附件 1

中俄数学挑战基金
2025 年度项目申报指南

一、总体目标

中俄数学挑战基金旨在推动数学领域的创新与发展，并促进数学成果在产业中的产生实际应用。产业界与学术界协同，共同凝练，发布数学难题；同时鼓励跨界，不同领域的研究学者来申请课题，共同解决产业中的数学难题。

二、难题方向

2025 年度数学挑战基金拟发布 17 个难题，根据申请方案择优资助，每个项目拟资助 50 万人民币（税前），项目实施周期一般为 1 年。

序号	难题方向
1	基于 1BIT 量化的高精度向量检索算法
2	低比特 Attention 加速算法
3	MoE 模型专家权重压缩算法
4	文档 KV 复用难题
5	OOD 向量检索难题
6	基于大容量 SSD 的低内存高性能 FTL 方案
7	面向高时延介质的数据预取算法设计与实现
8	Massive MIMO 算法到底层算子组合直接映射
9	Massive MIMO 信号张量压缩及压缩数据上的直接计算
10	高条件数矩阵分解&求逆低位宽计算
11	结构化矩阵求逆
12	基于函数插值的低复杂度并行计算
13	多层模型的近似误差回传/训练算法
14	动态干扰信道预测—非平稳环境下的参数估计
15	高效计算-芯片算法的低功耗实现： 低比特位宽前向网络
16	计算图调度与内存复用方案联合寻优
17	多用户多载波组合智选算法

注：受资助者在项目研究过程中形成的与项目相关的成果的著作权及专利等，包括但不限于论文、著作、源代码等，其知识产权权利归属，由资助方与受资助者及其所在单位共享，详细条款由项目资助协议具体协定。

指南方向 1：基于 1BIT 量化算法的高精度向量检索

背景

向量检索是一个经典的数学问题[1]，已有近 60 年的研究历史，当前已被广泛应用于互联网搜索推荐、大模型 RAG、向量数据库、视频图像检索等场景。向量检索是从一个向量集合中找到与给定查询向量最接近的向量的算法，定义如下：

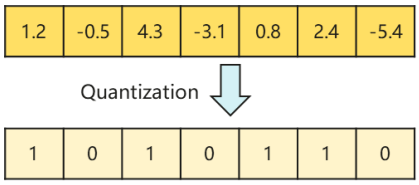
向量检索定义：
给定一个d维向量$q \in \mathbb{R}^d$，从一个包含n个d维向量的集合 $B \in \mathbb{R}^{n\times d}$ 中检索得到与q 向量最相似的向量v，即求解：

$$v = arg\max_{v\in B} sim(v,q)$$

其中 q 被称为查询向量，B 被称为向量底库。其中向量相似度通常有两种定义方式：
1) 欧式相似度：$sim(v,q) = -\|v-q\|_2$
2) 内积相似度：$sim(v,q) = \langle v,q\rangle$

现状

1BIT 量化算法[2][3]是向量检索领域的前沿技术，也是当前工业界和学术界的热门研究方向。但是，现有基于 1BIT 量化的向量检索算法的 Top-10精度通常小于 80%，而实际应用中通常要求向量检索的 Top-10精度达到 95%~99%。因此，如何提升 1BIT 量化后的向量检索精度，是业界亟待解决的技术挑战。



向量检索精度定义：
给定一个d维向量$q \in \mathbb{R}^d$，一个包含n个d维向量的集合 $B \in \mathbb{R}^{n\times d}$，定义向量检索的 Top-$k$精度为：

$$S_k = \frac{|R_k \cap R_k^*|}{k}$$

其中$R_k = \{r_1, r_2, \dots, r_k\}$ 是基于量化后的 1BIT 向量，通过暴力搜索得到的集合B中与向量q相似度最高的 k 个向量 ID；$R_k^* = \{r_1^*, r_2^*, \dots, r_k^*\}$是基于量化前的 FP32（32 位浮点数）向量，通过暴力搜索得到的集合B中与向量 q 相似度最高的 k 个向量 ID。

问题描述

问题输入：

1) 向量底库：由 n 个 d 维实数向量组成的向量集合 $B \in \mathbb{R}^{n\times d}$；
2) 查询向量：由 m 个 d 维实数向量组成的向量集合 $Q \in \mathbb{R}^{m\times d}$
3) 向量相似度函数：$sim(v,q)$，其中 v 和 q 分别是两个向量，限定$sim(v,q)$必定为欧式相似度或内积相似度，即$sim(v,q) = -\|v-q\|_2$ 或 $sim(v,q) = \langle v,q\rangle$。

优化目标：

4) 设计二值量化函数 f，其输入是向量底库 B 中的任意实数向量 $x \in B$，

$$f(x) = x' \in \{a,b\}^d$$

其中 a,b 是两个固定的常数。

5) 并设计量化后的向量相似度函数 g

$$g(x',q) = g(f(x),q) \in \mathbb{R}$$

使得对于全部查询向量 $q \in Q$，最小化量化后的向量相似度计算的总误差：

$$\min_{f,g} \sum_{q\in Q}\sum_{x\in B} |sim(x,q) - g(f(x),q)|$$

4 目标

设计二值量化函数 f 和量化后的向量相似度函数 g，使得基于 1BIT 量化的向量检索 Top-1, Top-10 和 Top-100 精度达到 95%以上，挑战 99%。

- 参考文献：

[1] Bruch, Sebastian. Foundations of Vector Retrieval. Springer, 2024.

[2] Gao, Jianyang, and Cheng Long. "RaBitQ: quantizing high-dimensional vectors with a theoretical error bound for approximate nearest neighbor search." Proceedings of the ACM on Management of Data 2.3 (2024): 1-27.

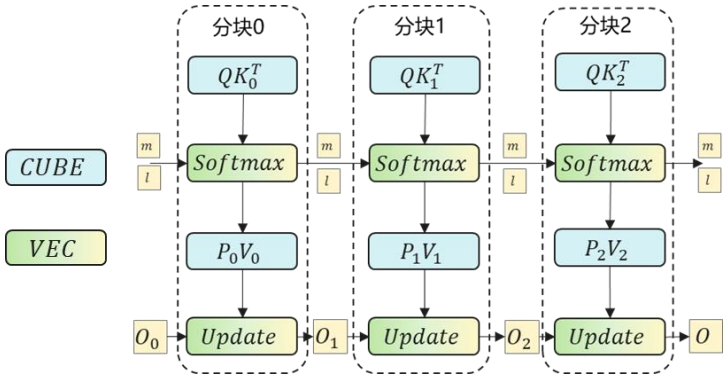
[3] Gao, Jianyang, et al. "Practical and asymptotically optimal quantization of high-dimensional vectors in euclidean space for approximate nearest neighbor search." Proceedings of the ACM on Management of Data 3.3 (2025): 1-26.

指南方向 2：低 Bit Attention 极致加速

背景

Attention 机制被广泛应用于各类机器学习任务中，在有效提升模型能力的同时也带来了具大的计算开销。由于其计算复杂度与输入序列长度的平方呈正比，随着应用向着更长序列发展，Attention 计算影响逐步提高，例如在典型多模态视频生成任务中，Attention 计算时间占比超过 80%。

Attention 计算为矩阵-向量复合计算，其 SoftMax 计算为向量计算。同时为亲和硬件实现，业界普遍采用 Flash Attention[1]方法分块迭代完成 Attention 计算，额外引入了 Update 计算也为向量计算。因此，对于 Attention 计算的加速需要同时考虑矩阵计算部分和向量计算部分。



现状

业界已有相关研究，对 Attention 中的 QKV 进行低比特量化，利用低比特矩阵乘算力加速 Attention。但是由于 Flash Attention 计算中包含了大量向量计算，无法被低比特矩阵乘加速。因此，在高 CV 算力比的芯片上，低比特 attention 由于陷入向量计算瓶颈导致无法真正实现加速。

业界算法	时间	计算精度			
		QK^T	$Softmax$	SV	$Update$
FlashAttention2.0[1]	2023.07	FP16	FP32	FP16	FP32
FlashAttention3.0[2]	2024.06	FP16/FP8	FP32	FP16/FP8	FP32
SageAttention1.0[3]	2024.10	INT8	FP32	INT8	FP32
SageAttention2.0[4]	2024.11	INT4/INT8	FP32	INT8	FP32
SageAttention3.0[5]	2025.05	MXFP4	FP32	MXFP4	FP32

问题描述

由于 Attention 计算中的矩阵乘法 QK^T 与Softmax操作在大模型和长序列下计算开销高，因此希望对 Attention 进行加速，可使用方法包括但不限于：低比特量化（例如 4/8-bit）、近似计算（例如分段近似等）。

但要求对最终 Attention 输出A_{approx}保持高度保真，即：

$$\cos\theta\big(\mathrm{vec}(\mathrm{Attention}(\mathrm{Q},\mathrm{K},\mathrm{V})),\mathrm{vec}(A_{approx}(Q,K,V))\big) > 0.999$$

输入：

Attention 计算输入三个矩阵$\mathrm{Q},\mathrm{K},\mathrm{V} \in R^{N\times d}$（N 为输入序列长度，d 为模型 Head dim）。同时，$\mathrm{Q},\mathrm{K},\mathrm{V}$矩阵为大模型中的激活值包含少量的 outlier 值。对于$\mathrm{Q},\mathrm{K},\mathrm{V}$的数值分布，可以建模为两个正态分布组成的混合分布，定义如下：

$$Q,K,V \sim 0.999\cdot\mathcal{N}(0,1) + 0.001\cdot\mathcal{N}(0,100)$$

输出：

Attention 计算包含两个矩阵乘计算和向量 SoftMax 计算，标准的 Scaled Dot-Product Attention 公式如下：

$$\mathrm{Attention}(\mathrm{Q},\mathrm{K},\mathrm{V}) = \mathrm{Softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V$$

其中 SoftMax 计算对矩阵的每行进行，为保证数值稳定性，通常采用如下的 Stable SoftMax 形式：

$$Softmax(x_{ij}) = \frac{exp^{x_{ij}-\max(x_i)}}{\sum_j exp^{x_{ij}-\max(x_i)}}$$

目标

精度目标：

在给定的$\mathrm{Q},\mathrm{K},\mathrm{V}$分布假设或基于模型实际数据情况下，满足加速方案输出与标准 Attention（FP16 矩阵乘，FP32 矢量计算）输出余弦相似度达到 0.999+：

$$\cos\theta\big(\mathrm{vec}(\mathrm{Attention}(\mathrm{Q},\mathrm{K},\mathrm{V})),\mathrm{vec}(A_{approx}(Q,K,V))\big) > 0.999$$

性能目标：

相比于标准 Attention，所提出的方法应在理论上或实践中**计算复杂度降低 3x+**，注意需要满足矩阵计算和向量计算复杂度同时降低 3x。

实现计算中不同类型的操作可能具有不同的计算开销，例如典型处理器处理 FP16 数据的能力是 FP32 的 2 倍，而同精度下 EXP 计算能力只有乘法。因此，对于计算复杂度的评估，应区分不同数据类型和不同操作类型，可以参考下表。

	operation	FP32	FP16	FP8	FP4
CUBE	**MAC**	/	1	0.5	0.25
VECTOR	**MUL/ADD/SUB**	2	1	0.5	/
	MAX/MIN	2	1	0.5	/
	EXP/Reciprocal	8	4	0.5	/
	DIV	8	4	0.5	/

- 参考文献

[1] FlashAttention-2:Faster Attention with Better Parallelism and Work Partitioning

[2] FlashAttention-3:Fast and Accurate Attention with Asynchrony and Low-precision

[3] SageAttention: Accurate 8-Bit Attention for Plug-and-play Inference Acceleration

[4] SageAttention2: Efficient Attention with Thorough Outlier Smoothing and Per-thread INT4 Quantization

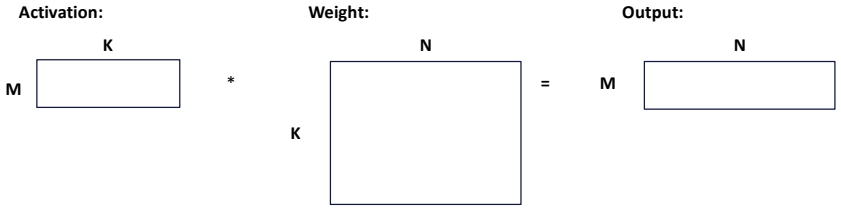
[5] SageAttention3: Microscaling FP4 Attention for Inference and An Exploration of 8-bit Training

指南方向 3：MoE 专家权重压缩

背景

向量检索 DeepSeek 应用火爆离不开其高效、经济的部署方式。当前基于 H800 推理部署采用专家并行(EP)和大 Batch_Size 策略，改变 MoE 层的 GeMM 算子的形状(M, N, K)，实现单节点算力打满，成为中心推理场景的主流方案。

然而，在一体机场景下，Batch_size 通常小于 10，推理阶段面临以下两个问题：受限，MoE 算子的并行度不足，GeMM 算子的形状(M, N, K)通常退化为(1, N, K)，性能受限于访存带宽。其次，提升 Batch_size 大小将导致更多专家被同时激活，访存瓶颈更加严重。总之，在一体机推理场景下，设计算法压缩被激活的专家权重降低带宽需求，迫在眉睫。



现状

当前大语言模型权重压缩相关研究包括以下方面：

- 单专家权重矩阵内(intra-matrix)压缩，比如W4A16,W4A8等量化方案，相比原始BF8权重可以达成1/2带宽需求削减。目前已经部署W4A16方案，仍无法解决带宽问题，需要更低bit的量化方案。
- MoE同层内(intra-layer)压缩，比如先分组再SVD压缩[1]、基于残差的Delta压缩[2]等方法。
- MoE不同层间(inter-layer)压缩，比如分析层间SVD分解[3]、基于重要性排序剪枝[4]。
- 其它在线剪枝策略等。目前尚无直接针对DeepSeek-R1/V3模型的分析工作。

设计 MoE 层专家权重压缩算法需要关注以下两方面挑战：

- 被激活的专家分布存在热点，与输入数据相关。若基于此特点设计权重压缩算法需要考虑泛化性。
- 权重压缩算法能接受有损压缩，但精度验证需要整网模型端到端测试。目前尚无单层MoE专家权重误差传播到整网精度误差的理论推导。

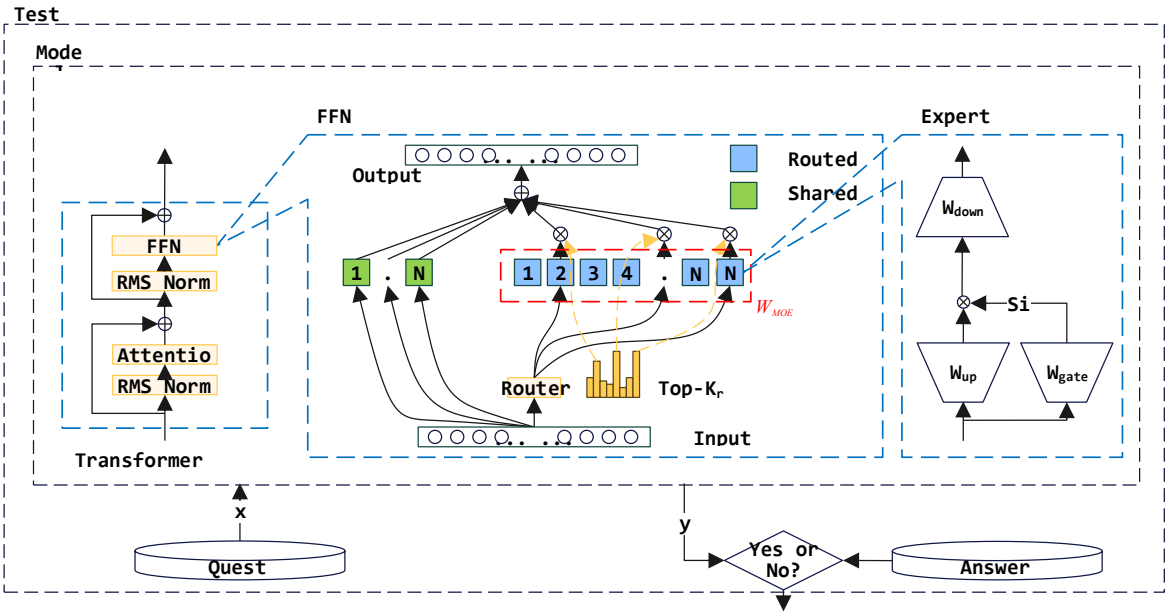
问题描述

MoE 模块W_{MoE}是 DeepSeek 模型$M(W_{\text{MoE}}, W_{\text{else}})$重要组成部分，它由一系列被激活的专家组成。

对于给定输入x权重$w_{down}, w_{gate}, w_{up}$单个专家计算遵循：

$$output = w_{down}\left(silu\left(w_{gate}(x)\right) \cdot w_{up}(x)\right)$$

其中，$w(x) = w \cdot x, w = w_{down}, w_{gate}, w_{up}$对应矩阵乘，$silu(x) = \frac{x}{1+e^{-x}}$对应于$x$逐元素激活处理。



问题输入：

DeepSeek 模型：由 671B 个实数构成的参数集合$\in \mathbb{R}^{671B}$（其中 654B 用于W_{MoE}，其余都在W_{else})和对应的单步推理函数$M(W_{\text{MoE}}, W_{\text{else}})$。

测试问题集合：由q个$m \times d$维实数向量构成的向量集合Quest $\in \mathbb{R}^{q \times m \times d}$。

优化目标：

设计 MoE 权重压缩算法 f 用于产生压缩后的权重$W'_{MoE} = f(W_{MoE})$。

基于压缩后的权重W'_{MoE}，设计新的单步推理函数$M'(W'_{MoE}, W_{else})$ 在给定测试问题集合Quest上极小化推理误差：

$$\min_{f,M'} \sum_{x \in \text{Quest}} |\mathrm{M}(\mathrm{W}_{\text{MoE}}, \mathrm{W}_{\text{else}})\mathrm{x} - \mathrm{M}'(\mathrm{W}'_{\text{MoE}}, \mathrm{W}_{\text{else}})\mathrm{x}|$$

目标

在 DeepSeek V3 量化权重 W4A16 或 W4A8 模型上，通过挖掘 MoE 层专家权重相似性，不引入量化方案，设计 MoE 专家权重压缩算法f和对应的单步推理函数M'使得模型推理性

能提升并能在给定测试数据集上保持良好精度。

精度目标：使能模型权重压缩算法以后，通过主流 benchmark 测试的精度掉点小于 1%。

性能目标：给定参数 Batch_size=10，输入序列长 2000，输出序列长 1000 条件下，通过理论分析 MoE 算子计算量、访存量等指标，论证使能权重压缩算法后，MoE 算子耗时减少 30%。

- 参考文献：

[1] Beyond Standard MoE: Mixture of Latent Experts for Resource-Efficient Language Models

[2] Delta Decompression for MoE-based LLMs Compression

[3] MoE-I^2: Compressing Mixture of Experts Models through Inter-Expert Pruning and Intra-Expert Low-Rank Decomposition

[4] Condense, Don't Just Prune: Enhancing Efficiency and Performance in MoE Layer Pruning

指南方向 4： RAG 系统中文档 KV 复用的关键挑战

背景

在传统的检索增强生成（RAG）系统中，工作流程包含两个阶段：

1. **检索阶段**: 从语料库中获取相关文档集D $= \{d_1, d_2, \dots, d_n\}$;
2. **生成阶段**: 将检索到的文档与用户查询 q 和系统提示 p 进行拼接，拼接后用于大语言模型推理.

效率瓶颈在于每次查询时都需要对检索到的文档重新计算注意力，Prefill 阶段需要 $O(n^2)$复杂度的自注意力计算。文档 KV 缓存技术提出存储注意力中间结果（KV cache）以供复用。

问题描述

1．变量定义：

定义 1（输入组件）

文档集合: $D = \{d_i\}_{i=1}^{n}$，其中每个文档$d_i = (t_1^i, t_2^i, \dots, t_m^i)$;

用户查询: $q = (t_1^q, t_2^q, \dots, t_k^q)$;

系统提示: $p = (t_1^p, t_2^p, \dots, t_l^p)$;

模型有 L 层，每层 H 个注意力头;

每个注意力头 Key/Value 维度为d_k, d_v.

定义 2（KV 缓存），对于文档d_i，其 KV 缓存为：

$$KV_i = \left\{KV_i^{l,h} | 1 \le l \le L, 1 \le h \le H\right\},$$

其中每个$KV_i^{l,h} = (K_i^{l,h}, |V_i^{l,h}) \in \mathbb{R}^{m\times d_k} \times \mathbb{R}^{m\times d_v}$

2．注意力计算：

问题 1 (理想注意力计算). 对于拼接输入 $x = [p; q; d_1; \dots; d_n]$, 第 l 层第 h 个注意力头的计算如下:

$$Attention^{l,h}(Q^{l,h}, K^{l,h}, V^{l,h}) = softmax\left(\frac{Q^{l,h}(K^{l,h})^T}{\sqrt{d_k}}\right),$$

其中 $Q^{l,h} = W_Q^{l,h}x, K^{l,h} = W_K^{l,h}x, V^{l,h} = W_V^{l,h}x$.

问题 2 (KV 复用计算). 使用缓存文档时，计算变为:

$$\tilde{K}^{l,h} = [k_p^{l,h}; k_q^{l,h}; k_1^{l,h}; \dots; k_n^{l,h}],$$

$$\tilde{V}^{l,h} = [V_p^{l,h}; V_q^{l,h}; V_1^{l,h}; \dots; V_n^{l,h}],$$

其中下标 p, q 表示实时计算的组件。

技术挑战

问题 3. 完全重算与 KV 复用的注意力权重差异:

$$A_{ij} = \frac{\exp(q_i \cdot k_j, /\sqrt{d_k})}{\sum_j, \exp(q_i \cdot k_j, /\sqrt{d_k})}$$

$$\tilde{A}_{ij} = \frac{\exp(q_i \cdot \tilde{k}_j, /\sqrt{d_k})}{\sum_j, \exp(q_i \cdot \tilde{k}_j, /\sqrt{d_k})}$$

其中 $\tilde{\mathrm{k}}_{\mathrm{j}}$ 来自缓存计算，导致$\mathrm{A}_{\mathrm{ij}} \neq \widetilde{\mathrm{A}}_{\mathrm{ij}}$.

问题 4.位置编码错位:

$$\textit{完全计算}: k_j = W_K(x_j + PE(j))$$

$$\textit{缓存复用}: \tilde{k}_j = W_K(x_j + PE(pos_{cache}))$$

缓存位置编码pos_{cache}与新上下文中的实际位置 j 不匹配.

问题 5. 定义第 l 层输出误差$\epsilon^l = \left\|\tilde{h}^l - h^l\right\|$. 多文档场景下:

$$\epsilon^{l+1} \leq \left\|W_O^l\right\| \cdot \|\sigma'\| \cdot (\epsilon^l + \delta^l)$$

其中δ^l为当前层注意力误差，导致误差跨层放大。

研究目标

- **跨文档注意力**: 开发保证$\left|\tilde{A}_{ij} - A_{ij}\right| < \epsilon$的机制.
- **位置编码:** 创建满足下式的调整方法:

$$PE_{adjusted}(pos) \approx PE(pos_{actual})$$

- **误差控制:** 建立误差传播上界:

$$\epsilon^{l+1} \leq C \cdot \epsilon^l,\ C < 1$$

- **效率:** 实现计算时间要求:

$$T_{reuse} \leq \alpha \cdot T_{recompute},\ \alpha \in (0,1)$$

指南方向 5： 分布外（OOD）向量检索难题描述

背景

向量检索（Approximate Nearest Neighbor Search, ANNS）是数据库、信息检索和大规模模型中的核心技术。传统的 ANNS 算法（例如 HNSW、IVF）基于“邻居的邻居仍然是

邻居”的假设，即查询向量（Query）与数据分布（Key）之间具有一致的空间局部性。然而，在实际应用中，查询可能来自分布外（OOD）场景，导致：

- **检索效率降低：**OOD 查询违反了空间局部性原则，其中相似数据点（KK）距离远小于查询-数据点（QK）的距离。
- **检索适应性不足：**现有方法（例如 RoarGraph）依赖已知的分布假设，这会损害分布内检索性能。

问题描述

1. 变量定义:

- **数据分布**: 点集 $K = \{k_i\}_{i=1}^{N} \subseteq \mathbb{R}^d$ 服从分布 P_k.
- **查询分布:** $q \sim P_Q$, 可能满足$P_Q \neq P_K$ (OOD 场景).
- **相似性度量**: 欧氏距离$D(q,k) = \|q-k\|_2$.

2. 输入/输出:

- **输入:** 数据集K, 图索引$G = (V,E)$, 查询向量$q \sim P_Q$.
- **输出:** Top- K 近似邻居$\widehat{N}_{k(q)}$ 召回率 $\geq \tau$.

3. 数学描述:

传统 ANNS 假设$P_Q = P_K$，局部性成立:

$$\forall q \sim P_Q, if\ k_i \in N_K(q), then\ N_K(k_i) \approx N_K(q)$$

OOD 场景下 $(P_Q \neq P_K)$, 局部性破坏:

$$\exists q \sim P_Q, s.t. \min_{k \in K} D(q,k) \gg \min_{k_i,k_j \in K} D(k_i,k_j)$$

此时遍历节点数T_{OOD} 显著增加:

$$T_{OOD} \approx \frac{\mathbb{E}_{q \sim P_Q}[D(q,K)]}{\mathbb{E}_{k_i,k_j \sim P_K}[D(k_i,k_j)]} \cdot T_{ID}$$

目标

设计 OOD 鲁棒的向量检索方法，要求:

1. **高效性**: $T_{OOD} \approx T_{ID}$.
2. **无分布假设:** 不依赖 P_Q先验.
3. **保持 ID 性能:** 不损害 P_K 内检索效率.

指南方向 6： 基于大容量 SSD 的低内存高性能 FTL 难题

背景

随着 SSD 容量的快速增长（如从 TB 级到 PB 级），闪存转换层（FTL）的设计面临严峻挑战。传统 FTL 的地址映射策略主要分为**块级映射**和**页级映射**

- **块级映射**以擦除块（Block，通常 128KB 2MB）为粒度管理逻辑地址到物理地址的映射，其映射表内存占用较小，但写入灵活性差（覆盖写需先擦除整个块），导致写放大（Write Amplification）严重，影响 SSD 性能和寿命。
- **页级映射**以页（Page，通常 4KB）为粒度管理映射关系，写入灵活性强，但映射表内存开销过大（例如 1TB SSD 需至少 1GB 内存存储映射表）。

现有折中方案（如混合映射、缓存部分映射表）仍存在以下问题：

1. **双重读取问题（Double Read Problem）：**若映射表存储在闪存中，读取数据需先访问映射表（第一次读），再访问数据（第二次读），显著降低读性能.
2. **学习型索引的局限性（Limitations of Learned Indexes）**：尽管学习型索引（如分段线性模型）可压缩映射表内存占用，但 SSD 有限的计算资源难以支持复杂模型推理，且动态更新效率低，无法适应频繁的写入负载。

因此，亟需设计一种**低内存开销、高性能、适应大容量SSD的FTL方案**。

问题描述

1. 标量定义:

C: SSD 物理容量（字节）

L: 逻辑地址空间大小（字节）

P: 物理页大小（如 4KB）

$B = N \times P$: 物理块大小(如，当 $N = 64$ 时 $B = 256$KB)

$l \in \{0,1,\dots,[L/P]-1\}$:逻辑页编号 (LPN)

$p \in \{0,1,\dots,[C/P]-1\}$: 物理页编号 (PPN)

M: LPN $\rightarrow$ PPU $\cup$ {NULL}: 逻辑页到物理页的映射函数

2. 目标函数:

设计映射策略 M，满足:

内存约束:

$$Mem(M) \leq \alpha \cdot [L/P] \cdot \log_2[C/P],\ \alpha \ll 1$$

性能约束:

$$Latency_{read} \leq Latency_{flash} + \delta$$

$$WA(M) \leq \beta$$

可靠性约束: 映射表需支持崩溃一致性.

数学挑战：

映射表压缩: 需找到函数$\mathcal{F}$ 将 M 编码为紧凑结构：

$$\mathcal{F}(M) = \langle \theta, \varepsilon \rangle,\ |\theta| + |\varepsilon| \ll |M|$$

双重读取避免:

$$Pr[M(l) = \mathcal{F}^{-1}(\theta, l)] \geq 1 - \epsilon,\quad (\epsilon \to 0)$$

目标

设计新型 FTL 方案，实现:

- **低内存映射表 Low Memory:** 内存占用降低至传统页级映射的 10% 以下 ($\alpha \leq 0.1$)
- **高性能地址转换:**
  - 读操作99% 以上请求可通过一次闪存访问完成 ($\epsilon \leq 0.01$)
  - 写放大系数 WA≤ 2
- **轻量级计算 Lightweight Computation:** 算法复杂度为 $O(1)$ 或者 $O(\log n)$

指南方向 7： 面向高时延介质的数据预取算法设计与实现

背景

高时延介质凭借其高容量/价格比，成为归档和数据保护场景的主流选择。然而，其时延问题限制了其在温冷数据场景的应用。为突破这一限制，结合 **SSD 缓存+ 磁带**的混合架构，结合数据预取算法，目标是将访问时延降低至百毫秒级。传统预取算法（如顺序流、间隔流预测）主要面向 DRAM、SSD 和 HDD 等多级存储，而高时延介质的独特特性（如极高预取开销、长时延、受限计算资源）要求设计新型预取算法，以在多样化负载下提升 SSD 缓存命中率并降低访问时延。

问题描述

1. 变量声明:

- 存储系统访问序列:$S = \{a_1, a_2, \dots, a_n\}$, 其中 a_i 表示第i次访问的逻辑块地址(LBA)。
- 与去窗口大小: w, 表示每次预取的最大连续数据块数。
- SSD 缓存容量: C, 单位为数据块数.
- 缓存命中函数: $H(a_i) \in \{0,1\}$，若 a_i 在 SSD 中命中则$H(a_i) = 1$，否则为 0。
- 预取触发阈值: θ, 当预测置信度超过该值时触发预取.
- 预取算法: $f: S \to P$, 其中$P = \{p_1, p_2, \dots, p_k\}$ 为预测的待预取数据块序列。

2. 优化目标:

最大化 SSD 缓存命中率，同时最小化平均访问时延T_{avg}:

$$Maximize \frac{1}{n}\sum_{i=1}^{n} H(a_i), \quad MinimizeT_{avg} = \frac{1}{n}\sum_{i=1}^{n} T(a_i)$$

其中$T(a_i)$ 为访a_i的时延，若命中则$T(a_i) = T_{SSD}$，否则需从磁带加载，$T(a_i) = T_{tape}$。

3. 约束条件:

- **预测长度约束**: 需预测更长的序列以隐藏时延：

$$|P| \geq w, \ w \gg 1$$

- **计算资源约束**: 预取算法的计算复杂度需满足实时性要求:

$$Time(f(S)) \leq T_{max}$$

- **缓存容量约束**: 预取数据总量不能超过 SSD 缓存容量:

$$\sum_{j=1}^{k} \mathbb{I}(p_j \notin SSD) \leq C - CurrentCacheUsage$$

4. 关键挑战的数学刻画:

挑战 1（预测泛化性）：算法需在线适应访问模式变化，即:

$$f \text{ 需满足 } \forall S_{new}, \lim_{|S_{new}|\to\infty} \frac{\sum_{a_i \in S_{new}} H(a_i)}{|S_{new}|} \geq \eta$$

其中η为目标命中率阈值。

挑战 2（复杂规律捕捉）：需建模高阶马尔可夫性或长程依赖:

$$P(a_{t+1}|a_t, a_{t-1}, \dots, a_{t-m}) \approx P(a_{t+1}|State(a_{1:t}))$$

其中m为历史窗口，State 可能由 RNN、Transformer 等结构编码。

挑战 3（预取时延敏感）：需优化预取触发时机以避免无效预取:

$$\text{PrefetchAt } t \Leftrightarrow \exists P, s.t. \frac{|P \cap S_{t+1:t+w}|}{|P|} \geq 0$$

挑战 4（计算资源受限）：算法复杂度需为亚线性:

$$\text{Time}(f) = O(\log n) \text{ or } O(1)$$

研究目标

设计一种面向高时延介质的在线自适应预取算法，满足：

- **高泛化性：**在多样化负载（随机/顺序/混合）下保持高命中率 $(\eta \geq 90\%)$。
- **长序列预测：**支持预测窗口$w \geq 10$以隐藏磁带时延。
- **低计算开销：**单次预测时间 $\leq$ 1ms。
- **在线学习：**无需离线训练，动态适应访问模式变化。

该问题的突破将推动高时延介质在温冷存储场景的广泛应用，显著降低存储成本。

指南方向 8： 跨领域矩阵计算在硬件加速算子上的最优映射

背景

无线通信 Massive MIMO 信号处理和 AI 的 Transformer 注意力的计算包含大量矩阵计算，矩阵计算已成为两类系统的核心计算负载。无线信号处理涉及矩阵求逆、特征值分解、快速傅里叶变换（FFT）等操作，而 AI 算法 Transfomer 算法依赖大规模矩阵乘加、归一化、Softmax 等计算。AI 芯片通过专用硬件算子（如矩阵乘加单元、向量计算单元）提供显著的加速能力，但现有研究缺乏对跨领域矩阵计算任务的统一优化框架。本问题旨在建立数学模型，将无线与 AI 的矩阵计算映射到底层硬件算子，实现精度约束下的计算代价最小化。

现状

1．无线信号处理：

矩阵运算优化数学原理进行优化。通过矩阵直接分解/求逆、迭代法（如 Neumann 级数）降低计算复杂度。常用计算精度有 FP32/16，int32/16 等多种计算精度。

2．AI 算法：

算子融合：将矩阵乘加与激活函数合并为单一算子（如 GEMM+ReLU）。

量化加速：通过低精度（FP16/INT8）减少计算量，需平衡精度损失。

硬件加速：昇腾芯片支持以下算子加速：矩阵乘加（GEMM）：100 倍加速（FP32 精度）非线性向量运算（如 Softmax、ReLU）：10 倍加速。标量运算（如常数乘法）：无加速。

3. 挑战：

1）无线与 AI 的矩阵计算优化独立进行，未针对同一硬件平台设计统一框架，导致算子调用冗余和计算资源浪费。

2）针对不同底层算子加速开发工作量大，非底层最优算子组合，计算性能受限等缺点。

问题描述

给定无线信号处理矩阵处理和 AI Transformer 注意力计算：

1．无线 Massive MIMO 信号处理矩阵计算：

上行 MIMO 检测：$\boldsymbol{W} = \boldsymbol{H}\,(\boldsymbol{H}\,\boldsymbol{H}^{H} + \boldsymbol{R_{uu}})^{-1}$

下行波束成型：$\boldsymbol{W}_n = \boldsymbol{V}_n(\boldsymbol{V}_n^{H}\boldsymbol{V}_n + \delta^2\boldsymbol{I})^{-1}$

2．AI Transformer 注意力计算：

$$\boldsymbol{Q} = \boldsymbol{X} \times \boldsymbol{W_Q}, \boldsymbol{K} = \boldsymbol{X} \times \boldsymbol{W_K}, \boldsymbol{V} = \boldsymbol{X} \times \boldsymbol{W_V}$$

$$Attention(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = softmax\left(\frac{\boldsymbol{Q}\times\boldsymbol{K}^{\boldsymbol{T}}}{\sqrt{\boldsymbol{d_k}}}\right)\times\boldsymbol{V}$$

$$\sigma(\boldsymbol{Z})_i = \frac{e^{\boldsymbol{Z}_i}}{\sum_{j=1}^{K} e^{\boldsymbol{Z}_j}}, \boldsymbol{Y} = \frac{\boldsymbol{X}-E[\boldsymbol{X}]}{\sqrt{Var[\boldsymbol{X}]+\varepsilon}}\times\gamma+\beta$$

将上述的计算任务分解为底层算子（GEMM、向量运算、标量运算）的组合，在满足下述要求约束条件下求解最优算子调用方案。

数学建模：

定义计算图$\boldsymbol{G}=(\boldsymbol{V},\boldsymbol{E})$，其中节点$\boldsymbol{V}$表示矩阵运算，边$\boldsymbol{E}$表示数据依赖。将每个节点$\boldsymbol{v}\in\boldsymbol{V}$分解为硬件算子组合$\boldsymbol{s_v}=\{\boldsymbol{s_1},\boldsymbol{s_2},\dots,\boldsymbol{s_k}\}$，其中$\boldsymbol{s_i}$为 GEMM、向量运算或标量运算。

目标函数：

$$min\sum_{v\in V}\sum_{s\in S_v} T(\boldsymbol{s}) \quad s.t.Error(\boldsymbol{S_v})\le\epsilon$$

其中，$T(\boldsymbol{s})$为算子$\boldsymbol{s}$的计算时间，$Error(\boldsymbol{S_v})$为整体算子组合的精度损失。

1. 精度约束：输出结果与浮点计算结果的相对误差 $\varepsilon \le 10^{-3}$。

2. 计算代价最小化：总计算耗时最小。

目标

1．建立跨领域矩阵计算的统一优化模型，量化硬件加速收益与精度损失的关系。

2．设计动态规划或整数规划等算法，求解最优算子组合。

3．在昇腾平台上实现无线（MIMO 检测和下行波束成型算法）与 AI（Transformer 注意力计算）任务优化，实验验证计算时间降低 ≥30%。

- **参考文献**

[1] W. H. Chen, “Matrix Computation for Wireless Communications,” IEEE Trans. Signal Process., 2020.

[2] T. L. Marzetta, “Large-Scale Antenna Systems,” Foundations and Trends® in Signal Processing, 2014.

[3] A. Vaswani et al., “Attention Is All You Need,” NeurIPS, 2017.

[4] S. Venkataramani et al., “Optimized GEMM for Deep Learning,” MLSys, 2021.

[5] Huawei Ascend AI Processor Architecture, Huawei White Paper, 2022.

[6] J. Pool et al., “What’s Inside NVIDIA A100 Tensor Cores?” arXiv:2103.05111, 2021.

[7] Y. Yan et al., “Approximate Matrix Multiplication for Wireless Systems,” IEEE JSTSP, 2021.

[8] M. Gupta et al., “Deep Learning with Low Precision,” ACM Comput. Surv., 2018.

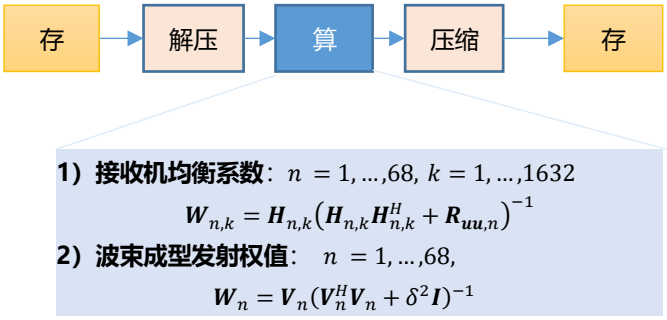
指南方向 9： Massive MIMO 信号张量压缩及压缩数据上的直接计算

背景

随着 5.5G 通信技术的演进，Massive MIMO（大规模多输入多输出）系统在 6GHz 及更高频段的应用成为关键技术方向。为实现更高的频谱效率和网络容量，天线数量呈现指数级增长（如数百至数千天线阵列）。然而，Massive MIMO 信号处理中的矩阵运算复杂度（如信道估计、预编码、检测等）随天线数呈立方增长$O(N^3)$，存储复杂度呈平方增长$O(N^2)$，导致计算资源需求剧增，成为系统部署的核心瓶颈。通过矩阵压缩技术降低存储，但现有压缩框架需将压缩后数据恢复为全维信号再进行计算，未能彻底解决复杂度问题。

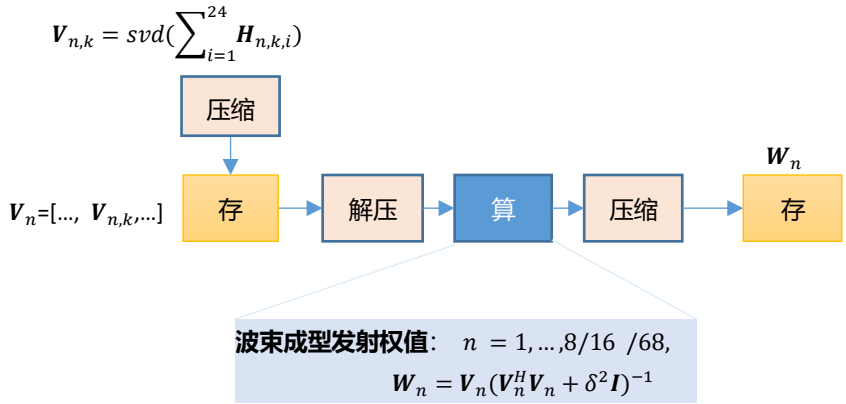
现状

压缩后数据需恢复为全维矩阵才能进行后续处理，导致计算复杂度未显著降低。压缩与计算环节割裂，未能充分利用压缩数据的结构特性优化计算过程。



问题描述

问题 1. 下行波束成型的计算和压缩问题：



其中：$\boldsymbol{V}_n \in \mathrm{C}^{256\times L}, n = 1,\dots,8/16/32/64$ ； $\boldsymbol{W}_n \in \mathrm{C}^{256\times L},\ n = 1,\dots,8/16/32/64$，$L$为下行调度流数，$L$为1~100。本问题取$L = 64$。

压缩数据量目标： $\boldsymbol{V}_n$、$\boldsymbol{W}_n$分别压缩后，存储量小于原始 FP32 数据的 1/10。

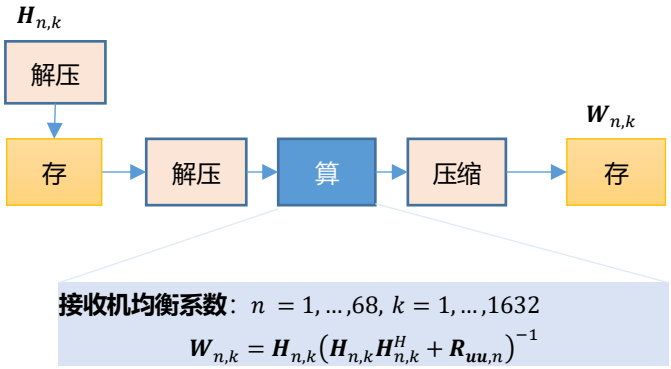
计算目标： $\boldsymbol{W}_n = \boldsymbol{V}_n(\boldsymbol{V}_n^H\boldsymbol{V}_n + \delta^2\boldsymbol{I})^{-1}$，$n = 1,\dots,8/16/32/64$

计算精度要求：$|\boldsymbol{W'}_n - \boldsymbol{W}_n|^2 < \varepsilon$，计算结果误差的归一化 MSE 小于-40dB (即归一化幅度误差<0.01)。

计算复杂度：包含压缩/解压缩及计算过程的复杂度小于原始不压缩计算基线的 1/3。

注：信道 **H** 可以用开源软件 Quadriga 生成，给定信道配置类型。

问题 2. 下行波束成型的计算和压缩问题：



其中：$\boldsymbol{H}_{n,k} \in C^{256\times L}, n = 1,\dots,8/16,\ k = 1,\dots,8/16*12$。$\boldsymbol{W}_{n,k} \in \mathrm{C}^{256\times 50}$，$n = 1,\dots,8/16,\ k = 1,\dots,8/16*12$。$\boldsymbol{R}_{\boldsymbol{uu},n} \in \mathrm{C}^{256\times 256}, n = 1,\dots,68$。$L$为上行调度流数，本问题$L$取 32。

压缩数据量目标： $\boldsymbol{H}_{n,k}$、$\boldsymbol{W}_{n,k}$分别压缩后，存储量小于原始 FP32 数据的 1/10。

计算目标： $\boldsymbol{W}_{n,k} = \boldsymbol{H}_{n,k}\left(\boldsymbol{H}_{n,k}\boldsymbol{H}_{n,k}^H + \boldsymbol{R}_{uu,n}\right)^{-1}$，$n = 1,\dots,8/16\ /64$

计算精度要求：$|\boldsymbol{W'}_n - \boldsymbol{W}_n|^2 < \varepsilon$，计算结果误差的归一化 MSE 小于-40dB (即归一化幅度误差<0.01)。

计算复杂度：包含压缩/解压缩及计算过程的复杂度小于原始不压缩计算基线 的 1/3。

注：信道 **H** 可以用开源软件 Quadriga 生成，给定信道配置类型。

压缩和解压缩：通过减小数据量的存储空间方法，可参考图像和视频的压缩。压缩和解压缩配套使用，最终计算满足精度要求即可。不限制压缩方法：考虑矩阵/张量的低秩分解，变换域等减小有效数据量的压缩方法；也可以考虑更低 bit 存储，减少数据量的压缩方案。

目标

满足计算结果误差的归一化 MSE 小于-40dB (即归一化幅度误差<0.01)的条件下：

1、存储量降低：针对上行接收机均衡和下行波束成型发送的主要存储矩阵，存储量小于原始 FP32 数据的 1/10。

2、计算复杂度降低：包含压缩/解压缩及计算过程的复杂度小于原始不压缩计算基线的1/3。

指南方向 10：高条件数矩阵分解&求逆低位宽计算

背景

无线通信中常见的矩阵算子包含 SVD、Cholesky、QR，以及共轭矩阵求逆，通常在收发信机中有较高的计算开销。尤其是在高干扰、高调制阶数等恶劣的场景下，为了保证矩阵分解的精度，高条件数的矩阵通常需要更大的位宽需求，也意味着更高的芯片面积和能耗开销。

问题描述

本问题希望从矩阵条件数识别、矩阵分解算法设计、数据表征方法入手，最终形成完整的底层计算和位宽设计方案，实现低复杂度的矩阵分解、求逆。相关的芯片底层矩阵计算，涉及 SVD、Cholesky、QR，以及矩阵求逆：

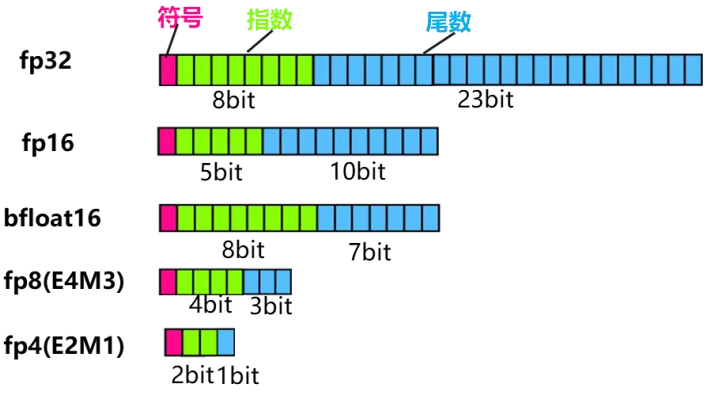
$$U * S * V^H = SVD(A)$$
$$L * L^H = CHOL(A)$$
$$Q * R = QR(A)$$
$$A^{-1} = inv(A)$$

矩阵的条件数识别旨在实现动态的位宽设计。通过低复杂度的识别矩阵条件数，对低条件数矩阵采用小位宽设计；对高条件数矩阵采用大位宽设计，最终达成系统级整体复杂度最优设计。

常见的矩阵分解算法包含 Jacobi、分块 Cholesky、Modified Gauss Seidel 等。本问题考虑在此基础上做优化创新，形成新的矩阵分解算法。在底层矩阵计算中不仅需要考虑算法本身的乘加计算复杂度，算法的位宽需求对整体的计算开销也需要考虑在内。另外，不同条件数下保持算法架构的统一，能够避免过多的硬件开销。

底层数据的表征形式是位宽设计的关键。常见的表征形式包括 FP、BFP、ANT、MX 等。不同的数据表征对应的位宽需求不同，从而影响整体的计算开销。

表征形式和位宽说明：以 FP16 为例，其位宽是 16bit，包括 1bit 符号位，5bit DAGC 因子和 10bit 数据位。详细信息可以参考文献[1]。其他形式和位宽可参考下图：



还有一类分块量化表征形式，如 MX INT8，下图所示是 8 个复数共用 1 个指数因子，可以节省指数的存储空间。详细信息可以参考文献[2]。

目标

本问题期望设计一套针对任意维度矩阵的分解/求逆算法，可以根据如下表所示的矩阵维度/条件数设定来验证方案的性能、复杂度收益。

针对 4 个矩阵算子，能够识别出高条件数场景，并在满足分解误差不超过 1%的前提下，使得计算复杂度相比基线降低 75%。

注：此处的计算复杂度考虑了位宽的大小；示例：相同复乘次数下，FP32 方案计算复杂度是 FP16 方案计算复杂度的 2 倍。

关于误差的定义：以 SVD 分解为例，误差的定义

$$\frac{||H - USV||_F}{||H||_F} + ||U^H U - I||_F + ||V^H V - I||_F$$

即综合考虑重构误差和酉矩阵的正交性。

矩阵分解/求逆算子	矩阵维度	矩阵条件数	基线算法/位宽	目标
SVD 分解	64*4,128*4,256*4	10^4～10^5	Jacobi(FP32)	分解误差不超过 1%，计算复杂度相比基线降低 75%
Cholesky 分解	共轭对称阵 64,128,256	10^4～10^5	分块 Cholesky 分解[3](FP32)	
QR 分解	80*16,160*32,320*64	10^2～10^3	Modified Gauss-Seidel[4](FP32)	
共轭对称求逆	共轭对称阵 64,128,256	10^4～10^5	分块 Cholesky 分解[3]+分块三角阵求逆(FP32)	

- 参考文献

[1] https://en.wikipedia.org/wiki/Half-precision_floating-point_format

[2] https://www.opencompute.org/documents/ocp-microscaling-formats-mx-v1-0-spec-final-pdf

[3] Chen, Jianping, et al. "Block Algorithm and Its Implementation for Cholesky Factorization." (2013).

[4] Barlow, Jesse L. "Block Modified Gram--Schmidt Algorithms and Their Analysis." SIAM Journal on Matrix Analysis and Applications 40.4(2019):1257-1290.

指南方向 11：结构化矩阵求逆

背景

常规矩阵求逆算法，如分块 Cholesky 求逆，在其串行计算过程中，多个子矩阵求逆可能存在大量相同分块的求逆运算。但由于其求逆结果往往与前后迭代的子阵求逆运算相关，因此是无法复用。如何减少矩阵间交叠部分的重复计算，降低总计算复杂度，是本问题的主要牵引目标。

问题描述

输入：已知共轭对称阵$\boldsymbol{R} \in \mathbb{C}^{N*N}$，复矩阵$\boldsymbol{H} \in \mathbb{C}^{N*L}$。

求解：结构化矩阵$\boldsymbol{P}_g, \boldsymbol{Q}_g$，$g = 1,2,\dots,G$，使得$\boldsymbol{W}_0 = \boldsymbol{R}^{-1} * \boldsymbol{H}$与如下矩阵$\boldsymbol{W}_1$有最大的列空间相似性：

$$\max tr(\boldsymbol{U}_0^H \boldsymbol{U}_1 \boldsymbol{U}_1^H \boldsymbol{U}_0)$$

$$\boldsymbol{W}_1 = [\boldsymbol{P}_1(\boldsymbol{P}_1^H \boldsymbol{R} \boldsymbol{P}_1)^{-1} \boldsymbol{P}_1^H * \boldsymbol{H}\boldsymbol{Q}_1, \dots, \boldsymbol{P}_G(\boldsymbol{P}_G^H \boldsymbol{R} \boldsymbol{P}_G)^{-1} \boldsymbol{P}_G^H * \boldsymbol{H}\boldsymbol{Q}_G] \in \mathbb{C}^{N*L}$$

其中，$\boldsymbol{U}_0, \boldsymbol{U}_1$分别为$\boldsymbol{W}_0, \boldsymbol{W}_1$列空间的正交基构成的矩阵。$\boldsymbol{P}_g, \boldsymbol{Q}_g$满足如下约束：

- $\boldsymbol{P}_g$维度为$N * M_g$，满足$N > M_g$，是一个列抽取矩阵，其每一列仅有一个取值为 1 的非零元，且列之间正交，即满足$\boldsymbol{P}_g(n,m) \in \{0,1\}, {\boldsymbol{P}_g}^H * \boldsymbol{P}_g = \boldsymbol{I}$。
- $\boldsymbol{Q}_g$维度为$L * L_g$，满足$L = \sum_g L_g$，是一个列抽取矩阵，其每一列仅有一个取值为 1 的非零元，且列之间正交，即满足$\boldsymbol{Q}_g(n,m) \in \{0,1\}, {\boldsymbol{Q}_g}^H * \boldsymbol{Q}_g = \boldsymbol{I}$。另外对于$t \neq g$，满足$\boldsymbol{Q}_g$和$\boldsymbol{Q}_t$的列空间正交。简单的来看，矩阵$[\boldsymbol{Q}_1, \dots, \boldsymbol{Q}_G]$可以由单位阵通过列重新排列得到。

目标

对于典型维度$N = 128, \mathrm{L} = 12$，以及$N = 256, \mathrm{L} = 24$，限制$M_g \leq \frac{3}{8}N$，达成如下目标：

a) 性能目标：$\frac{tr(\boldsymbol{U}_0^H \boldsymbol{U}_1 \boldsymbol{U}_1^H \boldsymbol{U}_0)}{L} > 95\%$

b) 复杂度目标：求解$\boldsymbol{P}_g, \boldsymbol{Q}_g$以及计算$\boldsymbol{W}_1$的复杂度，相比通过分块 Cholesky 分解求逆[1]和矩阵乘计算$\boldsymbol{W}_0$的复杂度降低 50%。

- 参考文献

[1] Chen, Jianping, et al. "Block Algorithm and Its Implementation for Cholesky Factorization." (2013).

指南方向 12： 基于函数插值的低复杂度并行计算

背景

无线通信系统中的射频收发信机电路，如功放，存在非线性特征，影响信号的质量与频谱。采用数学建模进行非线性系统的求逆与辨识，产生与模拟电路相反的非线性成分进行对消，可以有效降低模拟电路产生的非线性失真。高精度低复杂度的数学校正模型，可以帮助功率放大器（PA）提升效率 5%+，降低数字芯片功耗，面向 5G Massive MIMO，预期提升集成度 50%。

技术现状

对于非线性系统辨识，可表征为如下数学形式：

假设$\{x_{2n}\}_{\mathbb{Z}_{\geq 0}}$是$X \sim N(0, \sigma^2)$ 的$i.i.d.$ 采样，由于系统需求，$\{x_{2n}\}_{\mathbb{Z}_{\geq 0}}$经过线性插值获得$\{x_{2n+1}\}_{\mathbb{Z}_{\geq 0}}$，共同组成高速信号$\{x_n\}_{\mathbb{Z}_{\geq 0}}$。公式如下：

$$x_{2n+1} = \sum_{l=-L}^{L} s_l x_{2(n-l)}$$

$\{x_n\}_{\mathbb{Z}_{\geq 0}}$经过非线性函数$f$计算后输出$\{y_n\}_{\mathbb{Z}_{\geq 0}}$：

$$y_n = f(x_{n-M}, \cdots, x_{n-1}, x_n, x_{n+1} \cdots, x_{n+M})$$

其中，f代表系统的非线性函数，以记忆多项式模型为例，$a_{q,m}$为记忆多项式系数，其表达式如下：

$$y_n = \sum_{m=-M}^{M} \sum_{q=1}^{Q} a_{q,m} x_{n-m} |x_{n-m}|^{q-1}$$

典型配置：$Q = 7, M = 10$，其它常见的非线性模型形式见附录。

实际系统需要单位时间内计算并输出S个y_n，但由于硬件速率限制，单个函数单位时间只能运算出T个$y_n (T < S,$ 且 $T|S)$；常见解决方案是将高速串行计算转换为低速并行计算，以“空间换时间”。 $S/T \in \mathbb{Z}^+$倍并行计算过程为，使用$P = S/T$个模型$\{f_k = f, \ \forall k \in \{0, \dots, P-1\}\}$，单位时间内同时计算输出得到

$$y_{Pn+k} = f_k(x_{Pn+k-M}, \cdots, x_{Pn+k-1}, x_{Pn+k}, x_{Pn+k+1} \cdots, x_{Pn+k+M}),$$

而后等价实现单位时间内运算出$T * \frac{S}{T} = S$个y_n。然而，需要P倍的资源（对应所有模型中浮点计算次数等）去并行实现S/T个模型。

问题描述

以 2 倍并行为例，希望利用$\{y_{2n}\}$信息并结合函数插值理论，预测(或部分预测)$\{y_{2n+1}\}$，同时预测资源少于原先计算$\{y_{2n+1}\}$的资源且精度较高。

目标

令f为模型，$\{x_n\}$为输入信号，$y_n = f(x_n)$，$\mathcal{T}(.) \in \mathbb{R}^+$为函数/网络资源统计函数（通常以模型乘法次数度量）

寻找插值函数

$$\hat{f}:\{(y_{2(n-L)},\ \ldots,\ y_{2(n+L)})\}_{\mathbb{Z}_{\geq L}} \to \mathbb{C},\ s.t.\ L \in \mathbb{Z}_{\geq 0},\ 0 < \frac{\mathcal{T}(\hat{f})}{\mathcal{T}(f)} \leq 0.2,\ 且\ 10\log_{10}\left(\frac{\|\hat{f}-B\|}{\|U\|}\right)^2 \leq -65$$

其中，$B(y_{2(n-L)},\ \ldots,\ y_{2(n+L)}) = y_{2n+1}:\{(y_{2(n-L)},\ \ldots,\ y_{2(n+L)})\}_{\mathbb{Z}_{\geq L}} \to \{y_{2n+1}\}_{\mathbb{Z}_{\geq 0}}$，

$U(2n+1) = y_{2n+1}:2\mathbb{Z}_{\geq 0} \to \mathbb{C}$, $\|.\|$为函数的 2 范数。

附录：常见的非线性映射f数学表达式，如下所示：

(1) GMP [1]，典型配置：Q=6，M=10，L1=L2=2

$$f_{GMP}(x_n) = \sum_{m=-M}^{M}\sum_{q=1}^{Q} a_{q,m}x_{n-m}|x_{n-m}|^{q-1}$$

$$+ \sum_{m=-M}^{M}\sum_{q=2}^{Q}\sum_{l=1}^{L_1} b_{q,m,l}x_{n-m}|x_{n-m-l}|^{q-1} + \sum_{m=-M}^{M}\sum_{q=2}^{Q}\sum_{l=1}^{L_2} c_{q,m,l}x_{n-m}|x_{n-m+l}|^{q-1}$$

(2) DVR [2]，典型配置：K=8，M=10

$$f_{DVR}(x_n) = \sum_{m=-M}^{M}\sum_{k=1}^{K} a_{k,m}\big||x_{n-m}| - \beta_k\big|e^{j\theta_{n-m}}$$

$$+ \sum_{m=-M}^{M}\sum_{k=1}^{K} b_{k,m}\big||x_{n-m}| - \beta_k\big||x_n|e^{j\theta_{n-m}} \text{ where } e^{j\theta_{n-m}} = \frac{x_{n-m}}{|x_{n-m}|},\ \beta_k = \frac{k}{K}$$

(3) Multistage Cascaded Model [3]，典型配置：P=4，K=4，M=10，L1=L2=2

$$f_{MCM}(x_n) = f_{GMP}(f_{DVR}(x_n))$$

(4) RVTDNN [4]，典型配置：M=10, K=20

$$u_n^{(k)} = \tanh\left(\sum_{m=-M}^{M} a_{m,k} Real[x_{n-m}] + b_{m,k} Imag[x_{n-m}]\right)$$

$$v_n^{(p)} = \sum_{k=1}^{K} c_{k,p} u_n^{(k)}$$

$$f_{RVTDNN}(x_n) = v_n^{(0)} + i v_n^{(0)}$$

- 参考文献：

[1] D. R. Morgan, Z. Ma, J. Kim, M. G. Zierdt, and J. Pastalan, “A generalized memory polynomial model for digital predistortion of RF power amplifiers,” *IEEE Transactions on Signal Processing*, vol. 54, no. 10, pp. 3852–3860, 2006.

[2] A. Zhu, “Decomposed vector rotation-based behavioral modeling for digital predistortion of RF power amplifiers,” *IEEE Trans. Microw. Theory Techn.*, vol. 63, no. 2, pp. 737–744, 2015.

[3] R. Criado, W. Li, W. Thompson, G. Montoro, K. Chuang, and P. L. Gilabert, “Model-Order Reduction of Multistage Cascaded Models for Digital Predistortion,” *IEEE Journal of Microwaves*, vol. 5, no. 1, pp. 137–149, Jan. 2025.

[4] T. Liu, S. Boumaiza, and F. M. Ghannouchi, “Dynamic behavioral modeling of 3G power amplifiers using real-valued time-delay neural networks,” *IEEE Transactions on Microwave Theory and Techniques*, vol. 52, no. 3, pp. 1025–1033, Mar. 2004.

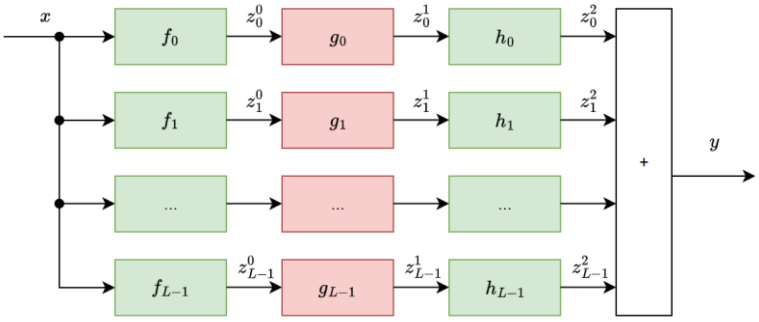
指南方向 13：多层模型的近似误差回传/训练算法

背景

为了降低基站能耗，信号传输的关键器件射频功率放大器（PA）的效率越来越高，导致校正其非线性所对应的数字预失真模型越来越复杂，层级越来越深，模型能力也随之提升，但是也导致训练/解算的代价也越来越大。

现状

记 x为数字域复信号 ，y为输出复信号，复杂级联模型为$x \to \sum_k (L^k(g^k(f^k(x)))$。模型中不同模块针对输入$x$的数学表达式如下，其中$c_i(n)$为模型迭代参数：



函数f和h表示线性函数，g表示非线性函数，公式分别如下：

$$f_i:\ z_i^0(n) \quad = \sum_m c_i^0\,(m) x_i(n-m)$$

$$h_i:\ z_i^2(n) \quad = \sum_k c_i^2\,(k) z_i^1(n-k)$$

$$g_i:\ z_i^1(n) \quad = \sum_p \sum_j \sum_q c_i^1\,(n) z_i^0(n-q) |z_i^0(n-j)|^{p-1}$$

各个参数意义：

- i: 表示本层中的第i个模块
- m, k, q, j：表示模块的对应记忆深度
- p：表示多项式阶次
- n: 离散时间

问题描述

训练采用的 SGD（随机梯度下降）的算法，分为两步：误差回传和系数求导。

目标函数：

$$argmin_c\ J$$

其中，

$$J = \sum_n e_n^* e_n = \sum_n (d_n - y_n)^* (d_n - y_n)$$

其中，d为期望，y为模型输出。

每层的误差回传为：

$$E^2 = \frac{\partial \sum_n e_n^* e_n}{\partial (z_i^2)^*} = \frac{\partial \sum_n e_n^* e_n}{\partial (z_i^2)^*}$$
$$E^1 = \frac{\partial \sum_n e_n^* e_n}{\partial (z_i^1)^*} = E^2 \cdot \frac{\partial (z_i^2)^*}{\partial (z_i^1)^*}$$
$$E_0 = \frac{\partial \sum_n e_n^* e_n}{\partial (z_i^0)^*} = E^1 \cdot \frac{\partial (z_i^1)^*}{\partial (z_i^0)^*}$$

每层系数求导：

$$\Delta_{c_i^2} = \frac{\partial \sum_n e_n^* e_n}{\partial (c_i^2)^*} = E^2 \cdot \frac{\partial (z_i^2)^*}{\partial (c_i^2)^*}$$
$$\Delta_{c_i^1} = \frac{\partial \sum_n e_n^* e_n}{\partial (c_i^1)^*} = E^1 \cdot \frac{\partial (z_i^1)^*}{\partial (c_i^1)^*}$$
$$\Delta_{c_i^0} = \frac{\partial \sum_n e_n^* e_n}{\partial (c_i^0)^*} = E^0 \cdot \frac{\partial (z_i^0)^*}{\partial (c_i^0)^*}$$

系数更新：

$$c_i^2(n) = c_i^2(n-1) - \mu \Delta_{c_i^2}$$
$$c_i^1(n) = c_i^1(n-1) - \mu \Delta_{c_i^1}$$
$$c_i^0(n) = c_i^0(n-1) - \mu \Delta_{c_i^0}$$

目标

前向模型和训练全 online 的条件下，在给定模型，性能损失<0.3dB、收敛速度、稳定性的情况下，回传和训练的资源降低 30%以上。

- **参考文献**

[1] Jaderberg, Max, et al. "Decoupled neural interfaces using synthetic gradients." International conference on machine learning. PMLR, 2017.

[2] Adamson, Reece. "The Forward-Forward Algorithm: Characterizing Training Behavior." arXiv preprint arXiv:2504.11229 (2025).

[3]. Li, Qinyu, Yee Whye Teh, and Razvan Pascanu. "NoProp: Training Neural Networks without Back-propagation or Forward-propagation." arXiv preprint arXiv:2503.24322 (2025).

[4]. Hinton, Geoffrey. "The forward-forward algorithm: Some preliminary investigations." arXiv preprint arXiv:2212.13345 2.3 (2022)

[5]. Criado R, Li W, Thompson W, et al. On the parameter identification of cascaded behavioral models for wideband digital predistortion linearization[C]//2024 IEEE/MTT-S International Microwave Symposium-IMS 2024. IEEE, 2024: 657-660.

指南方向 14：非平稳条件下的自适应回波对消

背景

全双工场景下，收发同频，接收通道受到来自发射信号的自干扰，导致接收灵敏度降低。传统的解决方案使用自适应滤波架构实时跟踪外部信道变化，消除回波对接收通道的干扰。

现状

针对传统的 LMS 提出了很多改进算法，1）将时域系数解算变换到频域，并针对每个频点设置不同步长，解决了信号在频域动态波动大等问题[1]；2）提出子带自适应滤波算法，降低输入信号相关性提升收敛速度[2]；3）提出 PMLMS 算法，提升在稀疏场景下的收敛性能[3]。但这些算法很难满足系统对收敛速度的要求，跟踪动态信道性能受限，并且面对干扰时稳定性不足。

RLS 类算法计算复杂度高，实际落地芯片存在困难。

问题描述

输入数据：

- 发射信号$\boldsymbol{x}_{ref}$为服从复高斯分布的随机信号，为一维向量；
- 接收信号$\boldsymbol{x}_{rx} = \boldsymbol{x}_{si} + \boldsymbol{x}_{ue} + \boldsymbol{n}$，为一维向量，其中，
  a) $\boldsymbol{x}_{si}$为自干扰信号$\boldsymbol{x}_{si} = \boldsymbol{x}_{ref} * \boldsymbol{h}(t)$，$\boldsymbol{h}(t)$表示发射到接收的实际信道，受到环境影响快速非平稳变化(未知，需要被实时估计);
  b) $\boldsymbol{x}_{ue}$为服从复高斯分布随机信号;
  c) $\boldsymbol{n}$为复高斯白噪声；

优化函数：基于发射信号$\boldsymbol{x}_{ref}$和接收信号$\boldsymbol{x}_{rx}$估计外部信道$\widehat{\boldsymbol{h}}(t)$，消除自干扰信号：

$$\min_{\widehat{\boldsymbol{h}}(\boldsymbol{t})} \|\boldsymbol{x}_{err}\|^2 = \left\|\boldsymbol{x}_{rx} - \boldsymbol{x}_{ref} * \widehat{\boldsymbol{h}}(\boldsymbol{t})\right\|^2$$

目标

1、 给定数据完成回波对消，对消能力大于 60dB；

$$10\log_{10}\left(\frac{\|x_{si}\|^2}{\left\|x_{si} - x_{ref} * \widehat{\boldsymbol{h}}(\boldsymbol{t})\right\|^2}\right) = 10\log_{10}\left(\frac{\|x_{si}\|^2}{\|x_{err} - x_{ue} - n\|^2}\right) > 60$$

2、 计算复杂度$O(N)$，N为 FIR 阶数。

- **参考文献**

[1] Soo J S, Pang K K. Multidelay block frequency domain adaptive filter[J]. IEEE Transactions on Acoustics, Speech, and Signal Processing, 1990, 38(2): 373-376.

[2] Lee K A, Gan W S. Improving convergence of the NLMS algorithm using constrained subband updates[J]. IEEE signal processing letters, 2004, 11(9): 736-739.

[3] Duttweiler D L. Proportionate normalized least-mean-squares adaptation in echo cancelers[J]. IEEE Transactions on speech and audio processing, 2002, 8(5): 508-518.

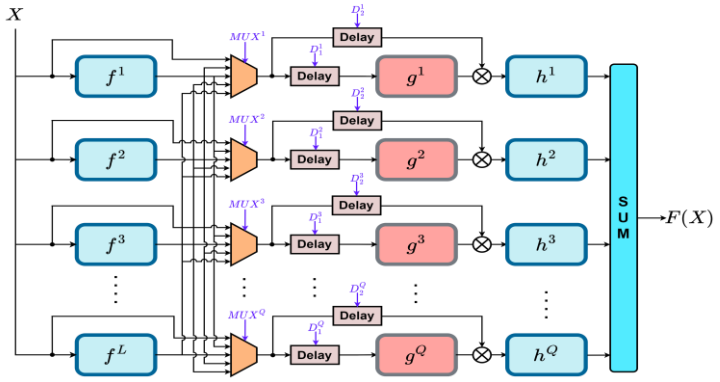
指南方向 15：高效计算-芯片算法的低功耗实现-低比特位宽前向网络

背景

通信算法需要支持的带宽和速率越来越大，算法所需功耗/芯片面积也越来越大，位宽设计是降低硬件开销的重要技术路径，希望研究典型的位宽设计算法，降低硬件开销。

现状

如下图所示模型是一种常见的建模复杂非线性的级联结构，建模能力通常较非级联结构强，模型中的位宽一般选取为 12~16bit，位宽的大小直接影响了芯片的资源。



问题描述

优化目标：

$$\min_{B_X,B_\alpha,B_c,B_\beta} Cost(F,X)$$

$$\text{st. } \|F(X) - F_\infty(X_\infty)\| \le e$$

参数说明：

其中，$X = [X(1), X(2), \ldots] \in \mathbb{C}^N$为输入信号序列，$N$为输入信号长度；$X_\infty$表示$X$全精度浮点版本；$e \in \mathbb{R}^+$为期望误差上限；$Cost(\cdot)$为代价函数

前向函数$F(\cdot)$由下列子函数按图所示拓扑组合而成；$F_\infty(\cdot)$表示 $F(\cdot)$的全精度浮点版本：

$$f^i: y^i(n) = \sum_{m=0}^{M} \alpha_m^i x^i(n-m), i = 1,2,\ldots,L$$

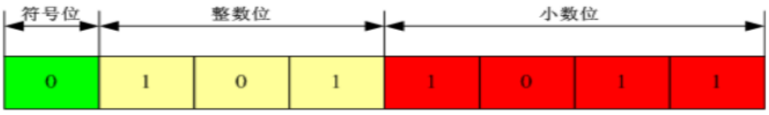
$$D_j^i: y^i(n) = x^i(n - D_j^i),\ i = 1,2,\ldots,Q, j = 1,2$$

$$g^i: y^i(n) = \sum_{p=0}^{P} c_p^i |x^i(n)|^{2p}, i = 1,2,\ldots,Q$$

$$h^i: y^i(n) = \sum_{m=0}^{M} \beta_m^i x^i(n-m), i = 1,2,\dots,Q$$

其中，$D_j^i, p \in \mathbb{z}^+$； $\alpha_m^i, c_p^i, \beta_m^i \in \mathbb{C}$；$B_X, B_\alpha, B_c, B_\beta$分别表示信号$X$，子函数系数$\alpha_m^i$，$c_p^i$，$\beta_m^i$的位宽。

案例：B =8



目标

1. 一种针对前向网络结构的位宽搜索和分析方法，位宽相对基线优化40%；
2. 使能最小化如上网络中各个节点信号和系数的位宽，同时保证位宽压缩后输出信号$F(X)$，与无限精度（浮点）输出信号$F_\infty(X_\infty)$之间误差小于设定值e(e<0.5dB)。

- **参考文献**

[1] Dinis D C, Cordeiro R F, Barradas F M, et al. Agile single-and dual-band all-digital transmitter based on a precompensated tunable delta–sigma modulator. IEEE Transactions on Microwave Theory and Techniques, 2016, 64(12): 4720-4730.

[2] Gholami A, Kim S, Dong Z, et al. A survey of quantization methods for efficient neural network inference. Low-Power Computer Vision. Chapman and Hall, CRC, 2022: 291-326.

[3] Nagel M, Fournarakis M, Amjad R A, et al. A white paper on neural network quantization. arXiv preprint arXiv:2106.08295, 2021

[4] X. Liu, W. Chen and Z. Feng, "Broadband Digital Predistortion Utilizing Parallel Quasi- Wiener-Hammerstein Model with Extended Dynamic Range," 2021 IEEE MTT-S International Wireless Symposium (IWS), Nanjing, China, 2021, pp. 1-3.

指南方向 16： 计算图调度与内存复用方案联合寻优

背景

无线通信的密集域算法计算图中包含数百个算子与张量，每个算子都需要被调度为合适的定制的张量/矢量指令，从而使得整张计算图的时延负载最低。单算子的平均调度方案10+，整图算子调度空间大于 10^100,导致无法快速完成算子调度方案寻优。同时由于硬件数据内存大小限制，计算图中的各个张量也需要进行内存复用，但各个张量占用的内存大小均是动态变化的，不同场景下存在显著差异，如何生成合适的内存复用方案使得在任意场景下内存总大小均不会超过硬件限制，也逐渐成为核心挑战。

现状

1：算子调度：

单算子的调度方案可以由软件工具自动选择，但算子间存在相互限制关系，硬件上并不能实现所有单算子的最优调度，因此计算图级别的算子调度方案寻优仍然依靠专家经验。

2：内存分配：

软件工具可以根据计算图上的数据依赖关系自动生成内存分配方案，并通过符号执行等方式评估出该内存复用方案是否超过硬件限制，但是无法直接生成不会超过内存上界的内存复用方案。

问题描述

给定以下输入条件：

1. 无线密集域算法计算图$G=(V,E)$， 其中 V 表示节点的集合（每个节点表示一个算子），E 表示边的集合（每条边表示一个张量）；
2. 定制张量/矢量指令集 $I=\{i_1, i_2,\ldots, i_m\}$ 以及当算子使用某个指令执行时的时延开销$L(v,i)$；
3. 计算图中表示各个张量大小以及控制流相关的参数集合为$x=(x_1,x_2,\ldots,x_n)$，给定每个参数的取值范围 $D=\{x \in Z^n \mid l_i \le x_i \le u_i, \forall i=1, \ldots, n\}$ 以及参数之间的约束关系；
4. 每个张量的内存大小计算函数$m(e)= f_e(x), e \in E$ 。

给定以下约束：

硬件内存大小限制M_{max}；

待寻优/决策变量：

1) 算子调度方案：$f: V \to I$；
2) 内存分配方案：$g: E \to \{1,\ldots,k\}$, $G_j=\{e \in E \mid g(e)=j\}$；

目标函数：

$$\min_{f,r} \sum_{v \in V} L\big(v,\ f(v)\big)\ \ with \sum_{j=1}^{k} \max_{e \in R_j} f_e(e)\ \leq\ M_{max}$$

目标

1．设计动态规划或整数规划等算法，评估内存复用方案的理论最优上界。内存上界评估值与理论误差低于 ＜10%；

2．基于评估算法，生成内存复用方案，使得内存上界不超过硬件限制（＜150Kb）；

3. 基于入参约束与取值范围，在有限时间内求解最优算子调度方案。

- **参考文献**

[1]] Zhang, X., Chen, Y., & Wang, M. (2021). IOS: Inter-Operator Scheduler for CNN Acceleration. In Proceedings of the MLSys Conference

[2] Liu, J., Zhao, T., & Li, K. (2020). Operator Fusion and Scheduling for Efficient Deep Neural Network Inference on Specialized Hardware. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 39(7), 1234–1245.

[3] Chaitin, G. J., Auslander, M. A., Chandra, A. K., Cocke, J., Kennedy, K., & Reddy, U. S. P. (1981). Register allocation via graph coloring. Computer Languages, 6(1), 47–57.

[4] Cooper, K., & Torczon, L. (2004). Engineering a register allocator. In Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI '04) (pp. 113–124)

[5] Chen, T., Moreau, T., Jiang, Z., Zheng, L., Yan, E., Shen, H., ... & Chen, Y. (2018). TVM: An Automated End-to-End Optimizing Compiler for Deep Learning. In OSDI '18: Proceedings of the 13th USENIX Symposium on Operating Systems Design and Implementation (pp. 578–594).

[6] Chen, H., Li, Y., & Zhang, D. (2021). Efficient Operator Scheduling via Full-Stack Compiler Techniques for DL Accelerators. In Proceedings of PLDI (pp. 234–245).

[7] Wang, M., Li, S., & Zhou, Q. (2022). Domain-Specific Operator Scheduling in Deep Learning: A Survey. IEEE Micro, 42(3), 45–57.

指南方向 17：多用户多载波组合寻优

背景

载波聚合(CA)是提升用户体验的关键技术，基站需要按照站级配置，小区信息和 UE 上报的能力组合信息进行载波组合寻优然后将最优组合配置给用户，但是随着基站频点规格，小区规格，邻区规格以及载波聚合规格的不断提升，用户上报的能力组合规格也不断增加，基站进行组合寻优的计算代价越来越大，在已知单个用户进行载波组合寻优的数学问题是冲突超图的背包问题，其是 NP-HARD 问题，而多个用户上报的能力组合存在相似和相同，其存在较大的优化空间，那么能否利用多用户的信息构建 UE 能力组合和最优载波组合之间的关系，从而降低多用户场景下载波组合寻优的计算代价。

多用户多载波组合寻优问题可抽象为如下路径规划问题，只在问题约束上会有些许简化；

问题描述

A 国的旅游业非常发达。每年接待上万游客。

A 国有若干个省。每个省有若干个城市。每个城市在 历史、地理、美食、文化 上有各自的大众评分。

对于每个游客，有若干个旅游计划。请帮每个游客选出总评分最高的计划。

a) A 国城市信息描述：

cityNum: A 国总的城市数.

对于每一个城市，给定如下信息：

城市 1	示例
city name	南京
province	江苏
history score	90
geograph score	95
food score	100
cultrue score	100

对应的输入描述如下：

city_num：城市个数

city_name[city_num]: 每个城市的名字

city_province[city_num]: 每个城市的省份

city_history_score[city_num]: 每个城市的历史评分

city_geography_score[city_num]: 每个城市的地理评分

city_culture_score[city_num]: 每个城市的文化评分

city_food_score[city_num]: 每个城市的美食评分

b) 游客计划信息描述：

planNum[i]:第 i 个游客的旅游计划数量.

对第 i 个游客的第 j 个旅游计划，给定如下信息：

计划 A		示例
province number		2
江苏	*history score set*	{70, 80, 90, 100}
	geograph score set	{85, 95}
	food score set	{95, 100}
	cultrue score set	{60, 70}
安徽	*history score set*	{70, 80, 90, 100}
	geograph score set	{85, 95}
	food score set	{95, 100}
	cultrue score set	{60, 70}

单个游客对应的输入信息如下：

plan_num：候选计划数

For i from 0 to plan_num-1

plan[i].province_num：第i个计划包含的省份的总个数

for j from 0 to province_num - 1

plan[i].province[j].province：第i个计划里面的第 j 个省份对应省的名称

plan[i].province[j].historyScoreSet ：第i个计划里面的第 j 个省份，可以去的历史评分的集合

plan[i].province[j].geographScoreSet：第i个计划里面的第 j 个省份，可以去的地理评分的集合

plan[i].province[j].foodScoreSet：第i个计划里面的第 j 个省份，可以去的美食评分的集合

plan[i].province[j].cultrueScoreSet：第i个计划里面的第 j 个省份，可以去的文化评分的集合

c) 输出

单个游客对应的输出信息如下：

target_plan_index：选中的旅游计划

for i from 0 to plan[target_plan_index].province_num -1

plan[target_plan_index].province[i].city_index：目标计划中，每个省选中的城市

d) 约束条件描述：

对于每个游客，其输出需要满足如下条件：

- 0 ≤ target_plan_index < plan_num; #target_plan_inde 的有效性判定
- 0 ≤ city_index < city_num # city_index 的有效性判定
- For each prov_index from 0 to plan[target_plan_index].province_num -1
  - city_index = plan[target_plan_index].province[prov_index].city_index
  - city_province[city_index] = plan[target_plan_index].province[prov].province #目标城市在该省范围内
  - city_history_score[city_index]∈plan[target_plan_index].province[prov]. historyScoreSet #目标城市的历史评分在范围内
  - city_geograph_score[city_index] ∈ plan[target_plan_index].province[prov]. geographScoreSet #目标城市的地理评分在范围内
  - city_food_score[city_index] ∈ plan[target_plan_index].province[prov]. foodScoreSet #目标城市的美食评分在范围内

- city_culture_score[city_index] ∈ plan[target_plan_index].province[prov]. cultureScoreSet #目标城市的文化评分在范围内

e) 评分函数描述：

对于每个游客，其评分函数如下：

$$planScore(plan\ A) = \sum_{for\ each\ province\ p\ in\ plan\ A} provinceScore(A,p)$$

$$provinceScore\,(plan\ A,\ province\ p) = MAX(score_{city\ 1,\ A,p},\ score_{city\ 2,A,p},\ \dots,\ score_{city\ k,A,p}),\quad k\ is\ number\ of\quad city\ in\ province\ p.$$

$$score_{city\ i,A,p} = \begin{cases} SCR_{city\ i,A,p} \begin{cases} if\ city_i's\ province = p \\ if\ city_i's\ history\ score \in A's\ p's\ history\ score\ set \\ if\ city_i's\ food\ score \in A's\ p's\ food\ score\ set \\ if\ city_i's\ culture\ score \in A's\ p's\ cultrue\ score\ set \end{cases} \\ 0, \quad otherwise \end{cases}$$

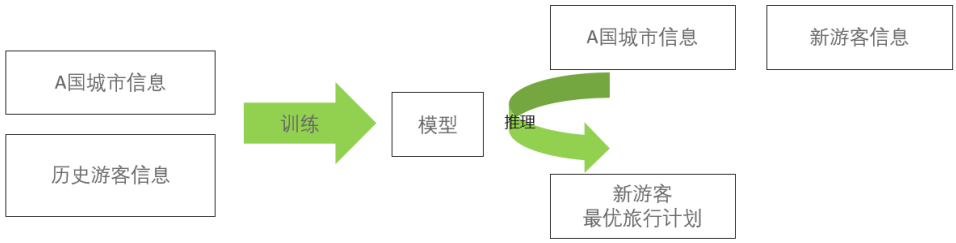
$$SCR_{city\ i,A,p} = city_i's\ history\ score\ *\ city_i's\ geograph\ score\ *\ city_i's\ food\ score\ *\ city_i's\ culture\ score$$

目标

该问题的传统解决思路是遍历所有可能的组合，然后选择最优的。该方案的弊端是遍历的 case 太多，时间复杂度太高。

寻找新的优化方法（不限于 AI）来更快的找对最优解。即，通过历史上众多游客及其选择的最优旅游计划来训练 AI 模型。训练完成后，在基于该新模型为新来的游客找最优旅游计划。

相较于传统算法，新方案的计算开销减少 75%的（为新游客找最优旅游计划的）。



致谢

衷心感谢在指南编制过程中给予指导和支持的专家学者和工作人员，包括：北京大学张继平、胡俊、范少锋、徐婷、于欣卉、王亚坤、张姣娜，华中师范大学刘宏伟、罗金泉、王国栋，兰州大学王记增、杨兵，湖南大学白敏茹、郑力彬，华东师范大学叶青杰，大湾区大学杨斯崑，山东大学冯新伟，吉林大学贾继伟，武汉大学向华、蒋维，中国科学技术大学张举勇、肖东，山东国家应用数学中心郭欣、严晓东、林一伟、韩昊辉，四川大学宋恩彬、陈刚，南京大学刘克勤，多伦多大学王林勃，莫斯科大学 Andrey Shafarevich、Alexander Ivanov、Alexander Zheglov、Vladimir Bogachev、Andrei Krylov，俄罗斯高等经济大学 Ivan Arzhantsev、Sergei Kuznetsov、Vasilii Gromov、Andrey Delitsyn，俄罗斯科学院索伯列夫数学研究所 Dmitry Tkachev、Adil Yerzin、Yuri Kochetov，俄罗斯国家人工智能发展中心 Evgeny Burnaev 等。