

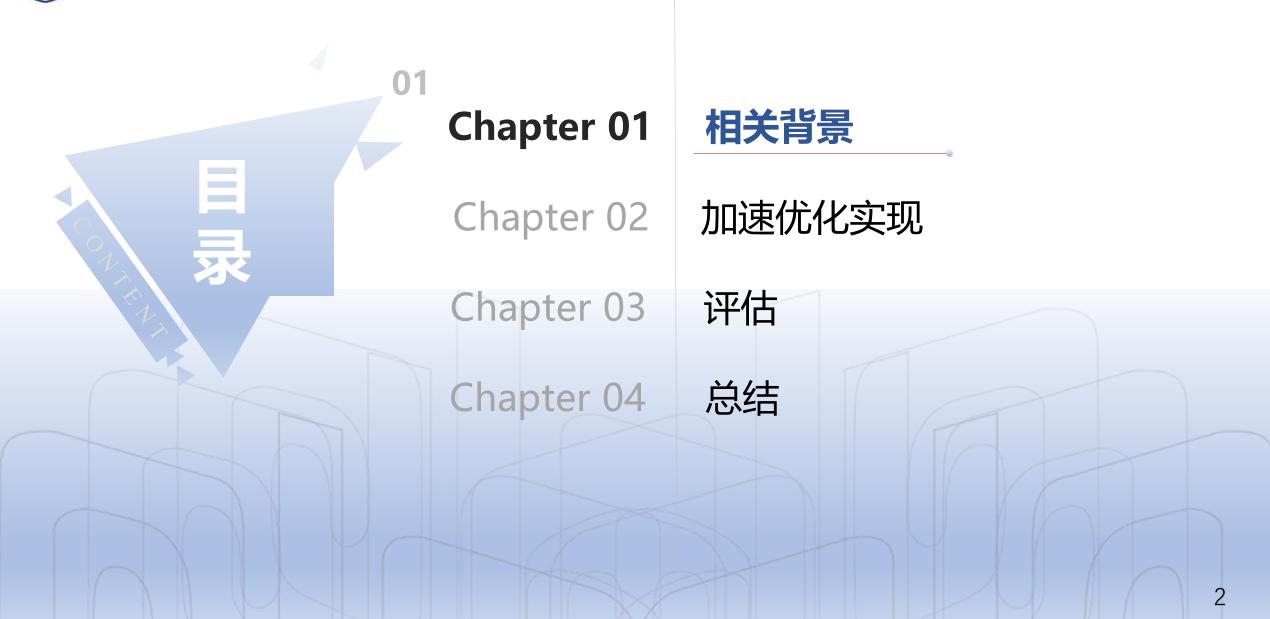
HI Kyber:基于GPU的高性能KYber实现

HI-Kyber: A novel high-performance implementation scheme of Kyber based on GPU

汇报人: 袁智健

指导老师: 董建阔





研究背景-符合密码安全需求



量子计算

•基于 Shor 和 Grover 算法的量子计算可以在多项式时间内求解离散对数、大整数因子分解等问题,这意味着在未来的量子计算时代,所有基于这些困难问题的公钥密码算法都不能安全使用

后量子 密码

• NIST 从 2016 年便启动了全球范围内的后量子密码标准征集,于 2022 年公布了后量子密码算法标准化过程的第三轮结果,包括四个选定的算法和四个候选算法; Kyber 是唯一的公钥加密算法

性能问题

• 软硬件平台上的性能优化实现是后量子密码算法能够成功应用到工业界的关键; 高复杂度算法增加了部署成本; **主流的后量子密码在吞吐性能、空间占用等方面存在诸多瓶颈与问题**

CRYSTALS-Kyber

CRYSTALS-DILITHIUM

Falcon

SPHINCS+

提高后量子密码计算效率,弥补密码迁移工作时空效率差距,是亟待解决的科学难题

GPU平台下的密码加速-符合业界研究需求





CPU VS GPU

- 强劲的算术逻辑单元,拥有复杂的ALU结构, 主要设计用于通用计算
- 复杂的控制器
- 较大缓存Cache, 采用更复杂的缓存管理策略
- 高效能的算术逻辑单元,拥有大规模的SIMD 结构,更适用于大规模并行计算
- 简单的控制器,适合流式处理、相似操作
- · Cache规模较小,局部性原理较为重要

业界已开展很多工作

Gupta 等人

 在 Volta 架构的 NVIDIA GPU Tesla V100中实现

了 Kyber1024子函数的 优化、特别是数论变换

NTT 和 Keccak

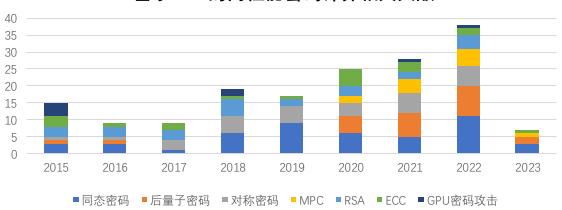
Lee 等人

• 在 NVIDIA GPU RTX2060上为数论变换 NTT 实现三种不同的细 粒度并行方法

• 利用AI加速器Tensor Core加速多项式乘法, 获得了更好的性能提高

Wan 等人

基于GPU的高性能密码计算相关文献



GPU平台的并行计算能力强大,适合加速密码算法等计算复杂度高的任务

背景知识-CRYSTAL-Kyber



Kyber 是一种 IND-CCA2 安全的密钥封装机制(KEM),用非对称加密的思想去封装密钥并进行密钥协商的算法,其安全性正是基于求解模格上错误学习问题困难假设。

算法 5 KYBER.CPAPKE.KeyGen(): 密钥生成

输出: 私钥 sk, 公钥 pk.

1: $d \leftarrow Random()$

▶ 随机数生成

2: $(\rho, \sigma) := G(d)$

▶ 通过确定性算法操作随机数以获得密钥参数

3: $\hat{\mathbf{A}} \leftarrow Gen_matrix_\hat{\mathbf{A}}(\rho), \hat{\mathbf{A}} \in R_{\rho}^{k \times k}$ in NTT domain

> 生成公钥参数Â

4: $\mathbf{s} \leftarrow Sample_s(\sigma), \mathbf{s} \in R_n^k \text{ from } B_n$

▶ 生成秘密 s

5: $\mathbf{e} \leftarrow Sample_e(\sigma), \mathbf{e} \in R_q^k \text{ from } B_{\eta_1}$

▶ 生成噪声参数e

6: $\hat{\mathbf{s}} := NTT(\mathbf{s})$

▶ 对秘密参数进行自定义 NTT 变换

7: ê := NTT(e)

▶ 对噪声参数进行自定义 NTT 变换

8: $\hat{\mathbf{t}} := \hat{\mathbf{A}} \circ \hat{\mathbf{s}} + \hat{\mathbf{e}}$

▶逐项乘法,再加上噪声

9: return $pk := Encode(\hat{\mathbf{t}}||\rho)$, $sk := Encode(\hat{\mathbf{s}})$

▶ 对公私钥对进行编码

算法 6 KYBER.CPAPKE.Enc(): 加密

输入: 公钥 pk, 消息明文 m, 随机数种子 r

输出:密文c

1: $(\hat{\mathbf{t}}, \rho) \leftarrow Decode(pk)$

▶ 对公钥进行解码

2: $\hat{\mathbf{A}}^T \leftarrow Gen_matrix_\hat{\mathbf{A}}^T(\rho), \hat{\mathbf{A}}^T \in R_{\theta}^{k \times k}$ in NTT domain

3: $\mathbf{r} \leftarrow Sample_r(r), \mathbf{r} \in R_a^k \text{ from } B_m$

4: $\mathbf{e}_1 \leftarrow Sample_e_1(r), \mathbf{e}_1 \in R_a^k \text{ from } B_m$

5: $e_2 \leftarrow Sample_e_2(r), e_2 \in R_a$ from B_m

6: $\hat{\mathbf{r}} := NTT(\mathbf{r})$

7: $\mathbf{u} := NTT^{-1}(\hat{\mathbf{A}} \circ \hat{\mathbf{r}}) + \mathbf{e}_1$

» 对相关参数进行逆自定义 NTT 变化

8: $v := NTT^{-1}(\hat{\mathbf{t}}^T \circ \hat{\mathbf{r}}) + e_2 + Decompress(m)$

» 对明文用到特定的处理方法

9: return $c_1 := Encode_u(\mathbf{u}), c_2 := Encode_v(v)$

▶ 对加密结果以特定方法编码

算法 7 KYBER.CPAPKE.Dec(): 解密

输入:密钥sk,密文c

输出:消息明文 m

1: $\mathbf{u} := Decode_u(c)$

2: $v := Decode_v(c)$

3: $\hat{\mathbf{s}} := Decode(sk)$

4: **return** $m := Compress(v - NTT^{-1}(\hat{\mathbf{s}} \circ NTT(\mathbf{u})))$

▶ 先对参数解码

背景知识-快速数论变换NTT



NTT(Number Theoretic Transform)在Kyber算法用于实现多项式的变换。 具体来说,Kyber使用NTT来进行多项式的前向变换和逆向变换,

多项式的乘法

1. 系数乘法

$$A(x) = a_0 + a_1 x + a_2 x^2 + \dots + a_{n-1} x^{n-1}$$

$$B(x) = b_0 + b_1 x + b_2 x^2 + \dots + b_{n-1} x^{n-1}$$

$$C(x) = c_0 + c_1 x + c_2 x^2 + \dots + c_{2n-2} x^{2n-2}$$

2. 点值乘法

给定n个点可以确定n-1次函数曲线的系数点值和系数存在映射关系,可以相互转化

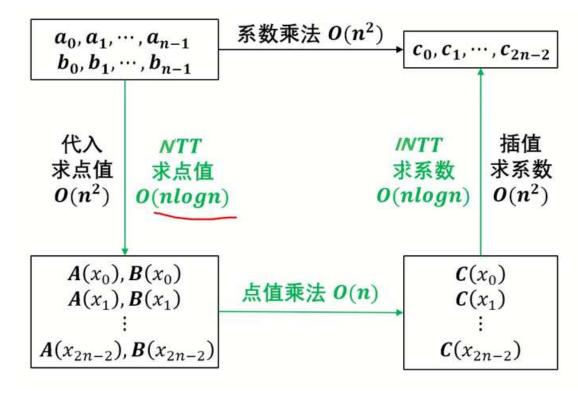


(1) 取 $x_0, x_1, x_2, \cdots x_{2n-2}$ 分别代入,求点值

$$A(x)$$
: $y_0, y_1, y_2, \dots, y_{2n-2}$
 $B(x)$: $y'_0, y'_1, y'_2, \dots, y'_{2n-2}$

(2) 相乘 C(x): $y_0y_0', y_1y_1', y_2y_2', \cdots, y_{2n-2}y_{2n-2}'$

(3) 插值 C(x): $c_0, c_1, c_2, \cdots, c_{2n-2}$



时间复杂度分析

背景知识-快速数论变换NTT



蝴蝶变换是在快速数论变换(FFT)和数论变换(NTT)等算法中的关键步骤之一,用于实现多项式的点值表示和系数表示之间的转换。

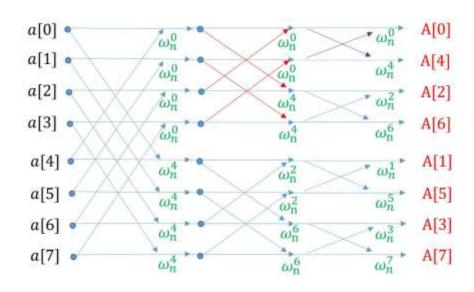
核心思想是将一个多项式的点值表示划分为两个子集,然后通过乘上旋转因子进行组合。

蝴蝶变换的数学表达式如下:

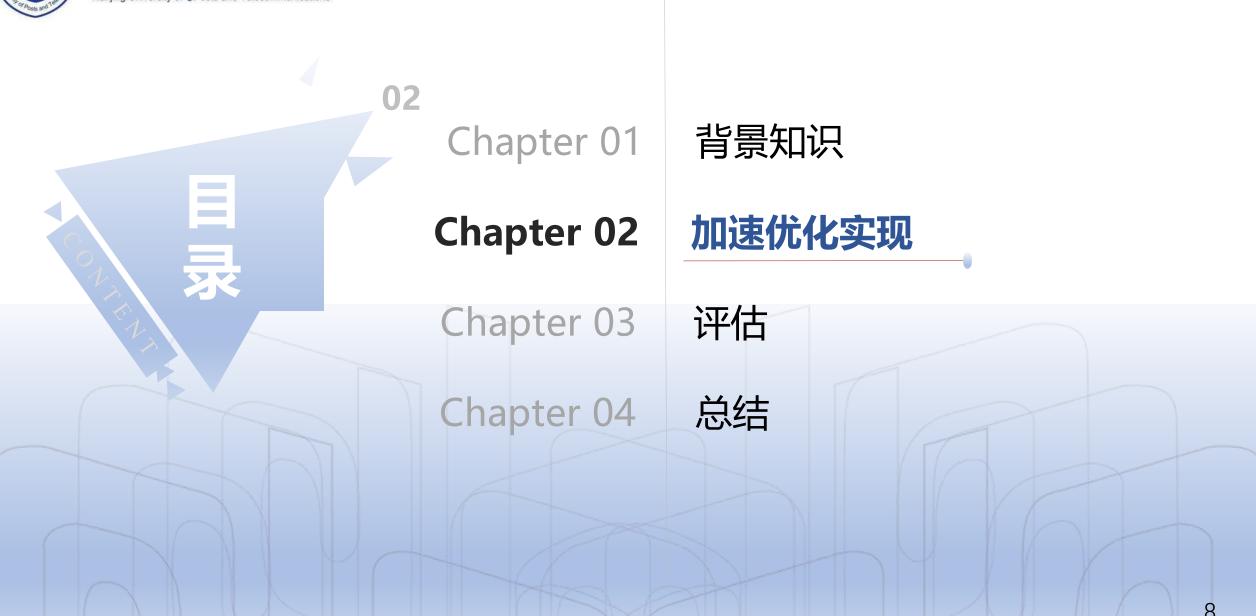
$$X_k = W_N^{jk} \cdot (X_j + W_N^{N/2} \cdot X_{j+N/2}) \ X_{k+N/2} = W_N^{jk+N/2} \cdot (X_j - W_N^{N/2} \cdot X_{j+N/2})$$

其中:

- X_j 和 $X_{j+N/2}$ 是原始多项式的两个子集。
- X_k 和 $X_{k+N/2}$ 是结果多项式的两个对应子集。
- W_N 是N次单位根,也就是旋转因子。





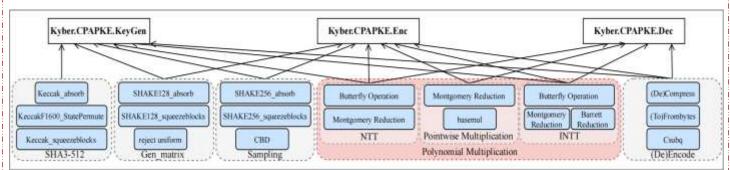




总体框架-基于 kernel 融合的设计



Kyber 整体的算法框架:



Kyber 分为三部分:

- 1) KeyGen
- 2) Encryption
- 3) Decryption

NIST Security Levels

Kyber512 1 Kyber768 3 Kyber1024 5 其中最耗时的算法之一: 多项式乘法

- 1) **数论变换 NTT**
- 2) 点乘算法 Pointwise Multiplication
- 3) 逆向数论变换 INTT

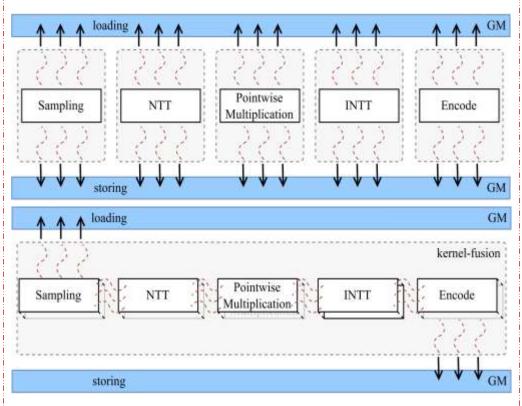
NTT 一直是研究热点 (Dilithium、NTRU、Saber等)

HI-Kyber 主要针对 NTT 的 GPU 实现提出优化

本工作将与:

- ✓ 基于相同指令架构的 GPU [1] 实现做对比 (TPDS 2020 Gupta 等人的工作)
- ✓ 目前性能最优的 GPU [2] 实现做对比 (ESORICS 2022 Wan 等人的工作)

多 kernel 模式 [2] VS kernel 融合模式:



不同的并行方式导致访问全局内存的次数:

Kernel 融合:避免全局内存的访问

Parallelization	KeyGen	Encryption	Decryption
multi-kernel	12	17	8
kernel-fusion	1	1	1

NTT - 最耗时的核心算法之一



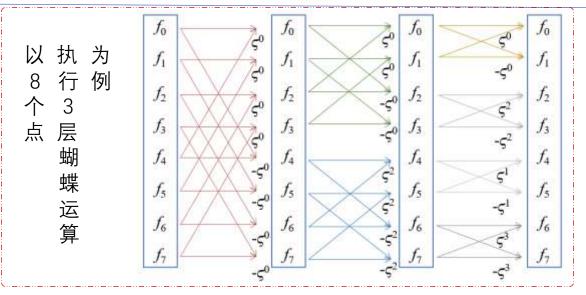
NTT:
$$\mathbf{v} = [v_0, v_1, \dots, v_{n-1}] \Longrightarrow \mathbf{V} = [V_0, V_1, \dots, V_{n-1}]$$

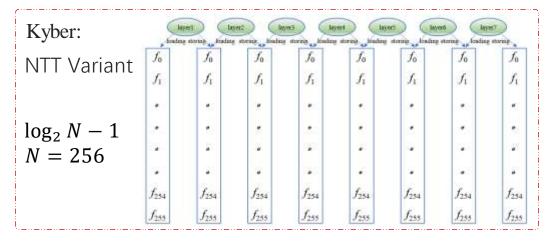
$$\mathbf{V_i} = \sum_{j=0}^{n-1} v_j \cdot \zeta^{ij}$$

Chinese Remainder Theorem(CRT):

$$NTT(f) = \hat{f}_0 + \hat{f}_1 X^1 + \dots + \hat{f}_{255} X^{255}$$

$$\begin{bmatrix} \hat{f}_0 \\ \hat{f}_2 \\ \hat{f}_4 \\ \vdots \\ \hat{f}_N \end{bmatrix} = \begin{bmatrix} 1 & \zeta^{0 \times 1} & \cdots & \zeta^{0 \times (\frac{N}{2} - 1)} \\ 1 & \zeta^{1 \times 1} & \cdots & \zeta^{1 \times (\frac{N}{2} - 1)} \\ 1 & \zeta^{2 \times 1} & \cdots & \zeta^{2 \times (\frac{N}{2} - 1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \zeta^{\frac{N}{2} \times 1} & \cdots & \zeta^{\frac{N}{2} \times (\frac{N}{2} - 1)} \end{bmatrix} \begin{bmatrix} f_0 \\ f_2 \\ f_4 \\ \vdots \\ f_N \end{bmatrix} \begin{bmatrix} \hat{f}_1 \\ \hat{f}_3 \\ \hat{f}_5 \\ \vdots \\ \hat{f}_{N-1} \end{bmatrix} = \begin{bmatrix} 1 & \zeta^{0 \times 1} & \cdots & \zeta^{0 \times (\frac{N}{2} - 1)} \\ 1 & \zeta^{1 \times 1} & \cdots & \zeta^{1 \times (\frac{N}{2} - 1)} \\ 1 & \zeta^{2 \times 1} & \cdots & \zeta^{2 \times (\frac{N}{2} - 1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \zeta^{\frac{N}{2} \times 1} & \cdots & \zeta^{\frac{N}{2} \times (\frac{N}{2} - 1)} \end{bmatrix} \begin{bmatrix} f_1 \\ f_3 \\ f_5 \\ \vdots \\ f_{N-1} \end{bmatrix}$$



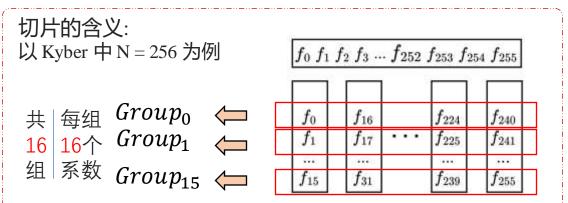


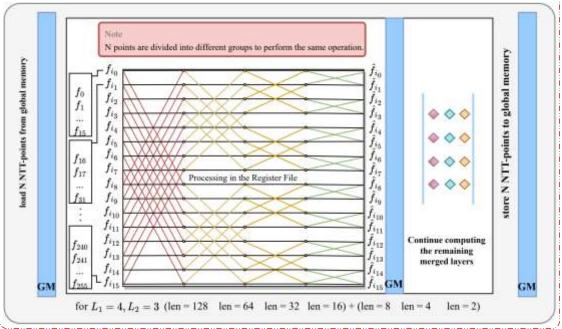


[1] E. Alkim, P. Jakubeit, and P. Schwabe, "Newhope on ARM Cortex-M," in *Security, Privacy, and Applied Cryptography Engineering: 6th International Conference, SPACE 2016, Hyderabad, India, December 14-18, 2016, Proceedings.* Springer, 2016, pp. 332–349.

方案一: 切片式层合并 SLM







```
\log_2 N - 1 = L_1 + L_2 + \dots + L_n
n:合并层的个数 L_i:每个合并层的层数
SLM:将 NTT 的系数量 N 分成 j 个切片
每个切片的第 k 个数据组成一组做蝴蝶运算
```

```
Algorithm 1: A sliced layer merging scheme of L_i
for NTT
    input: f(x) \in \mathbb{Z}_q[X]/(X^n+1), \zeta^n \in \mathbb{Z}_q
    output: \hat{f}(x) \in \mathbb{Z}_q[X]/(X^n+1), after L_i Layer
               Merging
     L_{finished} \leftarrow \sum_{i=1}^{i-1} L_i
     MAX\_Group \leftarrow N \gg (L_{finished} + L_i)
    for Group \leftarrow 0 to MAX\_Group do
        k \leftarrow \frac{N}{N \gg (L_{finished})} for len \leftarrow N \gg (L_{finished} + 1) to
          N \gg (L_{finished} + L_i) do
                     /* shift right one layer */
             for start \leftarrow Group \text{ to } N \text{ do } /* \text{ step by}
               i + len */
                  zeta \leftarrow \zeta^{k++}
                  for j \leftarrow start to start + len do
                   /* step by MAX_Group */
                      Butterfly Unit(f[j + len], fj])
```

当分组的数据达到一定程度时,数据可以直 接在寄存器中处理 切片的片数 j 是根据 合并层中最后一层 的蝴蝶运算距离来 决定的

寄存器是否冗余:

- 若分片数量太少,则每组需要处理的数据量可能会超过可用寄存器的数量;
- 若分片数量太多, 则每组需要处理 的数据量可能会 发生寄存器冗余

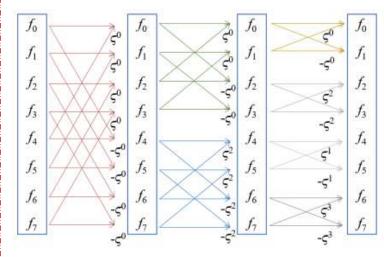
默认情况是选择最大 切片片数来使每组中 数据的加载量为**最少 可执行系数**

DFS-深度优先搜索



Preorder traversa

NTT 基于分而治之的思想



拥有相同旋转因子的 系数看成一个节点

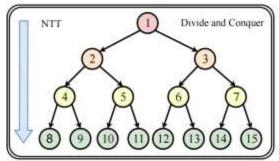
第一层 one node 第二层 two nodes 以此类推

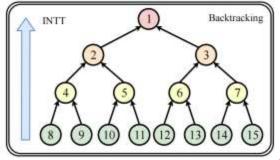
NTT



执行 BFS 的满二叉树

NTT: 将 解决大问题 分解成 解决小问题 INTT: 由 解决小问题 组成 大问题的解



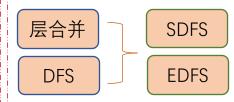


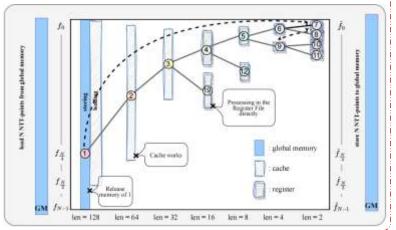
提出<mark>深度优先搜索策略(the depth-first search,DFS),以先</mark>序方式处理节点

从根节点出发,沿着左子树的方法进行纵向遍历,直到 找到叶子结点为止。然后回溯到前一个节点,进行右子 树节点的遍历,直到遍历完所有可到达节点止。

在 DFS 中,每层需要处理的系数量是逐层减半的

- ▶ 优化寄存器的使用
- ➤ 增加 Cache 击中率

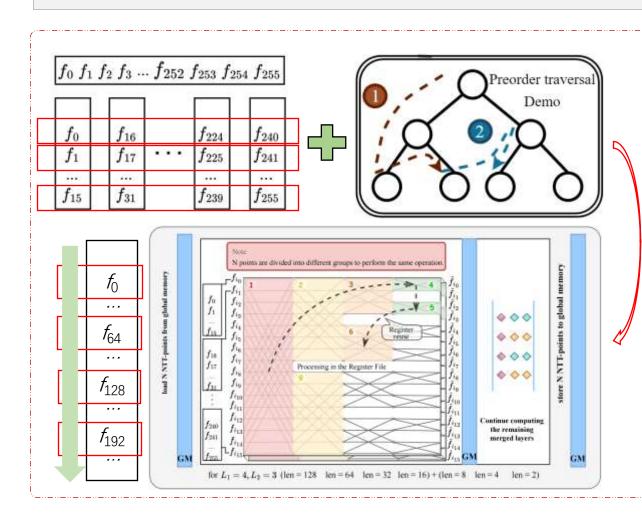




方案二: 切片式深度优先搜索SDFS



SLM DFS SDFS



DFS 与切片结合做切片遍历 即减少全局内存的访问次数 又能高效复用寄存器

当分组的数据达到一定程度时,数据 可以直接在寄存器中处理

有了闲置的寄存器,编译器就会提前安排 其他可用工作

寄存器是否冗余:

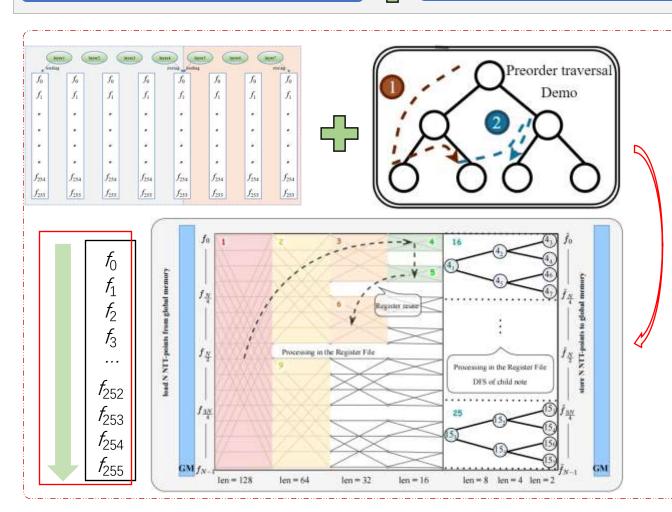
- 过载(编译器的工作)
- 冗余(大量实验证明)

```
input : f(x) \in \mathbb{Z}_q[X]/(X^n + 1), \zeta^n \in \mathbb{Z}_q
 output: f(x) \in \mathbb{Z}_q[X]/(X^n + 1), after L_c Layer
        Merging
\begin{array}{l} L_{fundad} \leftarrow \sum_{j=1}^{i-1} L_j \\ MAX\_Group \leftarrow N \gg (L_{fundad} + L_i) \\ \text{for } Group \leftarrow 0 \text{ in } MAX\_Group \text{ do} \end{array}
    a+- Group: /* the location where
     the butterfly begins */
     Leafnode_num -- U /- the number of
     leaf opdes traversed */
    k \leftarrow \frac{N}{N34(3\gamma_{\text{transled}}+3\gamma-2)}
         /. locate the C of the father
     len \leftarrow N \gg (L_{finished} + L_i - 1) /* visit
     the root node +/
     while Leafnode\_sum \neq \frac{N}{N \gg (L_{framew} + L_{g} - 2)}
                     /+ Jump nodes in layer
     (Louises + La) are not fully
     braversed 4/
        if len \neq N \gg (L_{finished} + L_i + 1) then
         /* Non-empty mode +/
           arts - Ch
            for start or a to a + light do /+ stap
             by j+lm +/
                for j +- start to start + less do-
                /* step by MAX_Group */
Butterfly Unit(f[j+len],fj[)
           Jen +- Jen € 3 /+ go to the left
            node +/
           k\mapsto k\gg 2
           Jen + Jen € 2 /+ backtrace the
             node «/
           k \leftarrow k \gg 2
            \operatorname{cris} \leftarrow \zeta^{(n+1)} \ / + \ \operatorname{qo} \ \operatorname{to} \ \operatorname{the \ right}
             node in Right-subtree */
           n = j + len
               /* hutterfly begins in the
            leaf node «V
            for j \leftarrow a m a + \frac{h m}{3} da /* step by
             MAX_Group +/
              Batterfly Unit(f[j+len], f/f)
            a = j + Jen
              /* butterfly begins in the
            hacktracing node +/
           k = 12 /* locate the C of the backtracing node */
            len = prov(2, jump, +1)
             /+ backtrace the sode +/
            Leafnusk_num++ /* add the
             number of visited node */
```

方案三:整体深度优先搜索EDFS



LM DFS EDFS



DFS 与分层的思想结合做整体遍历

即减少全局内存的访问次数 又能高效复用寄存器 还能访问连续的内存地址

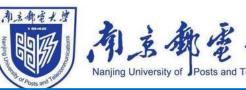
重点注重

- 内存层次结构优化
- ➤ 高效利用 GPU 的计算型资源

在具体的实验方案中,作为 GPU 方案的设计人员,尽管我们无法精确控制每个寄存器保存的具体数据,尽可能地重复使用寄存器是至关重要的。

```
Algorithm 3: EDFS-NTT scheme
  input : f(x) \in \mathbb{Z}_a[X]/(X^n + 1), \zeta^n \in \mathbb{Z}_a
  output: f(x) \in \mathbb{Z}_q[X]/(X^n + 1), after L_i Layer
   L_{finished} \leftarrow \sum_{i=1}^{i=1} L_i
   Leafnode\_num \leftarrow 0
                              /* the number of
   leaf nodes traversed */
                 /* the location where the
   butterfly begins +/
  while Leafnode\_num \neq \frac{N}{N \gg (L_{finished} + L_i = 2)} do
               /* Leaf nodes are not fully
  traversed +/
      if len \neq N \gg (L_{finished} + L_i + 1) then
       /* Non-empty node */
         zeta \leftarrow \ell^k
         for start \leftarrow a to a + \frac{len}{2} do /* step by
             for i + start to start + len do
              Butterfly Unit(f[j + len], f[j])
         len \leftarrow len \ll 2 /* go to the left
          node */
         k \leftarrow k \gg 2
         len \leftarrow len \ll 2
                           /* backtrace the
         k \leftarrow k \gg 2 \ zeta \leftarrow \zeta^{k+1} \ /* \ go \ to \ the
          right node in Right-subtree +/
         a = j + len
            /* CT butterfly begins in the
          leaf node +/
         for j \leftarrow a to a + len do
          Butterfly Unit(f[j + len], fj[)
         a = j + len
            /* CT butterfly begins in the
          next node */
         k = \frac{k+2}{posc(2,jump_s)} /+ locate the \zeta of
          the backtracing node ./
          len = pow(2, jump_i + 1)
          Leafnode_num++
                                      /* add the
          number of visited node */
```

Comparison-三种方案对比



对每一种方案都做了详细分析,考虑和设计;总的来说,设计了三种快速实现 NTT 的方案: SLM、SDFS 和 **EDFS**

LM的收益:

[1] 减少全局内存的访问次数

切片的收益:

[2] 当分组的数据达到一定程度时,数据可以直接在寄存器中处理

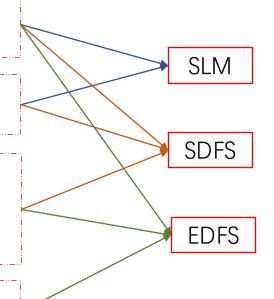
DFS的收益:

- [3] 当逐层减半的数据达到一定程度时,数据可以直接在寄存器中处理
- [4] 提前结束任务和高效复用寄存器
- [**5**] 增大 Cache 的击中率

整体遍历的收益:

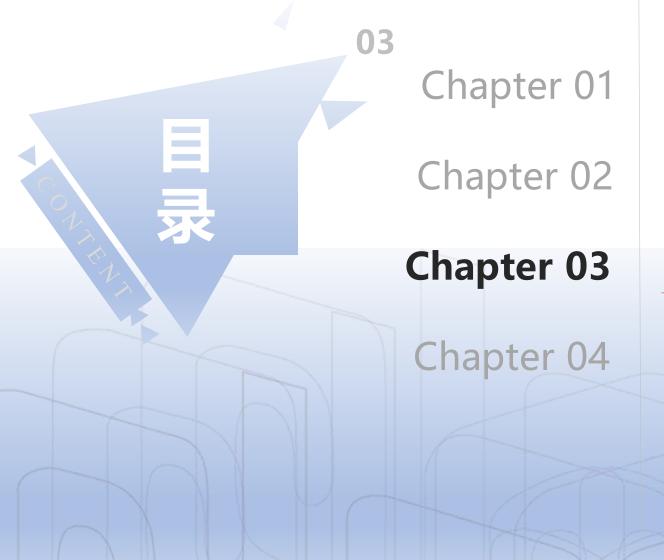
[6] 访问连续的内存空间

Benefit Scheme	[1]	[2]	[3]	[4]	[5]	[6]
SLM		√				
SDFS-NTT		√	√	√	√	
EDFS-NTT			√	√	√	√



旨在注重方案设计的合理性以及无 论从并行、内存和指令、探索最适 配于 GPU 的加速方案





背景知识

加速优化实现

评估

总结

实验测试



D 对 SLM做了 1+6,2+5,3+4,4+3,5+2,6+1,2+2+2+1,3+3+1的分层的数据吞吐测试;逐步增加数据量测试寄存器冗余

```
case 0:
      printf("TEST NTT SLM SPEED TEST\n");
     NTT SPEED TEST(g gridSize, g blockSize,g packSize);
     NTT ML1P6(g gridSize, g blockSize,g packSize);
     NTT_ML2P5(g gridSize, g blockSize,g packSize);
     NTT ML3P4(g gridSize, g blockSize,g packSize);
     NTT ML4P3(g gridSize, g blockSize,g packSize);
     NTT ML5P2(g gridSize, g blockSize,g packSize);
     NTT_ML6P1(g gridSize, g blockSize,g packSize);
     NTT_ML2P2P2P1(g gridSize, g blockSize,g packSize);
     NTT ML3P3P1(g gridSize, g blockSize,g packSize);
      break:
VTT native OUT TimeUsed: 8498.228880 ms, Throughput: 47274426.197419 ops/sec, Latancy: 0.549823 ms.
connectTest is 38728 in 38726
NTT GPU m CPU correctTest passed!
NTT MLIP6 OUT TimeUsed: 9936.648800 ms, Throughput: 30909636.803718 ops/sec, Latency: 8.993865 ms.
Checking resuts!!!
correctTest is 38728 in 38728
NTT_ML1P6 GPU = CPU correctTest passed!
NTT RL2P5 DUT TimeUsed: 8939.799000 ms, Throughput: 54363188.702565 ops/sec, Latency: 8.893980 ms.
Checking resuts!!!
correctTest is 39720 in 39720
NTT_ML2P5 SPU = CPU correctTest passed!
NTT ML3P4 DUT TimeUsed: 7772.317880 ms, Throughput: 39525481.825502 ops/sec, Latency: 8.777223 ms.
Checking resuts!!!
corectTest is 38720 in 38720
NTT ML3P4 GPU = CPU corectTest passed!
NTT ML4P3 DUT TimeUsed: 6726.127000 ms, Throughput: 45672643,489796 ops/sec, Latency: 8.672613 ms.
Checking resuts[[]
correctTest is 50720 in 30720
NTT_ML4P3 GPU - CPU correctTest passed!
NTT PLSP2 OUT TimeUsed: 6741.749888 ms. Throughput: 45566818.638298 ops/sec, Latency: 8.874175 ms.
Checking resutal!!
correctTest is 30720 in 30720
NTT MLSP2 BPU = CPU correctTest passed!
NTT HL6P1 DUT TimeUsed: 7011.396000 ms, Throughput: 43814384.467881 ops/sec, Latency: 8.781148 ms.
Checking resuts!!!
correctTest is 58720 in 38720
NTT MLSP1 GPU a CPU correctTest passed
VTT ML2P2P2P1 DUT TimeUsed: 5949.728866 ms, Throughput: 51632612.448838 ops/sec,
                                                                    atency: 0.594973 ms.
correctTest is 38728 in 38728
NTT RL2P2P2P1 GPU = CPU correctTest passed!
NTT MLSFSF1 OUT TimeSted: 6153.431000 ms, Throughput: 49921745.389023 ops/sec, Latency: 8.615363 ms.
Checking resurs!!!
correctTest is 38720 in 38720
NTT ML3F3P1 OPU = CPU correctTest passed!
```

```
case 1:
                                                                                            HTT HLDPS LOAD4 OUT Timelined: 8937.538660 ms, Throughput: 34375952.587776 ops/sec, Latency: 6.885753 ms.
        printf("TEST NTT SLM LOAD NUM SPEED TEST\n");
                                                                                            Ducking results!!!
                                                                                            correctTest is 50720 in 30720
                                                                                            MTT_MLSPS_LG4D4 GPU a CPU correctTest pessent
        NTT_SPEED_TEST(g_gridSize, g_blockSize,g_packSize);
                                                                                            HTT ML275 LOADS OUT Timelised: 8016.299000 ms, Throughput: 54453791 879007 operior; Latercy: 8.881628 ms.
        NTT ML1P6(g gridSize, g blockSize,g packSize);
                                                                                            Checking results!!!
                                                                                            correctTest is 30700 in 38716
                                                                                            UTT ML2PS LOADS OPU - CPU correctTest sessed!
        NTT_ML2P5_LOAD4(g_gridSize, g_blockSize,g_packSize);
                                                                                            NTT MICES LOADIS OUT Timeland: 9818 648660 ms. Throughout: $4892987 529146 005/int. Laterty: 8.961865 ms.
                                                                                            thecking results!!!
        NTT_ML2P5_LOADB(g_gridSize, g_blockSize,g_packSize);
                                                                                            correctTest in Series in Series
                                                                                            NTT_MLDPS_LDADGE OPU = CPU correctTest passed!
        NTT_ML2P5_LOAD16(g_gridSize, g_blockSize,g_packSize);
                                                                                            NTT MIZPS LOADEZ DIT Timebled 9007 880888 ms. Throughput: 85952724.822950 opr/set, Latercy: 8.928788 ms.
                                                                                            Checking results (1)
        NTT_ML2P5_LOAD32(g_gridSize, g_blockSize,g_packSize);
                                                                                            correctfest is surge in serge
                                                                                            NYT_HCRPK_LONDAID GPU - CPU correctTest gassed
        NTT ML2P5 LOAD64(g gridSize, g blockSize,g packSize);
                                                                                            MIT M.JFS Limbes Out Timesced 9958.173888 ms, Throughput: 82568324.563880 ops/sec, Latency: 8.853817 ms.
                                                                                            Checking Pessts !!!
        NTT_ML2P5_LOAD128(g_gridSize, g_blockSize,g_packSize);
                                                                                            correctTest is 36720 in 18720
                                                                                            HTT HUDGE LANDER SELL & CRU correctTest cassed)
        NTT ML3P4 LOAD8(g gridSize, g blockSize,g packSize);
                                                                                            NTT M.295 LDSD128 OUT Tameland: 18725.572600 ms, Throughput: 28681820.679120 dps/sec, Latency: 1.672557 ms.
                                                                                            Checking resutalli
        NTT_ML3P4_LOAD16(g_gridSize, g_blockSize,g_packSize);
                                                                                            correctTest is 36726 in 18720
                                                                                            NTT_NL3F1_LOADQ18 GPU = CPU torrectFast passed?
        NTT_ML3P4_LOAD32(g_gridSize, g_blockSize,g_packSize);
                                                                                            HTT ML3P4 LOADS OUT TimeUsed: 7776.338060 ms, Throughput: 39584455.696242 ops/sec, Latency: 0.777634 ms.
        NTT_ML3P4_LOAD64(g_gridSize, g_blockSize,g_packSize);
                                                                                            Checking results []
                                                                                            corectTest is 30720 in 30720
        NTT_ML3P4_LOAD128(g_gridSize, g_blockSize,g_packSize);
                                                                                            MIT_MLSP4_LDADS GPU = CPU corectTest passed!
                                                                                            NTT MLSP4 LDADIS OUT fimeUsed: 7772.284000 ms, Throughput: 20525162.800704 ops/sec, Latency: 0.777228 ms.
        NTT_ML4P3_LOAD16(g_gridSize, g_blockSize,g_packSize);
                                                                                            Checking resuts!!!
        NTT ML4P3 LOAD32(g gridSize, g blockSize,g packSize);
                                                                                            corectTest is 90720 in 30720
                                                                                            NTT HEAPA LOADIE SPU a CPU correctTest masked!
        NTT ML4P3 LOAD64(g gridSize, g blockSize,g packSize);
                                                                                            NTT ML3P4 (DADS2 OUT TimeUsed: 8257.245000 ms, Throughput: 37283692.050453 ope/ser, Latency: 8.625724 ms.
        NTT_ML4P3_LOAD128(g_gridSize, g_blockSize,g_packSize);
                                                                                            ESPECTTUSE IS 30720 In 30720
                                                                                            HTT_MLSP4_LDAD32 SPU - CPU corectTest passed!
        NTT_ML5P2_LOAD32(g_gridSize, g_blockSize,g_packSize);
                                                                                            NTT ML3P4 LOAD64 OUT TimeUsed: 8694.866800 mm. Throughput: 35334445.396177 ops/sec, Latency: 8.869487 ms.
        NTT ML5P2 LOAD64(g gridSize, g blockSize,g packSize);
                                                                                            Direcking results!!!
                                                                                            corectTest is 30720 in 30720
        NTT_ML5P2_LOAD128(g gridSize, g blockSize,g packSize);
                                                                                            MTT MLSP4 LDADG4 GPU s CPU corectTest massed!
                                                                                            MTT ML3P4 LDAD128 OUT fimeUsed: 16785.312000 ms, Throughput: 16483155.162175 ops/sec, Latency: 1.678532 ms,
        NTT_ML6P1_LOAD64(g gridSize, g blockSize,g packSize);
                                                                                            Checking resuta!!
                                                                                            corectTest is 90720 is 90720
        NTT_ML6P1_LOAD128(g gridSize, g blockSize,g packSize);
                                                                                           NTT ML3P4 LOAD128 GPU = CPU corectTest passed!
        NTT ML2P2P2P1 LOADNUM TEST(g gridSize, g blockSize,g packSize);
        break;
                                                                                                                       WTT 9L592 (0600) OUT Timested: $740.368000 on. Throughout $400000 800000 scotted. Letests: $.874000 co.
                                                                                                                        Decking results (1)
NTT ML4P3 LOAD16 OUT TimeUsed: 6725.024000 ms, Throughput: 45650134.375729 ops/sec. Latency: 0:672502 ms.
                                                                                                                       ancescitude ha sures in sures.
Checking resutal!!
                                                                                                                        NTT_PLSF2_LOADS2 ONL = OPE correctFoot percent.
                                                                                                                        MTT M.SF2 109D84 ONT Timeland: TWIT: 200000 ms, Throughput: 40121097 [78418 opc/set, Labourg: 9.763725 ms.
correctTest is 30720 in 30720
                                                                                                                       Checking retain(1)
NTT ML4P3 LOAD16 GPU = CPU correctTest passed!
                                                                                                                        increations in series in series
                                                                                                                        ST 16 SEC 176006 SEC 4 TEV connectifiest manual
NTT ML4P3 LCAD32 OUT TimeUsed: 7417.180800 ms, Throughput: 41417358.079486 ops/sec, Latency: 0.741718 ms.
                                                                                                                        077 MLSF2 LORCER DOT Time/sed: LANG.478000 mg, Toroughput: 28422235.000005 upr/sec, Lanescy: L.888046 mg.
Checking resuts!!!
                                                                                                                        percentlant is owner to settle
correctTest is 30720 in 30720
                                                                                                                        HTT HUSES LONDOUGH DRV - CRU correctTest persent
NTT ML4P3 LOAD32 GPU = CPU correctTest passed!
                                                                                                                       WIT PLEYS LIASSA OUT TIMESHAY! NOT COMM MI, Throughout: 40775296. MOXRB gal/sec, Catency: 6.76CNF mi.
NTT ML4P3 LOAD54 OUT TimeUsed: 8882.184860 ms, Throughput: 38009528.117647 ops/sec, Latency: 8:888218 ms.
                                                                                                                        Chebbe reputation
Checking resuts!!!
                                                                                                                       serventTest 15 38729 in 30728
correctTest is 38728 in 38728
                                                                                                                        WIT JUSTS USASSA OFC - ON correct lest passed
NTT ML4P3 LOAD64 GPU + CPU correctTest passed!
                                                                                                                       NOT PLEFE LIABILE OUT TIMELED LEGELS/1988 m., Throughput: MICHESS 198765 ($1/44), Libercy: LASSET m.
NTT ML4P3 LOAD128 OUT TimeUsed: 18828.548000 ms, Throughput: 28390429.024482 ops/sec, Latency: 1.082055 ms. Chailing restall?
Checking resutal!!
                                                                                                                        correctfact is 19729 in 19719
                                                                                                                        WIT JUSPS_LEARLISE BPU + CPG convectifiest pessed
correctTest is 30720 in 30720
NTT ML4P3 LOAD128 GPU = CPU correctTest passed!
```

实验测试



● 对SDFS、EDFS都做了1+6, 2+5, 3+4, 4+3, 5+2, 6+1, 2+2+2+1, 3+3+1的分层的数据吞吐测试; 最终选择EDFS进行Kyber性能测试

SDFS 性能测试

```
case 2:
    printf("TEST NTT_SlicedTREE_SPEED_TEST\n");
    NTT_ML1P6_TREE(g_gridSize, g_blockSize,g_packSize);
    NTT_ML2P5_TREE(g_gridSize, g_blockSize,g_packSize);
    NTT_ML3P4_TREE(g_gridSize, g_blockSize,g_packSize);
    NTT_ML4P3_TREE(g_gridSize, g_blockSize,g_packSize);
    NTT_ML5P2_TREE(g_gridSize, g_blockSize,g_packSize);
    NTT_ML6P1_TREE(g_gridSize, g_blockSize,g_packSize);
    NTT_ML2P2P2P1_TREE(g_gridSize, g_blockSize,g_packSize);
    NTT_ML3P3P1_TREE(g_gridSize, g_blockSize,g_packSize);
    break;
```

```
TREE NTT MLIP6 OUT TimeDied: 5954.062000 ms, Throughput: 51509829.951725 ops/sec, Latency: 0.595406 ms,
Checking results!!!
correctTest is 30728 in 38728
NTT PLIPS THEE GPU - CPU correctTest passed!
TREE NTT ML2PS OUT TimeUsed: 5384.267000 ms, Throughput: 57055120.037095 ops/sec, Latency: 0.538427 ms.
Checking results [1]
correctTest is 30720 in 30720
NTT PLOPS THEE GPU - CPU correctTest passed!
TREE NTT ML3P4 GUT TimeUsed: 5374.934800 ms, Throughput: 57154190.172382 opt/sec, Latency: 0.537493 ms.
Checking resuts!!!
corectTest is 10720 in 30720
NTT_PL3P4_TREE GPU - CPU corectTest passed!
TREE HTT ML4P3 OUT TimeUsed: 6245.925800 ms, Throughput: 49184868.012344 ops/sec, Latency: 0.624592 ms.
correctTest is 30728 in 38728
NIT FLAPS TREE GPU - CPU correctTest passed!
TREE NTT MLSP2 OUT Timelsed: 5270.103000 ms, Throughput: 58291000.838458 ops/sec, Latency: 0.527018 ms.
Checking results!!!
correctTest is 38728 in 38728
NTT PLSP2 TREE GPU - CPU correctText passed!
TREE NTT MLGP1 OUT TimeUsed: 5340.398888 ms, Throughput: 57427188.511438 ops/sec, Latency: 8.534039 ms.
Checking results[1]
correctTest is 38728 in 38720
NTT_PLOPI_TREE GPU = CPU correctTest pessed!
TREE NIT ML2929291 OUT TimeUsed: 5872.918888 ms, Throughput: 68556958.4 2142 opt/sec, Latency: 6.587291 ms.
Checking resuts!!!
correctTest is 30720 in 30720
NTT_PL2P2P2P1_TREE GPU = CPU correctTest passed!
TREE NTT ML393P1 OUT TimeUsed: 5480.066000 ms. Throughput: 56057710.012873 ops/sec. Latency: 0.548007 ms.
Checking results!!!
correctTest is 38728 in 38728
```

EDFS 性能测试

case 3:

correctTest is 38730 in 38730

NTT_SPSP1_TREE SPU a CPU correctTest passed!

```
printf("TEST NTT EntireTREE SPEED TEST\n");
       NTT 1P6 TREE(g gridSize, g blockSize,g packSize);
       NTT_2P5_TREE(g gridSize, g blockSize,g packSize);
       NTT 3P4 TREE(g gridSize, g blockSize,g packSize);
       NTT_4P3_TREE(g_gridSize, g_blockSize,g_packSize);
       NTT_SP2_TREE(g_gridSize, g_blockSize,g_packSize);
       NTT_6P1_TREE(g_gridSize, g_blockSize,g_packSize);
       NTT_2P2P2P1_TREE(g_gridSize, g_blockSize,g_packSize);
       NTT 3P3P1 TREE(g gridSize, g blockSize,g packSize);
       break;
TEST WIT THEE SPEED TEST
WIT 7602 OUT Timeland: 4718.175000 ms, Throughput: 45107160.033074 opt/zec, Latency: 0.471838 ms.
Checking resutall!
correctTest is 38738 in 38738
ATT TREE GPU a CPU conventTest passed!
TREE HTT 196 Out Timetoed: 4405.621000 ms, Throughput: 88253162.426281 opt/oot, Latercy: 8.440622 ms.
correctTest is 10720 in 10720
HTT IPG TREE GPU + CPU correctTest massed!
TREE NIT 295 OUT Timeline: 4862.420000 mg, Throughput: 82000200.145888 opp/sec, Latercy! 8.485242 mg.
Chathing resutatil
connectfact to 18719 to 18719
ATT 2PS TREE GPU a CPU correctTest opened!
TREE NIT 394 OUT Timelised: 6661.000000 ms., Throughput: 88600777.146563 opi/wet, Latency: 2.506181 ms.
Checking repotatil
oprectTest is $6726 in $6728
HTT_1PA_TREE GPU + CPU corectTeut passed!
THEE NIT 4P3 OUT Timelood: $836.226800 mg. Throughput: $8000000 878305 opines, Latercy: 8.583623 ms.
Checking results!!
correctTest is 38720 in 38728
ATT_4P3_TREE GPU = CPU correctTest passed!
TREE NIT 592 OUT Timeload: 4660.667990 ns., Throughput: 60055941.137130 ops; sec., Latency: 2.463047 ms.
Checking resots 11/
correctTest is 10720 in 30720
HTT_SP2_TREE GFW + CPU correctTest massed!
TREE WIT 671 OUT Timelised: 4811.285000 mg. Throughput: 83800000.256800 ops/sec, Latercy: 8.481120 ms.
Charleing respectiff
correctTest in $2710 in $2710
NTT SPI TREE SPU = CPU correctTest gassed!
TREE NTT 2F2F2F1 DUT Timeused: 5445.555800 ms, Throughput: 56412862.600733 ogs/swc, Latency: 0.544555 ms.
Checking recetail)
correctTest is 10710 in 10710
HTT 2P2P2P1 THEE SPU is CPU correctTest passed!
TREE MIT SP3P1 OUT TimeLoad: $180.000000 ms, Throughput: $0004818.000137 oph/sec, Latency: 0.518800 ms.
```

```
Kyber 性能测试
               common des millartes per rache ser samuts, qui esperter rache ser reports, que millarter turn ? hi, pubble penettiment in-
          pro-limit to the total persons their
                  intra pagainte ogni collecto gris sina pagainten dont sino ogni, ottorro gospi, gas follocco gospia, pagaintero gospia
          contributed a contribution of the relation of the contribution of 
           experience and first houseast. Man, recognist. Manager, Labour, Manager, Manager, and St.
                      timetred, butl. Company / test Incomin
          property and religious as, and religious as an adjusted that I have party neutronyment I constructed that a relative party
           indicated the collected of the collected and the pre-collected tests ? CORT DECK DESTRUCTORY ? signed contact the collected tests and the collected tests and the collected tests and the collected tests and the collected tests are collected to the collected tests and the collected tests are collected to the collected tests are collected to the collected tests and the collected tests are collected to the collected tests are collected tests are collected tests are collected to the collected tests are collected tests.
   nale have you see you haven't
           the Care & e. for E i beed dealers are
                    may according to the grid size, go, softener than according a filterer go, at, go, softener ga, racke hat, go, softener ga
                      Court of the Control of the Control
            comis timenat - recolation, real stanto per - errorei prento per + comismos pell stanto per - errorei stanto per - errorei stanto per -
          Note - deciletings, collecter hash - text bridge - new A timescopy
print; Wat May and Tanasant M. etc. broughter M operior, Learner M. etc., "
"Exempt, Volta, Learner / Note, Darrier,"
           common per reflective at, any millioning and at, any officially their discontinuous and applicable of the commonwealth
   end hertigocains, bellinerally
                  tokowanowanialista goldzin, goldzini diedzini ogranilista goldzin goldzin goldzini goldzini.
           permissionic base real area, marti-
           could be sent a contract or real story to our . See real stort to our a contract out your story to our . See real story to our . I this
          Tartit + triodiet pp. diffector, facts * hell feature. * liter / Themsel
                      Descript, Sport, Resided J took Scotters
          ment the indemning policitative, policitative gains and other transferon term there are no continuous days, access
Select compare pie keypair of host and device
GPU GEN DUT TimeUsed: 30389.423600 ms, Throughput: 1460410.688756 opx/sec, Latency: 19.144711 ms.
CPU GEN OUT Throughput 18561.764 op/sec, Latency: 0.894581 ms.
 Select compare like ent of host and device
SPU SMC CNT TimeUned: $4600.026000 mg, Throughput: [1775250.436607 ops/set] Latency: 17.304511 mg.
CPU ENC OUT Throughput 9329.885 op/sec, Latancy: 0.167192 ml.
 Select compare pay dec of host and device
DPU DEC DUT TimeUsed: 7685.427880 es, Throughput: 7984558.800987 ops/set, Latency: 5.842713 ms.
key exchange 1336382.826698 ops/sec
```

分层为 2+2+2+1 时, SDFS 性能最好

分层为 1+6 时。EDFS 性能最好

选取 EDFS 作为 NTT实现方案,测试 Kyber 性能

OPU ENC OUT Throughput 46524.528 op/sec, Latency: 8.821454 mi.

实验结果-三种 NTT 实现方案的性能



● 对 SLM、SDFS、EDFS都做了 1+6, 2+5, 3+4, 4+3, 5+2, 6+1, 2+2+2+1, 3+3+1 的分层的数据吞吐测试

TEST: 10000次取平均值

NTT基于GPU的原始吞吐 48318 kops/s

1. SLM于

SLM 最优实现 51924 kops/s <mark>提升 7.5%</mark> 寄存器冗余测试

以 SLM 中性能最优的分组 2+2+2+1 为 例

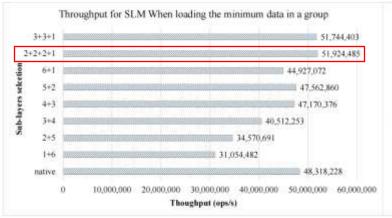
每组中数据加载量最少时, 性能最优 寄存器没有冗余

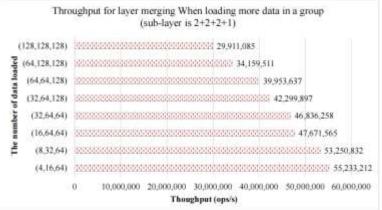
2. SDFS

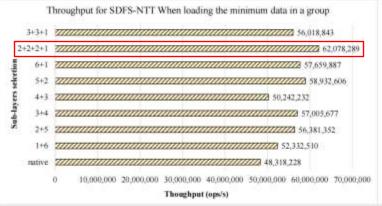
SDFS 最优实现 62078 kops/s 提升 28.5%

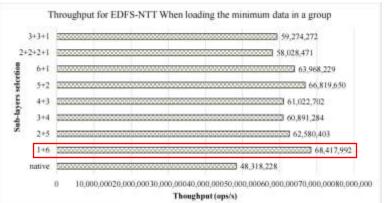
3. EDFS

EDFS 最优实现 68417 kops/s <mark>提升 41.6%</mark>









实验性能-HI-Kyber与其他工作对比

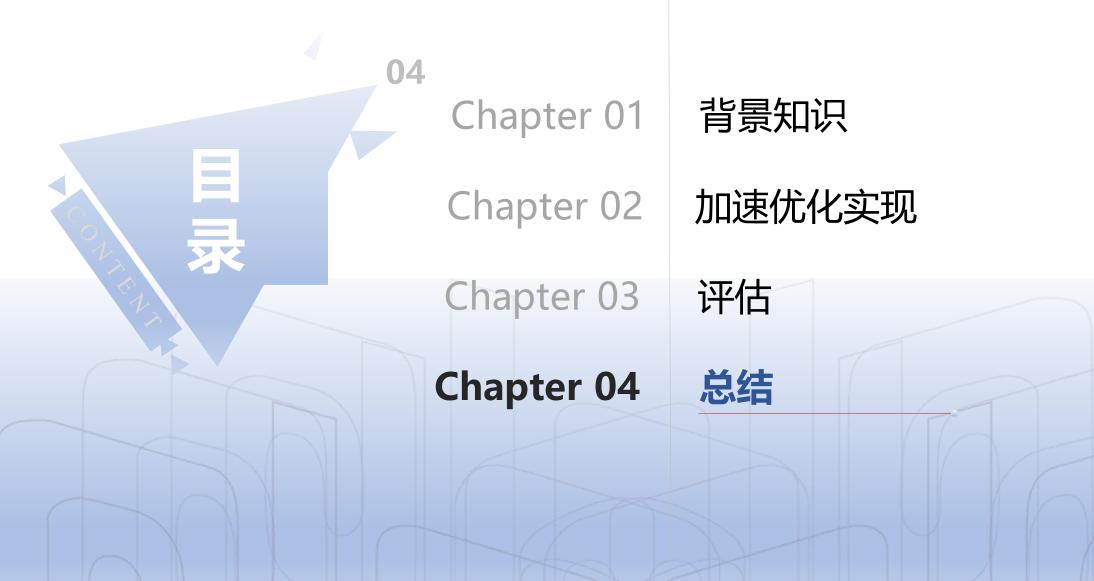


在相同的 GPU 平台上,HI-Kyber 的性能是 [4] 基于相同指令集 GPU 实现的 3.52 倍; 是 [5] (目前最优性能实现) 基于 AI 加速器 Tensor Core 实现的 1.78 倍

	平台	KeyGen/(ops/s)	Encryption/(ops/s)	Decryption/(ops/s)	Perf/(KX/s)
Sanal et al. [1]	Apple A12 2-cores Vortex at 2.490 GHz and 4-cores	26,157	26,774	27,360	13
Xing et al. [2]	Xilinx Artix7	17,182	14,728	11,600	7
C-Ref. [3]	Intel Core i7-4770K 3.5 GHz(Haswell), 4 Cores	11,390	10,101	8,826	5
AVX2-Ref. [3]	Intel Core i7-4770K 3.5 GHz(Haswell), 4 Cores	47,591	35,962	44,232	23
Gupta et al. [4]	NVIDIA Volta V100 GV100, 5120 CUDA cores	-	-	-	473
L. Wan et al. [5]	NVIDIA GeForce RTX 3080	1,250,000	1,298,701	2,380,952	820
Our work (EDFS-NTT)	NVIDIA Titan V Volta GV100, 5120 CUDA cores	1,639,949	1,763,898	7,885,157	1,358
	NVIDIA Tesla V100 Volta GV100,5120 CUDA cores	2,022,552	2,105,977	9,378,937	1,664
	NVIDIA GeForce RTX 3080	1,916,227	2,204,866	6,091,689	1,458
	NVIDIA Jetson AGX Xavier	160,130	187,832	539,450	123

同样适于 嵌入式 GPU







实验总结 - 基于 GPU 的后量子密码 Kyber 加速技术





基于 GPU 的后量子密码 Kyber 加速技术设计

- I. 构建 Kernel 融合架构 减少对全局内存的访问
- II. NTT 算法设计 GPU 寄存器的复用和减少对全局内存的访问
 - I. SLM 提升 7.5%
 - II. SDFS 提升 28.5%
 - Ⅲ. EDFS 提升 41.6%
- Ⅲ. 测试了**不同并行规模**下的 Kyber 最终性能吞吐以确定最佳并行参数(块数 = 80, 线程数 = 384, 寄存器数量 = 168)

最终密钥交换次数可达到 1358 kops/s

在相同的 GPU 平台上

- ✓ 是相同指令集架构 GPU 实现的 3.52 倍 (TPDS 2020 Gupta 等人的工作)
- ✓ 是基于AI加速器 GPU 实现(目前最优实现)的 1.78 倍(ESORICS 2022 Wan 等人的工作)



Thank you for listening