GMEM：面向领域加速器的通用内存管理异构融合调度

王彬 陈辉　华为OS基础软件技术专家

开放原子开源基金会 OPENATOM FOUNDATION | OpenEuler

GMEM：面向领域加速器的通用内存管理及异构融合调度

- **GMEM：面向领域加速器的通用内存管理**
  - 异构内存管理现状以及问题
  - GMEM设计理念
  - GMEM效果呈现
- **异构融合调度**
  - 粗放调度下的算力利用率问题
  - ms级抢占以及细粒度算力切分
  - 异构融合调度愿景

GMEM：面向领域加速器的通用内存管理

异构加速器的黄金时代，内存是异构生态核心组成

现在是体系结构（加速器发展）的黄金时代 -- David Patterson

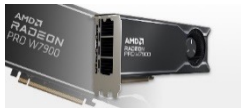
- 原因：CPU无法满足AI等异构领域的计算要求、而AI领域随着ChatGPT等的成功，跻身未来核心市场
- 例证：异构芯片市场正在不断扩张（NVIDIA市值已经突破万亿美元）

- **加速器应用广泛：**

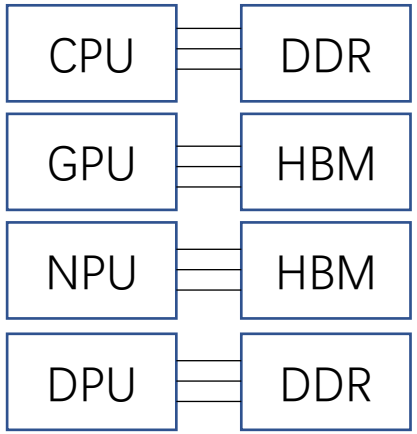
  人工智能 、图形处理、搜索推荐、大数据...



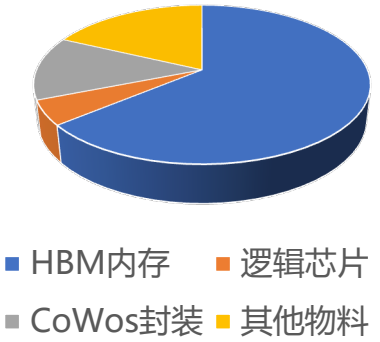
多种多样的加速器生态


- **内存是加速器架构的核心单元**
- **加速器成本组成，内存占比最大**

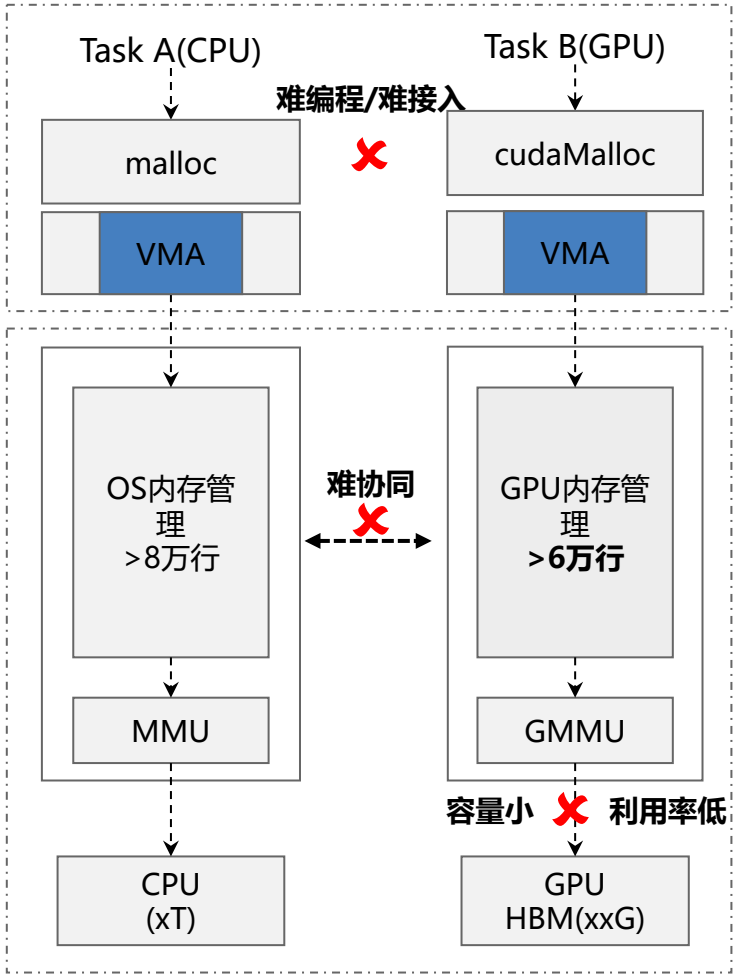


某加速器成本占比



当前异构内存生态体系割裂，软件栈走向烟囱化，与应用诉求不匹配

AS IS：体系分离



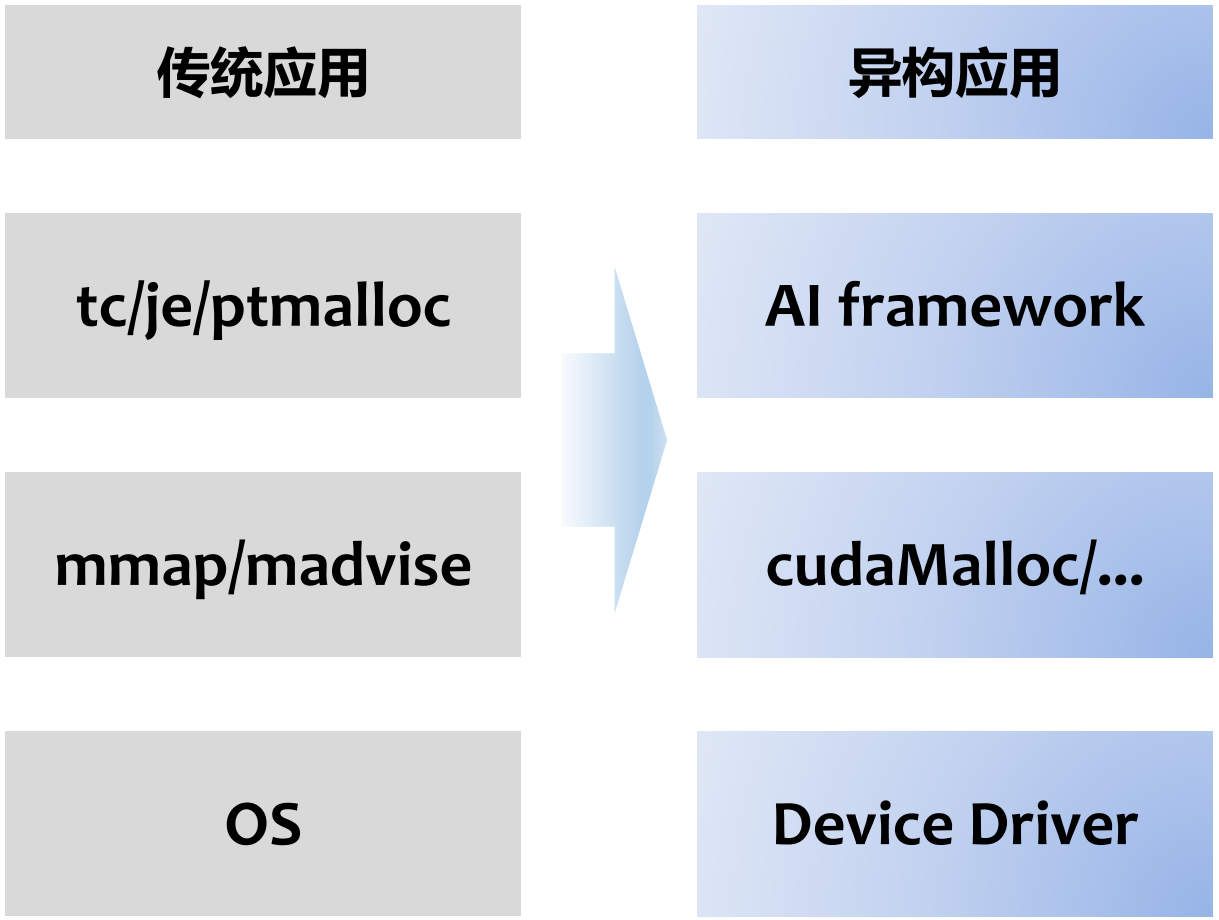
异构应用的诉求：

- ☐ XPU编程统一协同
- ☐ 更大的异构内存容量
- ☐ 更快的内存申请/释放
- ☐ 更高的内存利用率

然而，现实是 … …

- ☒ 编程性差：CPU缓存 OR GPU缓存？开发者深陷泥坛
- ☒ 内存不足：HBM容量低，OOM成为家常便饭
- ☒ 内存碎片化：内存紧缺雪上加霜
- ☒ 申请释放慢：实现不合理，重复开发

重复设计内存框架代价高昂，性能参差不齐



重复设计内存框架，代价高昂

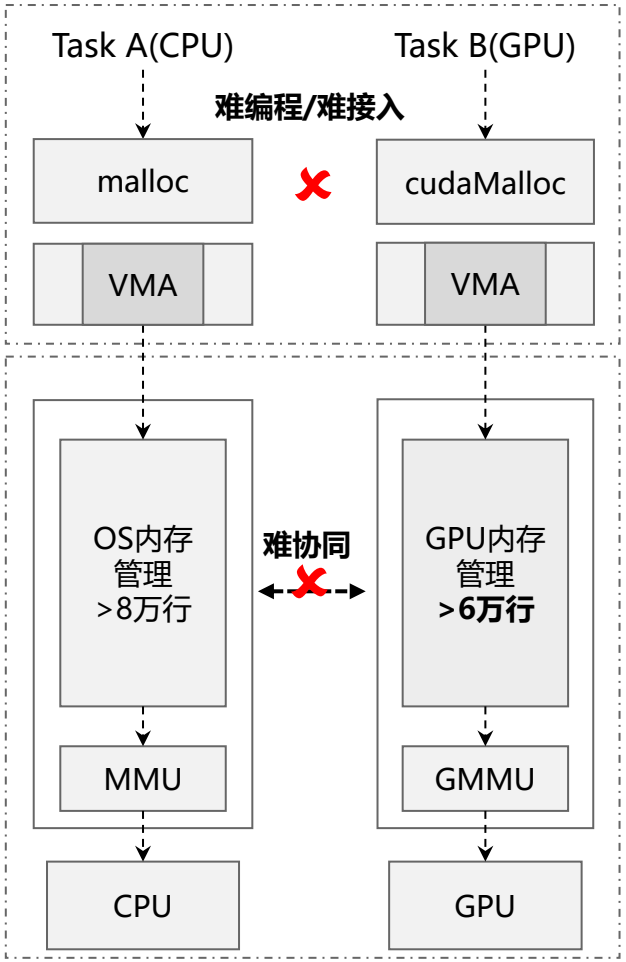
- 代码量大，数万行代码，维护困难
- 稳定性弱，相比经过数十年演进的Linux内存管理框架（功能趋向稳定，性能趋向极致）
- 内存碎片化严重，无法实现细粒度内存管理

加速器驱动(代码量)	内存管理框架	内核内存SWAP	总计
Nvidia	~34K	~70K	~104K
AMD	~14K	N/A	~14K
Huawei	~30K	N/A	~30K

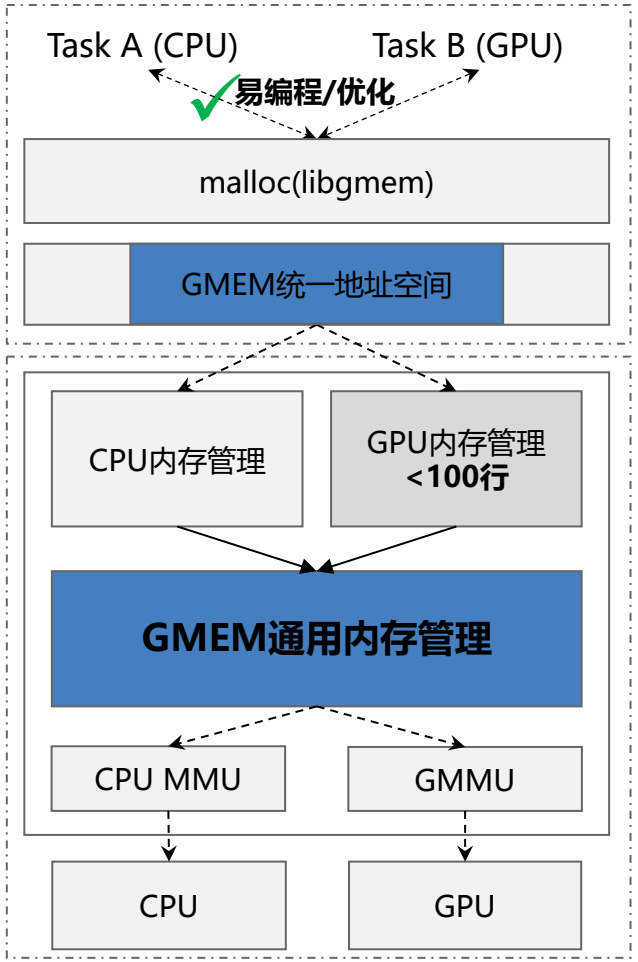
从CPU生态，机械式迁移至异构生态

GMEM使能下的异构生态演进，中心化协同打造极致性能

AS IS：体系分离



TO BE：中心化协同



GMEM的使命：

1. 加速器无需重复造轮子，专注内存优化策略
   - 无需重复开发用户态“malloc”方案
   - 无需重复开发内核态内存管理框架
2. 可编程性大幅增强
   - 统一的异构内存空间
   - 用户通过hint极易实现内存策略
   - 异构内存透明超分（Host DDR）
3. 更好的内存性能
   - 更快的内存申请/释放速度
   - 更高的内存利用率

易接入：GMEM实现加速器极易接入，享受Linux原生内存管理的极致性能

- GMEM提供统一的OS内存管理框架，实现加速器极易接入：
  - 加速器仅需百行代码可轻松接入GMEM生态，享受Linux原生内存管理极致性能

90%缩减

内存管理框架	代码量
昇腾NPU驱动	30,000
基于GMEM的NPU驱动	30 (调用GMEM API) 2000 (底层MMU实现)

- GMEM使加速器专注内存优化策略，无需重复造轮子
  - 更快的内存申请/释放速度
  - 更高的内存利用率

3倍+提升

用户态内存方案	平均时延
malloc (GMEM)	300ns
PyTorch-malloc	1000ns

高性能：GMEM实现内存透明超分，大幅提升训推性能

高性能训推

◆ 内存透明超分，使能高性能推理，超大模型单卡可训

- 单卡（32G-NPU）可训52B GPT大模型
- AI框架修改量 < 10 Loc
- 超分T级Host内存，大幅提升问题处理规模

✓ **LLAMA推理性能提升30%~70%**

模型吞吐量 (Token/s)	原始	GMEM	提升
LLAMA-7B	52	69	32%
LLAMA-13B	14	24	71%

✓ **性能相比NVIDIA-UVM提升60%+**

模型	Matmul	Resnet (large batch)	GPT2
NVIDIA-UVM	1	1	1
GMEM	1.7x	1.8x	2x

极低内存碎片

◆ 蛋白质折叠模型（等效AlphaFold模型）应用成果

- 内存利用率：HBM利用率从40%提升至90%+
- 问题处理规模：蛋白质折叠长度提升25%
- 端到端推理速度：提升20%~40%（更快的编译速度）

✓ **问题处理规模提升25%**

✓ **推理速度提升20%~40%**

易编程：GMEM赋能OS原生接口，基于malloc/mmap实现极易异构编程

GMEM赋能加速器，实现极简异构编程：

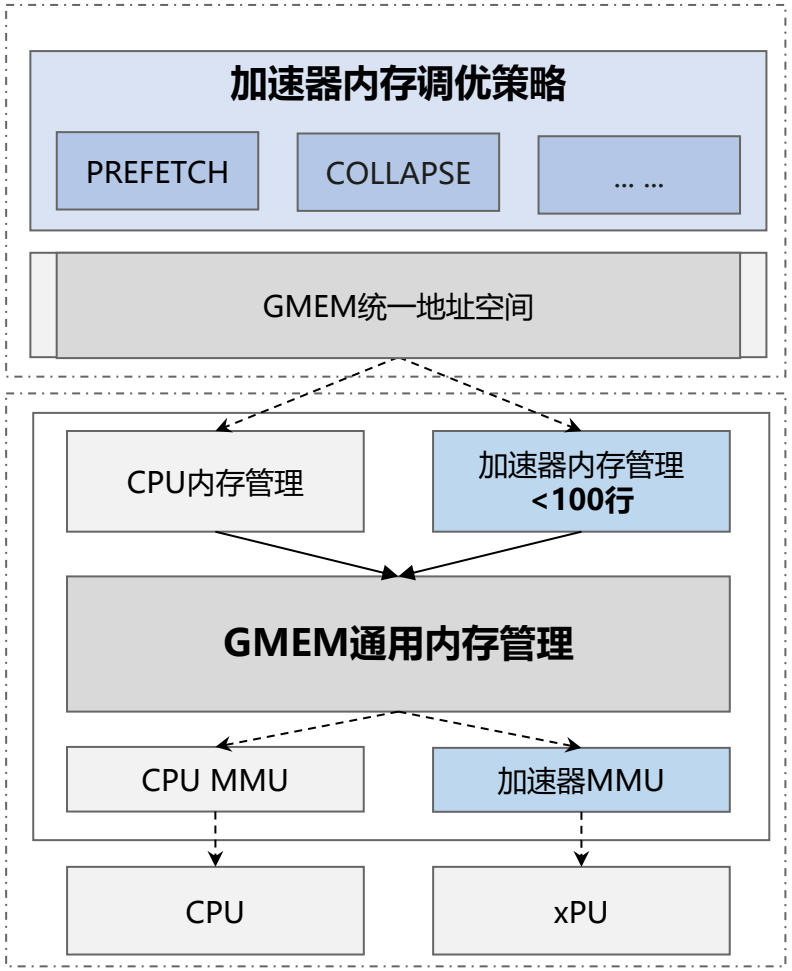
```
#1 A = malloc();
#2 prepare(A);
#3 B = aclrtMalloc();
#4 aclrtMemcpy(B, A, h2d);
#5 aclopExecute(op, B);
```

```
#1 A = malloc(MAP_PEER_SHARED); // A为统一虚拟地址
#2 prepare(A);
#3 // hmadvise(NPU_id, A, size, PREFETCH); // 预取，可选
#4 aclopExecute(op, A); // 触发缺页，预取则不触发
```

GMEM编程新范式：

- 按需动态分配内存
- CPU-XPU 统一虚拟地址编程（算子可直接下发）
- 用户无感扩容HBM （OS自动swap HBM-DDR）
- 预取性能优化

GMEM提供丰富性能优化语义，助力加速器打造极致竞争力



GMEM使能下的中心化协同架构

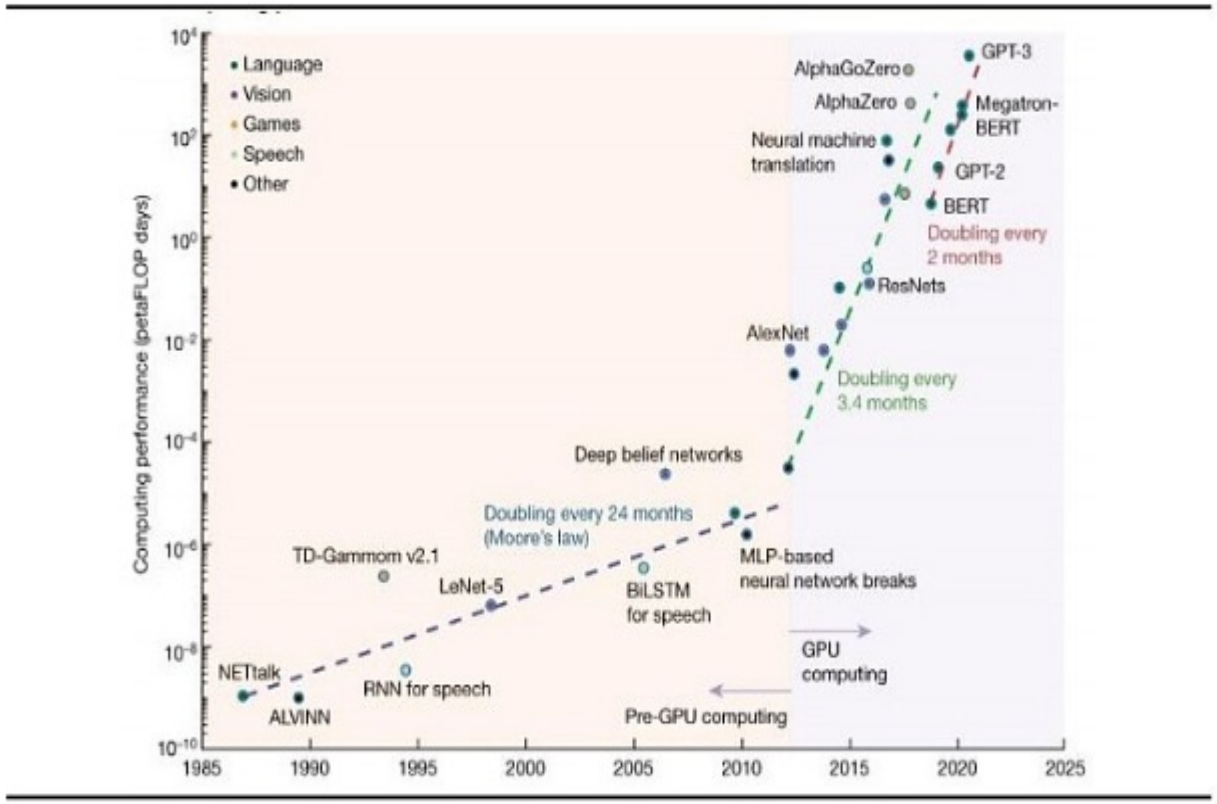
丰富的内存优化语义，助力加速器打造极致竞争力

- ✓ PREFETCH ：异步预取语义，交叠通信与计算
- ✓ COLLAPSE_SPARSE ：稀疏压缩语义，极致发挥硬件性能
- ✓ RANDOM ：随机访问语义，提升稀疏场景综合带宽
- ✓ LAZYFREE ：内存碎片快速回收语义，释放更多HBM资源
- ✓ More ... ...

异构融合调度

异构算力需求快速膨胀，AI模型训推成本高，算力利用率低

AIGC应用加速了算力需求，远超摩尔定律



ChatGPT引发新一轮AI算力需求爆发。根据OpenAI发布的《AIandCompute》分析报告中指出，自2012年以来，AI训练应用的算力需求每**3.4个月**就回会翻倍，从2012年至今，AI算力增长超过了**30万倍**。据OpenAI报告，ChatGPT的总算力消耗约为3640PF-days（即假如每秒计算一千万亿次，需要计算3640天），需要7-8个算力500P的数据中心才能支撑运行。

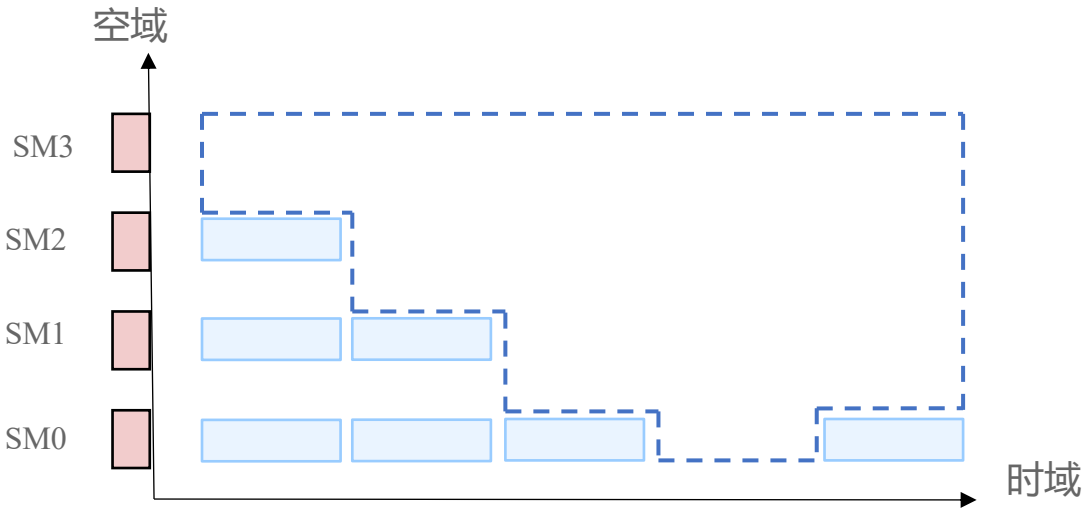
算力成本高

■ 自建IDC的推理成本估算

日活用户数（万）	单用户每日提问次数	每个问题平均字数（个）	A100 GPU对每个字的响应时间（毫秒）	每日消耗GPU计算时间（小时）	每天需要A100 GPU芯片（个）	NVIDIA DGX A100服务器（台）	NVIDIA DGX A100服务器价格（万美元）	推理成本（亿美元）
2000	10	20	350	388889	16204	2026	19.9	4.03

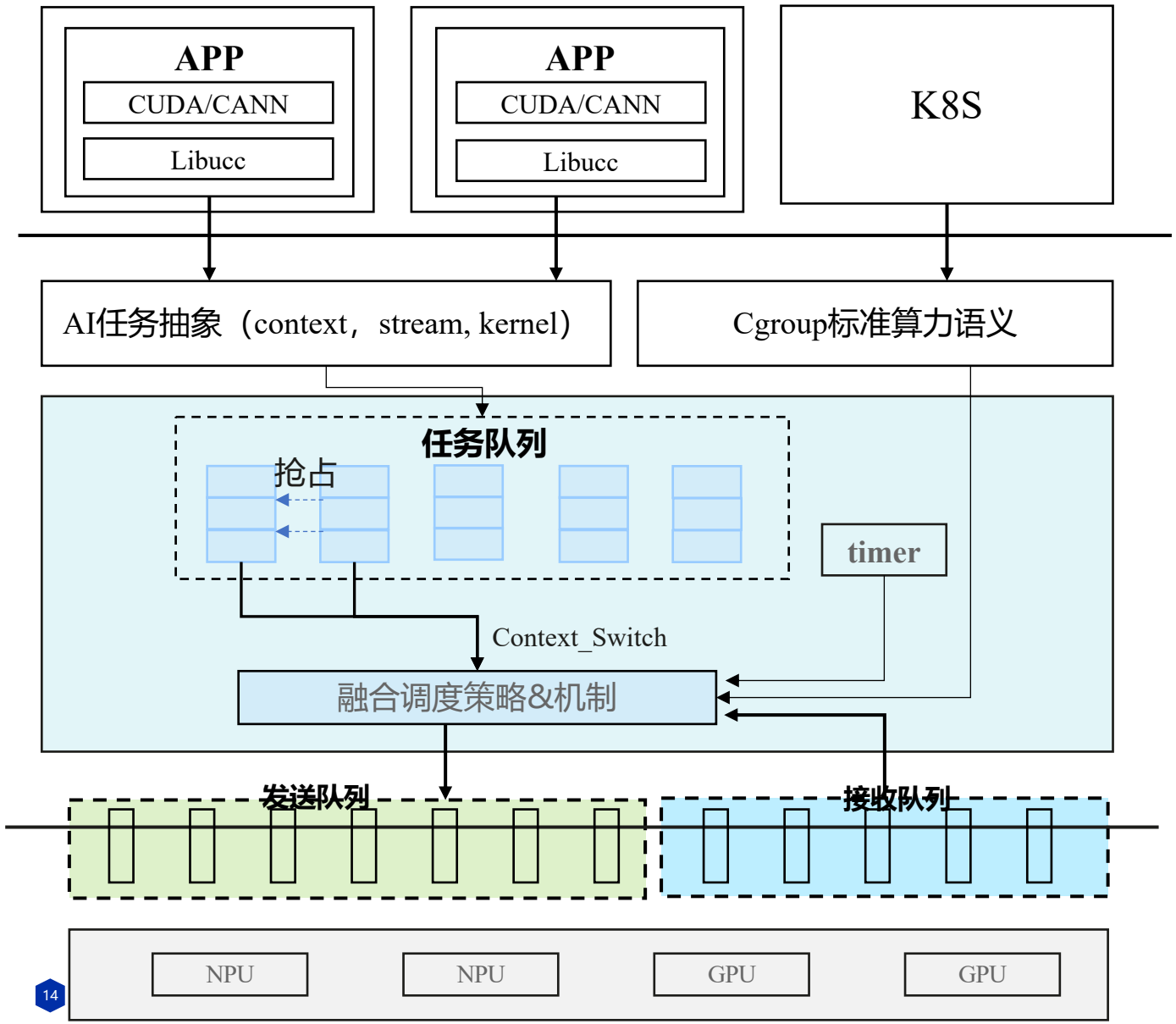
在训练千亿参数的盘古大模型时，华为团队调用了超过**2000块**的昇腾910芯片，进行了超2个月的数据训练能力。华为内部称，每年大模型训练调用GPU/TPU卡超过4000片，3年的大模型算力成本高达**9.6亿**元人民币。

粗放式的资源管理导致资源碎片严重，利用率低



从公开资料显示：时域上统计，公有云上GPU的利用率只有20%-30%左右，主要因为分池部署，业务量不足导致；在空域上统计，平均单个任务对GPU的有效利用率< 10%，主要因为业务平均负载小，资源独占等原因导致。

异构融合调度：支持ms级抢占能力，实现精细灵活的算力切分能力



1）任务抽象：

对AI任务进行抽象，包括context, stream, kernel进行抽象，同时包括任务的状态管理等。

2）算力抽象：

对NPU算力进行抽象，包括device, VF, RTSQ等，提供通用的能力；

3）调度模型：

调度接口：ucc_wake, ucc_wait, ucc_yield, ucc_set_attr, ucc_set_affinity接口；

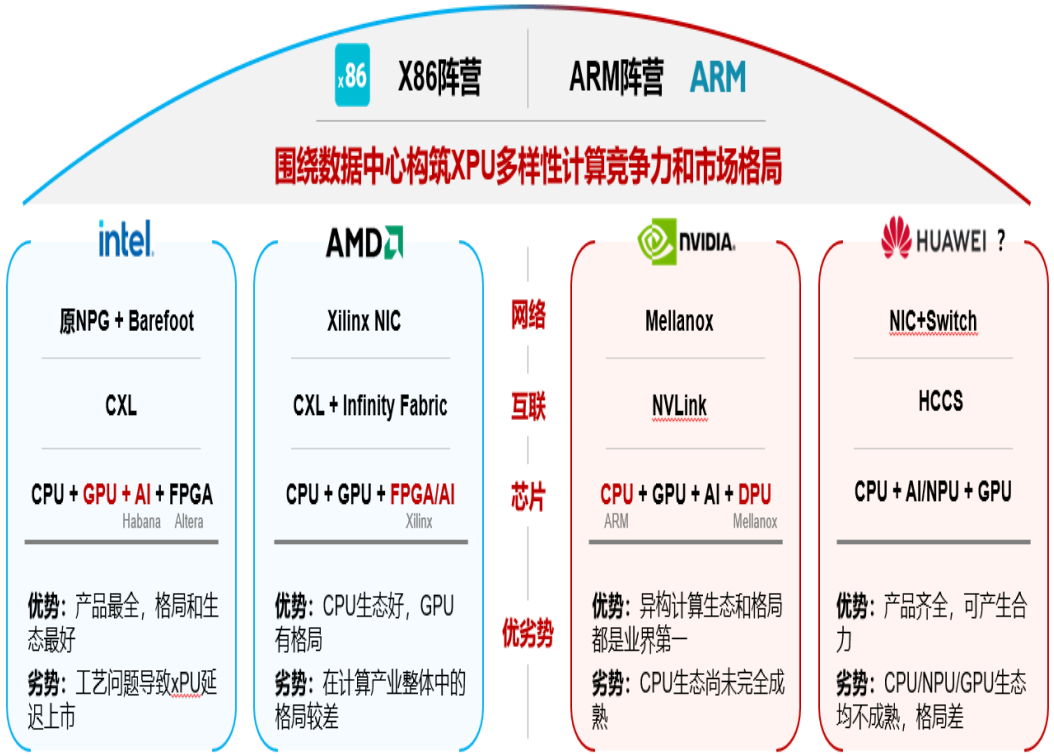
调度策略：提供ucc_rt，ucc_cfs和可编程调度策略，支持多种调度需求；

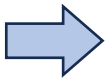
抢占机制：提供高优先级抢占低优先级任务和分时机制。

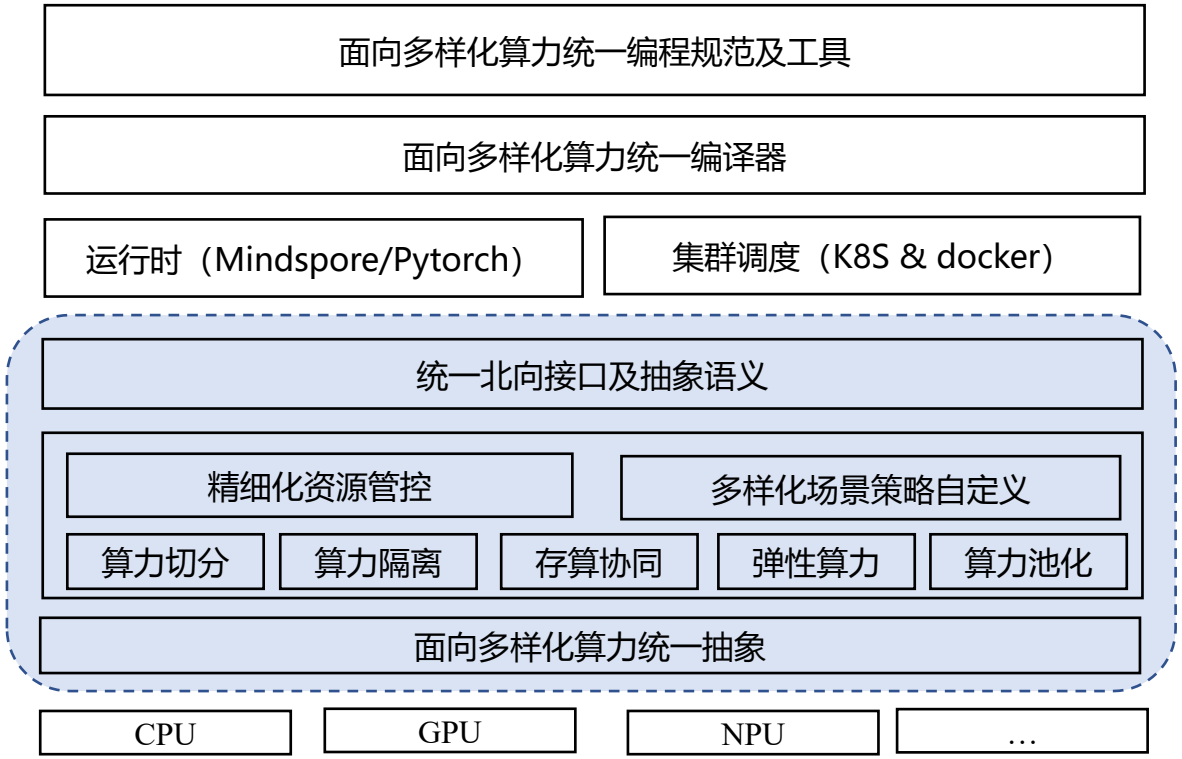
效果：针对算子下发模型，提供ms粒度的抢占能力

异构融合调度愿景

构建统一算力编程语义，打造精细化、弹性、服务化的异构算力管理方式，致力于消除算力碎片、提升算力利用率，降低算力成本。







THANKS

THANKS

THANKS

THANKS