计 算 机 研 究 与 发 展
Journal of Computer Research and Development
DOI: 10.7544/issn1000-1239.202440074
61(6): 1351−1369, 2024

机器学习辅助微架构功耗建模和设计空间探索综述

翟建旺[1] 凌梓超[2] 白 晨[3] 赵 康[1] 余 备[3]
[1](北京邮电大学集成电路学院 北京 100876)
[2](北京邮电大学计算机学院 北京 100876)
[3](香港中文大学计算机科学与工程系 香港 999077)
(zhaijw@bupt.edu.cn)

Machine Learning for Microarchitecture Power Modeling and Design Space Exploration: A Survey

Zhai Jianwang[1], Ling Zichao[2], Bai Chen[3], Zhao Kang[1], and Yu Bei[3]
[1] (*School of Integrated Circuits*, *Beijing University of Posts and Telecommunications*, *Beijing* 100876)
[2] (*School of Computer Science*, *Beijing University of Posts and Telecommunications*, *Beijing* 100876)
[3] (*Department of Computer Science and Engineering*, *The Chinese University of Hong Kong*, *Hong Kong* 999077)

Abstract Microarchitecture design is a key stage of processor development. It is at the upper level of the entire design flow and directly affects core metrics such as performance, power consumption, and cost. Over the past few decades, new microarchitecture solutions, coupled with advances in semiconductor manufacturing, have enabled newer generations of processors to achieve higher performance, lower power consumption and cost. However, as chip design enters the post-Moore era, the dividends from the evolution of semiconductor technology are increasingly limited, and power consumption has become a major challenge for energy-efficient processor design. Meanwhile, modern processors are becoming more complex in architecture and the design space is larger, requiring designers to make accurate design metrics tradeoffs to achieve the most desirable microarchitecture design. Moreover, the existing stage-by-stage decomposition of the development and validation flow is extremely lengthy and time-consuming, and it is difficult to achieve global energy efficiency optimization. Therefore, how to perform accurate and efficient power estimation and design space exploration at the microarchitecture design stage becomes a key issue. To tackle these challenges, machine learning has been introduced into the microarchitecture design process, providing efficient and accurate solutions for microarchitecture modeling and optimization. We firstly introduce the main design flow of processors, microarchitecture design and its major challenges, then amplify machine learning-assisted integrated circuit design, which focuses on research advances in the use of machine learning techniques to assist microarchitecture power modeling and design space exploration, and finally conclude with a summary and outlook.

Key words processor design automation; microarchitecture design; power modeling; design space exploration; machine learning

摘 要 微架构设计是处理器开发的关键阶段,处在整个设计流程的上游,直接影响性能、功耗、成本等核心设计指标.在过去的数十年中,新的微架构设计方案,结合半导体制造工艺的进步,使得新一代处理器

收稿日期: 2024-02-01; **修回日期**: 2024-03-19
基金项目: 国家重点研发计划项目(2022YFB2901100); 香港特别行政区研究资助局(CUHK14210723); 北京市自然科学基金项目(4244107)
This work was supported by the National Key Research and Development Program of China (2022YFB2901100), the Research Grants Council of Hong Kong SAR (CUHK14210723), and the Beijing Natural Science Foundation (4244107).
通信作者: 赵康(zhaokang@bupt.edu.cn)

能够实现更高的性能和更低的功耗、成本.然而,随着集成电路发展至“后摩尔时代”,半导体工艺演进所带来的红利愈发有限,功耗问题已成为高能效处理器设计的主要挑战.与此同时,现代处理器的架构愈发复杂、设计空间愈发庞大,设计人员期望进行快速精确的指标权衡以获得更理想的微架构设计.此外,现有的层层分解的设计流程极为漫长耗时,已经难以实现全局能效最优.因此,如何在微架构设计阶段进行精确高效的前瞻性功耗估计和探索优化成为关键问题.为了应对这些挑战,机器学习技术被引入到微架构设计流程中,为处理器的微架构建模和优化提供了高质量方案.首先介绍了处理器的主要设计流程、微架构设计及其面临的挑战,然后阐述了机器学习辅助集成电路设计,重点在于使用机器学习技术辅助微架构功耗建模和设计空间探索的研究进展,最后进行总结展望.

关键词 处理器设计自动化;微架构设计;功耗建模;设计空间探索;机器学习

中图法分类号 TP332

处理器芯片已经成为现代信息社会的基石,是推动新一轮科技革命和产业变革的关键力量[1],对于提升国家战略竞争力和国际地位具有重要意义[2].作为计算机系统的核心和大脑,处理器负责执行各类控制、计算任务,其性能和功耗等设计指标对整个系统至关重要.因此,处理器的设计制造在集成电路和计算机产业中占据了关键地位.

随着半导体工艺的发展,晶体管特征尺寸逐步逼近物理极限,摩尔定律所预测的等比例微缩定律已经难以维持,给处理器的设计制造带来了诸多挑战.随着平面光刻衍射极限[3]、短沟道效应[4]、冯·诺依曼瓶颈[5]等问题愈发突出,现代处理器面临尺寸缩小瓶颈、能耗瓶颈和算力瓶颈等问题.当特征尺寸进入量子效应显著的范围,诸多次级物理效应随之显现,比如源漏寄生电阻占比增大、栅极隧穿泄漏等,导致芯片的功耗密度快速上升,散热问题也限制了处理器的主频提升,集成电路已经进入功耗限制时代[2].

传统的超大规模集成(very large scale integration, VLSI)电路基于分阶段设计的思想,层层分解,在各个阶段进行局部最优化设计,有利于缩减问题规模、提高设计效率.然而,随着处理器规模的急剧攀升,传统设计流程愈发难以收敛于全局能效最优.各阶段的相对独立导致了设计鸿沟的产生,难以支撑跨层次联合设计,无法达到架构、电路、器件跨层优化的性能水平[2].因此,必须采取不同设计阶段和设计模式相协同的思想,考虑不同阶段的相互影响,实现跨层优化,使得最终结果收敛于全局最优,并减少设计迭代.研究人员已经尝试从全系统的角度进行设计,并利用机器学习(machine learning, ML)[6]强大的建模和优化能力打破设计壁垒.左移融合的新型设计范式[7]也对设计工具提出了更高的挑战,研发更加智能高效的新一代电子设计自动化(electronic design automation, EDA)工具势在必行.

对于商业处理器而言,能否在完成既定设计指标的前提下,尽可能缩减物料和开发成本成为产品竞争力的重要考量.因此,在早期架构设计阶段前瞻性地评估性能、功耗、面积(performance, power, and area, PPA)等指标,并探索符合需求的微架构设计显得尤为重要.一般而言,微架构设计方案形成设计文档,并通过工程开发的方式转换为寄存器传输级(register transfer level, RTL)代码,然后利用 EDA 工具完成逻辑综合和物理版图设计.微架构设计所描述的每一个简单语句都可以很轻易地转换为成千上万行 RTL 代码,导致设计复杂度急剧攀升、问题规模爆炸性增长.因此,如何在早期微架构设计阶段较为精确地预估 PPA 指标,并对不同设计方案进行探索和优化,对于提高开发效率至关重要.

目前,手机、汽车电子、AIoT 等各类设备发展迅猛,迫切需要定制化、高能效的处理器设计,以及系统性的芯片敏捷设计方案[8].作为开发流程中极为关键的一步,微架构建模和探索优化的质量和效率决定了能否快速获得理想的处理器芯片,直接影响开发周期和投入产出.本文将介绍微架构功耗建模和设计空间探索领域的研究进展,尤其是如何利用机器学习技术提高建模、优化过程中的性能和效率.

1 背景介绍

1.1 处理器设计流程

VLSI 的设计制造是一个复杂庞大的工程,为了应对不断发展的芯片集成度和半导体工艺,其设计流程也日趋多样化.目前,层次化设计是被采用最多的设计方法.

图 1 给出了处理器芯片的典型设计流程.系统定

义[9]主要根据客户需求定义产品的基本结构、目标和原则，确定系统功能、PPA 以及工艺选择等. 架构设计[10]则给出目标处理器的基本架构，包括模拟和混合数字模块的集成、软硬 IP 核的使用、内存管理、通信、电源要求等. 其中，微架构用于实现给定的指令集架构(instruction set architecture, ISA)，包括该实现的特定机制和硬件结构，并在一定约束条件下使用硬件电路结构进行描述. 功能和逻辑设计[11]则定义了目标设计的功能行为，并通过高层次硬件模型实现，比如使用 Verilog/VHDL 等高级硬件描述语言完成 RTL 代码开发. 在给定工艺库后，逻辑综合[12]将 RTL 代码转换为门级网表，并完成逻辑化简和优化. 在获得门级网表后，基于工艺库信息进行布图规划、布局、布线等后端物理设计[13]步骤，从而将电路元器件和互连关系映射到晶片上. 然后，对物理版图进行物理验证、测试等各项检查验证. 如果满足既定 PPA 设计指标，则通过签核形成标准版图文件交付半导体制造商，并经过一系列工艺制造、封装测试形成最终产品交付客户. 而如果不满足既定设计目标，则需要修改早期设计，并重新执行后续设计流程.

为了获得更加理想的设计结果，处理器开发往往需要大量的反馈迭代，导致开发周期长、全局优化不足、设计成本大幅度增加. 因此，为了避免复杂耗时的后期设计和验证流程，有必要在早期设计阶段从全系统的角度进行跨层优化，并开发更加高效智能的 EDA 工具. 总而言之，在早期的微架构设计空间中进行精确高效的建模和优化，有利于为处理器设计实现良好开端，从而缩短设计周期、降低开发成本.

1.2 微架构设计及其面临的挑战

在计算机系统中，微架构是处理器对特定 ISA 的实现，也称为微体系结构. 这种实现通常是指寄存器、存储器、控制器、算术逻辑单元和其它数字逻辑块的组合实现. 换而言之，微架构给出处理器中存在的所有电子元件和数据路径的逻辑设计，并以特定方式布局，从而实现指令的最佳执行. 微架构与 ISA 相结合组成了整个系统的计算机体系结构，设计人员可以使用不同的微架构设计来实现一个给定的 ISA，从而解决不同的问题，比如加速某种特定任务、降低功耗、提高成本效益等. 如图 1 所示，微架构设计处于整个设计流程的上游，对处理器芯片的 PPA 有关键影响，因此往往需要在多个维度进行权衡.

作为处理器设计方式的逻辑表示，通常将特定的微架构表示为数据流图，用以描述各种组件的互连和交互. 作为示例，图 2 给出了 RISC-V BOOM 处理器的微架构示意图. RISC-V 是一种开源 ISA，因其免费、简洁以及良好的可移植性等特点，受到了广泛关注和支持，并取得了显著发展[14−18]. BOOM[14−15]通过使用 Chisel[19]来构建内核生成器，能够提供一系列乱序 RISC-V 内核设计，以适应不同的应用场景. 如图 2 所示，BOOM 主要由前端(FrontEnd)、指令解码单元(IDU)、执行单元(EU)和加载存储单元(LSU)4 个部分组成. 得益于 BOOM 的可参数化微架构设计，设计人员可以通过配置不同的微架构设计参数，在性能和功耗等指标之间做出不同的权衡. 由于对开发人员友好、性能高、可配置等优点，BOOM 在开源社区备受追捧. 同时，BOOM 为微架构建模和探索优

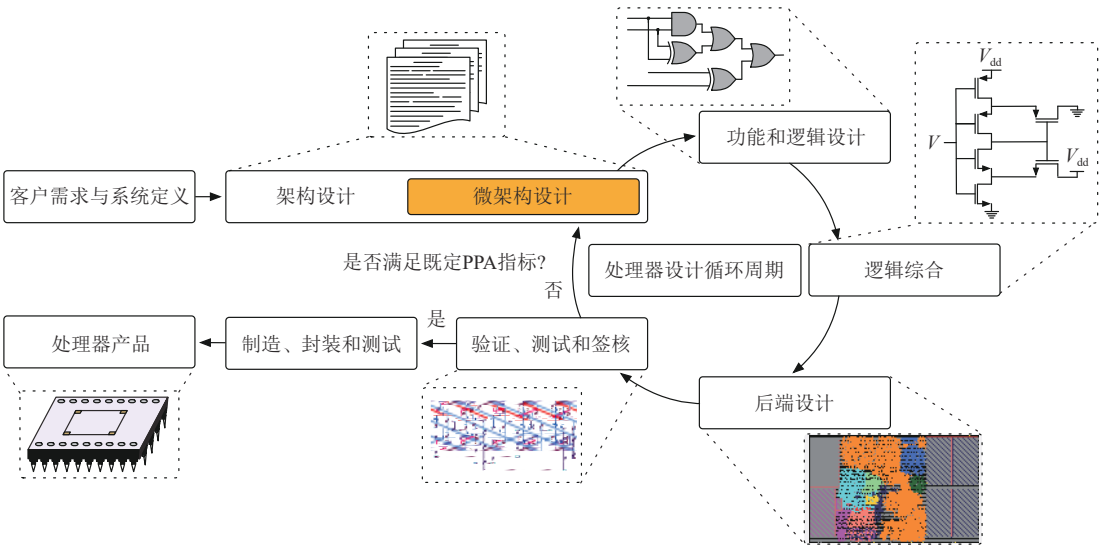


Fig. 1 Illustration of processor chip design flow

图 1 处理器芯片设计流程示意图

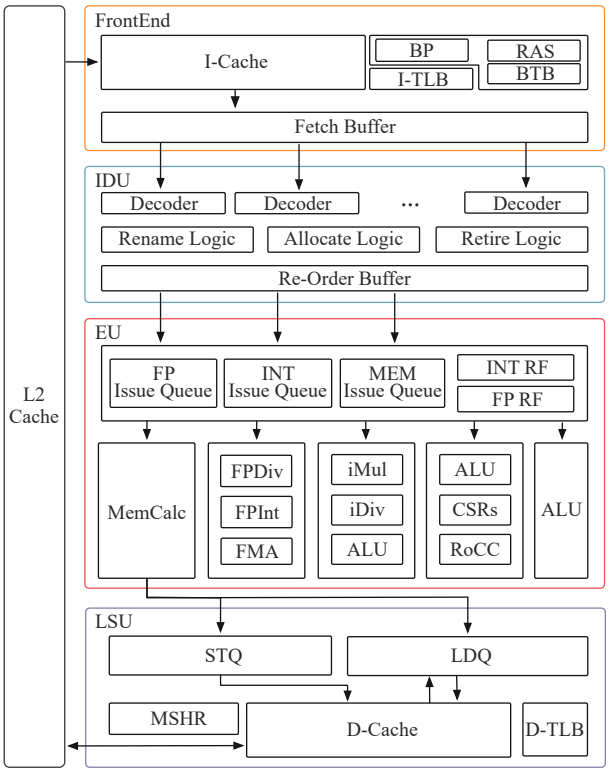


Fig. 2 Illustration of RISC-V BOOM microarchitecture

图 2 RISC-V BOOM 微架构示意图

化提供了很好的机会，本文所调研的很多工作都是基于 BOOM 展开的.

分支预测等各级流水线设计，执行单元的数量、延迟、吞吐量选择，存储器的大小、延迟、吞吐量和连接性等均是微架构设计和决策的核心任务. 随着设计需求的日益复杂和制造工艺的不断发展，设计人员必须最大程度地压缩开发周期，以加快上市时间. 重要的是，必须考虑能效、成本和可扩展性等诸多因素，对流水线和各个组件进行细致地分析和评估，以尽早评估目标设计的 PPA 指标，从而确定最适合特定应用场景的微架构设计方案，以提高处理器的性能和能源效率. 然而，互相冲突的设计指标、漫长耗时的开发流程、复杂庞大的设计空间，给微架构建模和探索决策带来了 3 个严峻的挑战：

1)处理器芯片所关注的设计指标，例如性能和功耗是负相关的，甚至是相互冲突的，无法实现全部指标的同时最优，必须考虑如何做出良好权衡.

2)传统的设计验证流程极为耗时，对给定微架构设计方案进行 PPA 评估通常需要花费大量时间，如何利用有限的早期特征信息进行较为精确的 PPA 预估成为挑战.

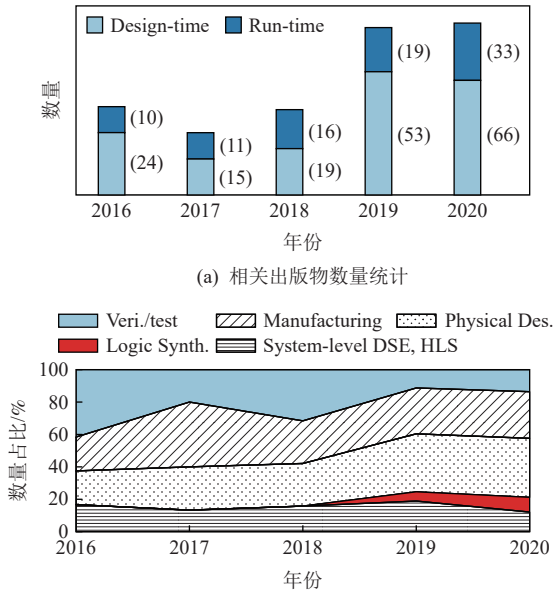
3)现代处理器包含许多复杂的组件，并由此产生了极为庞大的设计空间，几乎不可能遍历每种微架构设计以获得最佳解决方案，如何高效探索全局最优设计成为挑战.

1.3 机器学习辅助集成电路设计

为了解决集成电路设计领域面临的诸多挑战，研究人员尝试应用机器学习等人工智能(artificial intelligence, AI)[20]技术赋能芯片设计. 人工智能的概念于 1956 年首次被提出，并在棋类游戏、计算机视觉、自然语言处理等应用中取得了重大突破，是未来科技发展的重要方向. 机器学习使用计算机模拟人的学习行为，并从数据中学习知识以改进预测模型和智能系统的性能. EDA 应用中的很多问题可以表征为决策问题、回归问题与检测问题，具有多阶段、多目标、不连续、非线性等特点，且面临较大的不确定性和时间压力. 通过应用机器学习技术，可以有效利用历史数据和知识来构建更加准确的电子设计模型、开发处理器的跨层优化方法，提高 EDA 工具的自动化和智能化程度，从而帮助设计人员快速生成高质量的芯片设计.

目前，机器学习在 EDA 领域的应用仍处于探索阶段，但已经取得了一些显著进展. Cadence 发布了基于机器学习引擎的数字全流程，芯片设计吞吐量最高提升 3 倍，PPA 最多改善 20%. Synopsys 则推出了 AI 自主芯片设计系统 DSO.ai™，利用强化学习在芯片设计的庞大解空间中搜索优化目标，在降低设计成本的同时获得 PPA 提升. Mentor(现西门子 EDA)推出了用于更智能设计的 AI/ML 工具包，其中 AI 辅助的高层次综合(high level synthesis, HLS)工具能够快速找到神经网络加速器引擎的最佳 PPA 实现，基于机器学习的光学邻近校正工具和光刻仿真工具能在保持最佳精度的同时大幅度提高设计效率. 得益于数十年的积累，EDA 厂商拥有丰富的全流程工具和流片验证经验，为利用机器学习技术辅助 EDA 工具开发提供了充足的数据和实践积累.

学术界也对机器学习技术在 EDA 领域的应用开展了大量研究. 如图 3 所示，Rapp 等人[21]总结了 2016–2020 年间在五大主要 EDA 学术期刊和会议中，使用机器学习进行集成电路辅助设计的出版物数量及其在不同阶段的占比. 从图 3(a)不难看出，使用机器学习解决 EDA 问题正变得越来越流行. 如图 3(b)所示，在 2020 年，约有 65% 的工作是用于物理设计和制造阶段的，因为这些工作涉及物理版图的几何表示，有利于应用较为成熟的图像相关的机器学习算法. 而在早期设计阶段，如系统级设计空间探索(design space exploration, DSE)、HLS、逻辑综合等相关



(a) 相关出版物数量统计

(b) 不同设计阶段的研究占比

Fig. 3 Statistics of numbers and percentage on EDA publications based on machine learning[21]

图 3 基于机器学习的 EDA 出版物数量及占比统计[21]

研究则较少. 这些问题大多是组合优化问题, 在理论上更加难以求解, 且缺乏领域知识的引入和可扩展性研究. 根据上述统计, 机器学习用于后端物理设计和制造阶段已被广泛研究, 未来需要把更多的精力集中于较早期的设计阶段, 这也是本文的重点所在.

总的来说, 机器学习与 EDA 方法学的融合是革命性的一步, 使得芯片设计生产力产生质的飞跃. 首先, 机器学习强大的建模能力能够为设计流程中较早期阶段提供精准高效的指标评估, 有利于实现跨层优化. 其次, 数据驱动的机器学习模型能够为特定流程和场景提供定制化算法、减少人工干预, 具备更高的自动化和智能程度. 此外, 通过将复杂的 EDA 问题转化为机器学习问题, 能够有效降低 EDA 算法开发门槛, 吸引更多研究人员.

2 微架构功耗建模

集成电路的功耗建模是一个广泛而持久的研究问题. 随着摩尔定律的放缓和登纳德缩放定律的崩溃, 功耗问题已经成为高能效处理器设计的主要挑战, 集成电路已经进入功耗限制时代[2]. 因此, 迫切需要开发精确鲁棒的功耗模型以指导处理器芯片的设计和优化. 研究人员已经针对不同设计阶段提出了一系列建模方法. 然而, 现代处理器架构复杂、设计空间庞大, 现有模型依然难以满足不断增长的建模速度、通用性和准确性需求. 本节将介绍功耗建模问题、相关建模方法, 重点在于机器学习辅助微架构功耗建模, 并进行小结.

2.1 功耗建模问题

作为处理器芯片开发中的主要挑战与优化目标, 需要在设计阶段尽早地对功耗进行建模和估计, 以便对不同设计方案进行探索决策. 在这种情况下, 功耗模型需要具备对不同硬件设计, 以及所运行的不同工作负载进行精确建模的能力, 并助力设计空间探索和优化.

功耗通常指集成电路在单位时间内所消耗的能量, 即所需的电源功率. 一般而言, 集成电路的总功耗 P 包含 3 个主要部分: 开关功耗(switching power)、短路功耗(short circuit power)和漏电功耗(leakage power), 由式(1)给出:

$$P = \alpha C V_{\mathrm{dd}}^2 f_{\mathrm{clk}} + \alpha E_{\mathrm{S}} f_{\mathrm{clk}} + V_{\mathrm{dd}} I_{\mathrm{leak}}. \tag{1}$$

式(1)中第 1 项为开关功耗, 是指电路单元在驱动外部电容性负载切换状态时进行充放电所消耗的能量/功率, 也称为翻转功耗或者狭义的动态功耗. 开关功耗与总负载电容(C)、电源电压(V_{dd})、时钟频率(f_{clk})和活动因子(α)成正比, 其中 α 通常为每个周期的平均开关次数.

式(1)中第 2 项为短路功耗, 是指电路切换期间 PMOS 管和 NMOS 管同时打开形成的短路电流以及单元内部电容充放电所引起的内部功耗(internal power), 内部能量通常以脉冲形式耗散. E_{S} 为每个开关操作的短路能量、f_{clk} 为时钟频率, α 为活动因子.

式(1)中第 3 项为漏电功耗. 晶体管实际上只能充当“不完美”开关, 当未发生状态切换时, 通过晶体管的漏电流仍然会产生漏电功耗. 漏电功耗取决于漏电流(I_{leak})和电源电压(V_{dd}). 漏电流主要包括晶体管沟道和栅极之间的栅极漏电流, 漏极和源极之间的亚阈值漏电流, 以及漏极和衬底之间的反向偏置电流. 随着晶体管特征尺寸和氧化物厚度的缩小, 漏电功耗愈发不可忽视.

开关功耗和短路功耗均是电路开关活动导致的, 可将两项之和称为广义的动态功耗(dynamic power); 而漏电功耗与电路活动无关, 不依赖于开关切换, 相对保持恒定, 也可称之为静态功耗(static power). 通常而言, 在数字电路中, 动态功耗为总功耗的主要来源, 但随着特征尺寸的持续缩小, 漏电流所导致的漏电功耗持续增长, 并导致了登纳德缩放定律的崩溃, 成为了先进制程发展中的极大阻碍.

2.2 相关功耗建模方法

在传统设计流程中，作为黄金标准的功耗估计通常是由门级商业分析工具，例如 PrimeTime PX(PTPX)[22]完成的，这需要以长时间的门级设计及仿真为代价，并且只能在设计流程的较晚阶段使用，难以辅助处理器芯片的跨层优化.

为了避免门级功耗分析的高昂成本，研究人员尝试在更高的抽象层次(即更早的设计阶段)进行功耗建模. 一般而言，越在后期设计阶段进行的功耗建模会越精确，但相应的建模成本也会大幅度提高. 图 4 对用于不同抽象层次的功耗建模工作进行了统计，根据是否需要 RTL 实现和仿真，可以将现有功耗建模方法分为：微架构级和 RTL/门级. 一般而言，门级商业分析工具给出的估计结果被认为是黄金标准值，其他门级功耗模型的误差在 1%~5%；RTL 级模型通常依赖于仿真波形信号，能够实现较为精确的细粒度功耗估计，误差范围为 1%~10%；而微架构级模型通常只能利用架构级的有限信息对不同粒度的平均功耗进行估计，误差范围为 2%~20%.

2.2.1 微架构功耗建模

在整个设计流程中，微架构级别的功耗建模和优化有难以替代的优势，这是因为系统级别的功耗优化技术往往能够实现较高的节能收益. 文献 [23] 的研究显示，电源关断(power shut off, PSO)技术可以实现约 95% 的漏电功耗优化，动态电压频率调节(dynamic voltage and frequency scaling, DVFS)可以带来 30%~60% 的动态功耗节省，多电压域(multiple supply voltage, MSV)架构则可以带来约 40% 的功耗优化. 在主流的设计流程，如 UPF(unified power format)设计流程，首先需要在微架构阶段进行定义，并由综合工具完成低功耗单元(如 isolation cells, level shifter cells)的实现，且要求在后端设计中使用特殊的电源管理库文件(PMK Kits). 显然，快速精确的微架构功耗模型成为开发系统级功耗优化技术的关键前提，使其成为十分典型且亟需解决的跨层优化问题.

在微架构层面，功耗模型通常只能基于有限的微架构设计参数和性能仿真信息等来对不同时间粒度的平均功耗进行建模. 由于无需 RTL 开发和仿真，微架构级的功耗建模通常具有极快的建模速度，以及较高的通用性和可扩展性，但由于难以获得详细的后端电路实现细节，导致建模精度较低.

最为普遍使用的是各类架构级解析功耗模型[24−26]，它们试图建立不同的硬件表示和库函数，并使用来自性能模拟器[27−30]的事件统计信息获得最终的功耗估计. Wattch[24]根据微架构组件的电容模型和架构模拟获得的开关事件计算动态功耗. CACTI[25]使用 ITRS 器件模型和 MASTAR[31]获得不同工艺节点下的器件参数，重点支持基于 SRAM 和 DRAM 的缓存和内存阵列. PowerTimer[32]使用一组参数化的能量函数对微架构性能模拟中的分层组件进行功耗建模. Orion[33]是一种用于估计片上网络功耗的工具，使用重复布线模型进行互连，并使用 MASTAR 和其它方法根据 ITRS 获得器件参数. McPAT[26]基于 ITRS 预测的 CMOS 器件参数，支持 90~22 nm 节点下多核处理器的功耗、面积和时序建模，提供了从架构级别到工艺级别的完整层次式模型. McPAT 具备高度的灵活性和可扩展性，被广泛应用于处理器架构设计，但由于其内部模型与实际设计之间的错位，其建模误差高达 20%~40%[34−35]，且缺乏对先进工艺节点的支持. McPAT-PVT[36]和 McPAT-Monolithic[37]分别将 McPAT

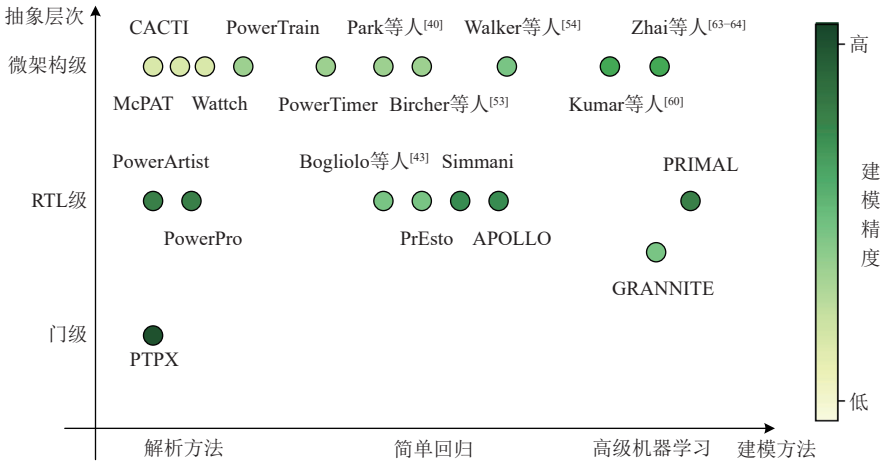


Fig. 4 Comparison of processor power modeling methods

图 4 处理器功耗建模方法对比

扩展到 20 nm 和 14 nm FinFET 工艺节点，但并没有完全解决精度不足的问题. Ravipati 等人[38] 提出了 FN-McPAT，集成了 FinFET 工艺下 BOOM 内核的综合结果，以支持 14 nm FinFET 及 NC-FinFET（负电容鳍式场效应晶体管）工艺. Van den Steen 等人[39] 提出了独立于微架构的机理模型完成处理器的功耗建模，从而加速大规模设计空间探索. Park 等人[40] 在更高的指令级别下进行建模，用于创建多个粒度级别的处理器功耗模型，从而快速映射到电子系统设计流程.

2.2.2 RTL/门级功耗建模

RTL/门级功耗模型需要进行 RTL 或门级网表设计和仿真，其通常是基于仿真所得的波形信号所建立的. 在 RTL 层次，PowerArtist[41] 和 PowerPro[42] 等行业工具可以提供粗粒度的 RTL 总功率估计. Bogliolo 等人[43] 使用基于回归的方法支持以更小的粒度构建功耗模型，但以有限的精度为代价. PrEsto[44] 使用线性模型通过 FPGA 加速来描述不同的电路模块. Yang 等人[45] 使用基于奇异值分解的特征选择技术来构建线性功耗模型. PRIMAL[46] 使用卷积神经网络来处理寄存器切换活动，为可重复使用的电路构件的功耗进行建模. Simmani[47] 使用 VCD dump 来构建一个切换模式矩阵，并通过聚类选择关键信号来构建功耗模型，能够提供更高精度的 RTL 功耗估计. GRANNITE[48] 将门级网表转换为图，并将 RTL 仿真中的寄存器状态和单元输入作为特征来构建图形神经网络模型，以预测门的切换率和平均功耗. APOLLO[49] 使用基于 MCP（minimax concave penalty）正则的线性回归实现特征选择，从而使用少量 RTL 信号来完成逐周期的功耗建模，以用作设计时功耗估计器和运行时片上功耗监控器. Fang 等人[50] 提出了一种用于 RTL 设计的综合前 PPA 估计框架 MasterRTL，首先将 HDL 代码转换为简单运算符图的 bit 级表示形式，在时序建模中捕获详细关键路径信息，并在功耗建模中集成切换率和模块级信息，从而实现对布局后 PPA 值的精确预测.

RTL/门级功耗模型依赖于更详细的硬件细节和仿真波形，能够实现相对较高的建模精度，甚至能够实现逐周期的精确建模. 尽管如此，2 个关键缺点导致这些模型难以应用于早期的微架构设计阶段：1）详细的逻辑设计和仿真需要投入高昂的人力和时间成本，且速度较慢，在早期微架构设计阶段难以使用；2）由于不同设计的硬件网表和仿真波形差别较大，导致所得的功耗模型大多是特定于硬件设计的，在不同的处理器设计间进行迁移面临挑战.

2.3 机器学习辅助微架构功耗建模

随着 AI 领域的飞速发展，机器学习模型的强大建模能力为早期阶段的精确指标评估提供了可能. 截止目前，研究人员已经开发出基于各类统计学习或机器学习的建模方法，实现了更加精确的微架构功耗模型，为处理器芯片的跨层优化提供了有效工具.

早期工作[51] 使用微架构设计参数进行统计回归建模来辅助设计空间探索，但由于缺乏与工作负载相关的事件统计等信息，难以准确地对不同的工作负载程序进行建模.

基于微架构仿真事件的设计时功耗模型[52] 和基于性能监视计数器（performance monitoring counter，PMC）的运行时功耗模型[53−55] 则被更加广泛地使用. Jacobson 等人[52] 使用相对较少的程序事件来进行功耗建模. Bircher 等人[53] 利用了处理器性能事件的“涓滴效应”，进而确定了 6 个 PMC 事件对整个系统功耗进行回归建模. Walker 等人[54] 建立了一个运行时功耗模型，并通过自动选择最佳的 PMC 事件来实现静态功耗和动态功耗的分离. Sagi 等人[55] 使用非线性变换来捕捉 PMC 事件和功耗值之间的关系，并使用最小角度回归来完成多变量的多项式功耗回归建模. Lebeane 等人[56] 提出了 WattWatcher，通过将性能事件计数和硬件描述文件传递到基于 McPAT 的可配置后端模型中，从而进行运行时功耗监控. 如图 5 所示，Reddy 等人[57] 将基于 PMC 的运行时经验模型[53] 转化为设计时微架构功耗模型，并与 gem5 模拟器[29] 相结合，从而实现了用于早期阶段的设计时功耗建模. 然而，机器活动所产生的事件信息通常与具体硬件配置密切相关，导致这些功耗模型多是特定于硬件设计的，难以对新型目标处理器做出良好预测.

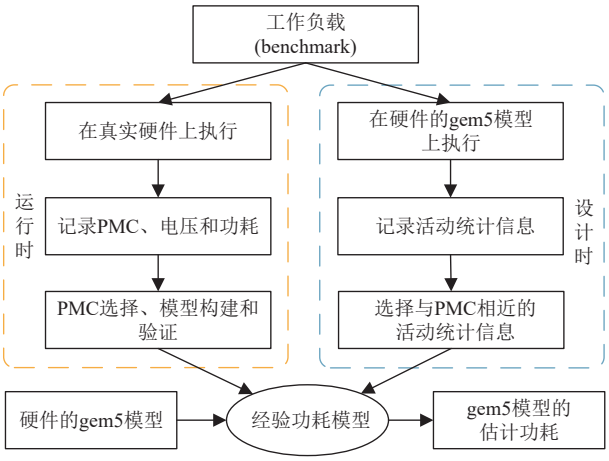


Fig. 5 Converting runtime models to design-time models[57]

图 5 运行时模型转换为设计时模型[57]

Ipek 等人[58−59]利用人工神经网络(artificial neural network, ANN)来捕获架构参数和性能/功耗之间的关系，以此构建准确的预测模型促进设计空间探索. 此外，为了减少所需采样架构的数量以构建满足特定精度约束的预测模型，使用智能采样来实现高效的训练过程. 他们在内存系统、处理器、芯片级多核处理器上验证了其方法的有效性.

如图 6 所示，Kumar 等人[60]从周期精确的微架构仿真中获得高级活动信号，例如每个模块的数据信号、控制信号、混合信号，并基于不同输入信号执行特征选择和特征工程，然后使用非线性回归模型对不同微架构模块构建功耗模型，从而形成分层组合的完整处理器功耗模型. 在实验中，他们使用 Verilator 工具[61]从处理器的 RTL 描述生成周期精确的 C++模型，并在有序执行的 RI5CY 处理器[62]和乱序执行的 BOOM 处理器上进行了验证.

由于其易用性和就绪性，解析功耗模型 McPAT 受到了广泛欢迎，但其建模误差愈发难以接受. 如图 7 所示，为了提高建模精度，PowerTrain[35]基于来自真实硬件的功耗测量结果，使用带有 L_1 正则的线性回归来重新加权 McPAT 中每个组件的估计功耗，从而提供更加精确的运行时功耗估计.

Zhai 等人[63−64]提出了一种将解析建模与机器学习校准相结合的微架构功耗建模框架 McPAT-Calib，其建模流程如图 8 所示. 首先对经典解析模型 McPAT 进行底层改进，然后确定了包含微架构设计参数、活动统计信息、解析功耗建模结果在内的广泛建模特征，并使用自动特征选择和非线性回归等机器学习方法做进一步校准，还设计了一种主动学习采样方法 PowerGS 来降低机器学习模型对标记数据规模的

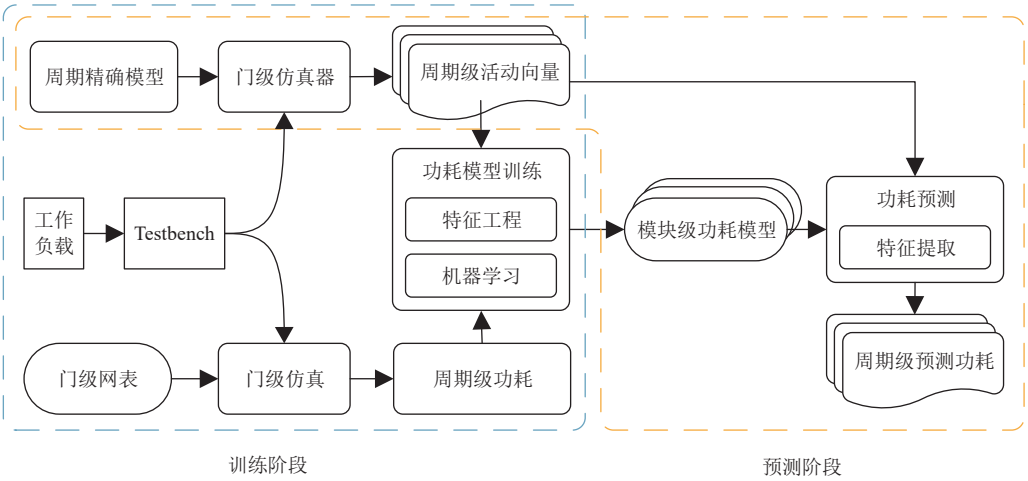


Fig. 6 Power modeling flow proposed by Kumar et al[60]

图 6 Kumar 等人[60]提出的功耗建模流程

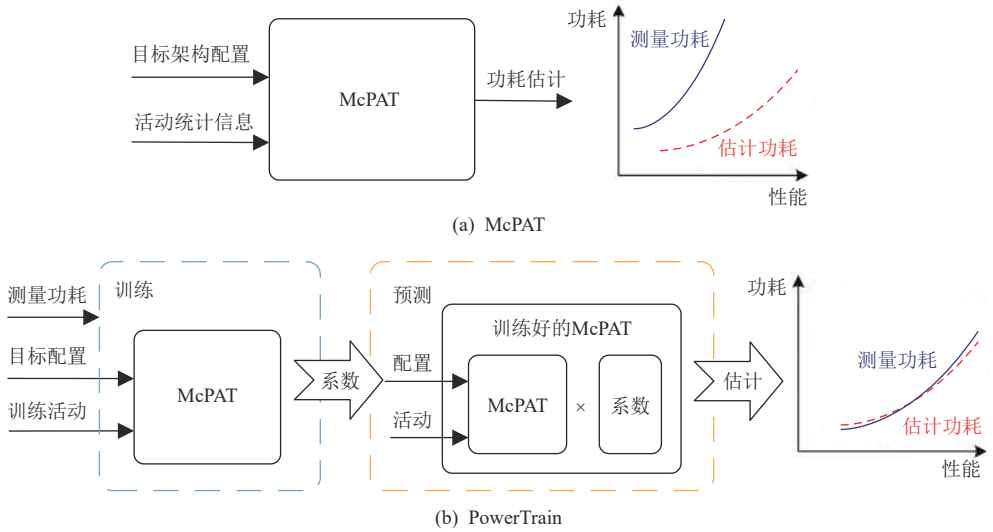


Fig. 7 Illustration of PowerTrain[35]

图 7 PowerTrain 示意图[35]

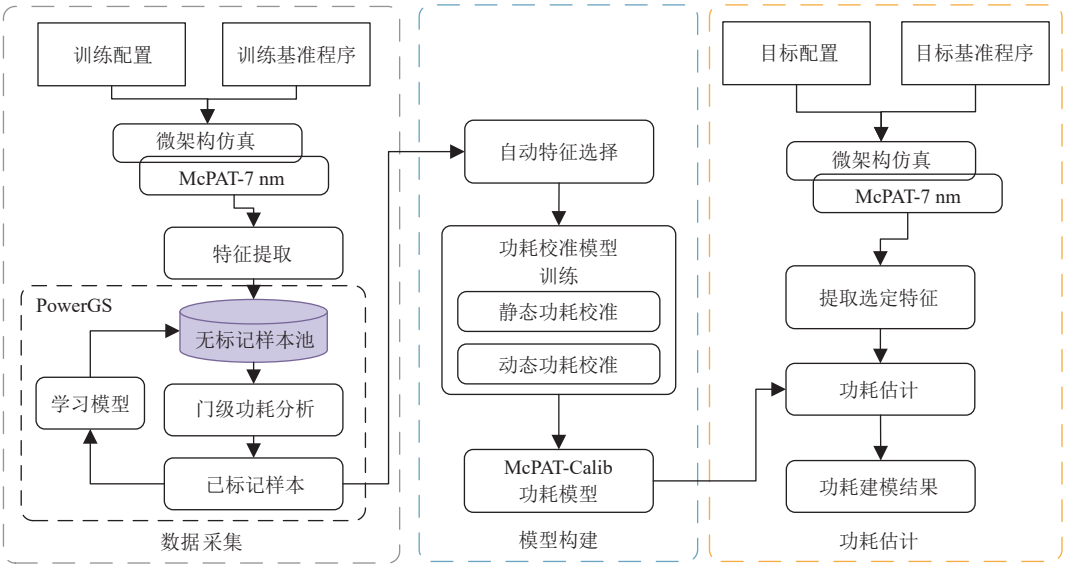


Fig. 8 Flowchart of McPAT-Calib framework[64]

图 8 McPAT-Calib 框架流程图[64]

需求，从而以较低代价实现建模精度和通用性的提升. Zhai 等人使用 BOOM 处理器的 15 种微架构配置和 80 种负载程序进行了验证.

Zhang 等人[65]得出了统一现有架构级功耗模型的通用公式及 PANDA 框架. 如图 9 所示，PANDA 框架结合了解析模型和机器学习模型的优点，在训练数据有限的情况下准确地预测功耗、面积、性能，该框架不依赖于特定解析功耗模型，并支持对未知新工艺节点的预测. PANDA 框架中，每个组件的 PPA 预测模型由 2 个子模型组成，其中非斜线部分表示解析性的资源函数模型，斜线部分则表示机器学习模型. 该框架也在 BOOM 处理器上进行了验证.

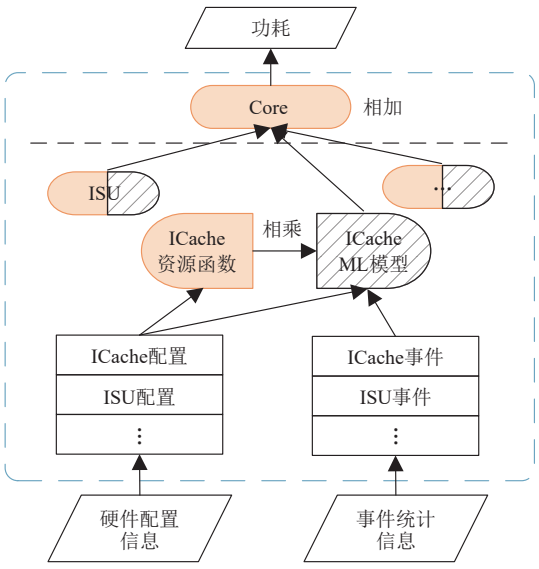


Fig. 9 Illustration of PANDA framework[64]

图 9 PANDA 框架示意图[64]

如图 10 所示，为了提高基于机器学习的微架构功耗模型的可迁移性，Zhai 等人[66]提出了一种基于神经网络和迁移学习的微架构功耗建模方法. 首先，根据先验知识设计基于 ANN 的微架构功耗模型，然后使用跨域混合生成接近于目标分布的辅助样本，并利用改进的域对抗训练完成知识迁移和模型构建. 在 BOOM 处理器不同微架构配置上的迁移实验证明了该方法的有效性.

Wang 等人[67]探索了跨工作负载的可迁移性，提出了一种迁移学习设计空间探索框架 TrEnDSE 来执行跨工作负载的性能预测. 首先，利用 Wasserstein 距离量化工作负载之间的黑盒可迁移性，并作为初始样本权重，设计了可迁移性感知的迁移学习算法自适应调整样本权重；此外，使用了集成装袋(bagging)学习模型和不确定性驱动的迭代优化方法，以利用这些样本权重来执行准确且鲁棒的性能和功耗预测.

如图 11 所示，Li 等人[68]提出了一种基于图神经网络的 PPA 估计框架 NoCeption，用于具有任意不规则拓扑和可变设计参数的片上网络(network on chips, NoC)的快速预测. 他们采用一种通用表示方法将 NoC 和应用任务图转换为属性图，并使用消息传递神经网络(message passing neural network, MPNN)转换为有效的图嵌入特征，从而实现 NoC 的 PPA 预测.

2.4 小 结

总的来说，在微架构设计阶段前瞻性地进行功耗估计有利于实现跨层优化、提升设计质量和效率. 然而，由于处在较高的抽象层次，微架构设计阶段难

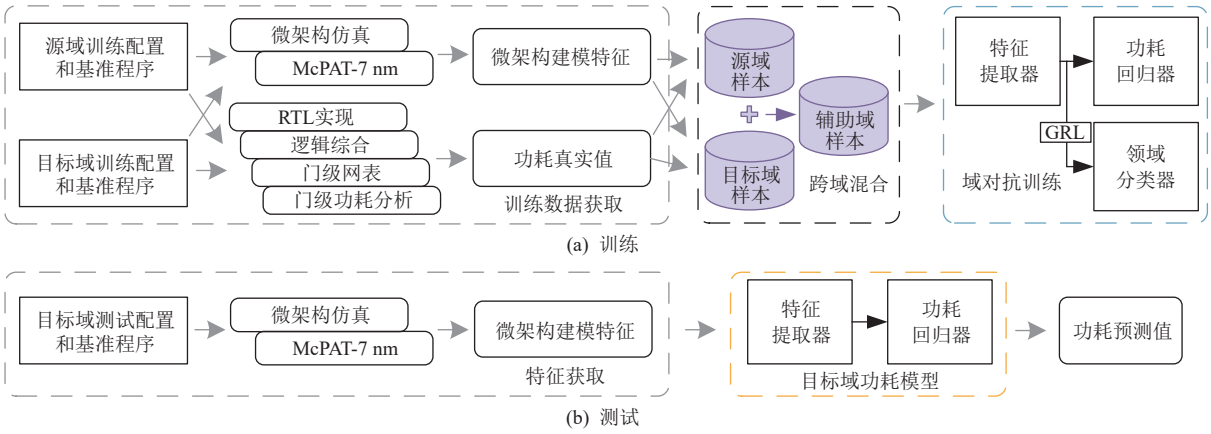


Fig. 10 Transfer learning-based microarchitecture power modeling flow proposed by Zhai et al.[65]

图 10 Zhai 等人[65] 提出的基于迁移学习的微架构功耗建模流程

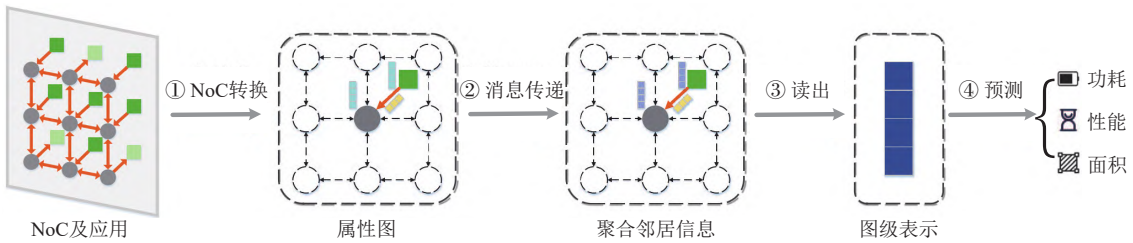


Fig. 11 Illustration of NoCeption framework[68]

图 11 NoCeption 框架示意图[68]

以捕获详细的硬件细节，给功耗建模带来了极大挑战. 为了导出数学上可用的预测模型，解析性功耗模型通常需要做出很多假设，且往往与特定的处理器架构高度耦合，导致通用性较差、建模精度较低，限制了解析模型的应用. 在这种情况下，各类机器学习模型被广泛应用微架构阶段的功耗及其他指标估计. 一般而言，基于机器学习的建模方法具有更好的特征提取和函数拟合能力，能够更好地捕捉复杂非线性关系，在估计精度上有可观的提升.

表 1 从使用阶段、适用范围、建模特征、建模方法及模型误差等方面总结了机器学习辅助的微架构功耗建模方法. 通过分析可以给出常见的功耗建模流程，大致包含 4 个步骤，即数据采集、特征选择、模型构建和功耗估计. 如何采集合适的训练数据、选择最佳的建模特征，并设计精确鲁棒的功耗模型是上述工作的重点，如图 12 所示.

未来，除了进一步提高功耗模型的建模精度、速度外，在数据获取、模型可迁移性、可解释性等方面仍需要进一步研究. 同时，机器学习模型的训练、推理成本不应抵消在设计早期进行功耗估计的速度优势，因此需要开发低复杂度、高精度、低成本的专用学习模型.

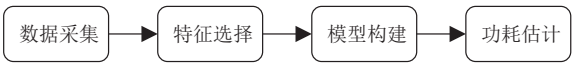


Fig. 12 Common power modeling flow

图 12 功耗建模一般流程

3 微架构设计空间探索

VLSI 的设计总是面临着诸多挑战，因为必须满足一系列严格且相互冲突的设计目标. 设计空间探索已经成为计算机系统设计过程中的基本问题之一[69]. 例如，集成电路的进步显著增加了处理器的复杂性，并产生了大量需要决定的设计参数，例如缓存大小、重排序缓冲区大小等，设计人员需要对不同的微架构设计方案进行探索和比较，从而确保最终处理器能够满足各项设计需求. 本节将介绍设计空间探索问题、相关探索方法，重点在于机器学习辅助微架构设计空间探索并进行小结.

3.1 设计空间探索问题

处理器芯片开发通常需要同时考虑多个优化目标，包括性能、功耗、成本等. 由于设计目标通常是冲突的，因此不可能存在一个同时优化所有目标的最优解决方案. 以图 2 所示的 BOOM 处理器为例，经

Table 1 A Summary of Machine Learning-Assisted Methods for Microarchitecture Power Modeling

表 1 机器学习辅助的微架构功耗建模方法总结

模型/文献	使用阶段	适用范围	建模特征	建模方法	模型误差
PowerTrain[35]	运行时	不同微架构、不同负载	PMC+硬件描述	线性回归	约 2%
WattWatcher[56]	运行时	不同微架构、不同负载	PMC+硬件描述	线性回归	平均 2.67%
文献 [53]	运行时	单一微架构、不同负载	PMC	线性回归	<9%
文献 [54]	运行时	单一微架构、不同负载	PMC	线性回归	2.8%~3.8%
文献 [55]	运行时	单一微架构、不同负载	PMC	非线性回归	平均 6.8%
文献 [51]	设计时	不同微架构	微架构设计参数	非线性回归	中位 5.4%
文献 [52]	设计时	单一微架构、不同负载	仿真活动统计	线性回归	约 2.5%
文献 [57]	设计时	单一微架构、不同负载	仿真活动统计	线性回归	平均 5.9%
文献 [58−59]	设计时	不同微架构	微架构设计参数	神经网络	<2%
文献 [60]	设计时	单一微架构、不同负载	外部输入信号	机器学习模型	约 3.6%
McPAT-Calib[63−64]	设计时	不同微架构、不同负载	架构参数、活动统计	解析模型+机器学习	3%~6%
PANDA[65]	设计时	不同微架构、不同负载	架构参数、活动统计	解析函数+机器学习	2%~8%
文献 [66]	设计时	不同微架构、不同负载	架构参数、活动统计	神经网络+迁移学习	平均 4.4%
TrEnDSE[67]	设计时	不同微架构、跨负载	架构参数	集成模型+迁移学习	<1%
NoCeption[68]	设计时	不同 NoC 配置及拓扑	架构参数	图神经网络	约 2.5%

过微架构设计参数的抽取、合法化以及简化后，可形成如表 2 所示的微架构设计空间，其规模超过 10^8. 显然，在如此庞大的设计空间内探索最优设计面临极大挑战.

Table 2 Microarchitecture Design Space of BOOM

表 2 BOOM 的微架构设计空间

模块	组件参数	描述	备选值
前端	FetchWidth	一次性可取回指令数	4,8
	FetchBufferEntry	取指缓冲条目数	8,16,24,32,35,40
	RasEntry	返回地址堆栈条目数	16,24,32
	BranchCount	同时推测分支数	8,12,16,20
	ICacheWay	ICache 组相连数	2,4,8
	ICacheTLB	ICache 地址翻译缓冲路	8,16,32
	ICacheFetchBytes	ICache 行容量	2,4
指令解码单元	DecodeWidth	一次性最多解码指令数	1,2,3,4,5
	RobEntry	重排序缓冲条目数	32,64,96,128,130
	IntPhyRegister	整型寄存器数	48,64,80,96,112
	FpPhyRegister	浮点型寄存器数	48,64,80,96,112
执行单元	MemIssueWidth	存储型指令发射宽度	1,2
	IntIssueWidth	整型指令发射宽度	1,2,3,4,5
	FpIssueWidth	浮点型指令发射宽度	1,2
加载存储单元	LDQEntry	加载缓冲条目	8,16,24,32
	STQEntry	存储缓冲条目	8,16,24,32
	DCacheWay	D-Cache 组相联数	2,4,8
	DCacheMSHR	缺失状态处理寄存器数	2,4,8
	DCacheTLB	DCache 地址翻译缓冲路	8,16,32

设计空间探索通常可以形式化为一个带约束的多目标优化问题. 给定一组 m 个决策变量，即探索自由度，适应度函数须优化 n 个目标值，其定义为：

$$f_i: R^m \to R^1, \tag{2}$$

其中，适应度函数 f_i 将解空间 X 中的点转换为第 i 个目标值，比如性能、功耗，潜在解 $\boldsymbol{x} \in R^m$ 是 m 个决策变量的赋值. 适应度函数的组合 $\boldsymbol{f}(\boldsymbol{x})$ 可以将解空 X 中的一个点映射到目标空间 Y 中. 在满足 k 个设计约束的前提下，使用 n 个适应度函数 f_i 来确定 m 个决策变量的解 $\boldsymbol{x}$，以最小化 n 个优化目标值，即：

$$\begin{aligned} &\min_{\boldsymbol{x}} \ \boldsymbol{y} = f(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \cdots, f_n(\boldsymbol{x})), \\ &\text{s.t. } g_i(\boldsymbol{x}) \geqslant 0, i \in [1, k_1], \\ &\quad h_j(\boldsymbol{x}) = 0, j \in [1, k_2], \end{aligned} \tag{3}$$

其中 $\boldsymbol{x} = (x_1, x_2, \cdots, x_m) \in X$，$\boldsymbol{y} = (y_1, y_2, \cdots, y_n) \in Y$；且 $k = k_1 + k_2$，$g_i(\boldsymbol{x})$ 和 $h_j(\boldsymbol{x})$ 分别代表第 i 和第 j 个设计约束. 对于单目标优化问题，不同解的比较是非常直观的，更好的适应度（即目标值）意味着更好的设计方案. 但对于多目标优化问题，不同解的比较变得非常复杂，因为涉及到多个目标值间的比较和权衡，从而对探索和优化方法提出了更高要求.

3.2 相关设计空间探索方法

在处理器开发的早期阶段，需要对微架构组件的组织和组合进行探索决策，以实现 PPA 指标之间的良好权衡，由此产生了微架构设计空间探索问题. 在工业界，微架构设计空间探索通常是依靠架构师经验和软件模拟[27−30]来完成的. 通过处理器架构师

的专家知识对设计空间进行修剪, 然后选择有限的候选设计在模拟器上进行仿真和比较权衡. 然而, 随着微架构设计变得越来越复杂, 参数空间越来越庞大, 这种方法愈发难以满足现代处理器的设计需求. 此外, 由于模拟器和真实设计之间的差异以及不同设计阶段之间的设计鸿沟问题, 所产生的仿真和建模误差以及人工偏见往往导致次优的解决方案. 学术界已经提出了各种方法来加速微架构设计空间探索, 这些方法可以分为解析方法和基于机器学习的黑盒方法.

解析方法旨在构建轻量级的可解释模型, 用函数公式描述微架构设计/工作负载与设计目标之间的关系, 从而辅助设计空间剪枝或实现快速探索. Karkhanis 等人[70]通过一阶模型获得超标量处理器的性能估计, 该模型结合了从功能级模拟中收集的缓存和分支预测器统计数据. 进一步地, Karkhanis 等人[71]提出了一个由性能和能量活动模型以及基于模型的优化过程组成的分析框架, 用以得出帕累托最优解的设计参数. RpStacks[72]有选择地从仿真中收集关键性能事件, 建立一些描述执行路径延迟的事件栈, 并估计整体性能以及失速事件构成. 解析方法通常需要大量的专家知识来构建轻量级模型, 且精度有限、可迁移性差, 难以处理不同设计下的大量设计点. Bai 等人[73]通过自动瓶颈分析来规避领域知识要求, 并提出了由新的微执行图表述、最佳关键路径构建算法和硬件资源重新分配策略组成的 Arch-Explorer 探索框架.

3.3 机器学习辅助微架构设计空间探索

构建解析模型需要大量的专业知识, 且难以扩展到新型处理器. 当专家知识难以获取时, 数据驱动的机器学习方法被引入. 与解析方法不同, 基于机器学习的方法构建数据驱动的学习模型来表征设计空间, 并进行基于学习的探索和优化, 由于具有更好的预测和搜索能力, 通常优于基于解析的探索方法.

早期工作将设计空间探索表述为回归问题, 通过各类学习模型预测给定处理器架构配置的 PPA 响应, 并基于 PPA 预测值进行决策优化. Lee 等人[51]将设计空间采样和统计学习结合起来, 无需详尽的模拟就能获得变化趋势, 从而在设计空间探索中实现新的功能. Ipek 等人[58−59]利用 ANN 来捕获硬件架构参数和 PPA 指标之间的关系, 以此构建准确的预测模型直接进行设计空间探索. Dubach 等人[74]使用以架构为中心的方法, 通过将离线训练的先验知识应用于各种基准程序, 从而预测整个微架构配置空间的性能和能耗.

如图 13 所示, Chen 等人[75]提出了 ArchRanker 探索框架, 他们将设计空间探索问题转换为排名问题, 通过训练 RankBoost 学习模型[76]来预测 2 种架构配置中哪一种的性能最佳, 从而确定整个设计空间中较优的架构设计. 在单核和多核设计场景上的实验表明, 相比于基于 ANN 回归模型的方法[59], 该排名模型需要更少的训练仿真时间.

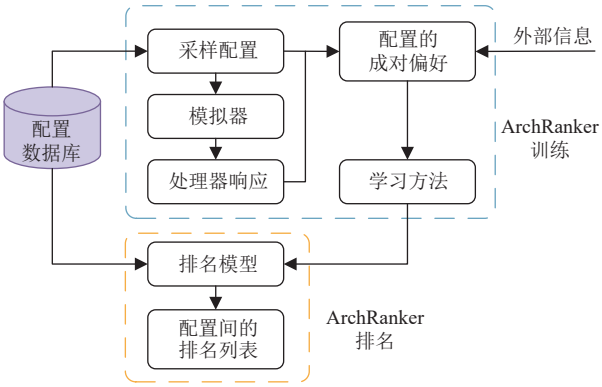


Fig. 13 Illustration of ArchRanker framework[75]

图 13 ArchRanker 框架示意图[75]

如图 14 所示, Li 等人[77]结合了统计采样和 AdaBoost 学习技术. 首先, 采用基于特征选择的正交阵列(orthogonal array, OA), 通过修剪对性能影响较小的参数以缩小设计空间; 其次, 引入基于 OA 的训练数据采样方法来选择代表性配置; 最后, 提出了一种主动学习方法 ActBoost 来构建预测模型, 并对未探索设计的性能进行预测. 基于 gem5 的实验结果表明, 在固定训练样本规模的情况下, 该方法能够实现更高的预测精度, 从而有助于实现高效、精确的设计空间探索.

如图 15 所示, Bai 等人[78−79]提出了一个基于贝叶斯优化(Bayesian optimization, BO)的自动框架来探索 BOOM 微架构设计, 称为 BOOM-Explorer, 实现了较高的探索质量和效率. 首先, 该框架利用先进的微架构感知主动学习 MicroAL 算法来生成多样化且具有代表性的初始设计集; 其次, 建立具有深度核学习函数的高斯过程模型 DKL-GP 来表征设计空间; 然后, 利用相关多目标贝叶斯优化来探索帕累托最优设计, 通过引入帕累托超体积期望提升来处理负相关的多目标(性能和功耗)优化; 最后, 提出了多样性引导的并行探索策略, 并与批量贝叶斯优化流程相结合. 该框架在 BOOM 处理器的微架构设计空间探索中展现了较高的性能, 并给出了如表 3 所示的具体参数选择, 其微架构参数取值与表 2 相对应.

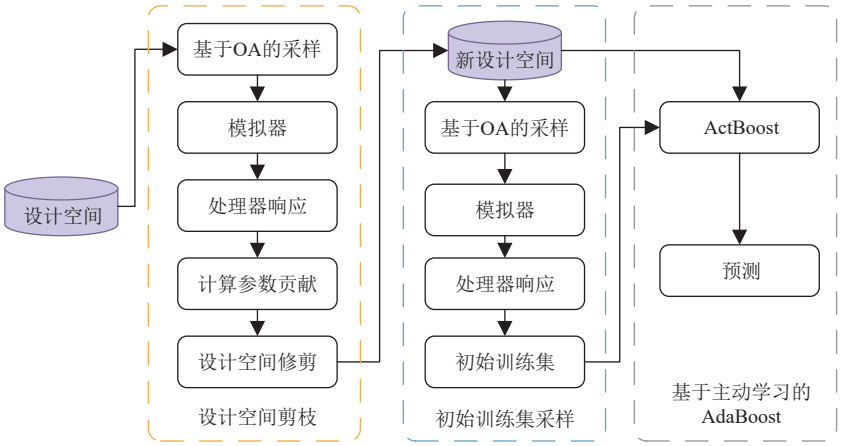


Fig. 14 Design space exploration methodology combining statistical sampling and AdaBoost learning[77]

图 14 结合统计采样和 AdaBoost 学习的设计空间探索方法[77]

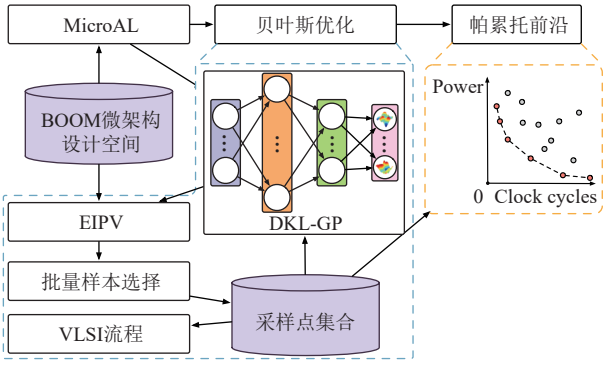


Fig. 15 Illustration of BOOM-Explorer framework[79]

图 15 BOOM-Explorer 框架示意图[79]

然而，微架构设计空间并不平滑，设计参数的微小改动有可能引起设计指标的较大波动，这与 GP 模型的先验假设并不完全相符. 因此，如图 16 所示，Bai 等人[80]将微架构设计空间探索形式化为一种多目标强化学习问题，利用微架构缩放图[81]将专家经验与强化学习方法紧密结合，使用智能代理（agent）依次确定各组件设计，并将架构师的设计偏好嵌入到方法论中，还使用轻量化的 PPA 评估框架构，建强化学习环境以加速收敛. Bai 等人[16]在 Rocket 处理器和 BOOM 处理器上验证了该方法的有效性.

Zhai 等人[82]则提出了一种帕累托驱动的主动学习方法，该方法结合参数重要性的先验知识自动选择更具代表性的初始设计，通过在迭代探索中构建基于树集成模型的动态性能和功耗模型，优先探索具有较大超体积贡献的预测帕累托设计，并允许接受较差的帕累托解来克服模型预测误差，同时使用并行策略来加速探索.

如图 17 所示，Yu 等人[83]提出了一种基于迁移学习的探索框架 IT-DSE，其代理模型经过预训练，可以吸收以往微架构设计任务中的知识. 他们将 Feature Tokenizer-Transformer（FT-Transformer）作为特征提取的骨干模型，以允许对具有不同设计空间的多个源任务进行预训练；同时，使用不变风险最小化（invariant

Table 3 Design Parameters Selectivity of Different Microarchitecture for BOOM Processors

表 3 BOOM 处理器的不同微架构设计参数选择

方法	微架构组件配置参数
原始两发射 BOOM[14−15]	{4, 16, 32, 12, 4, 8, 2, 2, 64, 80, 64, 1, 2, 1, 16, 16, 4, 2, 8}
BOOM-Explorer[78]	{4, 16, 16, 8, 2, 8, 2, 2, 32, 64, 64, 1, 3, 1, 24, 24, 8, 4, 8}
BOOM-Explorer[79]	{4, 16, 16, 8, 4, 8, 2, 2, 32, 64, 64, 1, 3, 1, 24, 24, 8, 4, 8}

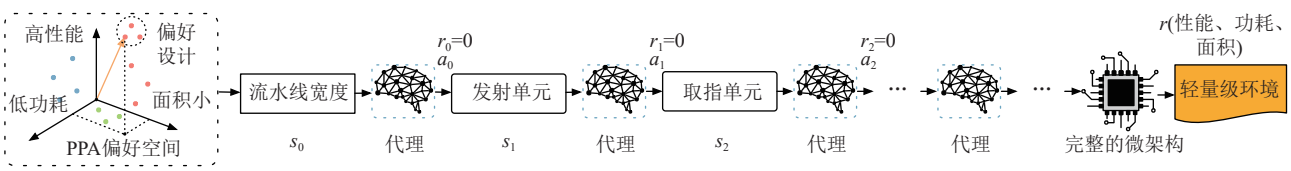


Fig. 16 RL-based microarchitecture design space exploration[80]

图 16 基于强化学习的微架构设计空间探索[80]

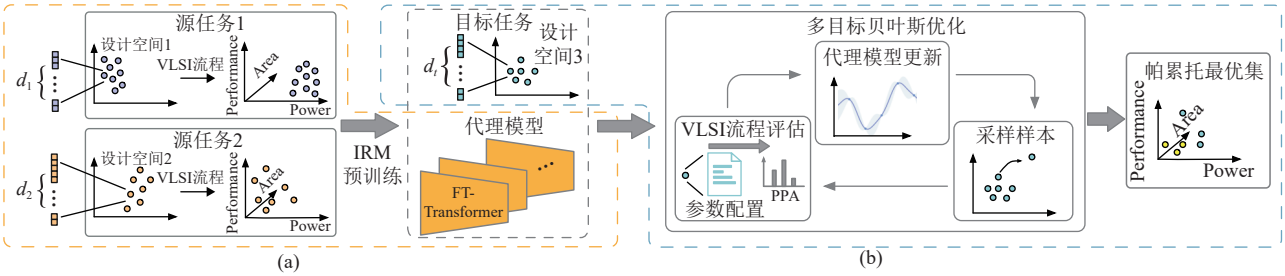


Fig. 17 Illustration of IT-DSE framework[83]

图 17 IT-DSE 框架示意图[83]

risk minimization, IRM)来增强学习模型在不同数据分布差异下的泛化能力;最后,利用多目标贝叶斯优化和预训练模型来探索帕累托最优设计.

Yi 等人[84]提出了一种基于图嵌入的微架构搜索框架 GRL-DSE. 如图 18 所示,GRL-DSE 使用一种有向无环图(directed acyclic graph, DAG)模型[85]来表示处理器微架构硬件组织和参数关系;其次,利用图表示学习来提取关键微架构特征并构建紧凑且连续的图嵌入空间,并使用变分图自动编码器(variational graph auto-encoder, VGAE)框架训练图神经网络模型;最后,使用基于集成学习策略的代理模型完成多目标贝叶斯优化,从而在图嵌入空间中进行微架构设计空间探索. 该方法同样在 BOOM 处理器上进行了验证.

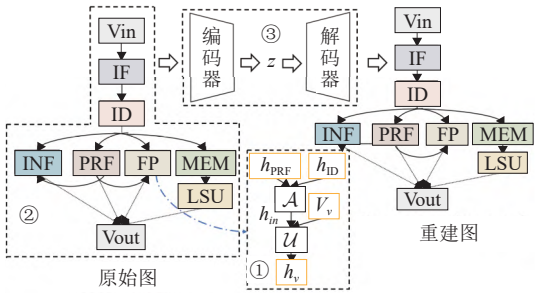


Fig. 18 Training methods of microarchitecture DAG and VGAE[84]

图 18 微架构 DAG 和 VGAE 的训练方法[84]

Wang 等人[86]使用集成模型进行高精度预测,以及一种上置信界超体积提升优化方法来逼近帕累托最优集,并提出了帕累托感知的过滤算法来减少探索时间. 进一步地,Wang 等人[87−88]通过集成学习增强模型 AdaGBRT 生成基于帕累托排序的样本权重以提高预测准确性,然后提出了一种基于超体积提升的优化方法来在多个目标之间进行权衡,并结合均匀性感知选择算法来跳出局部最优.

Esmaeilzadeh 等人[89]为硬件加速深度神经网络(DNN)和非 DNN ML 算法提出了一种物理设计驱动、基于学习的预测框架. 该框架采用统一的方法,将后端 PPA 分析与前端性能仿真相结合,从而实现对后端 PPA、运行时间和能耗等系统指标的实际估计,通过自动搜索架构和后端参数来优化物理设计和系统指标.

Chen 等人[90]提出了一种以 DNN 加速器为主要目标的探索框架 SoC-Tuner,用于高效探索片上系统(system on chip, SoC)配置的帕累托最优集. 该框架提出了一种基于重要性的 SoC 设计空间剪枝和探索初始化方法,并设计了一种采样算法来选择最具代表性的初始化点,以及一种信息增益引导的多目标优化方法,从而平衡 SoC 设计的多个设计指标. 该方法在由 Rocket/BOOM 处理器、Gemmini 加速器[91]等组成的 SoC 设计上进行了验证.

此外,为促进相关领域的发展,ICCAD 2022 举办了以处理器微架构设计空间探索为主题的竞赛[92]. 该比赛创建了一个包含 15 633 个不同 BOOM 微架构的数据集,并提供了设计空间探索算法基准测试平台,吸引了来自全球各地 166 个队伍的参与,在线比赛系统[93]也将长期维护下去.

3.4 小 结

在微架构设计阶段进行广泛的探索优化,有利于推动处理器设计收敛于全局最优,并减少设计迭代、降低开发成本,是实现处理器芯片跨层优化的重要途径. 然而,设计空间的复杂庞大、PPA 指标评估的不准确性,以及复杂耗时的设计验证流程使得微架构设计空间探索面临极大挑战. 在这种情况下,各类机器学习模型被用于 PPA 预估,贝叶斯优化、主动学习、强化学习等机器学习方法被用于探索最优设计. 得益于机器学习技术强大的建模和搜索能力,上述机器学习辅助的探索方法在很多情况下更加快速地获得了更好的微架构设计.

表 4 从探索目标、探索方法及 PPA 数据来源等方面总结了机器学习辅助的微架构设计空间探索工作. 总的来说,机器学习辅助设计空间探索的一般流程如图 19 所示. 对于给定的设计空间,初始化方法一

Table 4 A Summary of Machine Learning-Based Methods for Microarchitecture Design Space Exploration

表 4 基于机器学习的微架构设计空间探索方法总结

方法/文献	探索目标	探索方法	PPA 等数据来源
文献 [51]	帕累托前沿、流水线深度以及异构性分析	设计空间采样、统计学习	Turandot 仿真器、PowerTimer 工具
文献 [58−59]	设计空间的预测模型	使用 ANN 建模遍历子空间	SESC 仿真器、CACTI 等
文献 [73]	设计空间的预测模型	使用 ANN 和线性回归建模遍历子空间	仿真器、Wattch、CACTI 等
ArchRanker[75]	特定目标下最优设计	基于 RankBoost 排名模型遍历子空间	仿真器、Wattch、CACTI 等
文献 [77]	预测模型及最优设计	基于 AdaBoost.RT 模型和正交阵列采样	gem5 仿真器
BOOM-Explorer[78−79]	帕累托最优设计	基于贝叶斯优化和深度核高斯过程建模	商业 EDA 工具
文献 [80]	特定偏好下最优设计	基于微架构缩放图的强化学习	商业 EDA 工具
文献 [82]	帕累托最优设计	基于集成树建模和主动学习	商业 EDA 工具
IT-DSE[83]	帕累托最优设计	基于贝叶斯优化、不变风险最小化和 Transformer	商业 EDA 工具
GRL-DSE[84]	帕累托最优设计	基于图神经网络、集成模型、贝叶斯优化	商业 EDA 工具
文献 [86]	帕累托最优设计	基于 BagGBRT 和上置信界超体积提升	仿真器、McPAT 等工具
MoDSE[87−88]	帕累托最优设计	基于 AdaGBRT 和帕累托超体积提升	仿真器、McPAT 等工具
文献 [89]	ML 加速器最优设计	机器学习模型、图神经网络、贝叶斯优化	商业 EDA 工具
SoC-Tuner[90]	SoC 帕累托最优设计	设计空间剪枝、贝叶斯优化	商业 EDA 工具

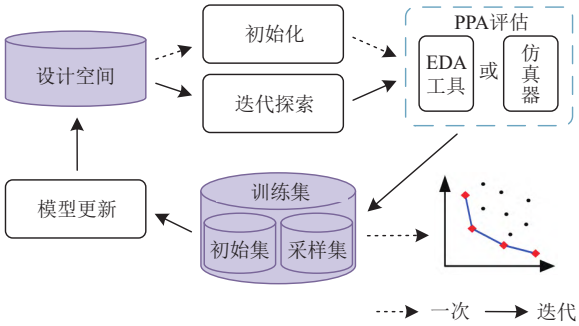


Fig. 19 The general flow of design space exploration

图 19 设计空间探索一般流程

次性地选择最具有代表性的初始设计，并依靠 EDA 工具或仿真器进行进行 PPA 评估，进而更新数据集并训练机器学习模型，然后依靠模型预测和采样策略进行迭代探索，在探索结束时给出目标设计(集).

未来，如何将微架构领域的专家知识(例如微架构组织和参数的依赖性等)与机器学习方法紧密结合，并努力提高探索框架的泛化性和可扩展性仍是需要进一步探索的问题. 除了单纯的设计参数调整外，如何利用机器学习技术反哺微架构设计，例如使用大模型辅助微架构设计生成，也是值得研究的问题.

4 总结与展望

本文首先回顾了现代处理器设计的主要流程，以及微架构设计在“后摩尔时代”所面临的建模和优化挑战. 在此基础上，介绍了如何运用机器学习技术辅助处理器微架构功耗建模和设计空间探索问题，并就相关的具体工作展开了讨论.

更小、更快、功耗更低的计算设备将是人们永远追求的目标，随着摩尔定律的进一步放缓，微架构设计将占有更重要的地位，且面临更加严峻的挑战. 展望未来，以下 3 个方面依然值得进一步研究：

1)更加快速精细的功耗估计和更高的模型可迁移性. 处理器规模的不断增长和工作负载的多样化，对建模速度和精度提出了更高要求. 现有工作大多利用各种机器学习技术对微架构设计在较长时间间隔内的平均功耗进行预测，建模的精细度仍有不足、建模代价也较为高昂，所得功耗模型的可迁移性不强. 因此，如何利用先验知识降低模型的训练成本，并进一步提高功耗模型的建模粒度、可迁移性，并降低建模代价至关重要. 对于建模粒度而言，首先需要精确估计不同工作负载在更短时间粒度内的功率消耗，并为动态电压频率调整等节能技术的设计提供参考；其次是更细的硬件粒度，即如何提供不同硬件层次的精确功耗估计，增强模型可解释性，并将其用于后续热问题的建模，允许设计人员更有针对性地进行估计和优化；最后，如何提高模型在多平台、多领域架构间的可迁移性和功能完善性，并实现对异质异构集成芯片的精确建模是未来的研究方向.

2)更加高效统一的设计空间探索优化框架. 处理器的规模在不断增长，产生了大量可参数化的微架构设计参数、EDA 工具参数以及诸多难以参数化的设计策略. 因此，如何将微架构探索与整个 EDA 设

计流程内的探索优化相结合,并结合先验知识实现高维设计空间内全局最优设计的快速探索成为关键.首先,可以考虑混合使用机器学习模型、软件仿真器和传统 EDA 设计流程工具进行建模及探索优化,并设计验证时间感知的调度策略.同时,现有的设计空间探索工作多是针对静态可参数化空间进行的,如何有效地对动态的难以量化的设计选项进行探索和优化也是一个尚未解决的问题.此外,目前的设计空间探索工具大多是面向特定设计或任务的,如何提高其在不同设计和工艺节点间的可迁移性,将不同阶段、不同任务的探索工具整合到一个统一的标准流程中,真正实现跨层次的全局优化也将是未来的研究方向.

3)高质量的公开数据集和数据获取方法.得益于 AI 领域的飞速发展,机器学习技术能够有效地辅助相关问题的求解.然而,机器学习模型通常需要大量的有标记训练数据,而传统 VLSI 设计流程漫长耗时,且 EDA 工具、先进 IP 核和工艺库需要昂贵的商业授权,导致缺乏针对相关任务的高质量公开数据,难以充分验证机器学习模型的泛化能力和迁移能力.目前已有针对集成电路后端设计等任务的公开数据集,但针对处理器微架构建模和探索的数据集仍比较欠缺,多数工作基于私有数据集,或使用开源的处理器设计自行生成数据集,数据规模小、代表性不强,阻碍了机器学习技术在相关领域的进一步应用.因此,如何利用主动学习等技术更加高效地获取数据,并建立涵盖不同处理器设计和 VLSI 设计阶段的公开数据集用于相关算法的设计和评估,也是值得进一步研究的问题.

考虑到相关算法的快速更新、持续增长的硬件支持能力,以及不断积累的应用数据,机器学习将与微架构设计的传统方法更加有机地结合,从而实现更加精确的建模和更加高效的探索优化.

作者贡献声明: 翟建旺和余备提出了写作思路;翟建旺、凌梓超和白晨负责调研并撰写论文;赵康和余备提出指导意见并修改论文.

参 考 文 献

[1] The State Council. Several policies to promote the high-quality development of integrated circuit industry and software industry in the new era[EB/OL]. [2023-12-25]. https://www. gov. cn/zhengce/content/2020-08/04/content_5532370. htm (in Chinese)
(国务院. 新时期促进集成电路产业和软件产业高质量发展若干政策[EB/OL]. [2023-12-25]. https://www.gov.cn/zhengce/content/2020-08/04/content_5532370.htm)

[2] Chen Yunqi, Cai Yimao, Wang Yu, et al. Integrated circuit technology: Future development and key issues–review of the 347th Shuangqing Forum (Youth)[J]. SCIENTIA SINICA Informationis, 2024, 54(1): 1−15 (in Chinese)
(陈云霁, 蔡一茂, 汪玉, 等. 集成电路未来发展与关键问题——第347期"双清论坛(青年)"学术综述[J]. 中国科学: 信息科学, 2024, 54(1): 1−15)

[3] Xiang Chengxiang, Yang Yongan, Penner R M. Cheating the diffraction limit: Electrodeposited nanowires patterned by photolithography[J]. Chemical Communications, 2009, 8: 859−873

[4] Chaudhry A, Kumar M J. Controlling short-channel effects in deep-submicron SOI MOSFETs for improved reliability: A review[J]. IEEE Transactions on Device and Materials Reliability, 2004, 4(1): 99−109

[5] Thimbleby H. Modes, WYSIWYG and the von Neumann bottleneck[C]//Proc of IEE Colloquium on Formal Methods and Human-Computer Interaction: II. London: IET, 1988: 4/1−4/5

[6] Zhou Zhihua. Machine Learning[M]. Singapore: Springer Nature Singapore , 2021

[7] Liang Yun, Zhuo Cheng, Li Yongfu. The shift-left design paradigm of EDA: Progress and challenges[J]. SCIENTIA SINICA Informationis, 2024, 54(1): 121−129 (in Chinese)
(梁云, 卓成, 李永福. EDA 左移融合设计范式的发展现状、趋势与挑战[J]. 中国科学: 信息科学, 2024, 54(1): 121−129)

[8] Bao Yungang, Chang Yisong, Han Yinhe, et al. Agile design of processor chips: Issues and challenges[J]. Journal of Computer Research and Development, 2021, 58(6): 1131−1145 (in Chinese)
(包云岗, 常轶松, 韩银和, 等. 处理器芯片敏捷设计方法: 问题与挑战[J]. 计算机研究与发展, 2021, 58(6): 1131−1145)

[9] Scheffer L, Lavagno L. EDA for IC System Design, Verification, and Testing[M]. FL: CRC Press, Inc, 2018

[10] Wu C M, Shieh M D, Wu C H, et al. VLSI architectural design tradeoffs for sliding-window log-MAP decoders[J]. IEEE Transactions on Very Large Scale Integration Systems, 2005, 13(4): 439−447

[11] Brown S, Vranesic Z. Fundamentals of Digital Logic with Verilog Design[M]. Translated by Xia Yuwen, Xu Yuxiao. 2nd. Beijing: China Machine Press, 2008 (in Chinese)
(Brown S, Vranesic Z. 数字逻辑基础与 Verilog 设计[M]. 夏宇闻, 须毓孝译. 原书第 2 版. 北京: 机械工业出版社, 2008)

[12] Rudell R L. Logic synthesis for VLSI design[R/OL]. Berkeley, California: University of California, Berkeley, 1989. [2023-12-25]. https://www2.eecs.berkeley.edu/Pubs/TechRpts/1989/1223.html

[13] Sherwani N A. Algorithms for VLSI Physical Design Automation[M]. New York: Springer Science & Business Media New York, 2013

[14] Celio C, David A P, Krste A. The Berkeley out-of order machine (BOOM): An industry-competitive, synthesizable, parameterized RISC-V processor[R]. Berkeley, CA: EECS Department, University of California, Berkeley, 2015

[15] Zhao J, Abraham G. SonicBOOM: The 3rd generation Berkeley out-of-order machine[C]// Proc of 4th Workshop Computer Architecture Research with RISC-V. New York: ACM, 2020:1−7

[16] Asanovic K, Rimas A, Jonathan B, et al. The rocket chip generator[R]. Berkeley, CA: EECS Department, University of California, Berkeley, 2015

[17] Chen Chen, Xiang Xiaoyan, Liu Chang, et al. , Xuantie-910: A commercial multi-core 12-stage pipeline out-of-order 64-bit high performance RISC-V processor with vector extension: Industrial product[C]//Proc of ACM/IEEE Annual Int Symp on Computer Architecture. New York: ACM, 2020: 52−64

[18] Xu Yinan, Yu Zihao, Wang Kaifan, et al. XiangShan Open-source high performance RISC-V processor design and implementation[J]. Journal of Computer Research and Development, 2023, 60(3): 476−493 (in Chinese)
(徐易难, 余子濠, 王凯帆, 等. 香山开源高性能 RISC-V 处理器设计与实现[J]. 计算机研究与发展, 2023, 60(3): 476−493)

[19] Bachrach J, Vo H, Richards B, et al. Chisel: Constructing hardware in a scala embedded language[C]//Proc of DAC Design Automation Conf. Piscataway, NJ: IEEE , 2012: 1212−1221

[20] Winston P H. Artificial Intelligence[M]. London: Addison-Wesley Longman Publishing Co. , Inc. , 1984

[21] Rapp M, Amrouch H, Lin Yibo, et al. MLCAD: A survey of research in machine learning for CAD keynote paper[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2021, 41(10): 3162−3181

[22] Synopsys. PrimeTime[CP/OL]. [2023-12-25]. https://www.synopsys.com/implementation-and-signoff/signoff/primetime.html

[23] Nesset S R. RTL Power Estimation Flow and Its Use in Power Optimization[M]. Norway: Norwegian University of Science and Technology, 2018

[24] Brooks D, Tiwari V, Martonosi M. Wattch: A framework for architectural-level power analysis and optimizations[C]//Proc of IEEE/ACM Annual Int Symp on Computer Architecture. New York: ACM, 2000: 83−94

[25] Thoziyoor S, Ahn J H, Monchiero M, et al, A comprehensive memory modeling tool and its application to the design and analysis of future memory hierarchies[C]//Proc of Int Symp on Computer Architecture. Piscataway,NJ: IEEE, 2008: 51-62

[26] Li Sheng, Ahn J H, Strong R D, et al. McPAT: An integrated power, area, and timing modeling framework for multicore and manycore architectures[C]//Proc of IEEE/ACM Int Symp on Microarchitecture. New York: ACM, 2009: 469−480

[27] Burger D, Todd M A. The SimpleScalar tool set, version 2.0[J]. ACM SIGARCH Computer Architecture News, 1997, 25: 13−25

[28] Alec R, Mircea R S. RISC5: Implementing the RISC-V ISA in gem5[C]//Proc of the 1st Workshop on Computer Architecture Research with RISC-V. Piscataway, NJ: IEEE, 2017: 1−7

[29] Binkert N L, Dreslinski R G, Hsu L R, et al. The M5 simulator: Modeling networked systems [J]. IEEE Micro, 2006, 26(4): 52−60

[30] Carlson T E, Heirman W, Eeckhout L. Sniper: Exploring the level of abstraction for scalable and accurate parallel multi-core simulation [C]//Proc of Int Conf for High Performance Computing, Networking, Storage and Analysis. Piscataway, NJ: IEEE, 2011: 1−12

[31] Semiconductor industries association. model for assessment of CMOS technologies and roadmaps (MASTAR)[EB/OL]. [2023-12-25] https://web.archive.org/web/20130709053354/http://www.itrs.net/models.html

[32] Brooks D, Bose P, Srinivasan V, et al. New methodology for early-stage, microarchitecture-level power-performance analysis of microprocessors[J]. IBM Journal of Research and Development, 2003, 47((5/6):): 653−670

[33] Wang Hangsheng, Zhu Xinping, Li-Shiuan P, et al. Orion: A power-performance simulator for interconnection networks[C]//Proc of IEEE/ACM Int Symp on Microarchitecture. Piscataway, NJ: IEEE, 2002: 294−305

[34] Xi S L, Jacobson H, Bose P, et al. Quantifying sources of error in McPAT and potential impacts on architectural studies[C]//Proc of IEEE Int Symp on High Performance Computer Architecture. Piscataway, NJ: IEEE, 2015: 577−589

[35] Lee W, Kim Y, Ryoo J H, et al. PowerTrain: A learning-based calibration of McPAT power models[C]//Proc of IEEE Int Symp on Low Power Electronics and Design. Piscataway, NJ: IEEE, 2015: 189−194

[36] Tang A, Yang Y, Lee C Y et al. McPAT-PVT: Delay and power modeling framework for FinFET processor architectures under PVT variations[J]. IEEE Transactions on Very Large Scale Integration Systems, 2015, 23(9): 1616−1627

[37] Guler A, Jha N K. McPAT-Monolithic: An area/power/timing architecture modeling framework for 3-D hybrid monolithic multicore systems[J]. IEEE Transactions on Very Large Scale Integration Systems, 2020, 28(10): 2146−2156

[38] Ravipati D P, Van S, Victor M, et al. Performance and energy studies on NC-FinFET cache-based systems with FN-McPAT[J]. IEEE Transactions on Very Large Scale Integration Systems, 2023, 31(9): 1280−1293

[39] Van den Steen S, De Pestel S, Mechri M, et al. Micro-architecture independent analytical processor performance and power modeling[C]//Proc of IEEE Int Symp on Performance Analysis of Systems and Software. Piscataway, NJ: IEEE, 2015: 32−41

[40] Park Y H, Pasricha S, Kurdahi F J , et al. A multi-granularity power modeling methodology for embedded processors[J]. IEEE Transactions on Very Large Scale Integration Systems, 2010, 19(4): 668−681

[41] Ansys. PowerArtist[CP/OL]. [2023-12-25]. https://www.ansys.com/products/semiconductors/ansys-powerartist

[42] Mentor. PowerPro RTL low-power[CP/OL]. [2023-12-25]. https://www.mentor.com/hls-lp/powerpro-rtl-low-power/

[43] Bogliolo A, Benini L, De Micheli G. Regression-based RTL power modeling[J]. ACM Transactios on Design Automation of Electronic Systems, 2000, 5(3): 337−372

[44] Sunwoo D, Wu G Y, Patil N A. PrEsto: An FPGA-accelerated power estimation methodology for complex systems[C]//Proc of IEEE Int Conf on Field Programmable Logic and Applications. Piscataway, NJ: IEEE, 2010: 310−317

[45] Yang Jianlei, Ma Liwei, Zhao Kang, et al. Early stage real-time SoC power estimation using RTL instrumentation[C]//Proc of IEEE/ACM Asia and South Pacific Design Automation Conf. Piscataway,

NJ: IEEE, 2015: 779−784

[46] Zhou Yuan, Ren Haoxing, Zhang Yanqing, et al. PRIMAL: Power inference using machine learning [C]//Proc of ACM/IEEE Design Automation Conf. New York: ACM, 2019: 1−6

[47] Kim D, Zhao J, Bachrach J, et al. Simmani: Runtime power modeling for arbitrary RTL with automatic signal selection[C]//Proc of IEEE/ACM Int Symp on Microarchitecture. Piscataway, NJ: IEEE, 2019: 1050−1062

[48] Zhang Yanqing, Ren Haoxing, Khailany B. GRANNITE: Graph neural network inference for transferable power estimation[C]//Proc of ACM/IEEE Design Automation Conf. New York: ACM, 2020: 1−6

[49] Xie Zhiyao, Xu Xiaoqing, Walker M, et al. APOLLO: An automated power modeling framework for runtime power introspection in high-volume commercial microprocessors[C]//Proc of IEEE/ACM Int Symp on Microarchitecture. Piscataway, NJ: IEEE, 2021: 1−14

[50] Fang Wenji, Lu Yao, Liu Shang, et al. MasterRTL: A pre-synthesis PPA estimation framework for any RTL design[C]//Proc of IEEE/ACM Int Conf on Computer Aided Design. Piscataway, NJ: IEEE, 2023: 1−9

[51] Lee B C, Brooks D M. Illustrative design space studies with microarchitectural regression models[C]//Proc of IEEE Int Symp on High Performance Computer Architecture. Piscataway, NJ: IEEE, 2007: 340−351

[52] Jacobson H, Buyuktosunoglu A, Bose P, et al. Abstraction and microarchitecture scaling in early-stage power modeling[C] // Proc of IEEE Int Symp on High Performance Computer Architecture. Piscataway, NJ: IEEE, 2011: 394−405

[53] Bircher W L, John L K. Complete system power estimation: A trickle-down approach based on performance events[C] // Proc of IEEE Int Symp on Performance Analysis of Systems & Software. Piscataway, NJ: IEEE, 2007: 158−168

[54] Walker M J, Diestelhorst S, Hansson A, et al. Accurate and stable run-time power modeling for mobile and embedded CPUs[J]. IEEE Transactios on Computer-Aided Design of Integrated Circuits and Systems, 2017, 36(1): 106−119

[55] Sagi M, Doan N A V, Rapp M, et al. A lightweight nonlinear methodology to accurately model multicore processor power[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2020, 39(11): 3152−3164

[56] Lebeane M, Ryoo J H , Panda R, et al. WattWatcher: Fine-grained power estimation for emerging workloads[C]//Proc of Int Symp on Computer Architecture and High Performance Computing. New York: ACM, 2015: 106−113

[57] Reddy B K, Walker M J , Balsamo D et al. Empirical CPU power modelling and estimation in the gem5 simulator[C] // Proc of IEEE Int Workshop on Power and Timing Modeling, Optimization and Simulation. Piscataway, NJ: IEEE, 2017: 1−8

[58] Ipek E, McKee S A, Caruana R, et al. Efficiently exploring architectural design spaces via predictive modeling[C]//Proc of ACM Int Conf on Architectural Support for Programming Languages and Operating Systems. New York: ACM, 2006: 195−206

[59] Ipek E, McKee S A, Singh K, et al. Efficient architectural design space exploration via predictive modeling[J]. ACM Transactions on Architecture and Code Optimization, 2008, 4(4): 1−34

[60] Kumar A K A, Al-Salamin S, Amrouch H, et al. Machine learning-based microarchitecturelevel power modeling of CPUs[J]. IEEE Transactions on Computers, 2023, 72(4): 941−956

[61] Wilson S Verilator [CP/OL]. [2023-12-25]. https://www.veripool.org/wiki/verilator

[62] Rossi D, Conti F, Marongiu A, et al. PULP: A parallel ultra low power platform for next generation IoT applications[C]//Proc of IEEE Hot Chips Symp. Piscataway, NJ: IEEE, 2015: 1−39

[63] Zhai Jianwang, Bai Chen, Zhu Binwu, et al. McPAT-Calib: A microarchitecture power modeling framework for modern CPUs[C]//Proc of IEEE/ACM Int Conf on Computer-Aided Design. Piscataway, NJ: IEEE, 2021: 1−9

[64] Zhai Jianwang, Bai Chen, Zhu Binwu, et al. McPAT-Calib: A RISC-V BOOM microarchitecture power modeling framework[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2023, 42(1): 243−256

[65] Zhang Qijun, Li Shiyu, Zhou Guanglei, et al. PANDA: Architecture-level power evaluation by unifying analytical and machine learning solutions[C]//Proc of IEEE/ACM Int Conf on Computer Aided Design. Piscataway, NJ: IEEE, 2021: 1−9

[66] Zhai Jianwang, Cai Yici, Yu Bei. Microarchitecture power modeling via artificial neural network and transfer learning[C]//Proc of IEEE/ACM Asia and South Pacific Design Automation Conf. Piscataway, NJ: IEEE, 2023: 1−6

[67] Wang Duo, Yan Mingyu, Teng Yihan, et al. A Transfer learning framework for high-accurate cross-workload design space exploration of CPU[C]//Proc of IEEE/ACM Int Conf on Computer Aided Design. Piscataway, NJ: IEEE, 2023: 1−9

[68] Li Fuping, Wang Ying, Liu Cheng et al. NoCeption: A fast PPA prediction framework for network-on-chips using graph neural network[C]//Proc of Design, Automation & Test in Europe Conf & Exhibition. Piscataway, NJ: IEEE, 2022: 1035−1040

[69] Guo Qi, Chen Tianshi, Chen Yunji, et al. Accelerating architectural simulation via statistical techniques: A survey[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2016, 35(3): 433−446

[70] Karkhanis T S, Smith J E. A first-order superscalar processor model[C]//Proc of IEEE/ACM Int Symp on Computer Architecture. Piscataway, NJ: IEEE, 2004: 338−349

[71] Karkhanis T S, Smith J E. Automated design of application specific superscalar processors: An analytical approach[C]//Proc of IEEE/ACM Int Symp on Computer Architecture. Piscataway, NJ: IEEE, 2007: 402−411

[72] Lee J, Jang H, Kim J. RPStacks: Fast and accurate processor design space exploration using representative stall-event stacks[C]//Proc of IEEE/ACM Int Symp on Microarchitecture. Piscataway, NJ: IEEE, 2014: 255−267

[73] Bai Chen, Huang Jiayi, Wei Xuechao, et al. ArchExplorer: Microarchitecture exploration via bottleneck analysis[C]//Proc of Annual IEEE/ACM Int Symp on Microarchitecture. Piscataway, NJ: IEEE, 2023: 268−282

[74] Dubach C, Jones T, O'Boyle M. Microarchitectural design space exploration using an architecture-centric approach[C]//Proc of IEEE/ACM Int Symp on Microarchitecture. Piscataway, NJ: IEEE,

2007: 262−271

[75] Chen Tianshi, Guo Qi, Tang Ke, et al. ArchRanker: A ranking approach to design space exploration[J]. ACM SIGARCH Computer Architecture News, 2014, 42(3): 85−96

[76] Freund Y, Iyer R, Schapire R E, et al. An efficient Boosting algorithm for combining preferences[J]. Journal of Machine Learning Research, 2003, 4(9): 933−969

[77] Li Dandan, Yao Shuzhen, Liu Yuhang, et al. Efficient design space exploration via statistical sampling and AdaBoost learning[C]//Proc of ACM/IEEE Design Automation Conf. New York: ACM, 2016: 1−6

[78] Bai Chen, Sun Qi, Zhai Jianwang, et al. BOOM-Explorer: RISC-V BOOM microarchitecture design space exploration framework[C]// Proc of IEEE/ACM Int Conf on Computer-Aided Design. Piscataway, NJ: IEEE, 2021: 1−9

[79] Bai Chen, Sun Qi, Zhai Jianwang, et al. BOOM-Explorer: RISC-V BOOM microarchitecture design space exploration framework[J]. ACM Transactions on Design Automation of Electronic Systems, 2024, 29(1): 1−23

[80] Bai Chen, Zhai Jianwang, Ma Yuzhe, et al. Towards automated RISC-V microarchitecture design with reinforcement learning[C]//Proc of AAAI Conf on Artificial Intelligence. Menlo, CA: AAAI, 2024: 1−9

[81] Eyerman S, Eeckhout L, Karkhanis T, et al. A mechanistic performance model for superscalar out-of-order processors[J]. ACM Transactions on Computer Systems, 2009, 27(2): 1−37

[82] Zhai Jianwang, Cai Yici. Microarchitecture design space exploration via Pareto-driven active learning[J]. IEEE Transactions on Very Large Scale Integration Systems, 2023, 31(11): 1727−1739

[83] Yu Ziyang, Bai Chen, Hu Shoubo, et al. IT-DSE: Invariance risk minimized transfer microarchitecture design space exploration[C]// Proc of IEEE/ACM Int Conf on Computer Aided Design. Piscataway, NJ: IEEE, 2023: 1−9

[84] Yi Xiaoling, Lu Jialin, Xiong Xiankui, et al. Graph representation learning for microarchitecture design space exploration[C]//Proc of ACM/IEEE Design Automation Conf. New York: ACM, 2023: 1−6

[85] Zhang Muhan, Jiang Shali, Cui Zhicheng, et al. D-VAE: A variational autoencoder for directed acyclic graphs[J]. arXiv preprint, arXiv: 1904.11088, 2019

[86] Wang Duo, Yan Mingyu, Teng Yihan, et al. A high-accurate multi-objective ensemble exploration framework for design space of CPU microarchitecture[C]//Proc of the Great Lakes Symp on VLSI. New York: ACM, 2023: 379–383

[87] Wang Duo, Yan Mingyu, Teng Yihan, et al. A high-accurate multi-objective exploration framework for design space of CPU[C] // Proc of ACM/IEEE Design Automation Conf. Piscataway, NJ: IEEE, 2023: 1−6

[88] Wang Duo, Yan Mingyu, Teng Yihan, et al. MoDSE: A high-accurate multi-objective design space exploration framework for CPU microarchitectures[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and System, 2024, 43(5): 1525−1537

[89] Esmaeilzadeh H, Ghodrati S, Kahng A B, et al. An Open-source ML-based full-stack optimization framework for machine learning accelerators[J]. arXiv preprint, arXiv: 2308.12120, 2023

[90] Chen Shixin, Zheng Su, Bai Chen, et al. SoC-Tuner: An importance-guided exploration framework for DNN-targeting SoC design[C] // Proc of IEEE/ACM Asian and South Pacific Design Automation Conf. Piscataway, NJ: IEEE, 2024: 1−6

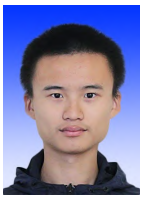
[91] Genc H, Kim S, Amid A, et al. Gemmini: Enabling systematic deep-learning architecture evaluation via full-stack integration[C] // Proc of ACM/IEEE Design Automation Conf. New York: ACM, 2021: 769–774

[92] Li Sicheng, Bai Chen, Wei Xuechao, et al. 2022 ICCAD CAD contest problem C: Microarchitecture design space exploration[C] // Proc of IEEE/ACM Int Conf on Computer-Aided Design. Piscataway, NJ: IEEE, 2022: 1−7

[93] Bai chen. ICCAD contest platform [EB/OL]. [2024-01-02]. http://47.93.191.38/

Zhai Jianwang, born in 1996. PhD, assistant professor. His main research interests include machine learning-assisted electronic design automation (EDA) algorithms, including microarchitecture power modeling, design space exploration, and physical design.

翟建旺, 1996 年生. 博士, 特聘副研究员. 主要研究方向为机器学习辅助的 EDA 算法, 包括微架构功耗建模、设计空间探索、物理设计.



Ling Zichao, born in 2000. Bachelor. His main research interests include computer architecture and power modeling.

凌梓超, 2000 年生. 学士. 主要研究方向为计算机体系结构、功耗建模.

Bai Chen, born in 1998, PhD candidate. His main research interests include computer architecture and electronic design automation.

白　晨, 1998 年生. 博士研究生.主要研究方向为计算机体系结构、电子设计自动化.

Zhao Kang, born in 1982, PhD, professor, PhD supervisor. Senior member of CCF. His main research interests include electronic design automation (EDA), compiling optimization for FPGA, and heterogenous computing systems.

赵　康, 1982 年生.博士, 教授, 博士生导师. CCF 高级会员.主要研究方向为电子设计自动化、面向 FPGA 的编译优化、异构计算系统.

Yu Bei, born in 1983. PhD, associate professor, PhD supervisor. His main research interests include electronic design automation (EDA) and machine learning.

余　备, 1983 年生. 博士, 副教授, 博士生导师. 主要研究方向为电子设计自动化、机器学习.