1. 盲计算库简介

该盲计算库是一种基于模分量同态的盲计算算法库，用来设计与盲计算相关的隐私保护应用，包含了一些常用的运算操作，可供用户实现多场景调用完成盲计算操作。

1. 盲计算库常用调用手册

声明：此处主要列举出该盲计算库相关的主要运算函数，不当之处，请大家批评指正。

**Blind.h:**

函数原型：void create\_BPS\_Paras(UserKey \*key, BPS\_PubParas\* pub\_paras, BPS\_PriParas\* private\_paras, int amp\_size);

功能说明：生成公私钥。

函数原型：int Encrypt(BPS\_PriParas \*private\_paras, int data, CipherText\* cipher);

功能说明：根据秘密参数加密，生成密文

函数原型：big Decrypt(BPS\_PriParas \*private\_paras, CipherText\* cipher);

功能说明：根据秘密参数解密

函数原型：int Decrypt2(BPS\_PriParas \*private\_paras, CipherText\* cipher);

功能说明：根据秘密参数解

函数原型：void BlindNegify(BPS\_PubParas\* pub\_paras, CipherText\* x, CipherText\* r);

功能说明：求密文相反数r=-x

函数原型：void BlindNegifyInplace(BPS\_PubParas\* pub\_paras, CipherText\* x);

功能说明：求密文相反数（代替）x=-x

函数原型：void BlindAdd(BPS\_PubParas\* pub\_paras, CipherText\* x, CipherText\* y, CipherText\* r);

功能说明：盲加法 r=x+y

函数原型：void BlindAddInplace(BPS\_PubParas\* pub\_paras, CipherText\* x, CipherText\* y);

功能说明：盲加法（代替）x=x+y

函数原型：void BlindSub(BPS\_PubParas\* pub\_paras, CipherText\* x, CipherText\* y, CipherText\* r);

功能说明：盲减法 r=x-y

函数原型：void BlindSubInplace(BPS\_PubParas\* pub\_paras, CipherText\* x, CipherText\* y);

功能说明：盲减法（代替） x=x-y

函数原型：void BlindMul(BPS\_PubParas\* pub\_paras, CipherText\* x, CipherText\* y, CipherText\* r);

功能说明：盲乘法 r=x\*y

函数原型：void BlindMulInplace(BPS\_PubParas\* pub\_paras, CipherText\* x, CipherText\* y);

功能说明：盲乘法（代替） x=x\*y

函数原型：void BlindPow(BPS\_PubParas\* pub\_paras, CipherText\* x, int y, CipherText\* r);

功能说明：盲幂运算 r=x^y

函数原型：void BlindPowInplace(BPS\_PubParas\* pub\_paras, CipherText\* x, int y);

功能说明：盲幂运算（替代）x=x^y

函数原型：void HalfBlindAdd(BPS\_PubParas\* pub\_paras, CipherText\* x, int y, CipherText\* r);

功能说明：半盲加法 r=x+y

函数原型：void HalfBlindAddInplace(BPS\_PubParas\* pub\_paras, CipherText\* x, int y);

功能说明：半盲加法(替代) x=x+y

函数原型：void HalfBlindSub(BPS\_PubParas\* pub\_paras, CipherText\* x, int y, CipherText\* r);

功能说明：半盲减法 r=x-y

函数原型：void HalfBlindSubInplace(BPS\_PubParas\* pub\_paras, CipherText\* x, int y);

功能说明：半盲减法(替代) x=x-y

函数原型：void HalfBlindMul(BPS\_PubParas\* pub\_paras, CipherText\* x, int y, CipherText\* r);

功能说明：半盲乘法 r=x\*y

函数原型：void HalfBlindMulInplace(BPS\_PubParas\* pub\_paras, CipherText\* x, int y);

功能说明：半盲乘法(替代) x=x\*y

函数原型：void HalfBlindMulInplace(BPS\_PubParas\* pub\_paras, CipherText\* x, int y);

功能说明：盲连加

函数原型：void BlindMulMany(BPS\_PubParas\* pub\_paras,int num, CipherText\* x[], CipherText\* r);

功能说明：盲连乘

**Blind\_parallel.h：**

函数原型：big ComposeToBig(BPS\_PriParas \*private\_paras, int num, int plain[]);

功能说明：合成大数

函数原型：int MultiEncrypt(BPS\_PriParas \*private\_paras, big data, CipherText\* cipher);

功能说明：对大数进行加密

函数原型：big MultiDecrypt(BPS\_PriParas \*private\_paras, CipherText\* cipher);

功能说明：根据秘密参数解密

函数原型：void DecomposeToInt(BPS\_PriParas \*private\_paras, int num, big cipher, int res[]);

功能说明：根据秘密参数解密

函数原型：void MultiBlindNegify(BPS\_PubParas\* pub\_paras, CipherText\* x[], CipherText\* r[]);

功能说明：求密文相反数r=-x

函数原型：void MultiBlindNegifyInplace(BPS\_PubParas\* pub\_paras, CipherText\* x[]);

功能说明：求密文相反数(替代)x=-x

函数原型：void MultiBlindAdd(BPS\_PubParas\* pub\_paras, CipherText\* x[], CipherText\* y[], CipherText\* r[]);

功能说明：盲加法 r=x+y

函数原型：void MultiBlindAddInplace(BPS\_PubParas\* pub\_paras, CipherText\* x[], CipherText\* y[]);

功能说明：盲加法(替代) x=x+y

函数原型：void MultiBlindSub(BPS\_PubParas\* pub\_paras, big x, big y, big r);

功能说明：盲减法 r=x-y

函数原型：void MultiBlindSubInplace(BPS\_PubParas\* pub\_paras, CipherText\* x[], CipherText\* y[]);

功能说明：盲减法(替代) x=x-y

函数原型：void MultiBlindMul(BPS\_PubParas\* pub\_paras, CipherText\* x[], CipherText\* y[], CipherText\* r[]);

功能说明：盲乘法r=x\*y

函数原型：void MultiBlindMulInplace(BPS\_PubParas\* pub\_paras, CipherText\* x[], CipherText\* y[]);

功能说明：盲乘法(替代) x=x\*y

函数原型：void MultiBlindPow(BPS\_PubParas\* pub\_paras, CipherText\* x[], int y[], CipherText\* r[]);

功能说明：盲幂运算 r=x^y

函数原型：void MultiBlindPowInplace(BPS\_PubParas\* pub\_paras, CipherText\* x[], int y[]);

功能说明：盲幂运算(替代) x=x^y

函数原型：void MultiHalfBlindAdd(BPS\_PubParas\* pub\_paras, CipherText\* x[], int y[], CipherText\* r[]);

功能说明：半盲加法 r=x+y

函数原型：void MultiHalfBlindAddInplace(BPS\_PubParas\* pub\_paras, CipherText\* x[], int y[]);

功能说明：半盲加法(替代) x=x+y

函数原型：void MultiHalfBlindSub(BPS\_PubParas\* pub\_paras, CipherText\* x[], int y[], CipherText\* r[]);

功能说明：半盲减法 r=x-y

函数原型：void MultiHalfBlindSubInplace(BPS\_PubParas\* pub\_paras, CipherText\* x[], int y[]);

功能说明：半盲减法(替代) x=x-y

函数原型：void MultiHalfBlindMul(BPS\_PubParas\* pub\_paras, CipherText\* x[], int y[], CipherText\* r[]);

功能说明：半盲乘法 r=x\*y

函数原型：void MultiHalfBlindMulInplace(BPS\_PubParas\* pub\_paras, CipherText\* x[], int y[]);

功能说明：半盲乘法(替代) x=x\*y

**Matrix.h：**

函数原型：void EncryptArray(BPS\_PriParas \*private\_paras, int rows, int cols, int\*\* data, CipherText\*\*\* cipher);

函数功能：将二维数组加密成密文矩阵

函数原型：void BlindAddMatrix(BPS\_PubParas\* pub\_paras, int rows, int cols, CipherText\*\*\* x, CipherText\*\*\* y, CipherText\*\*\* r);

函数功能：矩阵盲加法

函数原型：void BlindSubMatrix(BPS\_PubParas\* pub\_paras, int rows, int cols, CipherText\*\*\* x, CipherText\*\*\* y, CipherText\*\*\* r);

函数功能：矩阵盲减法

函数原型：void BlindMulMatrix(BPS\_PubParas\* pub\_paras, int rows, int cols, CipherText\*\*\* x, CipherText\*\*\* y, CipherText\*\*\* r);

函数功能：矩阵盲乘法

函数原型：void HalfBlindMulMatrix(BPS\_PubParas\* pub\_paras, int rows, int cols, CipherText\*\*\* x, int\*\* y, CipherText\*\*\* r);

函数功能：矩阵盲点积

函数原型：void BlindConvolution(BPS\_PubParas\* pub\_paras, CipherText\*\*\* x, CipherText\*\*\* y, CipherText\*\*\* r);

函数功能：矩阵盲卷积

函数原型：int\*\* DecryptArray(BPS\_PriParas \*private\_paras, int rows, int cols, CipherText\*\*\* cipher);

函数功能：将密文矩阵解密成二维数组