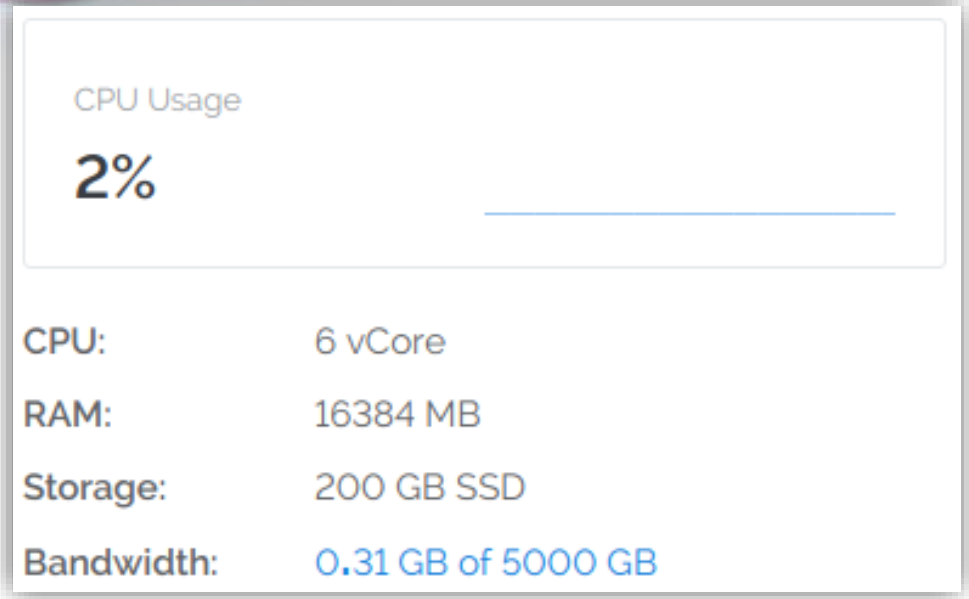
2019-01-03

Nankai-Baidu
Joint Laboratory

Parallel and Distributed
Software Technology Lab



关于HEAAN的总结

```
tree@tree: ~
File  Edit  View  Search  Terminal  Help
  1  [|||||||||||||||||||||||||||||||||||||||||||||||||99.3%]   4  [|||||||||||||||||||||||||||||||||||||||||||||||||100.0%]
  2  [|||||||||||||||||||||||||||||||||||||||||||||||||100.0%]   5  [|||||||||||||||||||||||||||||||||||||||||||||||||100.0%]
  3  [|||||||||||||||||||||||||||||||||||||||||||||||||100.0%]   6  [|||||||||||||||||||||||||||||||||||||||||||||||||100.0%]
  Mem[|||||||||||||||||||                          4.67G/15.7G]   Tasks: 22, 11 thr; 7 running
  Swp[                                               0K/0K]   Load average: 6.20 6.17 4.74
                                                              Uptime: 01:01:03
```

```
149.248.59.66
149.248.59.66
Enc NLGD time = 28550.7 ms
-----------------
encWData.logq after: 242
 !!! STOP 6 ITERATION !!!
 !!! START 7 ITERATION !!!
encWData.logq before: 242
-----------------
Start Enc NLGD
Enc NLGD time = 24961.3 ms
-----------------
encWData.logq after: 79
 !!! STOP 7 ITERATION !!!
----ENCRYPTED-----
w:0.0164058,0.150264,0.00216331,0.027966,0.00890167,0.0326707,0.00315831,0.0138809,0.0034814,0.0103655,0.00576676,-0.0210041,0.00158
041,-0.011796,-0.00698489,0.00437142,0.0333612,-0.0246181,0.0195439,0.00324137,0.0162196,-0.00800723,-0.00609741,-0.0253816,-0.00410
726,-0.0193205,0.00119126,0.0360199,0.0431541,0.00920888,-0.025231,-0.000594751,0.0116909,-0.000834049,-0.00183872,0.000804182,0.001
1451,0.00663709,0.0136006,0.000449622,0.00715692,-0.00119842,-0.00276409,0.00692596,0.00255863,0.00644382,-0.000643898,-0.0224966,-0
.00123865,0.00167096,-0.000178631,0.00576916,0.0199624,-0.00231451,0.0150486,0.0145144,-0.0188257,-0.00354363,-0.0117113,-0.00024511
,-0.00733171,-0.0192445,0.0077089,0.010513,0.0108393,0.00385812,-0.015641,-0.0145321,0.0113047,-0.0109755,-0.00388248,-0.000760774,0
.0133055,0.00339216,0.00257104,0.0143263,0.00291485,0.0154321,-0.00565885,0.0231084,0.00586879,0.00491765,-0.00547436,0.00739722,0.0
160957,-0.00761852,-0.0169673,0.000845321,-0.00322828,-0.00875158,0.0252244,-0.00137244,0.00849906,-0.00545511,0.00643193,-0.0001659
29,-0.00299338,-0.029969,0.0187531,0.0183589,0.0329634,0.0103479,0.00563936,0.010644,
Correctness: 54.2857 %.
AUC: 0.608268
-----------------
-------TRUE------
w:0.00675531,0.15165,0.00291801,0.0277866,0.00806619,0.0318684,0.00278064,0.012995,0.00307522,0.00946607,0.00479423,-0.0216599,0.000
604896,-0.0123591,-0.00778621,0.00352665,0.0247787,-0.0222425,0.0192013,0.0027827,0.0152482,-0.00862207,-0.00669233,-0.0258352,-0.00
489067,-0.0196852,0.000668965,0.0350206,0.042419,0.00838237,-0.0253378,-0.00137873,0.00422837,0.00141878,-0.00186224,-0.000284045,0.
000627714,0.00578338,0.0131597,-0.000529477,0.00639533,-0.00199692,-0.00342299,0.00592393,0.00214622,0.00552424,-0.00138821,-0.02315
44,-0.00902051,0.00365676,-0.000120347,0.0049936,0.0187372,-0.00322742,0.0145275,0.0137381,-0.0191393,-0.00429582,-0.0121181,-0.0008
75986,-0.00786588,-0.0195447,0.00708575,0.00995374,0.00289399,0.00566061,-0.0152887,-0.0146475,0.0107319,-0.0111199,-0.00428033,-0.0
0158697,0.0123018,0.0030173,0.00155763,0.013463,0.00204687,0.0146836,-0.00607847,0.0221053,-0.00229739,0.00707741,-0.00510883,0.0065
5329,0.015379,-0.00814613,-0.0170278,9.25131e-05,-0.00361483,-0.00891884,0.0238204,-0.00205995,0.00778649,-0.00613448,0.00602609,-0.
00107197,-0.0110664,-0.0272277,0.0190325,0.0175581,0.0317513,0.00942297,0.00491432,0.00990335,
Correctness: 53.6508 %.
AUC: 0.61505
-----------------
 !!! STOP 5 FOLD !!!
-----------------
Average Encrypted correctness: 263.175%
Average Encrypted AUC: 3.02143
Average True correctness: 260.317%
Average True AUC: 3.03519
root@vultr:/home/sly/IDASH2017/IDASH2017/Debug#
```

Nankai-Baidu Joint Laboratory

Parallel and Distributed Software Technology Lab

密文下的比较操作

- 密文之间的大小 ≫ 对应的明文之间的大小 （X）

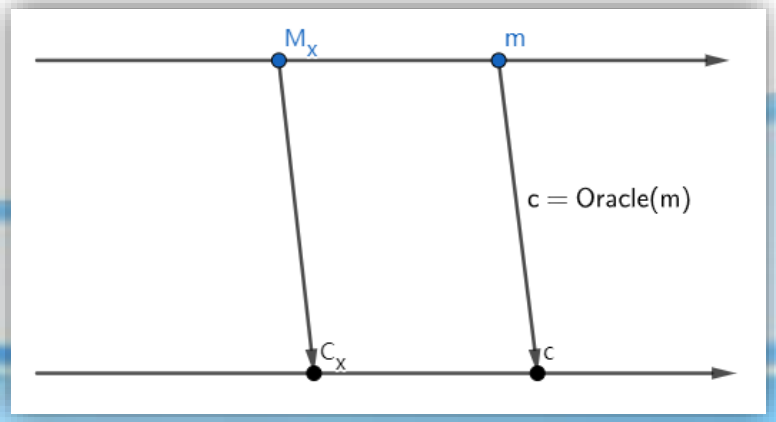
  假设：可以根据密文的大小推断出对应明文之间的大小，

  则　：一个对手可以在$O(n)$内确定一个密文所加密的明文。

  公开密钥系统E：pk, sk

  用户B：　$C_x = Enc(pk,\ M_x)$

  对手A：根据已有的信息构建一个预言机$Oracle(m) \approx Enc(pk, m)$



-输入任意明文输出系统E下对应的密文

-求用户B的M_x

-应该可以设计出$O(lgn)$的算法

密文下的比较操作

- 一种解决方案：多方计算

  涉及到密钥的传输 设计实验比较麻烦、比较困难

  多方计算存在若干问题

  同态加密的优势：在计算过程中不对密文进行解密，保证数据的安全

  也有人研究多方计算

密文下的比较操作

- 把明文的二进制的每一位加密

  IBM的HElib库实现了binaryCompare.h　compareTwoNumbers(...)

Doing Real Work with FHE: The Case of Logistic Regression

5.4 Comparing Two Integers

The procedure for integer comparison is somewhat similar to integer addition. We have two integers in binary, $a = (a_{t-1}, \ldots, a_1, a_0)$ and $b = (b_{t-1}, \ldots, b_1, b_0)$, and we want to compute the two integers $x = \max(a, b) = (x_{t-1} \ldots x_0)$ and $y = \min(a, b) = (y_{t-1} \ldots y_0)$, as well as the two indicator bits $\mu = (a > b)$ and $\nu = (b > a)$ (note that when $a = b$, both μ, ν are zero).

We begin by computing for every $i < t$ the bits $e_i := a_i + b_i + 1$ (which is 1 iff $a_i = b_i$) and $g_i := a_i + a_i b_i$ (one iff $a_i > b_i$). We then compute the products $e_i^* = \prod_{j \geq i} e_j$ and $g_i^* = g_i \cdot \prod_{j>i} e_j$, and the bits $\tilde{g}_i = \sum_{j \geq i} g_1^*$ (one iff $a_{t-1\ldots i} > b_{t-1\ldots i}$). Computing the products e_i^*, g_i^* is done using a recursive procedure somewhat similar to ComputeAllProducts from Section 4. Finally we compute the results by setting $\mu := \tilde{g}_0$, $\nu := 1 + \tilde{g}_0 + e_0^*$, and for $i = 0, \ldots, t-1$ we set $x_i := (a_i + b_i)\tilde{g}_i + b_i$ and $y_i := x_i + a_i + b_i$.

Note that we use all the g_i^*'s but only e_0^* for computing the output results, hence we somewhat optimized our procedure for computing these products by skipping the computation of e_i^*'s that are never used.

We remark that the last product $(a_i + b_i)\tilde{g}_i$ means that our procedure may use depth one more than the minimum possible. Using the absolutely smallest possible depth is challenging, straightforward solutions would take $O(t^2)$ multiplications (vs. $O(t)$ multiplications in the procedure above). While getting minimal depth with $O(t)$ multiplications is possible in theory, the procedure for doing this is overly complex (and extremely hard to parallelize), so we opted for a simpler procedure with slightly non-optimal depth. (Also, as opposed to the addition procedure from above, the simple procedure that we implemented here does not vary depending on the level of the input ciphertexts for a_i, b_i.)

明文空间{0,1}　明文模数为2

模2运算：加法、乘法

$$1 + 1 + 1 = 1 \ (mod\ 2)$$

$max(a, b)$　$min(a, b)$

$u = (a > b)$

$v = (a > b)$