FHE 導入
(L-III ゼミ 第2回ミーティング)

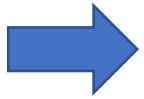
2021/07/23

FHE による演算

- 格子ベースのFHEで扱える演算は手法により異なる
  - 任意回の演算を行うために「ブートストラッピング」と呼ばれる処理を行う
- BFV, BGV(第2世代)
  - バイナリ,整数の加法・乗法が扱える
  - SIMD な演算もサポート
- TFHE(第3世代)
  - バイナリの加法(OR), 乗法(AND) が扱える
  - ブートストラッピングが速い(~10ms)
- CKKS(第4世代)
  - 固定小数点の加法,乗法が扱える
  - NNへの応用では SoTA な手法

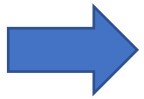
BFV,BGV vs TFHE

- 計算内容を工夫したり暗号学的なトリックで様々な計算を実現してきた
  - if (s == 0) then
    x * 2 + y
    else if (s == 1) then
    x * 4



(x * 2 + y)*s + (x * 4)*(s-1)

- ReLU 関数など実現できそうにない計算も存在
  - if (x > 0) then
    x
    else
    0



?

BFV, BGV vs TFHE

- TFHE は論理ゲートと等価な処理をFHE上で実現できる
- 整数 → 複数個の bit の集まり
  - 正負のある整数(符号付き整数) は2の補数で表現可能
  - 8bit 符号付き整数(int8_t)とか
    - 00000101 → 5
    - 11111001 → -7
- ReLU 関数も容易に表現可能
  - 符号bit を見る
    - 0 ならそのまま出力
    - 1 なら全部 0
      → 符号bit を反転して AND を取ればOK

Logical circuit over TFHE

- 論理ゲートは計算機を構成する基本要素
  → 論理回路に関する技術を転用できるのでは？
- 高級言語(Verilog, Chisel)で処理を記述→論理回路に変換 は容易
  - 最近はC 言語で書いた処理も論理回路に変換できる(HLS)
- TFHE に応用した事例
  - Cingulata(CEA LIST, **2015**)
  - FHE-Transpiler(Google, 2021)

Logical circuit over TFHE

- Chisel ならReLU関数も簡単に記述可能!
- **なんなら暗号を知らなくても実装できる**

```
class ReLU8bit extends Module {
    val io = IO(new Bundle{
        val in = Input(UInt(8.W))
        val out = Output(UInt(8.W))
    })

    when(!io.in(7)){
        io.out := io.in
    }.otherwise{
        io.out := 0.U(8.W)
    }
}
```

入力(整数8bit)

出力(整数8bit)

符号ビットが0(=正の数)なら
そのまま出力

符号ビットが1(=負の数)なら0を出力

Logical circuit over TFHE

- あとは論理回路に変換して TFHE に処理させればOK
  - 論理回路への変換は専用のOSS(yosys)
  - TFHE にどうやって処理をさせる？
- 回路を読み込んでいい感じに実行してほしい → Iyokan(VSP, 2019)
- (たぶん)世界最速のTFHEゲート並列実行エンジン