



通智超量一体化算力融合框架研究

刘鹏¹, 武振宇¹, 梁焱瑞¹, 顾慧琼¹, 刘申易¹, 刘畅¹, 涂艳丽², 郑坤松¹

(1. 中国移动通信集团设计院有限公司, 北京 100080;

2. 中国移动通信集团设计院有限公司重庆分公司, 重庆 401123)

摘要: 通算、智算、超算与量子计算的异构特性及分散管理, 是制约数字经济高质量发展的关键瓶颈。提出通智超量一体化算力融合框架 H-Cube 及链式架构 SMART 链, 构建“资源-任务-调度”三维协同体系以实现异构算力全生命周期管理。资源维定义异构算力向量空间, 以 FP16 为统一基准完成算力度量标准化; 任务维围绕三类核心场景, 通过凸优化建立任务与算力的精准匹配模型; 调度维基于马尔可夫决策过程 (MDP) 设计动态机制, 平衡近远期调度收益。该框架突破传统单一维度调度局限, 填补异构算力协同理论空白, 为算力网络工程化落地提供技术路径。

关键词: 通智超量一体化; H-Cube 框架; SMART 链; 马尔可夫决策过程

中图分类号: TP393

文献标志码: A

doi: 10.11959/j.issn.1000-0801.2025284

0 引言

算力作为数字经济时代的核心生产力, 其规模与协同能力直接决定数字产业的发展上限。近年来, 随着人工智能、大数据、量子计算等技术的突破, 算力形态呈现多元化演进, 通用计算 (通算) 支撑基础信息处理, 智能计算 (智算) 赋能 AI 模型训练与推理, 超级计算 (超算) 解决复杂科学问题, 量子计算则在特定场景展现指数级加速潜力。然而, 四类算力长期处于“各自为阵”的状态: 通算与智算资源调度割裂导致 AI 赋能成本居高不下, 超算与通用算力协同不足造成资源闲置, 量子计算与经典算力缺乏标准化接口难以落地应用。这种异构算力的分散化管理, 已成为制约数字经济高质量发展的关键瓶颈。

政策层面, 国家高度重视算力协同问题。2024 年, 中国移动在多个行业峰会中提出“两个新型”升级计划, 明确算力基础设施向“通智超量一体化”演进的目标; 2025 年, 工业和信息化部印发《算力互联互通行动计划》, 要求建立统一算力标识体系, 促进跨域算力资源互用。在此背景下, 构建一套能够整合通智超量资源、实现高效协同的技术框架, 成为算力网络演进的核心课题。

本文立足于异构算力协同需求, 提出通智超量一体化融合框架 H-Cube 及链式架构 SMART 链, 通过三维立体设计 (资源、任务、调度) 实现算力资源的全生命周期管理。本研究不仅填补了异构算力协同理论的空白, 更为算力网络的工程化落地提供了可参考的技术路径。



1 通智超量算力发展现状

1.1 算力形态的多元化演进

通算、智算、超算、量子计算在技术特征与应用场景上呈现显著差异，形成能力互补但协同分散的算力生态。

通算：以通用CPU为核心，支撑云计算、数据库、网站门户服务等基础业务，具有普适性强、部署成本低的特点。截至2024年，我国通算算力规模占整体算力规模接近70%，但资源利用率仅为50%~60%，存在严重的闲置浪费。

智算：以GPU、ASIC等加速芯片为核心，专注于AI模型训练与推理。随着大模型技术的爆发，智算需求呈指数级增长，2025年中国智能算力规模将达到1 037.3 EFLOPS，预计2028年将达到2 781.9 EFLOPS，但智算资源与通算资源的跨域调度成本高，导致AI赋能传统行业的门槛居高不下。

超算：以大规模集群为基础，聚焦气象预报、航空航天等科学计算，具有高并行、低延迟的网络特性。我国超算中心算力利用率约70%，但多局限于科研领域，与产业应用的协同不足。

量子计算：基于量子叠加与纠缠原理，在密码破解、材料模拟等场景具有潜在优势。目前处于NISQ（嘈杂中等规模量子）阶段，需与经典算力协同才能发挥价值，但缺乏标准化的融合接口。

1.2 算力协同的核心痛点

当前算力体系面临三大核心问题。

(1) 资源异构性障碍：通算、智算、超算、量子计算的硬件架构、编程模型、性能指标差异显著，缺乏统一的度量标准与标识体系，导致跨类型算力调度困难。例如，智算任务难以直接调用超算资源，量子计算结果无法高效融入经典计算流程。

(2) 任务适配性不足：不同任务对算力的需求存在显著差异（如实时性、精度、成本），现

有调度机制多基于单一算力类型设计，难以实现任务与算力的最优匹配。例如，金融反欺诈分析既需要通算的稳定性，又需要量子计算的加速能力，但现有框架无法协同两类资源。

(3) 动态调度效率低：算力资源状态（如利用率、成本）与任务需求（如并发量、时延要求）均处于动态变化中，传统静态调度策略难以应对复杂环境。例如，电商促销期间的突发算力需求，常因调度滞后导致服务中断。

2 一体化融合框架

2.1 通智超量一体化融合体系 H-Cube

本文提出了通智超量一体化融合框架H-Cube，该框架以“Heterogeneous（异构）”与“Cube（立方体）”为核心设计理念，通过三维度协同实现异构算力的一体化管理。其核心思想是将通智超量资源视为一个有机整体，通过资源维（x轴）的标准化度量、任务维（y轴）的精准映射、调度维（z轴）的动态优化，构建异构算力“度量-映射-调度”的闭环体系，通智超量一体化融合框架H-Cube如图1所示。

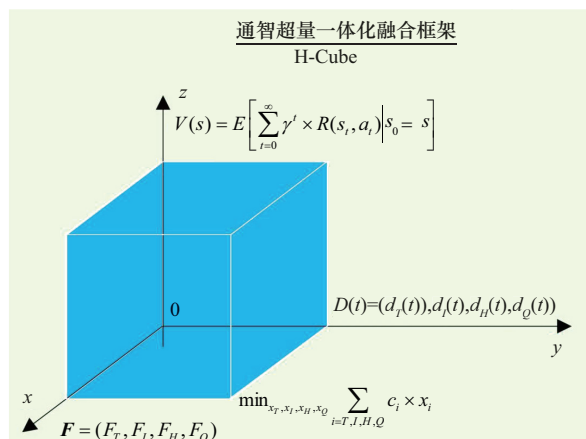


图1 通智超量一体化融合框架 H-Cube

x轴（资源维）：定义异构算力向量空间 $\mathbf{F} = (F_T, F_I, F_H, F_Q)$ ，其中 F_T 为通算算力（如CPU核数、内存容量）， F_I 为智算算力（如GPU算力、

AI 模型吞吐量), F_H 为超算算力 (如浮点运算能力、网络带宽), F_Q 为量子算力 (如量子比特数、相干时间)。通过异构向量空间, 实现算力资源直观呈现、直接调度。

y 轴 (任务维): 基于任务对算力的需求特征, 将其划分为通智一体、量子融合、超算并网三类场景, 通过凸优化方法实现任务需求与算力能力的静态匹配, 确定理论上的最优算力组合。

z 轴 (调度维): 基于马尔可夫决策过程 (MDP), 实时感知算力状态与任务变化, 通过动态调整资源分配策略, 最大化调度效率与收益, 实现理论匹配向实际执行的落地。

H-Cube 的创新性在于突破传统单一维度调度的局限, 通过立体维度的相互支撑形成了稳定的通智超量异构算力的融合架构——资源维为任务映射提供量化基础, 任务维为调度策略提供目标导向, 调度维通过实时感知与动态优化策略进行目标导向, 三者形成自迭代的融合协同机制。

2.2 通智超量一体化融合 SMART 链

为了动态描述 H-Cube 的融合协同机制, 本文基于 H-Cube 框架提出了通智超量一体化融合 SMART 链, 动态描述了“资源-状态-任务-动作-决策-奖励”的链式协同流程, 实现异构算力资

源的全生命周期管理, 通智超量一体化融合 SMART 链如图 2 所示。

S (State, 资源与状态): 实时采集通智超量资源的状态数据, 包括利用率、时延、成本等, 形成可量化的状态向量 $\vec{S} = (s_T, s_I, s_H, s_Q)$, 其中 s_T 代表通算状态, 以此类推。

M (MDP, 马尔可夫决策): 基于状态向量 S , 通过 MDP 模型生成调度策略。MDP 的核心是通过价值函数 $V(s)$ 评估当前状态下的最优动作, 平衡短期收益 (如任务完成率) 与长期收益 (如资源利用率)。

A (Action, 一体化调度动作): 根据 MDP 决策结果, 执行具体调度动作, 包括算力分配 (如将智算任务分配至空闲 GPU)、资源调整 (如超算资源临时支援通算业务)、模式切换 (如“量子-经典”融合模式的切换) 等。

R (Reward, 奖励与价值): 通过奖励函数 $R(s, a)$ 对调度动作进行评价, 例如任务完成率提升记为正奖励, 资源闲置记为负奖励, 形成反馈信号。

T (Task, 一体化任务): 待分配算力的任务需求, 通过任务与资源的映射反映该任务“任务需要什么算力、需要多少”的问题, 进行资源理

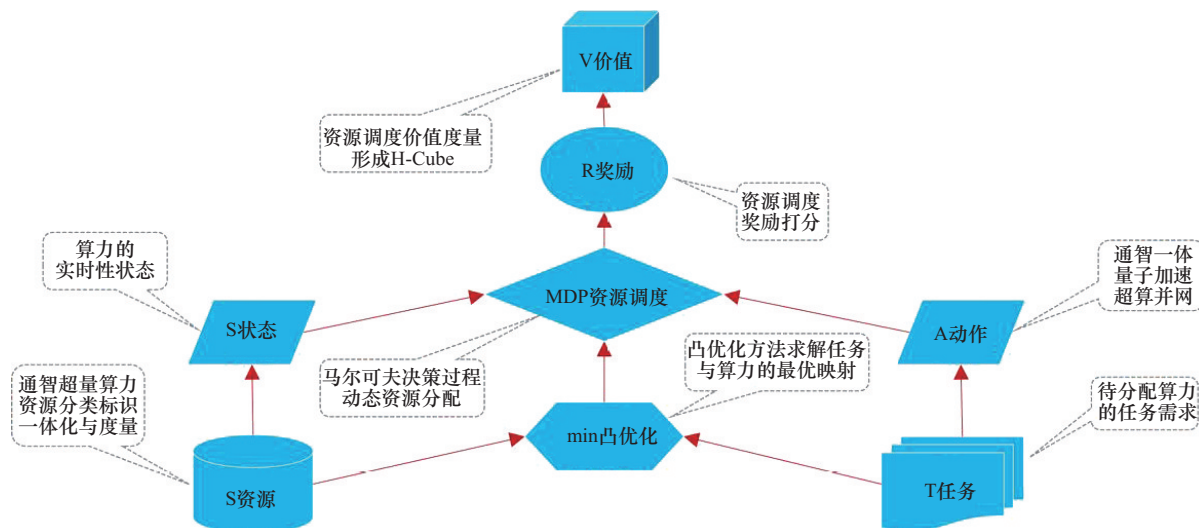


图2 通智超量一体化融合 SMART 链



论规划分配。

SMART 链的优势在于通过“资源-状态-任务-动作-决策-奖励”的闭环设计,实现算力资源的动态适配,既保证了任务需求的精准满足,又最大化了资源利用效率。

下文将针对 H-Cube 框架及 SMART 链的关键要素逐一展开,进行阐述。

3 框架解析

3.1 x轴:资源维度——异构算力的标准化度量

资源维度需要解决两个问题,一是资源直观可视、可用可调,二是资源异构可比、统一度量。

3.1.1 异构算力向量空间

为实现算力资源能够直观呈现、直接调度,通过资源异构算力向量空间直接表征各类算力的特征与调度颗粒度。

$$\mathbf{F} = (F_T, F_I, F_H, F_Q) \quad (1)$$

其中, F_T 是通用算力,调度颗粒度为核数或 vCPU; F_I 是智算算力,调度颗粒度为卡数; F_H 是超算算力,调度颗粒度为核数或 vCPU; F_Q 是量子算力,调度颗粒度为量子比特数×门保真度。

3.1.2 算力度量归一化基准选择

为解决“异构资源不可比”的问题,需要将异构算力进行归一化度量,算力度量归一化的核心挑战在于如何将不同精度、不同架构的算力统一到相同量纲,当前业界存在多种精度标准,包括双精度 FP64、单精度 FP32、半精度 FP16、脑半精度 BF16 以及整数精度 INT8/INT4 等。选择合适的归一化基准需要综合考虑技术趋势、产业共识和实用价值三方面因素。

从技术趋势来看,智算是当前算力需求的主流方向,而 FP16 是智算场景的主导精度,主流 AI 芯片均以 FP16 作为核心算力指标,这种技术趋势使得 FP16 成为最具代表性的智算精度。

在产业共识方面,MLPerf、SPEC 等主流基准测试已将 FP16 作为智算场景的标准精度,截至 2024 年,MLPerf Inference (v3.1) 统计显示,85% 的工业级提交采用 FP16/BF16 精度 (MLCommons, 2023)^①,进一步巩固了 FP16 的产业地位。

从实用价值角度,FP16 数据体量适中,在精度与存储效率之间取得平衡,与 FP32 相比,FP16 可减少一定的内存带宽需求,同时保持足够的精度用于深度学习训练和推理。这种特性使得 FP16 能够适应从边缘设备到数据中心的多样化场景需求。

综合考虑上述因素,本文选择 FP16 (半精度浮点数) 作为统一归一化基准,所有类型算力最终均折算为 FP16 等效值进行比较。

3.1.3 算力度量归一化折算方法

根据不同的算力类型及芯片的特征,分别进行算力归一化度量折算。

(1) 通算场景

Intel Xeon 系列 CPU 基于 AVX512 指令集,其 FP16 性能为 FP64 的 4 倍。

$$\text{TFLOPS@FP16} = \text{TFLOPS@FP64} \times 4 \quad (2)$$

其中, FLOPS@FP64 表示该芯片的 FP64 算力值, TFLOPS@FP16 表示归一化为 FP16 的算力值。

华为鲲鹏 920 系列 CPU,其 FP16 性能约为 FP64 的 8 倍。

$$\text{TFLOPS@FP16} = \text{TFLOPS@FP64} \times 8 \quad (3)$$

其中, FLOPS@FP64 表示该芯片的 FP64 算力值, TFLOPS@FP16 表示归一化为 FP16 的算力值。

(2) 智算场景

NVIDIA 系列 GPU 直接采用专为 AI 优化的 Tensor Core 算力值 (FP16),无需折算。

昇腾 910、寒武纪思元系列芯片也可采用官方直接公布的 FP16 算力值,无需折算。

^① MLCommons 是由 NVIDIA、Intel、Google 等企业主导的开源联盟,其基准结果被行业视为权威标准

(3) 超算场景

由 Intel Xeon 系列 CPU 组成的超算集群，单节点算力折算参照通算折算方法，集群算力采用如下方法折算。

$$\text{TFLOPS@FP16} = \text{TFLOPS@FP64} \times 4 \times n \quad (4)$$

其中，FLOPS@FP64 表示该芯片的 FP64 算力值，TFLOPS@FP16 表示归一化为 FP16 的算力值， n 表示超算集群的节点数量。

由 GPU 节点构成的超算集群，直接累加单节点的算力值即可。

国家超级计算机无锡中心“神威·太湖之光”的申威 26010 处理器未专设 FP16 硬件单元，半精度计算依赖 FP64 单元的“双倍数据并行”，理论峰值为 FP64 的 2 倍。^②

(4) 量子计算场景

量子体积法 (QV) 基于典型量子化学模拟加速比，根据经验系数，在理想无噪声情况下，量子体积与 FP16 算力的换算关系为：

$$\text{FP16}_{\text{equiv}} = \text{QV}_{\text{actual}} \times 1.2 \times 10^{-3} \quad (5)$$

其中， $\text{FP16}_{\text{equiv}}$ 为量子比特等效 FP16 算力， $\text{QV}_{\text{actual}}$ 为实际量子体积，考虑门保真度和电路深度，实际量子体积 $\text{QV}_{\text{actual}}$ 与理论量子体积 $\text{QV}_{\text{theoretical}}$ 的换算关系为：

$$\text{QV}_{\text{actual}} = \text{QV}_{\text{theoretical}} \times F_{\text{arg}} \times \frac{D_{\text{max}}}{D_{\text{ideal}}} \quad (6)$$

其中， $\text{QV}_{\text{theoretical}}$ 为理论量子体积； F_{arg} 为门保真度，取值 (0~1)； D_{max} 为最大可执行电路深度； D_{ideal} 为理想电路深度。

$$\text{QV}_{\text{theoretical}} = 2^n$$

其中， n 为量子比特数量。

该换算基于量子化学模拟中量子计算对经典 FP16 的典型加速比 (约 10^3 倍)，参考美国 IBM 量子团队与 MIT 的协同计算研究，并结合了国家超级计算机中心的工程实践。

量子比特等效法中量子比特与 FP16 算力的

换算关系为：

$$\text{FP16}_{\text{equiv}} = N_q \times F_{\text{arg}} \times T_{\text{coh}} \times 1.2 \times 10^{-3} \quad (7)$$

其中， N_q 为可用量子比特数； F_{arg} 为门保真度，取值 (0~1)； T_{coh} 为平均相干时间，单位为 μs 。

该换算方法基于 NISQ 时代量子性能的多维度衡量理论，结合量子化学场景下的“量子-经典”加速比实验，以及国内量子计算工程实践的经验校准，通过有效量子资源与经典 FP16 算力的场景化映射推导得出。

算法加速比法通过解决问题的时间对比折算与 FP16 算力的换算关系。

$$\text{FP16}_{\text{equiv}} = \frac{T_{\text{classical}}}{T_{\text{quantum}}} \times \text{FP16}_{\text{classical}} \quad (8)$$

其中， $T_{\text{classical}}$ 为经典计算机解决目标问题的时间； T_{quantum} 为量子计算机解决同一问题的时间； $\text{FP16}_{\text{classical}}$ 为经典计算机的 FP16 算力。

该方法基于计算机体系结构中“加速比”的经典定义，结合量子化学、随机电路等场景的“量子-经典”时间对比实验，以及 IBM、国家超级计算机中心的工程实践，通过“时间反比于算力”的假设推导量子的 FP16 等效值。

3.2 y轴：任务维度——任务与算力的精准匹配

任务维度以任务需求与计算能力的适配为核心，通过场景分类拆解任务特征、数学优化构建映射模型，为算力动态调度奠定理论基础。其中，任务映射机制是衔接供需的关键纽带。下文将围绕通智一体（通用与智能算力协同）、量子融合（量子与经典算力耦合）、超算并网（超级算力与公共算力集群互联）三大算力场景，结合任务映射方法展开阐述。

3.2.1 通智一体：智算与通算的协同优化

通智一体场景聚焦智算资源与通算资源的共址部署与协同调用，核心优势在于降低跨域开销、复用基础设施。典型应用包括以下内容。

(1) AI IDE 云服务：通过 AI IDE（如 Cursor、

② 国家并行计算机工程技术研究中心,国家超级计算机无锡中心《神威·太湖之光并行程序设计与优化》技术手册(2025 年)。



VS Code)集成大模型API,用户无需掌握Linux指令,仅通过自然语言prompt即可完成云服务器部署。智算资源提供推理服务,通算资源提供云主机支撑,二者通过本地网络交互,有效解决用户使用云主机而对于Linux指令操作不熟悉的痛点,AI IDE云服务如图3所示。

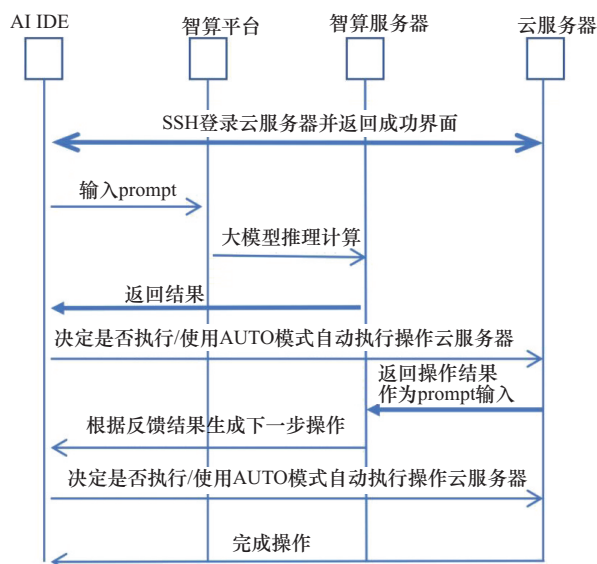


图3 AI IDE云服务

(2) 数据库智能操作: MCP终端通过大模型解析用户prompt,自动生成SQL指令操作数据库。智算服务器负责语义理解,通算服务器(MCP节点)提供数据库工具,协同实现“自然语言&数据库操作”的无缝转换,有效降低SQL技能门槛,数据库智能操作如图4所示。

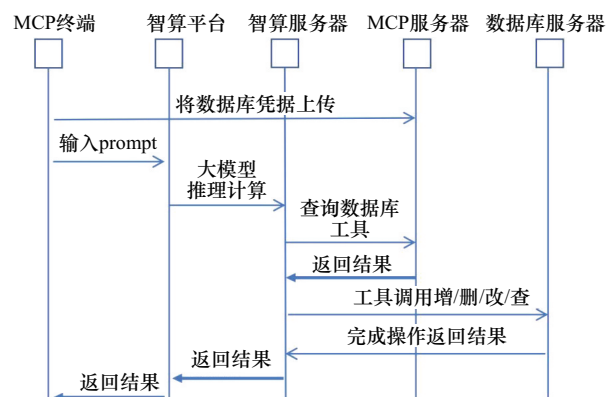


图4 数据库智能操作

(3) 网站AI赋能:在门户网站中集成大模型功能(如DeepSeek),通算资源支撑网页服务,智算资源提供推理能力,数据本地化存储避免信息泄露,减少云外开销,有效提升安全合规性,网站AI赋能如图5所示。

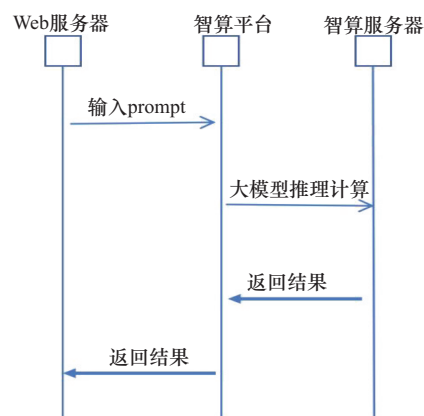


图5 网站AI赋能

总之,通智一体任务的技术路径就是通过智算与通算尽量做到网络共构(如共享交换机、存储池),实现流量本地闭环;通过任务拆分(智算负责AI推理,通算负责业务执行),最大化资源复用效率。

3.2.2 量子融合:量子与经典算力的协同加速

量子融合场景旨在通过“量子-经典”融合计算,发挥量子计算的加速优势与经典计算的稳定性。

(1) 量子融合的必要性

量子计算在特定问题(如大数分解、量子模拟)中具有指数级加速能力,但受限于硬件噪声与相干时间,需与经典计算协同。

- 量子优势:通过量子叠加态同时处理 2^n 个数据(n 为量子比特数),在组合优化、密码分析等场景效率远超经典计算。
- 经典优势:冯·诺依曼架构成熟稳定,硬件错误率 $<10^{-15}$,适合处理逻辑运算、数据存储等任务。
- 融合价值:例如,在金融反欺诈分析中,量子算法将异常交易识别复杂度从 $O(N^2)$

降至 $O\sqrt{N}$, 经典计算负责数据预处理与结果验证, 整体效率提升 1 000 倍以上。

(2) 融合模式

通过对现有各类量子算法加速经典计算的综合分析, 基于量子算力和经典算力的交互形式与频率, 本文将“量子-经典”融合工作模型分为瀑布模型和螺旋模型两大类。

- 瀑布模型: 经典计算负责任务分解与结果处理, 量子计算执行单次核心计算 (如量子搜索), 适用于流程固定的场景。例如大模型超参数搜索, 经典计算机生成候选参数, 量子计算机一次性评估最优解, “量子-经典”融合瀑布模型如图6所示。

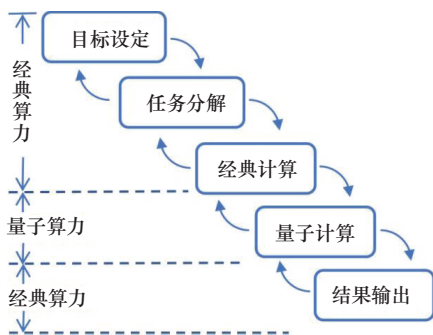


图6 “量子-经典”融合瀑布模型

- 螺旋模型: 量子与经典计算交替迭代, 经典优化器根据量子计算结果调整参数, 适用于持续优化场景 (如变分量子算法 VQE)。例如材料基态能量计算, 量子电路生成近似解, 经典计算机优化电路参数, 多轮迭代后收敛至最优解, “量子-经典”融合螺旋模型如图7所示。

螺旋式融合模式因其能利用经典算力弥补当前量子硬件的设备噪声大、相干时间短等不足, 将大问题分解为多轮小量子任务, 尤其适合匹配组合优化、机器学习等多轮迭代的应用场景, 将产生具有现实意义的算法加速效果。因此, 本文建议经典算力与量子算力应采用螺旋模型作为主要融合方式。

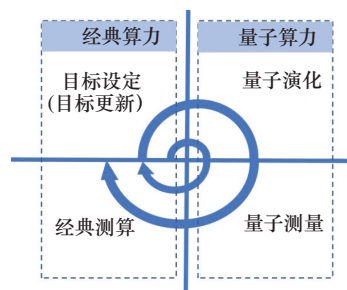


图7 “量子-经典”融合螺旋模型

(3) 融合云平台架构

量子计算机的运行依赖于低温、高真空和电磁屏蔽等苛刻条件, 这使得其操作复杂性和运维成本大大增加, 无法满足用户本地部署的要求。依托经典的云计算基础设施资源构建融合算力云平台, 在现有数据中心部署量子计算机或基于经典计算机构建分布式量子模拟系统, 通过云服务对外提供真实的量子算力或量子模拟算力, 是量子计算从验证探索逐步走向大规模应用落地的关键。构建“量子-经典”融合算力云平台, “量子-经典”融合算力云平台如图8所示。

- 物理资源层: 包含超导/离子阱量子计算机、CPU/GPU 服务器、分布式量子模拟器。
- 管理调度层: “量子-经典”融合计算程序编译过程中会将任务分为经典任务和量子任务, 经典任务交给经典计算硬件 (CPU 或 GPU) 处理; 量子任务交给量子处理单元 (quantum processing unit, QPU) 处理。这两种任务不是完全割裂的, 需要任务间的协同调度。
- 编程框架层: 处于“量子-经典”融合算法和量子/经典算力硬件之间的中间层, 是一个承上启下的重要层级。“量子-经典”融合算法/程序的开发依托编程语言, 算法/程序代码经多层编译后转化为量子/经典硬件可识别的指令集传递给底层物理硬件。

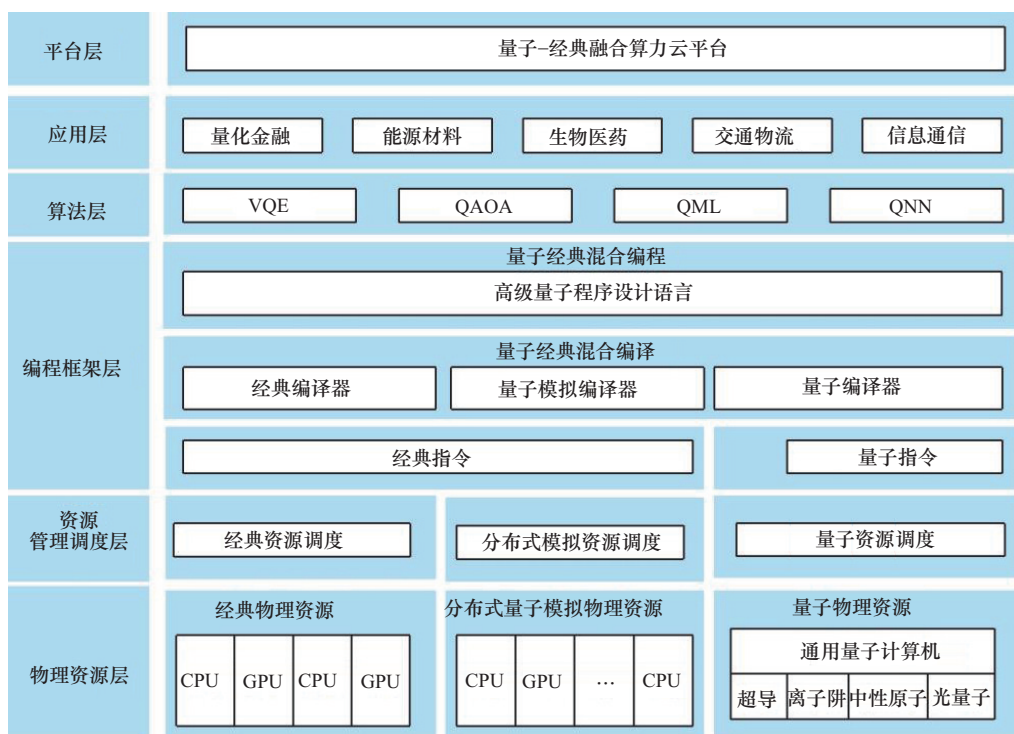


图8 “量子-经典”融合算力云平台

- 算法层：集成VQE、QAOA（量子近似优化算法）、QNN（量子神经网络）等融合算法。
- 应用层：面向金融、材料、生物医药等领域提供加速服务，如量子增强的药物分子模拟。
- 平台层：提供“量子-经典”混合编程接口（如Qiskit、Cirq）、资源调度器（优先分配量子资源）、转换网关（量子测量结果转经典数据）。

3.2.3 超算并网：超算资源的跨域赋能

超算并网旨在突破传统科学计算的局限，通过高速互联网络与通用算力协同，支撑产业级应用，主要有3个方面。

- 通算业务卸载：超算的高速互联特性（如Infiniband网络）可支撑高并发通算任务，例如，电商促销期间，将部分交易处理任务卸载至超算集群，提升响应速度。
- 智算训练加速：超算的大规模存储与并

行计算能力可支撑PB级训练数据的预处理，在模型训练中，使用超算集群提升数据清洗效率，再交由智算节点完成模型训练。

- 行业解决方案：超算与区域算力网络协同，为智慧城市、气象预警等场景提供一体化服务。例如，超算中心生成的高精度气象数据，通过边缘节点实时推送至城市交通调度系统。

3.2.4 任务映射：凸优化方法的应用

为实现“任务需求”与“计算能力”的精准映射，达成供需匹配，本文采用凸优化方法实现任务与算力的最优匹配，其理论基础源于凸优化在资源分配中的经典模型，并在异构算力协同场景中得到验证。通过将任务需求与算力能力建模为凸目标函数与凸约束集，可确保全局最优解的高效求解。凸优化方法是针对目标函数为凸函数、约束条件形成凸集的优化问题，通过系统性算法高效求解全局最优解的数学工具。

(1) 凸优化数据建模

① 目标函数

构建任务算力需求函数：

$$D(t) = (d_T(t), d_I(t), d_H(t), d_Q(t)) \quad (9)$$

其中, $D(t)$ 表示任务在时刻 t 对各类算力的即时需求; $d_T(t)$ 表示任务在时刻 t 对通算的需求; $d_I(t)$ 表示任务在时刻 t 对智算的需求; $d_H(t)$ 表示任务在时刻 t 对超算的需求; $d_Q(t)$ 表示任务在时刻 t 对量子计算的需求。

资源分配目标函数为：

$$\sum_{i=T,I,H,Q} c_i \times x_i \quad (10)$$

其中, c_i 是单位算力成本; x_i 是分配给任务 $d_i(t)$ 的第 i 类算力的资源量, $c_i, x_i \in \mathbf{R}^+$ 。可见, 资源分配的目标函数为线性函数, 而线性函数为凸函数。

② 约束函数

以上目标函数的约束条件为: $x_i \geq d_i(t)$, 即分配的算力资源不低于任务对该类算力的需求, 所以目标函数的定义域为:

$$x_i \in [d_i(t), +\infty) \quad (11)$$

多个 x_i 构成的定义域是凸集。

约束函数可以写成:

$$g_i(x) = d_i(t) - x_i \leq 0 \quad (12)$$

可见 $g_i(x)$ 是线性函数, 为凸函数, 约束形式属于凸约束。

综上, 任务与算力资源映射的优化问题是凸优化问题 (目标函数为凸函数, 约束条件为凸约束, 定义域为凸集), 根据凸优化理论, 该问题可通过凸优化方法求解, 且能保证找到全局最优解:

$$\min_{x_T, x_I, x_H, x_Q} \sum_{i=T,I,H,Q} c_i \times x_i \quad \text{s.t.} \quad c_i, x_i \in \mathbf{R}^+, \quad x_i \geq d_i(t) \quad (13)$$

(2) 凸优化必要性

使用凸优化法求解任务与算力的最优匹配。可以满足实时性需求、多资源协同、成本与需求平衡适配等多种资源映射需求。

① 实时性需求

在算力调度场景中, 任务对算力的需求 $d_i(t)$ 是实时变化的, 凸优化算法具有较好的收敛效率。对于实时更新的任务需求 $d_i(t)$, 可以快速重新优化问题, 得到算力分配方案 x_i , 满足资源分配实时性的要求

② 多资源协同需求

资源分配涉及通算、智算、超算、量子计算四类算力资源的协同分配, 凸优化方法能天然处理多变量、多约束的协同优化场景。将各类算力的成本、需求约束整合, 一次性求解出全局最优的资源分配策略, 避免了不同算力资源单独分配时可能出现的局部最优、资源冲突问题, 保障多类型算力高效协同。

③ 成本与需求平衡

目标函数 $\sum_{i=T,I,H,Q} c_i \times x_i$ 体现了成本考量, 约束 $x_i \geq d_i(t)$ 保障任务需求。凸优化方法求解时, 会在满足任务需求的前提下, 依据成本系数 c_i 合理分配资源, 优先使用成本低的算力, 同时平衡各类算力的使用, 实现“满足需求+成本最小”的双重目标, 适配实际场景中成本控制与业务需求的平衡诉求。

(3) 任务映射与资源分配的差异

在通智超量一体化融合框架 H-Cube 里, 任务映射和资源分配是不同阶段、不同逻辑的环节, 虽然都涉及将算力分给任务, 但目标、逻辑、操作深度有本质区别, 任务映射与资源分配的差异见表 1。

表 1 任务映射与资源分配的差异

维度	任务映射	资源分配
目标	建立“任务需求”与“算力能力”的静态匹配	实现“动态环境”下资源的实时优化调度
逻辑	用如凸优化方法建立任务&资源的理论对应	用动态算法 (如 MDP) 应对环境变化, 调整资源分配
操作	确定任务需要哪类算力、需要多少, 进行理论分配	落地执行, 动态调整, 执行具体分配多少、何时给、分配给谁



3.3 Z轴：资源调度——基于MDP的动态资源调度

马尔可夫决策过程是一种用于在动态环境（本文指通智超量四类算力资源调度场景）中，依据当前算力状态，通过选择动作（分配、调整各类算力资源）并考虑状态转移（执行动作后算力状态如何变化）和奖励（如成本、任务完成率等收益），以优化近远期收益（如最小化成本、最大化算力利用效率）的数学决策框架。调度维通过MDP实现动态环境下的资源优化分配，核心是通过状态感知、动作选择、奖励反馈实现策略迭代。

3.3.1 MDP的核心要素

（1）状态

状态（S）为当前算力的“快照”，本文聚焦算力调度，选择调度相关的算力利用率或闲置率，由通智超量四类算力的当前利用率或闲置率构成算力状态空间S：

$$S = (l_T, l_I, l_H, l_Q) \quad (14)$$

其中， l 表示各类算力当前的闲置率。

（2）动作

动作（A）为调度决策采取什么样的动作，构建动作空间A：

$$A = \{a_1, a_2, \dots, a_n\} \quad (15)$$

其中， a_i 表示某个调度动作，如下：

a_1 表示通智一体加速

a_2 表示量子计算加速通算

a_3 表示量子计算加速智算

a_4 表示量子计算加速超算

a_5 表示超算加速通算

a_6 表示…

根据通智一体、量子加速、超算并网等算力特征，可以构造丰富的动作空间。

（3）奖励

奖励（R）用打分机制优化调度策略，通过

奖励函数R来评价调度策略的优劣，例如：

任务完成率高，+n分

某类算力闲置率 $\geq x\%$ ，-n分

用高成本的量子计算，-n分

（4）状态转移

状态转移概率 $P(S'|S,A)$ 描述执行动作A后，状态从S变到S'的可能性，用来预判执行该调度后，未来资源状态的变化情况，以便于规划更优的策略。

3.3.2 资源调度奖励函数R

资源调度奖励函数综合考虑算力利用率、任务延迟、成本等指标，给调度策略打分：

- 策略好（利用率高、任务快、成本低）
→ 奖励高（分数高）
- 策略差（资源闲置、任务超时、成本爆炸）
→ 奖励低（分数低）

调度算法会朝着“奖励最大化”的方向，自动优化调度策略（比如怎么分配通算、智算资源）。本文设计的资源调度奖励函数如下：

$$R(s_t, a_t) = w_1 \times (1 - \max(l_i)) + w_2 \times C_r - w_3 \times C_s, \quad i \in \{T, I, H, Q\} \quad (16)$$

其中， $R(s_t, a_t)$ 表示在时刻 t 的状态 s_t 下，执行动作 a_t 获得的奖励； l_i 为某类算力的闲置率， $\max(l_i)$ 表示找出最闲置的哪类算力， $1 - \max(l_i)$ 表示闲置越严重，这项得分越低； C_r 为任务完成率，取值范围一般 $C_r \in [0, 1]$ ， C_r 越接近1表示任务完成得越好； C_s 为调度成本，表示调度过程中资源损耗、经济成本等量化值； $w_{1,2,3}$ 为权重系数，用来平衡效率与成本：

- w_1 为利用率为权重，如果重视资源利用率，就将 w_1 调大些，让算法优先减少资源闲置；
- w_2 为任务权重，如果重视任务按时完成，就将 w_2 调大些；
- w_3 为成本权重，如果想控制成本，就把 w_3 调大些；

3.3.3 资源调度的价值 V

用 MDP 的状态价值函数 $V(s)$ 对调度维进行量化评价, $V(s)$ 越大, 说明“处于状态 s 时, 未来能拿到的总奖励越高”, 当前状态越“有价值”。

$$V(s) = E \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) | s_0 = s \right] \quad (17)$$

其中, s 为当前的 MDP 状态, 当前时刻 $t=0$ 的状态是 s (如当前算力闲置率 (l_T, l_I, l_H, l_Q)), 从这个状态开始计算未来奖励; R 为奖励函数, 如上节所述; γ 为折扣因子, 即未来奖励的权重, $\gamma \in [0, 1]$, 用来平衡近期奖励和远期奖励:

- γ 越接近 1: 越重视长期奖励 (如调度策略愿意“牺牲当前成本, 换取未来更高收益”);
- γ 越接近 0: 越重视短期奖励 (如优先完成当前任务, 不管后续影响);

求解价值函数 $V(s)$, 如果预先知道 MDP 的状态转移概率, 可以使用贝尔曼方程通过价值迭代法直接推导计算 $V(s)$, 进而推导最优调度策略。但是算力调度环境复杂, 执行一次调度后, 算力资源的状态难以预测, 状态转移概率难以预先构建, 所以算力资源调度属于无模型特性, 无法预先知道 MDP 的状态转移概率 P , 而是直接与环境进行试错学习, 通过间接优化推导求解 $V(s)$ 。经典的无模型强化学习算法 Q-Learning 就是基于 MDP 框架, 通过学习 $Q(s, a)$, 间接优化 $V(s)$, 最终得到 MDP 问题的最优策略。

MDP 解决动态资源调度的核心逻辑是不断尝试不同的调度策略 (动作), 观察奖励和状态变化; 通过价值函数 $V(s)$ 评价当前状态下选择哪个动作未来获得的总奖励最大, 得到 MDP 的最优策略; 通过折扣因子 γ 平衡短期/长期收益。

4 结束语

本文针对通算、智算、超算与量子计算“各

自为阵”的异构算力管理瓶颈, 提出通智超量一体化算力融合框架 H-Cube 及链式架构 SMART 链, 通过“资源-任务-调度”三维协同, 系统解决了异构算力的标准化度量、任务与算力的精准匹配及动态调度难题——资源维以 FP16 为基准实现算力度量统一, 任务维借凸优化达成供需全局最优匹配, 调度维依托 MDP 平衡近远期调度收益。

该研究不仅突破传统单一维度调度的局限, 填补了异构算力协同的理论空白, 更实现了引言中“构建高效协同技术框架”的目标, 为算力网络的工程化落地提供了可直接参考的技术路径, 助力算力资源从“分散管理”向“智能协同”跨越, 为数字经济发展的算力支撑提供核心保障。

[作者简介]

刘鹏 (1985-), 男, 中国移动通信集团设计院有限公司高级工程师, 资深研究专员, 主要研究方向为智算、云计算等。

武振宇 (1975-), 男, 中国移动通信集团设计院有限公司正高级工程师, 研究总监, 主要研究方向为算力网络、智算中心、云计算等。

梁焱瑞 (1984-), 男, 中国移动通信集团设计院有限公司高级工程师, 高级研究专员, 主要研究方向为云计算、算力网络与绿智算力关键技术等。

顾慧琼 (1982-), 女, 中国移动通信集团设计院有限公司高级工程师, 资深研究专员, 主要研究方向为算力网络、IT 支撑、互联网信息安全等。

刘申易 (1980-), 男, 中国移动通信集团设计院有限公司高级咨询设计师, 主要研究方向为算力网络编排调度、算网大脑等。

刘畅 (1995-), 男, 中国移动通信集团设计院有限公司咨询设计师, 主要研究方向为云计算、智算中心、算力网络等。

涂艳丽 (1981-), 女, 中国移动通信集团设计院有限公司重庆分公司高级工程师, 工程总监, 主要研究方向为云计算、算力网络、智算中心等。

郑坤松 (1998-), 男, 中国移动通信集团设计院有限公司助理研究专员, 主要研究方向为云智算、量子计算等。