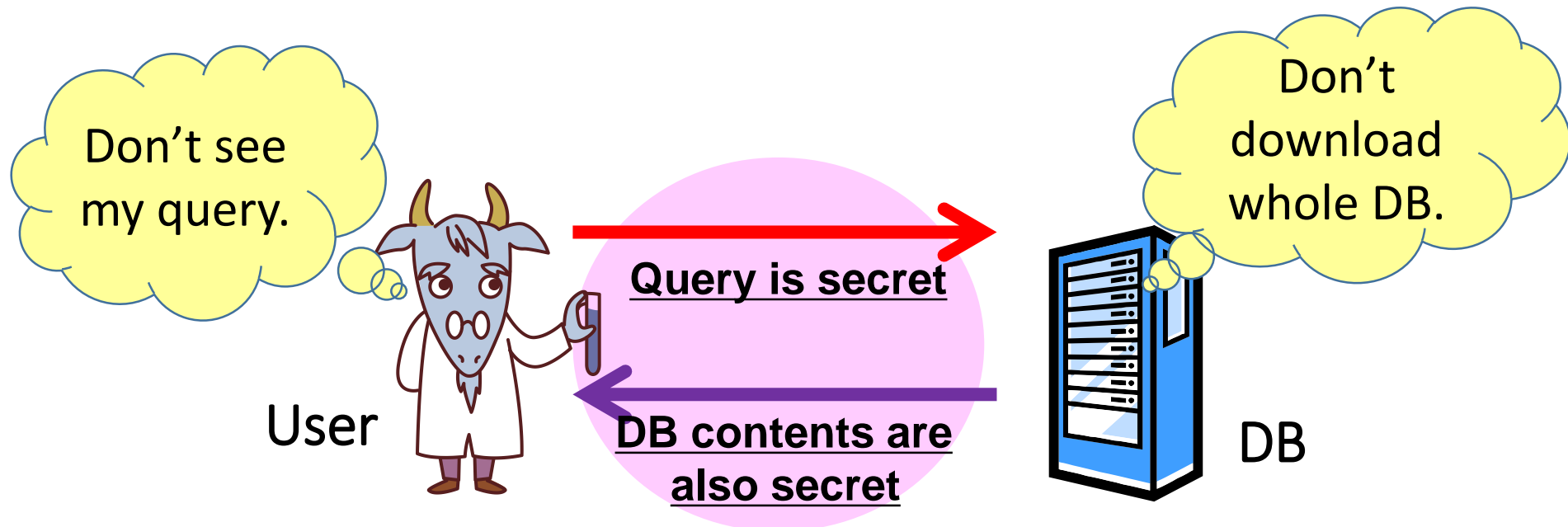
$$c = \text{Enc}(v[t])$$

Outline

- Background
- Additively homomorphic encryption
- Beacon search by Oblivious transfer
- **Genome sequence search**
 - Overview of the proposed method
 - Recursive oblivious transfer
 - Burrows Wheeler Transform
 - Results
- Conclusion

Problem Setup

- Our goal is to achieve following requirement:
 - A user would like to search a genomic sequence in a database to know whether or not his query matches to the DB.
 - For privacy reasons, the user wants to conceal the query, and the server wants to return only the result, and do not want to return any other information.



Related Works

- Computation of edit distance (Jha+2008, etc)
- Fixed-length keyword match (Blanton+2010)
- Finding similar sequence based on hamming distance (Baldi+2011, Cristofaro+2013)
- PIR of variable length keyword (Naganuma+2012)

Our goal:

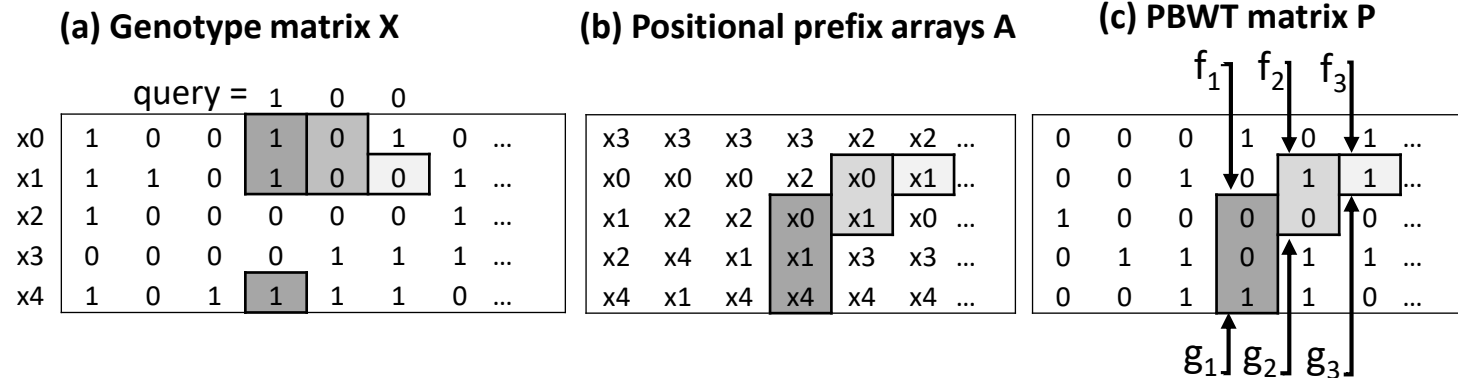
- Search variable length keyword match while keeping both sides' privacy.

Adversary Model

- Semi-honest
 - Both parties follow the protocol, but an adversarial one attempts to infer additional information about the other party's secret input from the legally obtained information.
- Malicious
 - An adversarial party cheats even in the protocol (e.g. by inputting maliciously chosen invalid values) in order to illegally obtain additional information about the secret.

Our Approach

- To combine
 - An efficient data structure such as (P)BWT
 - Cryptographic technique (Recursive Oblivious Transfer)
- (P)BWT stores string information very efficiently and still allows computations (Ferragina+2005, Durbin2014)
 - **k-prefix match b/w a query and DB is reported as an interval $[f_k, g_k]$ on the data structure.**
 - **An efficient algorithm is known to compute f_{k+1} from f_k and $q[k+1]$.**
 - Those values are **precomputable**.



Searching (P)BWT by Lookup tables

- The updates can be written in the form of referring a large, static look-up table v .

$$f_{K+1} = v_c[f_K]$$

$$g_{K+1} = v_c[g_K]$$

- Match is obtained by:

$$g_{K+1} - f_{K+1} + 1$$

OT is used to update f, g

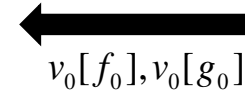
1st iteration:

$$q = (0, 1, 1, 0, \dots)$$

$$q[1] = 0$$

$$v_0 = (0, 1, 1, 2, \dots)$$

$$v_1 = (10, 10, 11, 11, \dots)$$


 f_0, g_0

 $v_0[f_0], v_0[g_0]$

2nd iteration:



$$q[2] = 1$$

$$f_1 = v_0[f_0]$$

$$g_1 = v_0[g_0]$$


 $v_1[f_1], v_1[g_1]$

K-th iteration:


 \vdots

$$q[k] = 1$$

$$f_{k-1}, g_{k-1}$$


 $v_1[f_{k-1}], v_1[g_{k-1}]$

Suffix Array (Manber91)

- Sorted array of all suffixes of a string.

S="ATGAATGCGA\$"

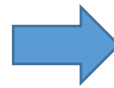
			SA	
S1	ATGAATGCGA\$	S11	\$	11
S2	TGAATGCGA\$	S10	A\$	10
S3	GAATGCGA\$	S4	AATGCGA\$	4
S4	AATGCGA\$	S1	ATGAATGCGA\$	1
S5	ATGCGA\$	S5	ATGCGA\$	5
S6	TGCGA\$	S8	CGA\$	8
S7	GCGA\$	S9	GA\$	9
S8	CGA\$	S3	GAATGCGA\$	3
S9	GA\$	S7	GCGA\$	7
S10	A\$	S2	TGAATGCGA\$	2
S11	\$	S6	TGCGA\$	6

Searching on SA

- Conduct binary search.

(Example) Search "ATG".

Greater than "ATG"?



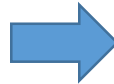
S11	\$
S10	A\$
S4	AATGCGA\$
S1	ATGAATGCGA\$
S5	ATGCGA\$
S8	CGA\$
S9	GA\$
S3	GAATGCGA\$
S7	GCGA\$
S2	TGAATGCGA\$
S6	TGCGA\$

Searching on SA

- Conduct binary search.

(Example) Search "ATG".

Greater than "ATG"?



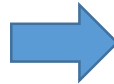
S11 \$
S10 A\$
S4 AATGCGA\$
S1 ATGAATGCGA\$
S5 ATGCGA\$
~~S8 CGA\$~~
~~S9 GA\$~~
~~S3 GAATGCCGA\$~~
~~S7 GCCGA\$~~
~~S2 TGAATGCCGA\$~~
~~S6 TGCCGA\$~~

Searching on SA

- Conduct binary search.

(Example) Search "ATG".

Greater than "ATG"?



~~S11~~ \$
~~S10~~ A\$
~~S4~~ AATGCGA\$
S1 ATGAATGCGA\$
S5 ATGCGA\$
~~S8~~ CGA\$
~~S9~~ GA\$
~~S3~~ GAATGCGA\$
~~S7~~ GCGA\$
~~S2~~ TGAATGCGA\$
~~S6~~ TGCGA\$

Searching on SA

- Conduct binary search.

(Example) Search "ATG".

Greater than "ATG"?



~~S11~~ \$
~~S10~~ A\$
~~S4~~ AATGCCGA\$
S1 ATGAATGCCGA\$
S5 ATGCCGA\$
~~S8~~ CGA\$
~~S9~~ GA\$
~~S3~~ GAATGCCGA\$
~~S7~~ GCCGA\$
~~S2~~ TGAATGCCGA\$
~~S6~~ TGCCGA\$

Searching on SA

- Conduct binary search.

(Example) Search "ATG".

Greater than "ATG"?



~~S11~~ \$
~~S10~~ A\$
~~S4~~ AATGCGA\$
S1 ATGAATGCGA\$
S5 ATGCGA\$
~~S8~~ CGA\$
~~S9~~ GA\$
~~S3~~ GAATGCGA\$
~~S7~~ CCGA\$

An efficient construction of SA (Nong+09):
 $O(N)$ time, $O(N(\log N + \log |\Sigma|))$ space

Searching on SA

- Conduct binary search.

(Example) Search "ATG".

		SA
	S11 \$	11
	S10 A\$	10
	S4 AATGCCGA\$	4
	S1 ATGAATGCCGA\$	1
Greater than "ATG"? →	S5 ATGCCGA\$	5
	S8 CCTA\$	8
	S9 GA\$	9
	S3 GAATGCCGA\$	3
	S7 GCCGA\$	7
	S2 TGAATGCCGA\$	2
	S6 TGCCGA\$	6

Finding SA interval [4, 5]

How do we find each SA interval with $O(1)$? → FM-index

Preparation

- i -th character of a string S is denoted by $S[i]$.
- Rank dictionary:

$$\text{Rank}_c(S, t) = |\{j \mid S[j] = c, \quad 1 \leq j \leq t\}|$$

(Example)

$i : 1 2 3 4 5 6 7 8 9$

$S : \text{ATGCTAGCT}$

$$\text{Rank}_A(S, 6) = 2$$

$$\text{Rank}_T(S, 3) = 1$$

$$\text{CF}_c(S) = \sum_{r < c} \text{Rank}_r(S, N)$$

$$\text{CF}_A(S) = 0$$

$$\text{CF}_T(S) = 6$$

Preparation

- i-th character of a string S is denoted by S[i].
- Rank dictionary:

$$\text{Rank}_c(S, t) = |\{j \mid S[j] = c, \quad 1 \leq j \leq t\}|$$

(Example)

i : 1 2 3 4 5 6 7 8

S : A T G C T A T A

$$\text{Rank}_c(S, 6) = 2$$

Wavelet tree(Grossi+03):

Time: $O(\log |\Sigma|)$

Space: $O(n \log |\Sigma|)$

$$\text{CF}_c(S) = \sum_{r < c} \text{Rank}_r(S, n)$$

$$\text{CF}_T(S) = 6$$

Burrows-Wheeler Transform (Burrows+94)

- $B[i] = S[SA[i]-1]$

$S = \text{"ATGAATGCGA\$"}$

A	\$
G	A\$
G	AATGCGA\$
\$	ATGAATGCGA\$
A	ATGCGA\$
G	CGA\$
C	GA\$
T	GAATGCGA\$
T	GCGA\$
A	TGAATGCGA\$
A	TGCGA\$

Burrows-Wheeler Transform (Burrows+94)

- B is *Reversible* transformation of S
 - No need to store additional data.
- Searchable (FM-index)
- Good fit to compression
 - Identical characters tends to be located near.

Let's start from an extreme case

$$B[i] = S[SA[i]-1]$$

`S="bfcgahejid$"`

`S[SA[1]=11]`

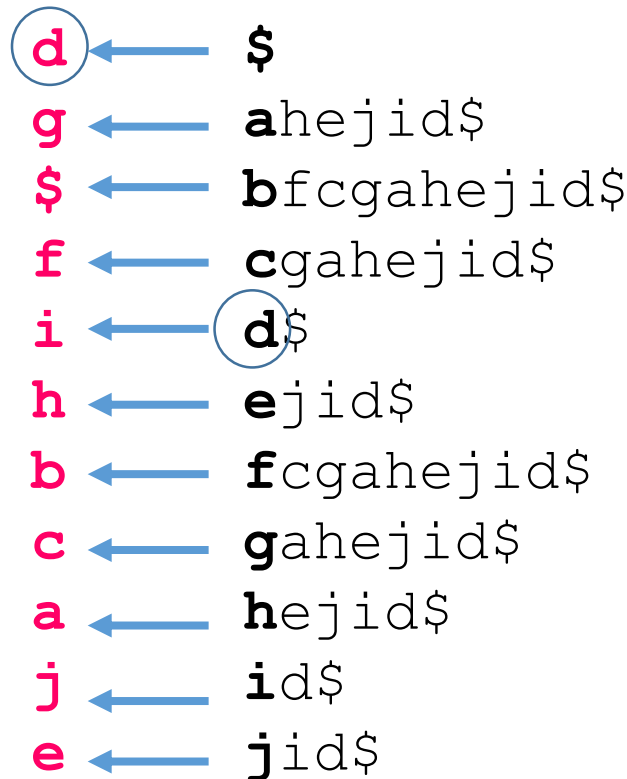
`B[1] = S[11-1]`

d	←	\$
g	←	a hejid\$
\$	←	b fcgahejid\$
f	←	c gahejid\$
i	←	d \$
h	←	e jid\$
b	←	f cgahejid\$
c	←	g ahejid\$
a	←	h ejid\$
j	←	i d\$
e	←	j id\$

Let's start from an extreme case

$$B[i] = S[SA[i]-1]$$

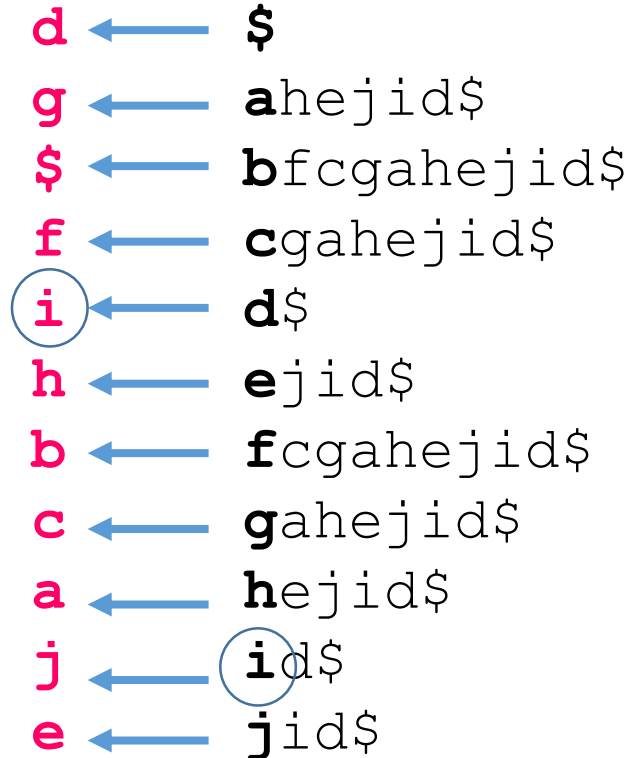
S="bfcgahej**id**\$"



Let's start from an extreme case

$$B[i] = S[SA[i]-1]$$

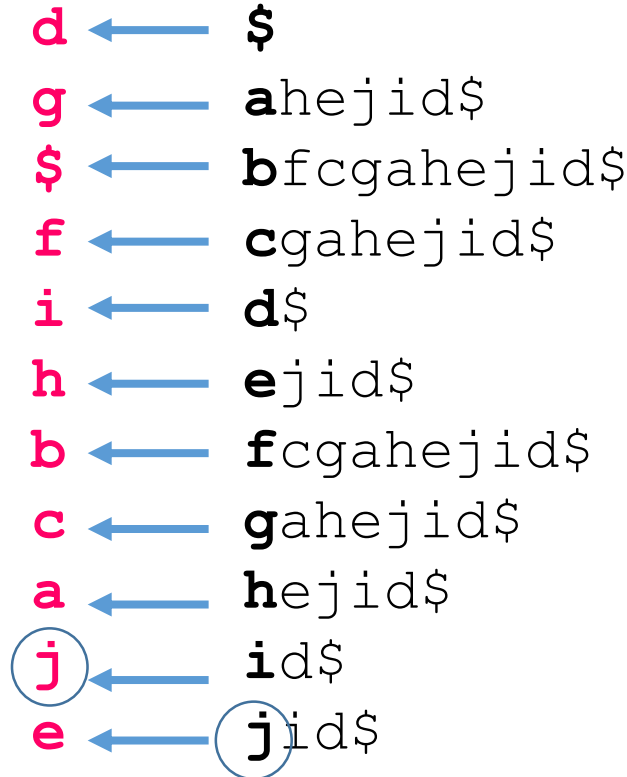
`S="bfcgahejid$"`



Let's start from an extreme case

$$B[i] = S[SA[i]-1]$$

`S="bfcgahejid$"`



What about identical characters?

S="ATGAATGCGA\$"

A	\$
G	A\$
G	AATGCGA\$
\$	ATGAATGCGA\$
A	ATGCGA\$
G	CGA\$
C	GA\$
T	GAATGCGA\$
T	GCGA\$
A	TGAATGCGA\$
A	TGCGA\$

What about identical characters?

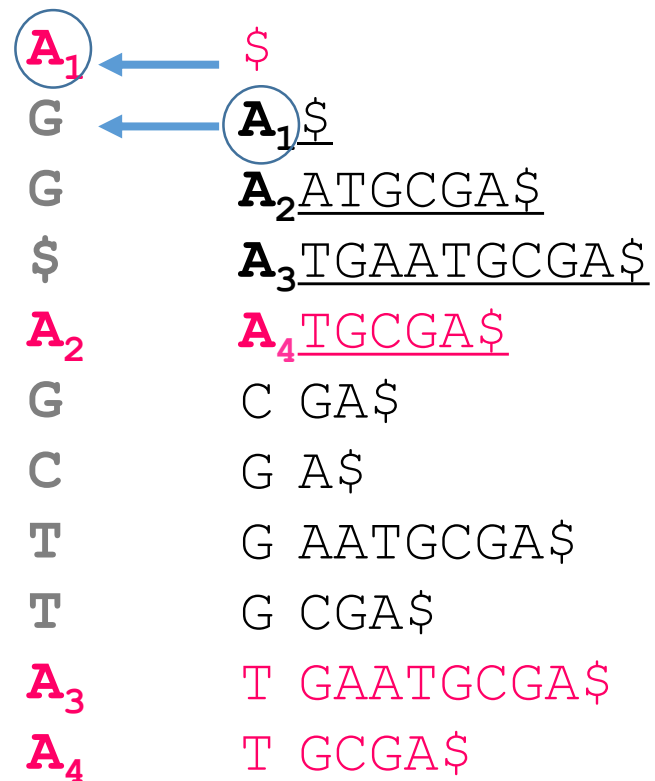
S="ATGAATGCGA\$"

A₁	\$
G	A₁ <u>\$</u>
G	A₂ <u>ATGCGA\$</u>
\$	A₃ <u>TGAATGCGA\$</u>
A₂	A₄ <u>TGCGA\$</u>
G	C GA\$
C	G A\$
T	G AATGCGA\$
T	G CGA\$
A₃	T GAATGCGA\$
A₄	T GCGA\$

The position of the same 'A' is determined by the same substring.

What about identical characters?

S="ATGAATGC**GA**\$"



The position of the same 'A' is determined by the same substring.

What about identical characters?

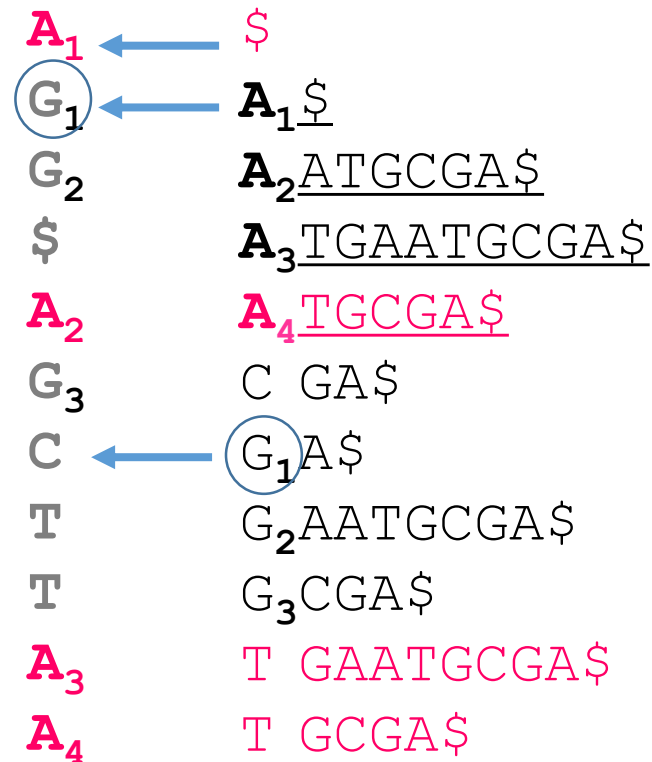
S="ATGAATG**CGA**\$"

LF-Mapping:

$$LF(i) := CF_{B[i]}(B) + Rank_{B[i]}(B, i)$$

```

P = 1
for i = 1 to N
  S[N-i] = B[p]
  p = LF(p)
end for
    
```



The position of the same 'A' is determined by the same substring.

FM-index (Ferragina+00)

- Searching on BWT
 - Using a rank dictionary on BWT of S
- Backward search
 - Searching from the last character and extend the match one by one, in a similar way to LF-mapping.

$$f' = CF_C(B) + \text{Rank}_C(B, f - 1) + 1$$

$$g' = CF_C(B) + \text{Rank}_C(B, g)$$

- Time complexity
 - $O(1)$ by naïve dictionary, $O(\log |\Sigma|)$ by Wavelet tree

FM-index (Ferragina+00)

$S = \text{"ATGAAATGCGA\$"}$

Finding substring ending with 'A'.

$i = |q|$

$f=1, g=N$

While $f \leq g$

$c = q[i--]$

$f = CF_c(B) + Rank_c(B, f-1) + 1$

$g = CF_c(B) + Rank_c(B, g)$

end for

$f=1, g=11 \rightarrow f'=2, g'=5$		SA
A ₁	\$	11
<hr/>		
G ₁	A ₁ \$	10
G ₂	A ₂ ATGCGA\$	4
\$	A ₃ TGAATGCGA\$	1
A ₂	A ₄ TGCGA\$	5
<hr/>		
G ₃	C ₁ GA\$	8
C	G ₁ A\$	9
T ₁	G ₂ AATGCGA\$	3
T ₂	G ₃ CGA\$	7
A ₃	T ₁ GAATGCGA\$	2
A ₄	T ₂ GCGA\$	6

FM-index (Ferragina+00)

$S = \text{"ATGAATGC GA\$"}$

Extending the match by 'G'
from 'A'.

$i = |q|$

$f=1, g=N$

While $f \leq g$

$c = q[i--]$

$f = CF_c(B) + Rank_c(B, f-1) + 1$

$g = CF_c(B) + Rank_c(B, g)$

end for

$f=2, g=5 \rightarrow f'=7, g'=8$		SA
A_1	\$	11
G_1	A_1 \$	10
G_2	A_2 ATGCGA\$	4
\$	A_3 TGAATGCGA\$	1
A_2	A_4 TGCGA\$	5
G_3	C_1 GA\$	8
C_1	G_1 A\$	9
T_1	G_2 AATGCGA\$	3
T_2	G_3 CGA\$	7
A_3	T_1 GAATGCGA\$	2
A_4	T_2 GCGA\$	6

FM-index (Ferragina+00)

$S = \text{"A**TGA**ATGCGA\$"}$

Extending the match by 'T'
from "GA".

```

i = |q|
f=1, g=N
While f<=g
  c = q[i--]
  f = CFc(B) + Rankc(B, f - 1) + 1
  g = CFc(B) + Rankc(B, g)
end for
    
```

$f=7, g=8 \rightarrow f'=10, g'=10$ SA

A ₁	\$	11
G ₁	A ₁ \$	10
G ₂	A ₂ ATGCGA\$	4
\$	A ₃ TGAATGCGA\$	1
A ₂	A ₄ TGCGA\$	5
G ₃	C ₁ GA\$	8
C ₁	G ₁ A\$	9
T ₁	G ₂ AATGCGA\$	3
T ₂	G ₃ CGA\$	7
A ₃	T ₁ GAATGCGA\$	2
A ₄	T ₂ GCGA\$	6

Searching (P)BWT by Lookup tables

- The updates can be written in the form of referring a large, static look-up table v .

$$f_{K+1} = v_c[f_K]$$

$$g_{K+1} = v_c[g_K]$$

- Match is obtained by:

$$g_{K+1} - f_{K+1} + 1$$

OT is used to update f, g

1st iteration:

$$q = (0, 1, 1, 0, \dots)$$

$$q[1] = 0$$

$$v_0 = (0, 1, 1, 2, \dots)$$

$$v_1 = (10, 10, 11, 11, \dots)$$


 f_0, g_0

 $v_0[f_0], v_0[g_0]$


2nd iteration:

$$q[2] = 1$$

$$f_1 = v_0[f_0]$$

$$g_1 = v_0[g_0]$$


 $v_1[f_1], v_1[g_1]$


K-th iteration:

 \vdots

$$q[k] = 1$$

$$f_{k-1}, g_{k-1}$$


 $v_1[f_{k-1}], v_1[g_{k-1}]$


Conceal intermediates

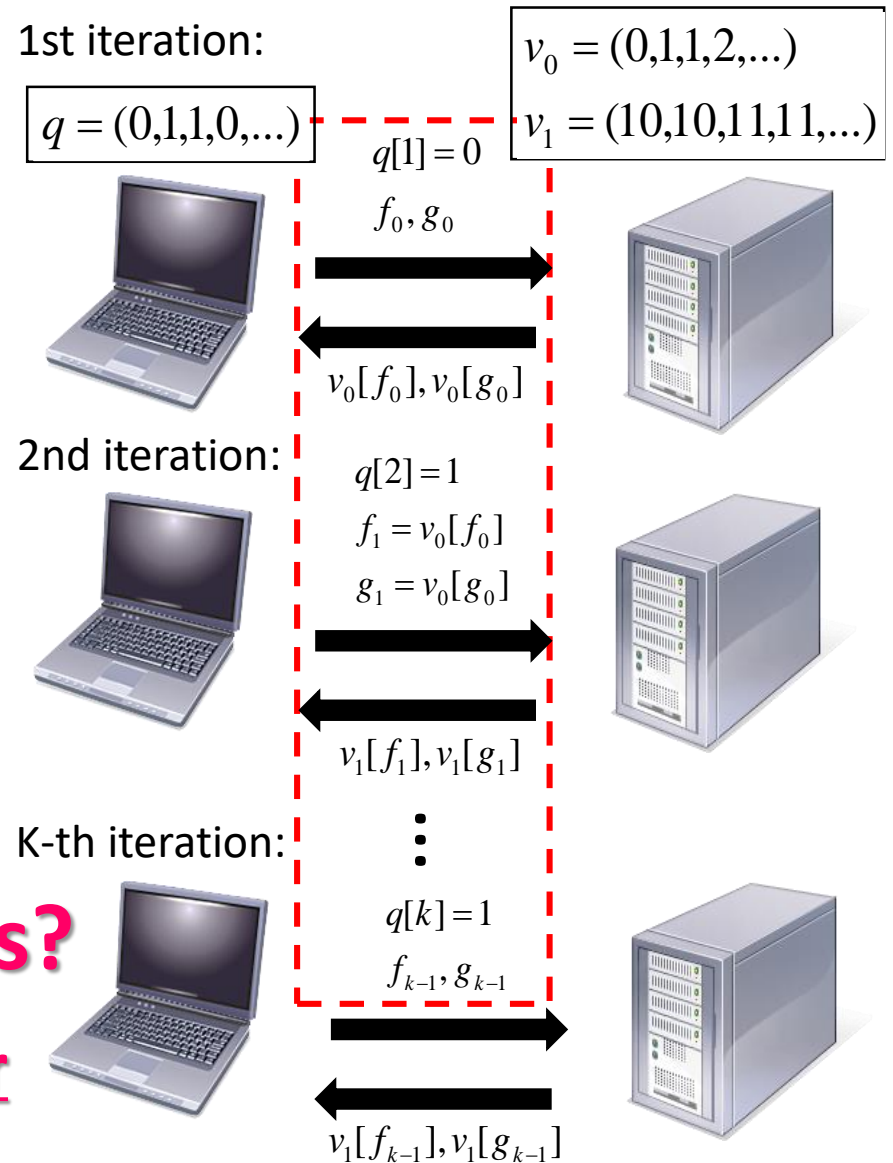
- It is ideal to conceal all the intermediates for protecting server's privacy more rigorously.

$$f_{k+1} = v_{q[k+1]}[v_{q[k]}[\dots v_{q[1]}[f_0]\dots]]$$

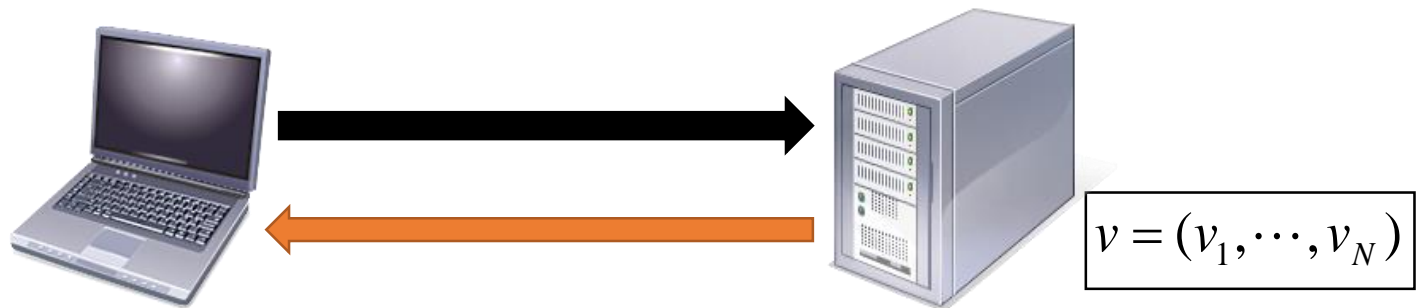
$$g_{k+1} = v_{q[k+1]}[v_{q[k]}[\dots v_{q[1]}[g_0]\dots]]$$

How do we achieve this?

Recursive Oblivious Transfer



(1 out of N) Recursive OT by AHE



$$c = \text{Enc}((v[t] + \underline{r})_{\text{mod } N})$$

Add a random value

(1 out of N) Recursive OT by AHE

(v[t] + r)-th cyphertext

[Step 1] Key setup

Secret key 
(for decryption)

Public key 
(for encryption)

$(\text{Enc}(0), \dots, \text{Enc}(1), \dots, \text{Enc}(0))$

[Step 2] Query entry

$(\text{Enc}(q_1), \dots, \text{Enc}(q_N))$

$$q_i = \begin{cases} 1 & (i = v[t] + r) \\ 0 & (i \neq v[t] + r) \end{cases}$$



$$v = (v_1, \dots, v_N)$$



(1 out of N) Recursive OT by AHE


Server makes an r-rotated permutation of the query to recover the correct query.

[Step 1] Key setup

Secret key 
(for decryption)

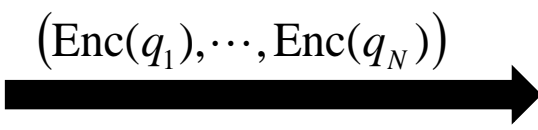
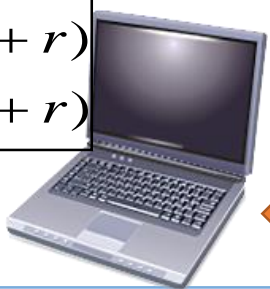
Public key 
(for encryption)

[Step 3] Computation of an encrypted result

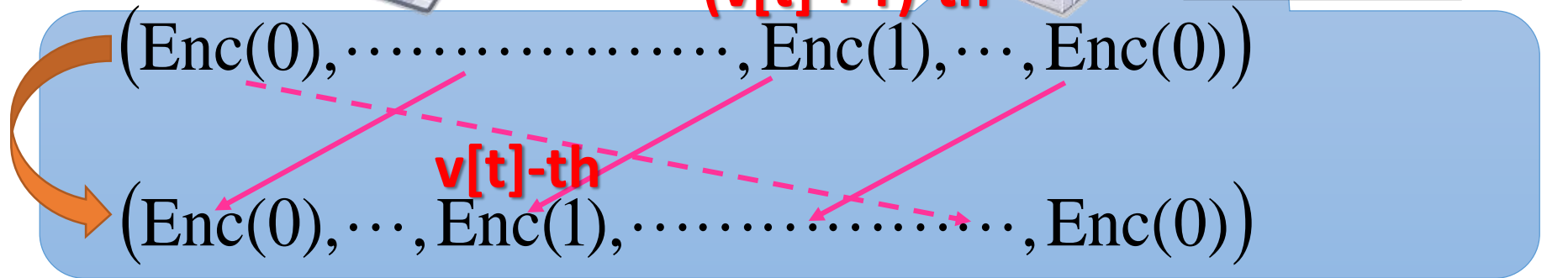
$$\text{Enc}(v[v[t]]) = \left(v[1] \otimes \text{Enc}(q_{(1-r)\text{Mod}_N}) \right) \oplus \dots \oplus \left(v[N] \otimes \text{Enc}(q_{(N-r)\text{Mod}_N}) \right)$$


[Step 2] Query entry

$$q_i = \begin{cases} 1 & (i = v[t] + r) \\ 0 & (i \neq v[t] + r) \end{cases}$$



$$v = (v_1, \dots, v_N)$$




(1 out of N) Recursive OT by AHE

[Step 1] Key setup

Secret key 
(for decryption)

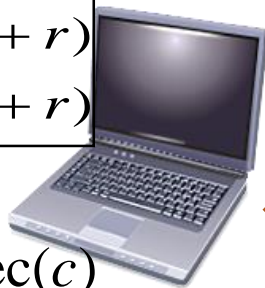
Public key 
(for encryption)

[Step 3] Computation of an encrypted result

$$\text{Enc}(v[v[t]]) = \left(v[1] \otimes \text{Enc}(q_{(1-r)\text{Mod}_N}) \right) \oplus \dots \oplus \left(v[N] \otimes \text{Enc}(q_{(N-r)\text{Mod}_N}) \right)$$


[Step 2] Query entry

$(\text{Enc}(q_1), \dots, \text{Enc}(q_N))$



$v = (v_1, \dots, v_N)$

$v[v[t]] = \text{Dec}(c)$

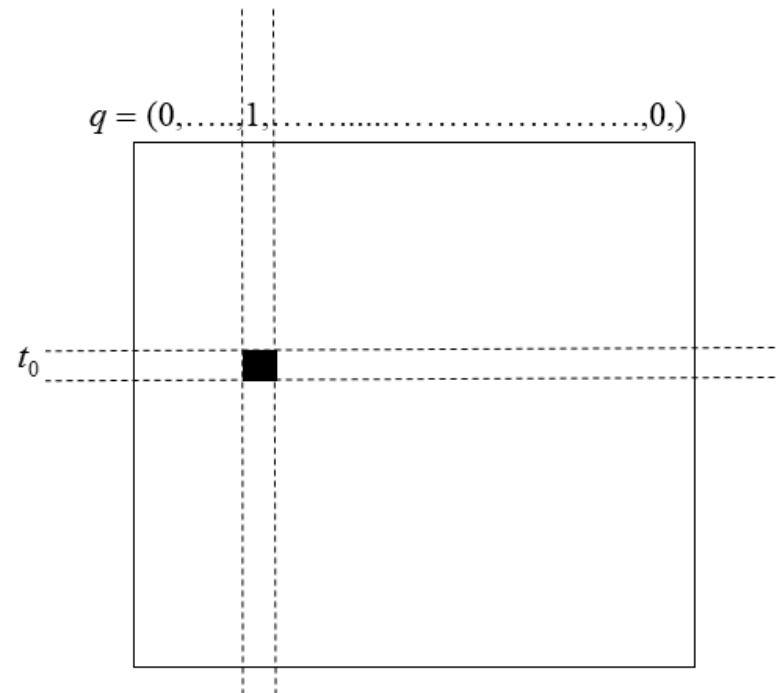
$c = \text{Enc}(v[v[t]])$

[Step 4] Decryption of the encrypted result

The user obtains $v[v[t]]$ w/o knowing $v[t]$

A communication efficient algorithm

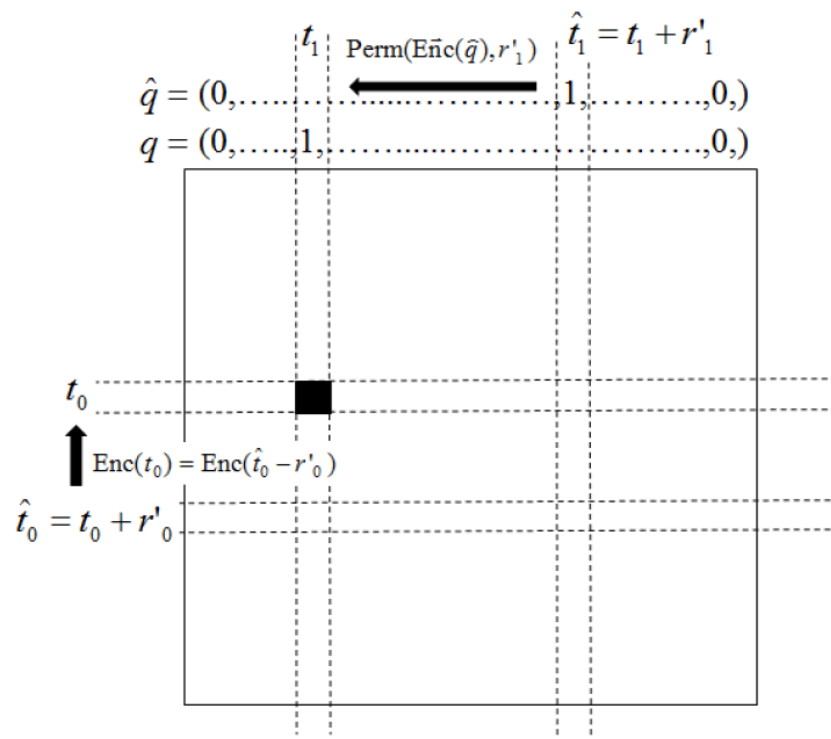
- Sublinear communication OT (Zhang+2013)
 - $O(\sqrt{N})$ communication
- Use 2-dimension representation of t :
 - $t_0 = t/\sqrt{N}$, $t_1 = t\% \sqrt{N}$
 - Computing:
 $v[i \times \sqrt{N} + t_1] + (t_0 - i) \times r$
for $i = 0, \dots, \sqrt{N}$
 - $(t_0 - i) \times r = 0$ iff. $T_0 = i$
(can leak $v[i \times \sqrt{N} + t_1]$
only t_0 -th row.)



A communication efficient algorithm

- Sublinear communication OT (Zhang+2013)
 - $O(\sqrt{N})$ communication
- Use 2-dimension representation of t :
 - $t_0 = t/\sqrt{N}$, $t_1 = t\% \sqrt{N}$
 - Computing:

$$v[i \times \sqrt{N} + t_1] + (t_0 - i) \times r$$
 for $i = 0, \dots, \sqrt{N}$
 - Use similar technique to recursive version.



PBWT-sec (shimizu+2016)

- PBWT (Durbin, 2014) + Recursive OT

Algorithm: PBWT-sec

Server creates a look-up table v

User initialize $[f, g]$

for $k = 1, \dots, L$:

 // updating $[f, g]$

 User sends $f = f + q[k] \times M, g = g + q[k] \times M$

 Server returns $V(f, k), V(g, k)$

 User updates $f = V(f, k), g = V(g, k)$

 User knows # of k -prefix matches by $(g - f + 1)$

 if $g - f < 0$: then exit;

**Computing
matches by ROT**

Shimizu+, Bioinformatics, 2016

<https://github.com/iskana/PBWT-sec>

Recursive search data structure for genomic data

- Our approach is applicable for the data structure enabling recursive search such as..
- BWT (Burrows+94, Ferragina+00)
 - A popular algorithm for NGS read alignment.
 - BWA(Li&Durbin10)
 - Bowtie(Langmead+09)
 - SOAP(Li+08)
 - etc...
- PBWT (Durbin14)
 - BWT like structure for searching aligned sequences.

Searching on aligned sequences

Match is computed by the data structure (pBWT) similar to BWT.

query: **GCA...GAAA**

s1: **ATGCA...AGCTA**

s2: **ATGTC...TATGT**

s3: **TTGCC...AGCGA**

s4: **TTGTC...TATGT**

s5: **GT**GCA**...**GACTA****

s6: **CTGTC...TATGT**

...

sM: **CTGTC...TATGT**

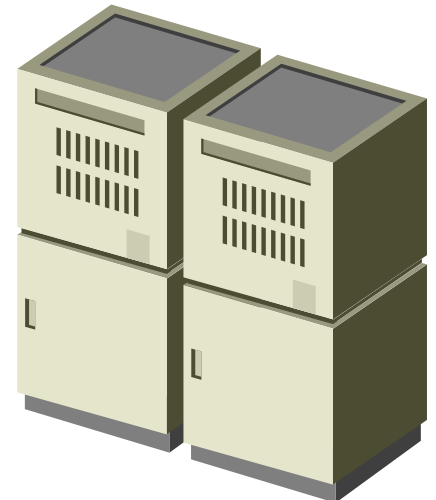
GCA,...,GAAA
from 3rd SNP



Sequence
k



of matches from k-
th SNP for every



Complexity

- PBWT-sec
 - Linear to the query length l
- Standard (exhaustive) approach
 - Sending every suffix of a query to check matches
 - Exponential to the query length l

	Time	Communication	Space
CP (user)	$O(\ell\sqrt{MD \Sigma })$	$O(\ell\sqrt{MD \Sigma })$	$O(\sqrt{MD \Sigma })$
CP (server)	$O(\ell MD \Sigma)$	$O(\ell\sqrt{MD \Sigma })$	$O(MD \Sigma)$
EX (user)	$O(\sqrt{D \Sigma ^\ell})$	$O(\sqrt{D \Sigma ^\ell})$	$O(\sqrt{D \Sigma ^\ell})$
EX (server)	$O(D \Sigma ^\ell)$	$O(\sqrt{D \Sigma ^\ell})$	$O(D \Sigma ^\ell)$

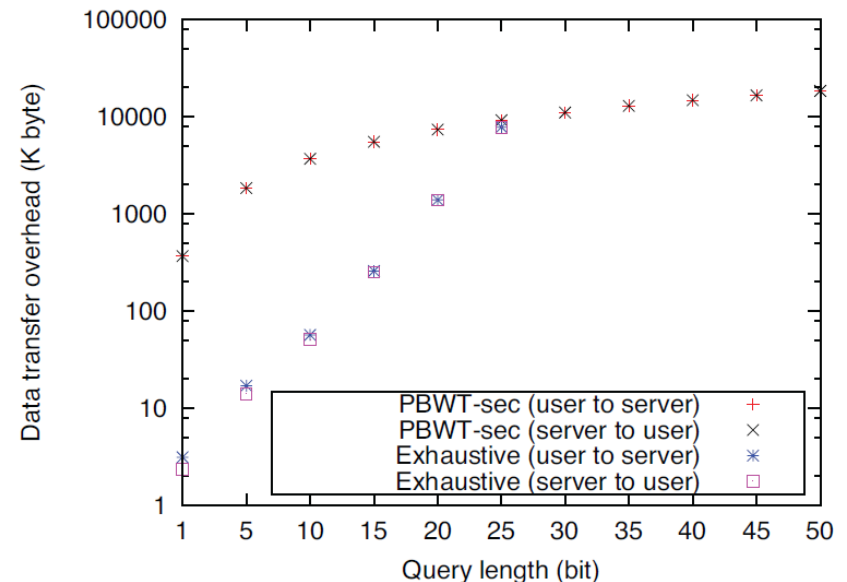
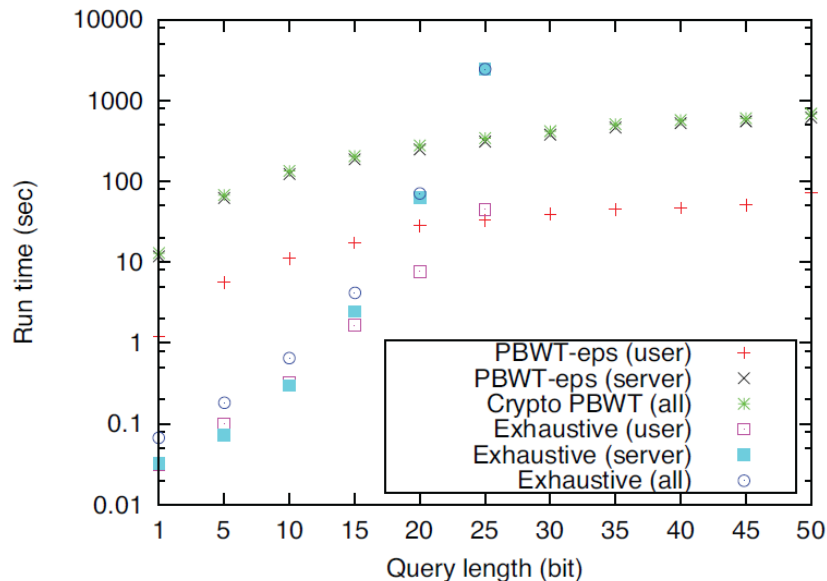
✂ Alphabet friendly algorithm has been developed (Sudo+, *in preparation*)

Experimental setup

- Implementation of PBWT-sec
 - C++ using AISTCRYPT (Open source C++ library of EC Elgamal).
- 2,184 haploid genomes from the chrom. 1 of the 1,000 Genomes Project (phase 1 data release).
- Tested on:
 - Laptop
(Intel Core(TM) i7 3.00GHz CPU; total 4 cores with HT)
 - A compute node
(Intel Xeon 2.40GHz CPU; total of 32 cores with HT)

Performance on laptop computers

- The observed run time and data transfer size of PBWT-sec is linear in the query length, while that of the exhaustive approach is exponential.



Run time

- Combined user's and server's run time was 15 sec for searching on 2,184 genomes by laptop (D=1)
- A compute node took between 7 and 132 seconds depending on the level of privacy.

	Laptop	Compute node		
Parallel Compute Cores	4	4	8	16
Run time (sec) with D = 1	15	22	15	7
Run time (sec) with D = 5	43	47	39	18
Run time (sec) with D = 10	78	84	68	31
Run time (sec) with D = 20	141	154	113	56
Run time (sec) with D = 50	338	386	260	132

D is a parameter for privacy level of the server.

Conclusion

- We have proposed a novel approach for searching string in a privacy-preserving manner.
- It achieves high utility and has strong security features and requires acceptable compute and communication resources.
- The algorithm can be used to facilitate sharing of genetic information across institutions and countries in order to identify large enough cohorts with a similar genetic backgrounds.

Acknowledgements

- Co-authors
 - Koji Nuida (AIST)
 - Gunnar Rätsch (ETHZ)
 - Hiroki Sudo (Waseda Univ)
 - Masanobu Jimbo (Waseda Univ)
- Shigeo Mitsunari (Cybozu) for developing AISTCRYPT.
- Members of Computational Biology Research Center at AIST