加速库在业务性能调优中的应用

薛永辉

HUAWEI

目录

HUAWEI

计算机为什么这么慢?

问题：设有一对新生的兔子，从第三个月开始他们每个月都生一对兔子，新生的兔子从第三个月开始又每个月生一对兔子。按此规律，并假定兔子没有死亡，1年后共有多少个兔子？10年后，又如何?

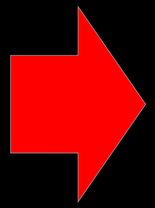
实质：斐波那契数列（0，1，1，2，3，5，…）

计算机的表现：观察一下不同的算法在1 GHZ的计算机下的表现

```
int64_t fib(int n) {

    if (n < 2) {
        return (int64_t)n;
    }

    return fib(n - 1) + fib(n-2);
}
```

令n=64，需要1天才能计算出来
令n=90，需要3世纪才能计算出来

时间复杂度 $O(2^n)$



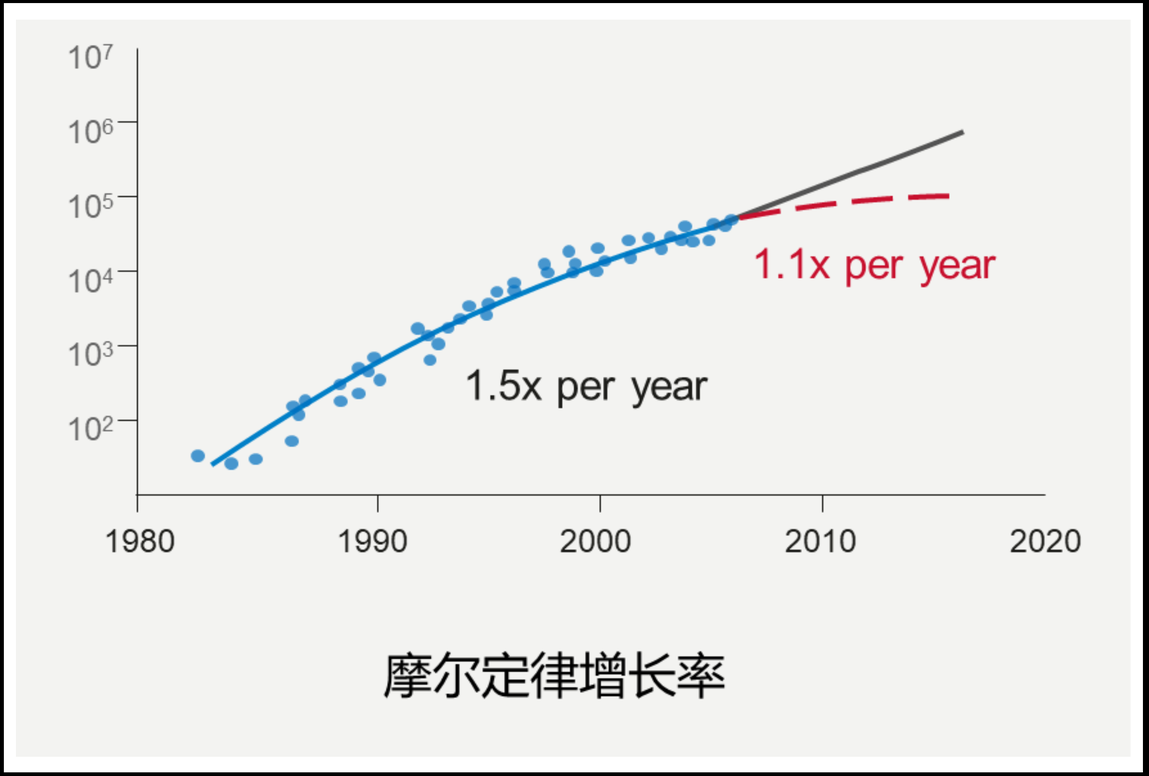
```
int64_t fib(int n) {
    if (n < 2) {
        return (int64_t)n;
    }
    int64_t f =0, g = 1;
    while(0 < n--) {
        g = g + f;
        f = g - f ;
    }
    return g;
}
```

令n=64或n=90，计算过程均无感知

时间复杂度 O(n)

HUAWEI

加速库开发所能带来的价值



加速库实践的意义

- 加速库提升软件计算效率
  加速库通过改进软件实现流程或算法，从而充分利用芯片计算能力，提升代码执行效率，使用户获得更好的性价比。

- 加速库重构摩尔定律
  软&硬件加速库的应用使得摩尔定律增长率继续有效。

- 加速库是软件系统工程的重要组成部分
  IT系统是多组件的，不可分割的，任何一个组成部分的改件对于整体性都会带来改进，而加速库在其实的提升作用更加重要。

加速库，即高性能软件库，是应用软件性能调化的最佳实践方法之一。

HUAWEI

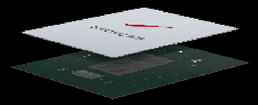
加速库是解决方案的基石

金融　互联网　运营商　政府　电力　交通　教育　…

大数据　分布式存储　数据库　HPC　ARM原生　AI　虚拟化

鲲鹏加速库

操作系统


处理器


SSD


鲲鹏主板


网卡


AI加速卡

HUAWEI

目录

- 加速库能够带来什么样价值
- **加速库开发的技术概览**
- 探索硬件加速的设计思路
- 揭开高性能软件编码的秘密
- 加速库在业务调优中的方法

HUAWEI

加速库开发技术栈

HUAWEI

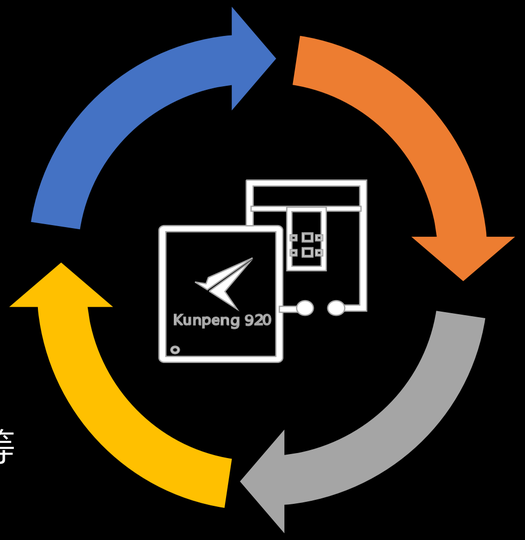
加速库开发的方法

软件基本库性能验证

- 通用Bench工具性能验证测试
- 基于软件原理定制化性能验证工具
- 解决方案集成性能验证测试

软件基础库编码调优

- NEON向量化，循环展开，分支预测等
- 数据预取，数据对齐，结构重排等
- 流水线，分支预测等



软件基础库性能瓶颈分析

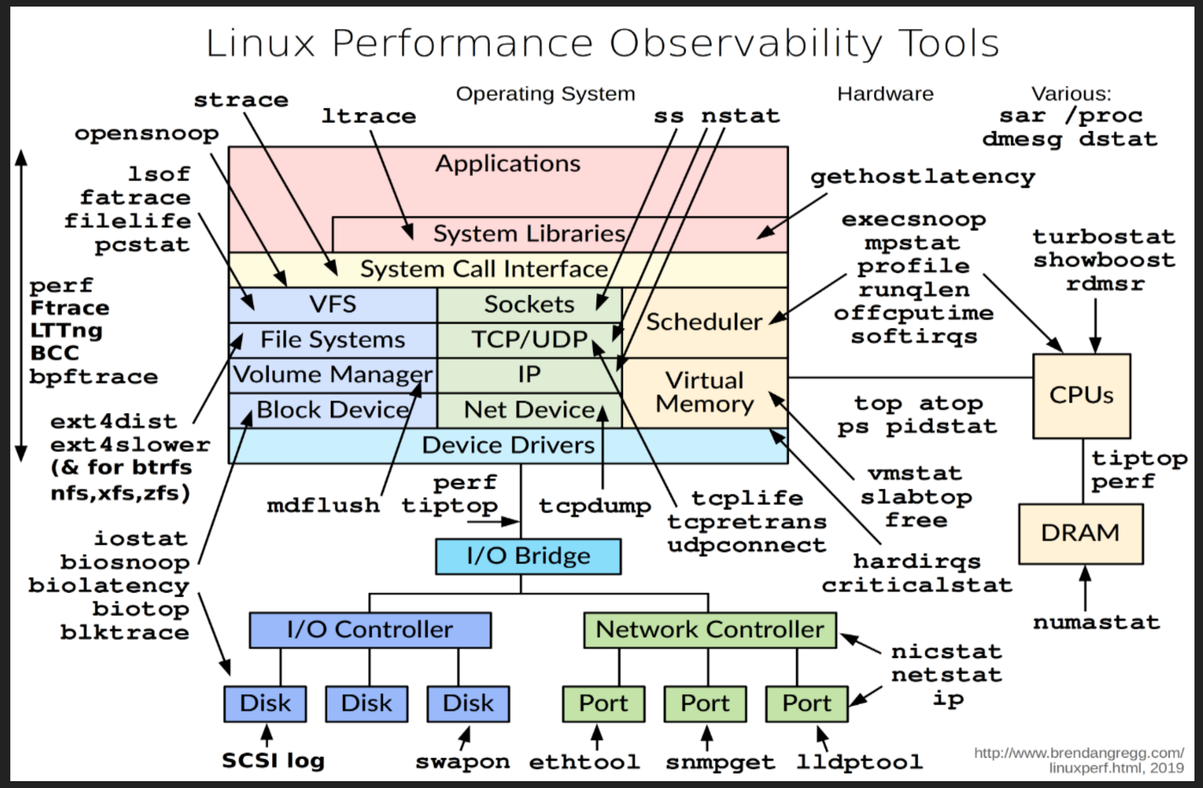
- 通过性能分析工具集识别热点函数，性能瓶颈
- 通过硬件平台与软件架构，识别潜在瓶颈点
- 通过分析软件代码实现，识别低效代码

软件基础库调优算法设计

- 基于软件算法的优化设计
- 基于存储结构的优化调计
- 基于软件运行场景的优化调设计

HUAWEI

工欲善其事，必先利其器



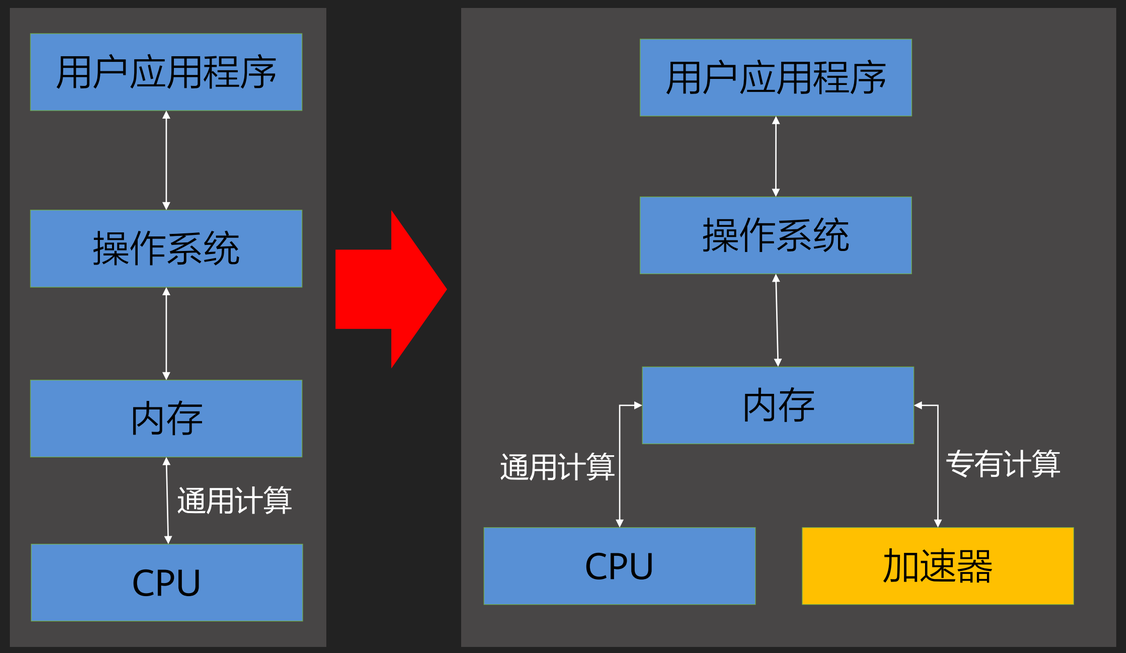
HUAWEI

目录

HUAWEI

探索硬件加速的有效方法

硬件加速架构



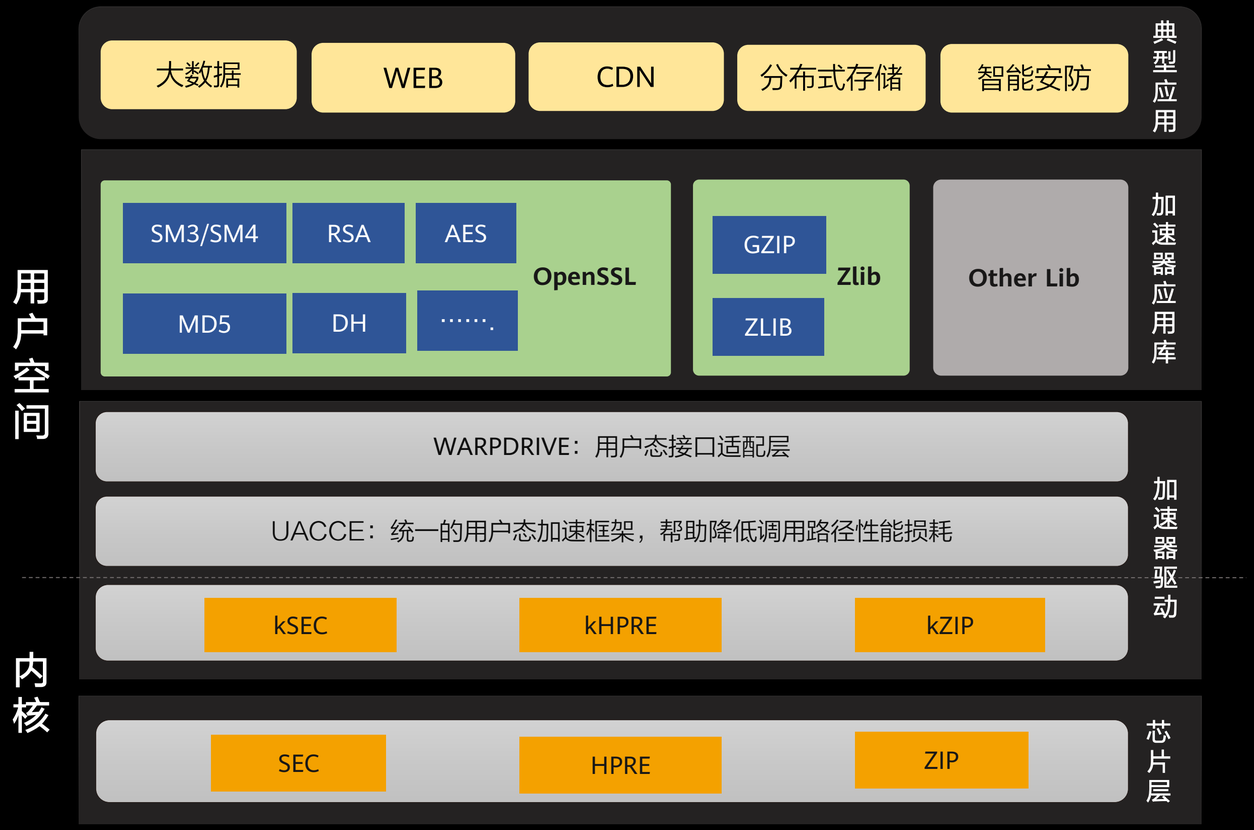
硬件加速关键方案

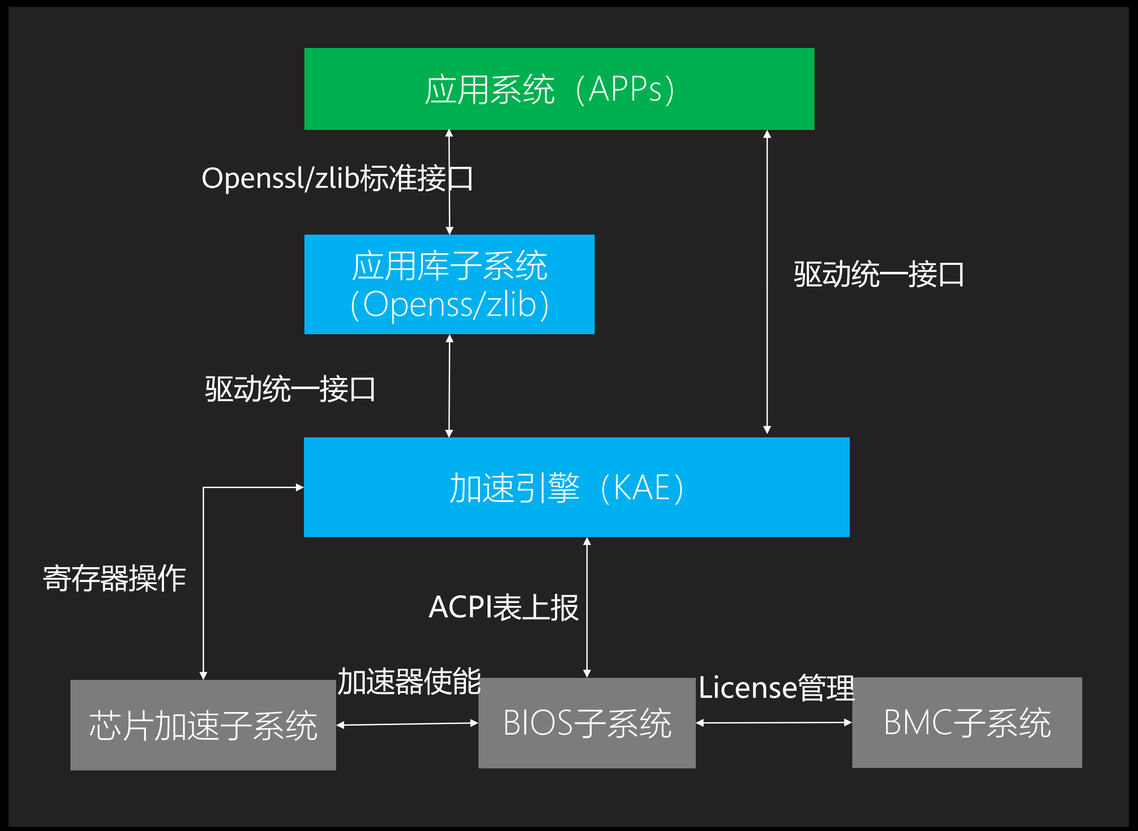
- 芯片面积有限，哪些能力有可做到加速器中？
  性能影响较大的，调用频率较高的算法。
- 识别到的算法，是完整的进行硬化吗？
  不是， 提取算法中的关键要素
- 加速器主要是硬件支持，软件无需做任何工作？
  基于UIO完成驱动开发；
  池化进行资源调用；
  数据切片，匹配硬件资源；
  与业界通用软件接口集成；

HUAWEI

鲲鹏加速引擎介绍

- 基于华为自研UACCE加速器内核框架，避免传统内核调用方式的路径损耗
- 100%继承OpenSSL和Zlib接口，应用层代码逻辑无需调整，使用简单

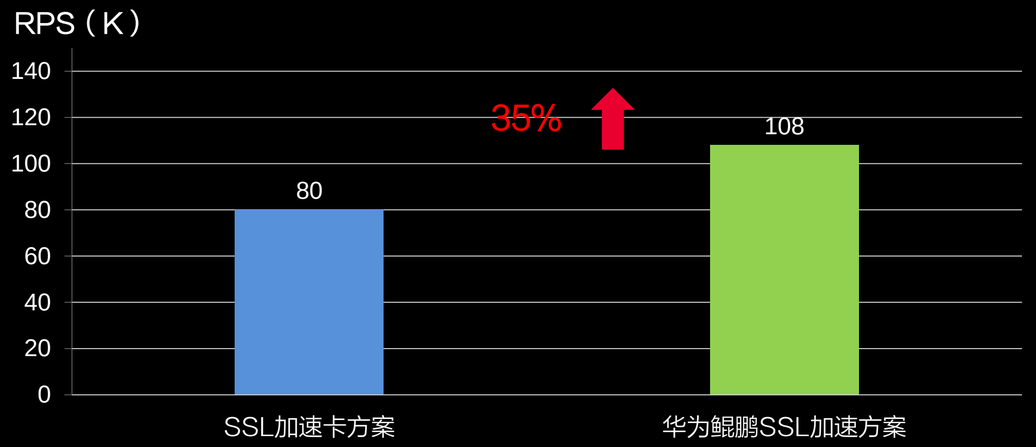




HUAWEI

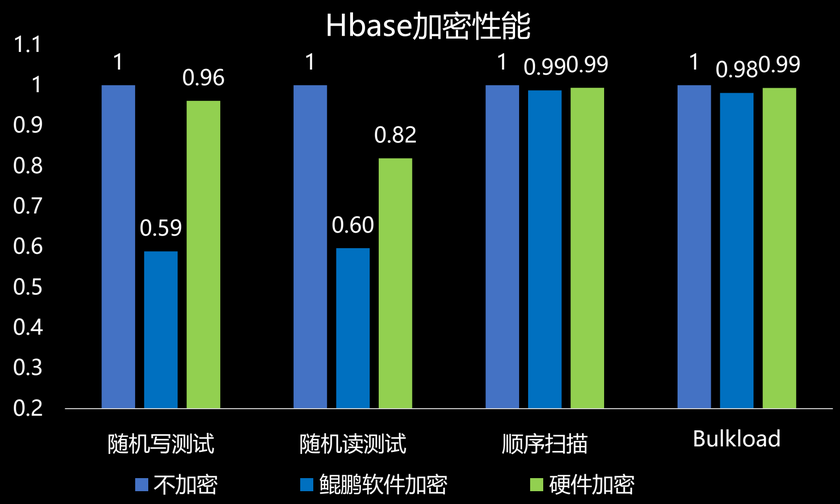
鲲鹏加速引擎优化效果

Web应用：使用RSA硬加速Nginx，相比主流加速卡性能提升35%

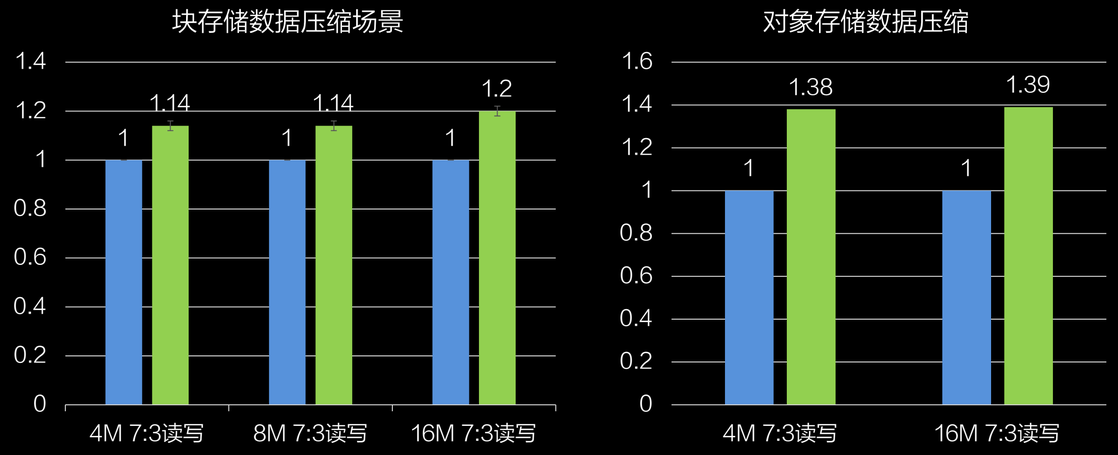


备注：由于影响因素较多，具体性能数据以环境实际测试为准，该性能数据仅供参考

大数据：启用数据安全加解密，CPU性能损耗< 5%



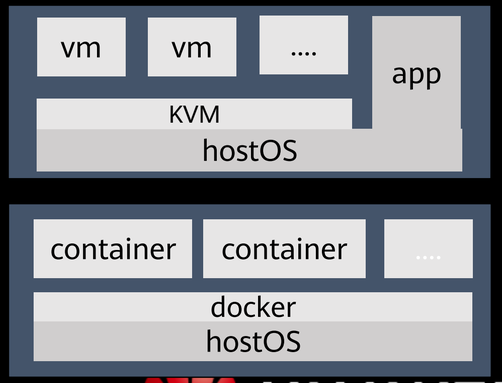
分布式存储：混合读写（7:3）带宽性能提升最高40%



云平台：支持SR-IOV直通，虚拟化&容器场景下物理机加速效果

关键规格

- 支持SR-IOV
- 支持Host与VM混合部署场景下使用加速器
- 支持容器场景下使用
- 加速通道分配支持Numa自适应
- 虚拟机PF/VF队列配置可迁移

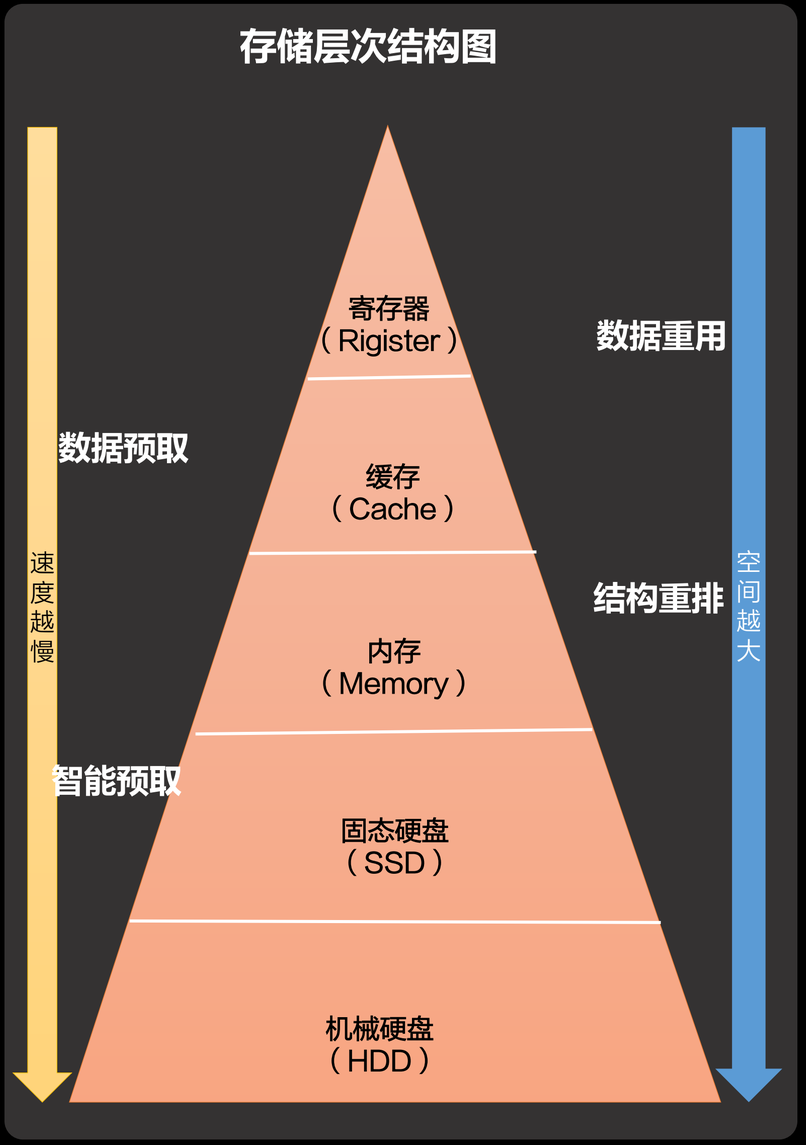


HUAWEI

目录

HUAWEI

如何通过解决IO（访存）瓶颈提升性能



典型优化代码

```
stuct List {
    char  MyID;              32位机：12bytes
    struct List* next;       64位机：24bytes
    char  Alias;
}
```

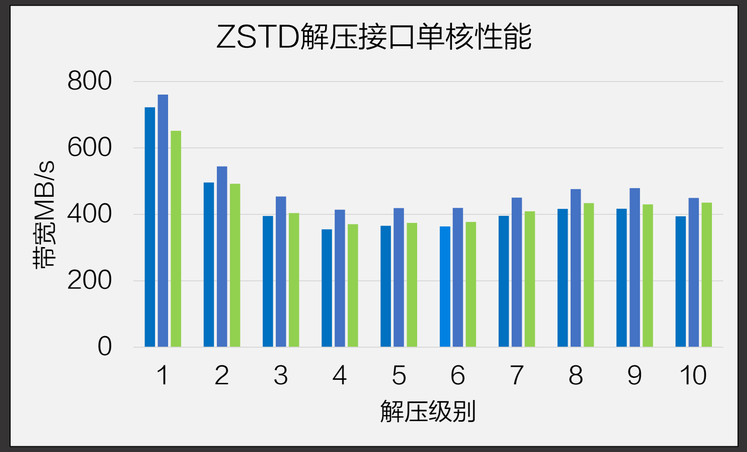
结构重排

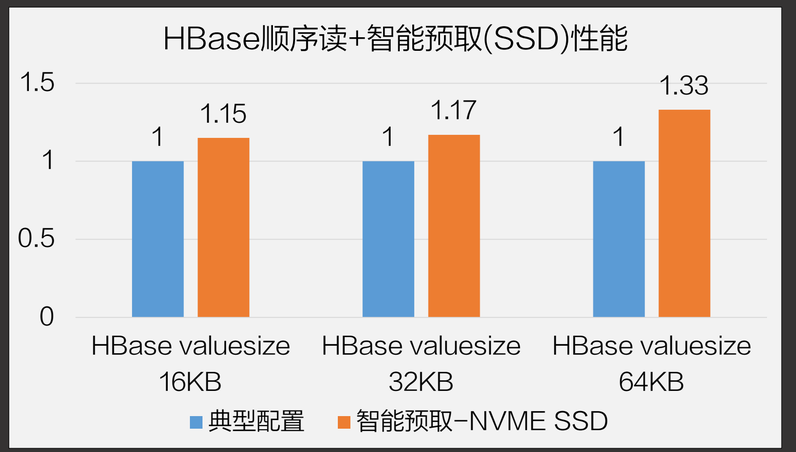
```
stuct List {
    char  MyID;              32位机：8 bytes
    char  Alias;             64位机：16bytes
    struct List* next;
}
```

```
static inline void Prefetch(const void* data)
 {
  __asm__ __volatile__(
    "prfm PLDL1STRM, [%[data]]          \n\t"
    :: [data] "r" (data));
}
int Vector_Add(int *dst, int *src1, int *src2, int size)
{
  for (int index =0 ; index < size; index += 4) {
     Prefetch(src1 + 256);
     Prefetch(src2 + 256);
     ...
  }
}
```

数据预取

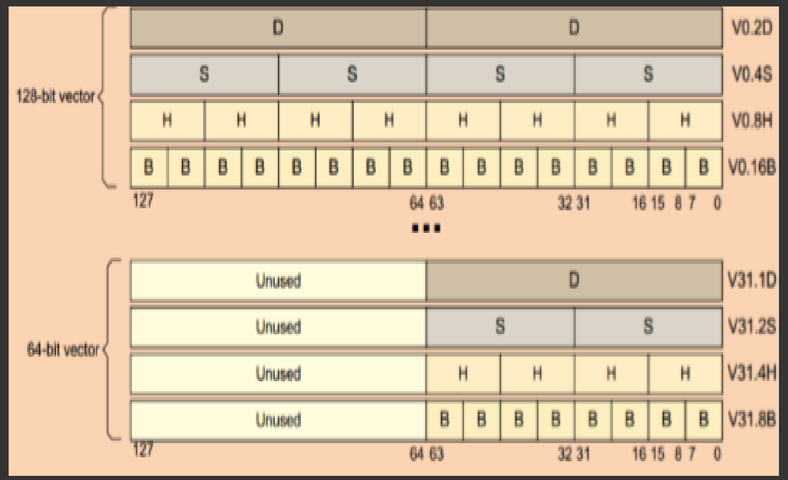
优化效果

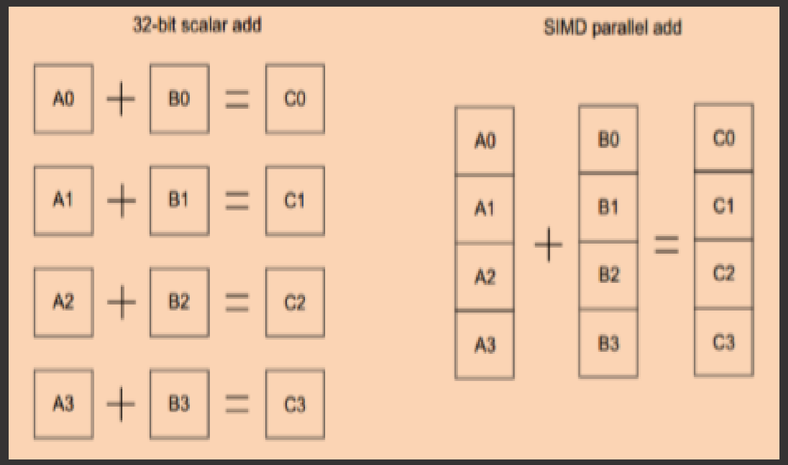




如何改进大规模数据的计算效率

硬件设计



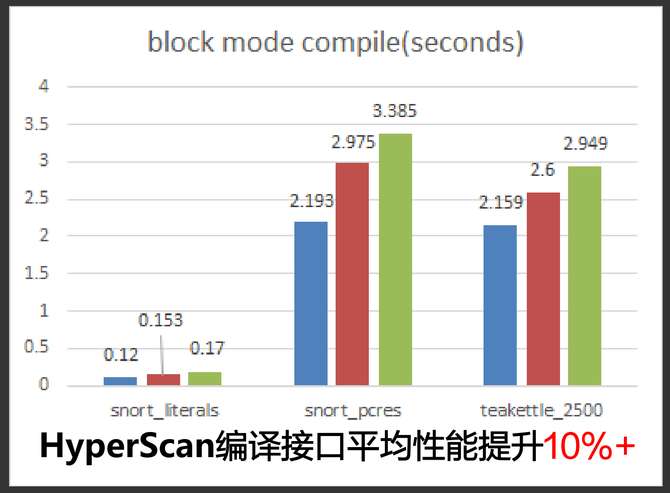


典型优化代码

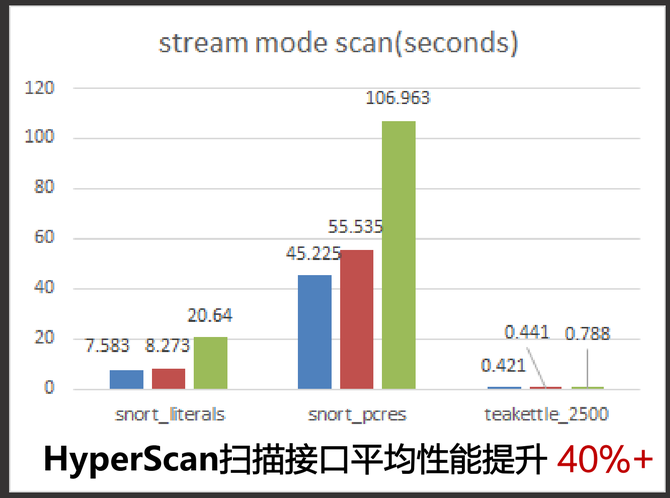
```
int Vector_Add(int *dst, int *src1, int *src2, int size)

{

        for (int index =0 ; index < size; index += 4) {

              dst[index] = src1[index] + src2[index]

        }

}
```

```
int Vector_Add(int *dst, int *src1, int *src2, int size)

{

        for (int index =0 ; index < size; index += 4) {

            int32x4_t vsrc1 = vld1q_s32(src1 + index);

             int32x4_t vsrc2 = vld1q_s32(src2 + index);

             int32x4_t vdst = vaddq_s32(vsrc1, vsrc2);

              vst1q_s32(dst+index, vdst);

         }

}
```

优化效果



HyperScan编译接口平均性能提升10%+

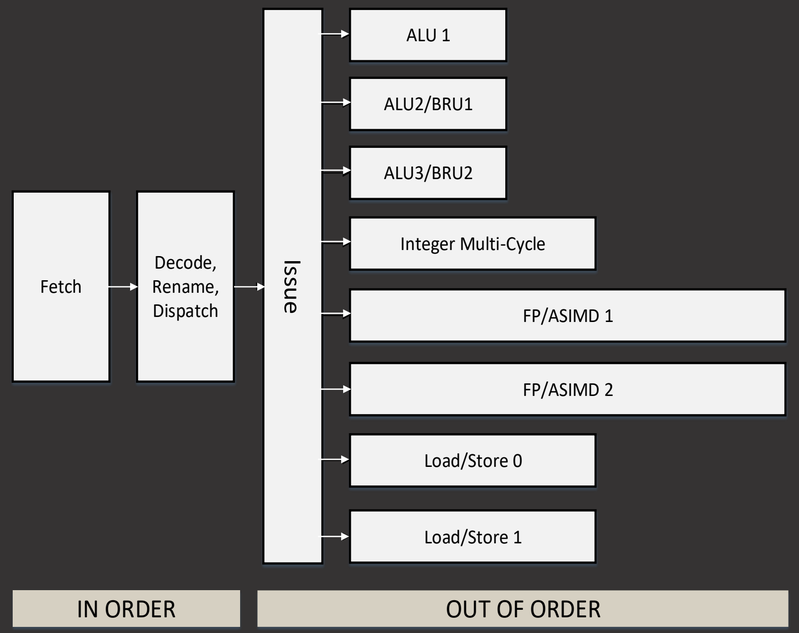


HyperScan扫描接口平均性能提升 40%+

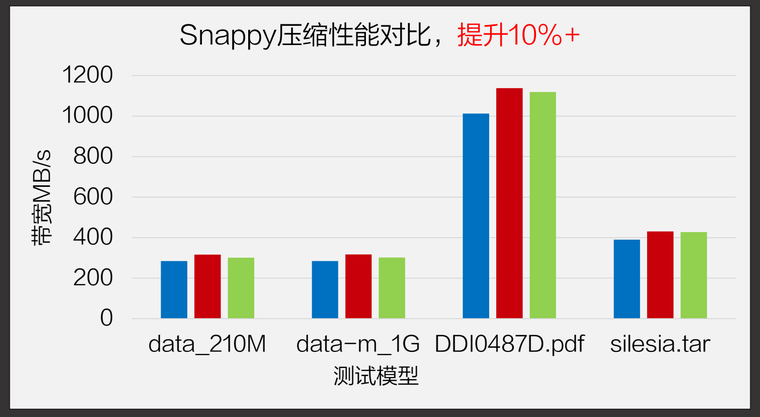
HUAWEI

如何通过改善流水线提升性能

流水线硬件示意图

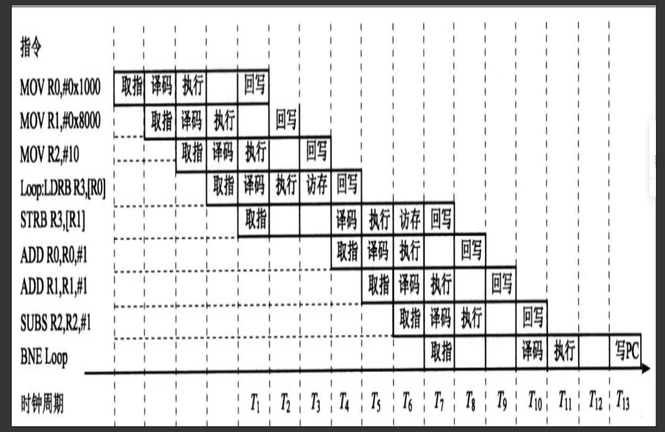


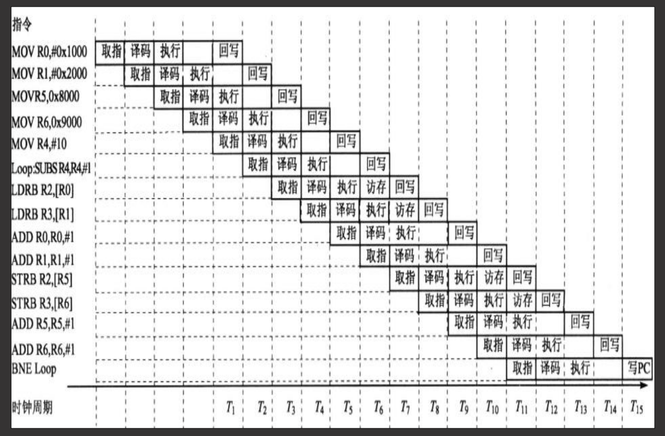
优化效果



优化代码

把内存地址0x1000和Ox2000处的数据分别拷贝到0x8000和0x9000处。
0x1000内容:1,2,3,4,5,6,7,8,9,10
0x2000内容:H, e, l, l, o, W, o, r, l, d






```
int  array[10000];
int threshold = 4000;
for(int index = 0; index <  10000; ++index) {
    if(array[index] < threshold) {
        DoSomething();
    }
}
```

场景1：array数据有序排列，执行代码；

场景2：array数据乱序排序，执行代码。

```
#define likely(x) __builtin_expect(!!(x),1)
#define unlikely(x) __builtin_expect(!!(x),0)
if (likely(x)):
        告诉编译器x表达式更有可能是真的if
(unlikely(x)):
        告诉编译器 x表达式更有可能是假的
```

如何通过算法优化提升性能

算法优化：减少计算指令

原始实现

```
float theta = 2 * PI * freq;
for(int index = 0; index < N; ++index) {
    float cos_y = cos(x);
    float sin_y = sin(x);
    //using cos_y, sin_y do something;
    x += theta;
}
```

阶段1优化：减少复杂计算的频度

```
float theta = 2 * PI * freq;
float cos_theta = cos(theta);
float sin_theta = sin(theta );
float last_cos_y = 1.0;
float last_sin_y = 0.0;
for(int index = 0; index < N; ++index) {
   float cos_y = last_cos_y * cos_theta - last_sin_y * sin_theta;
   float sin_y = last_sin_y * cos_theta + last_cos_y * sin_theta;
   //using cos_y, sin_y do something;
   last_cos_y = cos_y;
   last_sin_y = sin_y;
}
```

阶段2优化：提取同类项{x * y + p * q => (x + z) * y}

```
float theta = 2 * PI * freq;
float cos_theta = cos(theta);
float sin_theta = sin(theta );
float last_r = last_i = 0.0;
for(int index = N-1; index >= 0; ++n) {
   float cur_i, cur_r;
   cur_r = x[index] + last_r * cos_theta + last_i * sin_theta;
   cur_i = x[index] + last_i * cos_theta - last_r * sin_theta;
   last_i = cur_i;
   last_r = cur_r;
}
```

算法优化：查表法

计算计算三解函数的值必须依赖泰勒级数，如下正切函数：

$$\tan x = \sum_{n=0}^{\infty} \frac{B_{2n}(-4)^n(1-4^n)}{(2n)!}x^{2n-1} \qquad \forall x : |x| < \frac{\pi}{2}$$

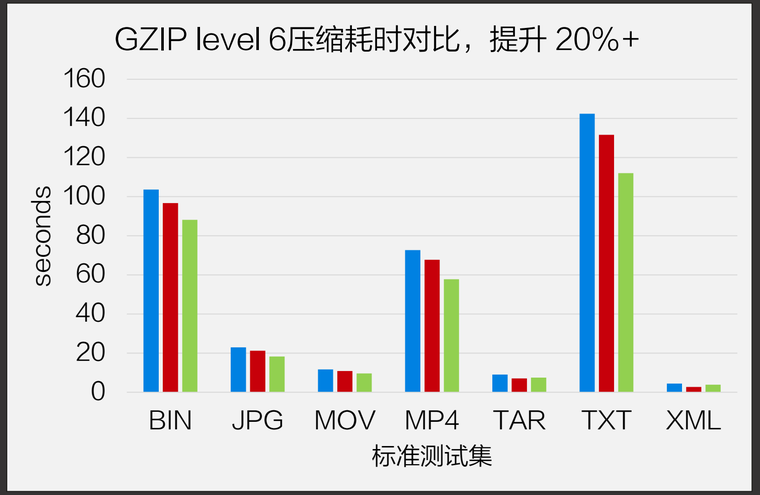
从上式中可以看到，显然，当|x|>1时，计算次数不可控，因此对tanx变换为

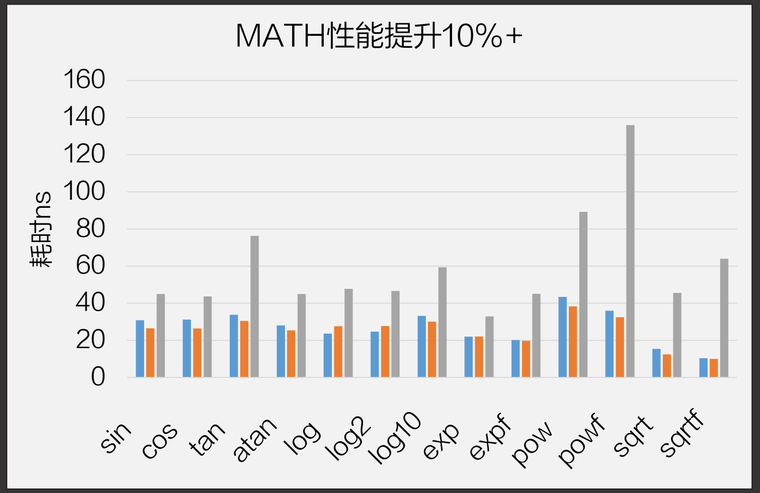
$$tan(x) = \tan(\alpha + \beta) = \frac{tan\alpha + tan\beta}{1 - tan\alpha \cdot tan\beta}, \beta = [x]$$

将上式中的tanβ通过提前计算，保存到数组中（即表中），需要计算时，通过索引查表获取，表的划分需使得tanα在运用泰勒级数计算时可计算。

最后，再通过tanα，tanβ两个值的代入到上式中通过四则运算即可得到tanx的值。

优化效果





其他常用的优化技巧

- 指令重排

  通过微调代码，使得汇编指令执行顺序产生变化，指令执行过程更符合流水线结构；

- 循环展开

  通过将循环体内的单次执行修改为多次执行，减少循环条件在执行过程中的比例；

- 标量替换

  通过将数组中的元素赋值给一个标量，用于计算过程，减少对数组元素的访问次数；

- 循环分块

  通过将大循环的计算过程，拆分成多个小循环的计算过程；

- 强度削弱

  通过合并计算过程，使得计算的总次数大幅度减少；

- 重复利用

  通过将一次计算结果，或一次缓存加载的数据多次使用，减少计算或加载；

目录

- 加速库能够带来什么样价值
- 加速库开发的技术概览
- 探索硬件加速的设计思路
- 揭开高性能软件编码的秘密
- 加速库在业务调优中的方法

HUAWEI

鲲鹏DevKit一站式开发套件提升全流程开发效率

极简高效、极智精准、极致性能

迁移	性能加速	编译	调优	无源码迁移
Porting Advisor 代码迁移工具	Kunpeng library 加速库	BiSheng Compiler 毕昇编译器	Hyper Tuner 性能分析工具加速库	ExaGear 二进制指令翻译
扫描效率每分钟5万行 应用迁移效率 45人天->6人天/应用	加速库性能提升 > 10%	ARM原生编译 性能提升20+%	性能可视精准调优	无源码软件 平滑迁移，业务无损

工具插件化与IDE开发环境集成

VS Code – Intellij – MindStudio

HUAWEI

鲲鹏应用使能套件BoostKit

提供全栈优化的应用加速能力

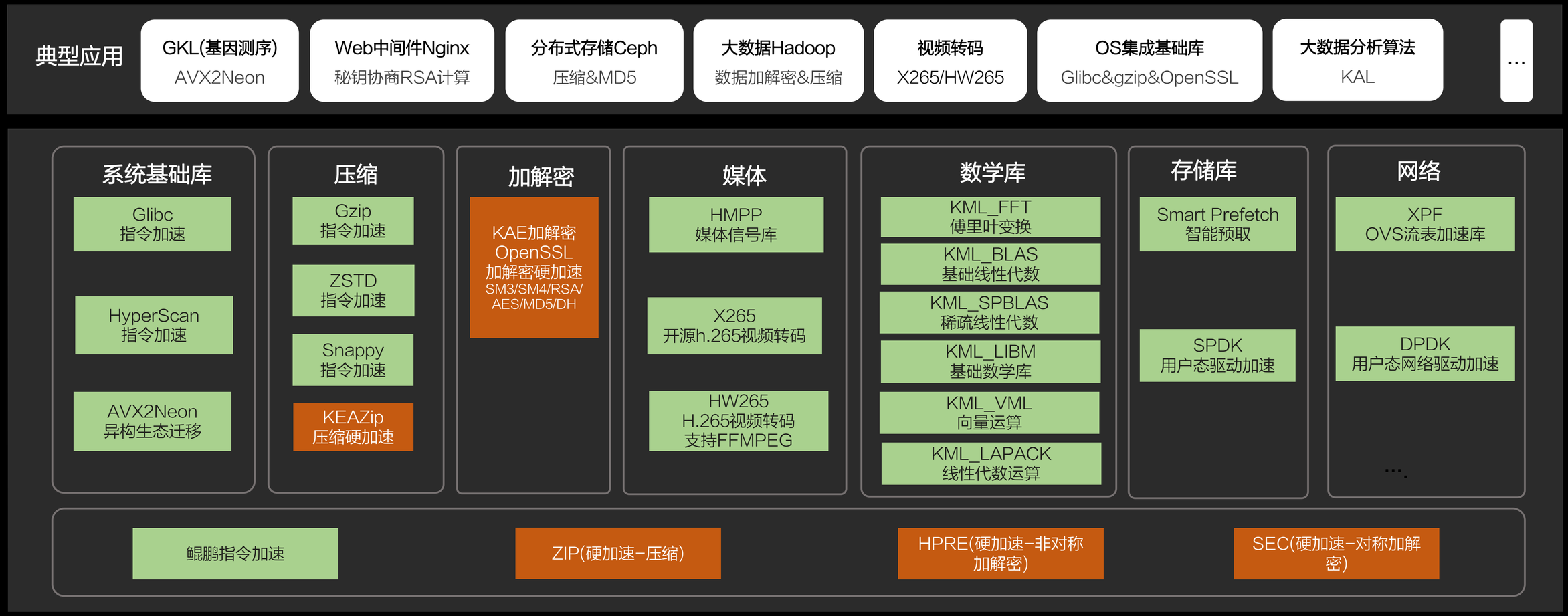
- 应用加速：应用性能倍级优势
- 基础加速：应用性能超越业界水平
- 开源使能：开源软件可用、好用

提供八大场景化应用使能套件：大数据、分布式存储、数据库、虚拟化、 ARM原生、Web/CDN、NFV和HPC

HUAWEI

鲲鹏加速库

鲲鹏加速库：基于鲲鹏平台指令优化和软硬件结合的加速技术



软加速库：基于鲲鹏指令的软加速库　　硬件加速：基于鲲鹏加速引擎的加速库

备注：KAE（Kunpeng Accelerator Engine）鲲鹏加速引擎；HPRE（High Performance RSA Engine）高性能RSA加速引擎；SEC（Security Engine）硬件安全加速引擎

HUAWEI

鲲鹏加速库在大数据解决中的应用

01 解决方案 | 02 热点功能 | 03 关键组件 | 04 加速库应用 | 05 优化效果

- 大数据
- 大数据平台
- 大数据组件
- 基础软件库
- 操作系统
- 鲲鹏服务器

热点功能	关键组件	加速库应用	优化效果
加密解密（AES/SM3/SM4）	HBase	加速引擎	• 传统软件加密，占用CPU资源，性能损耗大：20% • 鲲鹏硬件加密性能损耗小：＜5%
数学计算（GEMM/GEMV）	Spark	数学库	• 鲲鹏数学库使得Spark在进行矩阵运算时，性能有10%的提升
磁盘访问（顺序读）	HDFS	智能预取	• 智能预取使得大数据组件HDFS访问磁盘的性能有10%~50%性能提升

HUAWEI

鲲鹏加速库在分布式存储中的应用

01 解决方案

02 热点功能

03 关键组件

04 加速库应用

05 优化效果

分布式存储

应用平台

Ceph
对象/块/文件

基础软件库

操作系统

鲲鹏服务器

数据压缩
(Zlib)

磁盘访问
(顺序读)

Ceph

加速引擎

智能预取

混合读写（7:3）带宽性能提升**最高40%**

- 智能预取使得分布式存在访问磁盘的性能有20%~100%性能提升

HUAWEI

鲲鹏加速库在WEB中的应用

01 解决方案

02 热点功能

03 关键组件

04 加速库应用

05 优化效果

WEB解决方案

WEB网站

WEB容器（NGINX）

基础软件库

操作系统

鲲鹏服务器

- HTTPS作为WEB网站的主流选项，加入了鲲鹏加速引擎后，SSL处理能力从20K提升到108K

HUAWEI

鲲鹏加速库在人工智能中的应用

01 解决方案

02 热点功能

03 关键组件

04 加速库应用

05 优化效果

人工智能

语音识别平台

Python框架

DNN框架

操作系统

鲲鹏CPU

矩阵计算（GEMM）

Pytorch

数学库

- 语音识别平台Pytorch集成鲲鹏数学库KML-BLAS后，优化后的sgemm表现良好，识别时间缩短10%+

HUAWEI

鲲鹏加速库在软件迁移中的应用

01 解决方案

02 热点功能

03 关键组件

04 加速库应用

05 优化效果

软件迁移

X86高性能软件

C/C++软件库

Intrinsic指令

操作系统

鲲鹏CPU

SIMD指令 → intrinsics指令 → Avx2Neon →

- Avx2Neon指令库使得高性能软件库从x86迁移到鲲鹏平台过程中工作量大幅度减少，**修改代码从成千上万行减少为1行代码**，同时迁移后的软件在鲲鹏平台上性能最佳。

HUAWEI

总结

- 为什么要做加速库?

  提升软件计算效率，改善业务应用综合性能。

- 怎么做好加速库?

  通过加速库的技术栈，分类，合理运用工具进行分析和设优。

- 如何通过硬件实现加速?

  将常用，性能影响较大的算法设计到芯片中，通过驱动使能到业务应用中。

- 如何通过编码技巧实现加速?

  数据预取，流水线，算法优化，向量化，循环展开等。

- 如何挖掘加速库在业务中的价值?

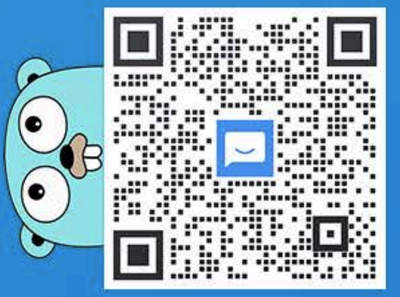
  解决方案性能测试->分析热点功能（函数）->识别功能组件->匹配加速库->确认加速效果

Thank you.

把数字世界带入每个人、每个家庭、
每个组织，构建万物互联的智能世界。

Bring digital to every person, home and organization for a fully connected, intelligent world.

抢占先机，成为未来 3 年抢手的后端开发人才



扫码获取详细大纲
并咨询课程详情

【Go 进阶训练营】带你成为字节跳动 2-2 级别 Go 工程师

- 12 大模块教学
- 3 大企业级实战案例
- 13 周视频课程教学
- 大厂助教 1v1 答疑
- 班主任全程贴心督学
- 简历直推一线互联网公司