对计算机体系结构研究的一点认识

钱学海
美国加州大学伯克利分校

关键词：计算机体系结构　研究方法

编者按：美国斯坦福大学最近在一份报告中分析了过去20年间出现的几乎所有的处理器，发现处理器性能提高了约1万倍。这使得像深度学习等20多年前还是遥不可及的技术，在今天强大的计算能力的支持下，释放出巨大的潜力。该报告进一步指出，在1万倍的性能提升中，半导体工艺贡献了100多倍，计算机体系结构贡献了80多倍。由于处理器在信息领域的基础性与普适性，计算机体系结构技术的进步对整个信息领域的快速发展起到了重要的推动作用。而该领域的四大国际顶级会议(ISCA, HPCA, MICRO, ASPLOS)[1]则是技术进步的思想源泉，几乎所有计算机体系结构领域的重大技术突破都是最早发表在这四大会议上。加州大学伯克利分校的钱学海博士是体系结构领域的新星，过去5年在四大会议上发表了7篇论文，对计算机体系结构研究有相当的理解和造诣。为此，本期专栏特邀钱学海与读者分享他对计算机体系结构研究的认识。

应*CCCF*之邀，很高兴能在此分享自己的一些研究体会和见闻，希望对大家的工作和研究有所帮助。

1964年阿姆达尔(C. M. Amdahl)在介绍IBM360系统时提出：计算机体系结构是程序员看到的计算机的属性，即概念性结构与功能特征。虽然这个定义简明准确，但若没有一定的实践经验，则会觉得其抽象含糊。我相信这也是一些学生在修完本科或者研究生的计算机体系结构课程后的体会。其实计算机体系结构是一门充满活力和贴近实际的重要学科，是软件和硬件的桥梁，同时也是信息产业的支柱之一。

研究心得

什么是好的研究？这可以从两个方面衡量，首先看研究的问题是否重要，其次看是否提出了有挑战性或者新颖的解决方案，若具备两者之一就是好的研究。对于什么是“重要”的问题，不同学者有不同的看法，但我们可以用一个通俗的标准来概括，即解决方案能否提高“效益”(提高性能或降低能耗)。这个标准相对简单，也得到大家认可。在第二个标准中，同行对于有挑战性的问题通常都有统一的认识，解决方案的新颖性也不难被专家发现。

谈到研究方法，最重要的是对研究问题有深刻的认识。这不仅

[1] ISCA: International Symposium on Computer Architecture，计算机体系结构国际研讨会；
HPCA: International Symposium on High Performance Computer Architecture，高性能计算机体系结构国际研讨会；
MICRO: International Symposium on Microarchitecture，微体系结构国际研讨会；
ASPLOS: International Conference on Architectural Support for Programming Languages and Operating Systems，编程语言和操作系统的体系结构支持会议。

意味着对现有工作了如指掌，知道它们的不同和不足，还需要思考它们为何会有这些差异和不足？**做研究和心理分析有相似之处，一篇论文读的次数多了，往往可以从字里行间发现作者的思维方式，如果能做到这一步，就离自己做出成果不远了**。相信读者有类似的体会，在开始读论文时常常觉得复杂或者难以分辨差别，但如果有了更深层次认识，即了解了作者为什么这样想，就可以相对轻松地提出更好的方案。这里不妨做一个类比，我们可以把对一个问题的解决空间看成是一个从根节点开始的有很多分支的程序，而发表的论文看成是叶节点的集合。如果只关注表象，就只能看到不同叶节点集合之间的重叠和差异。但是如果关注本质，回溯到离根节点较近的分支，就会发现虽然同一个人写了不同的论文，但其思路大都是从一个分支出发的。而不同的人写论文往往从最初就选择不同的分支。在理解现有工作的过程中，我们要尽量离根节点更近些，即看到论文的本质，然后从这个节点寻找不同的分支（解法）。这样做出的研究就不会过于增量(incremental)。类似的原则也应用于软件测试中，好的测试需要尽量保证覆盖靠近根节点的分支，以免造成许多测试都集中于一个子分支的叶节点（或底层节点）的情况。

读论文要涉猎广泛，不要局限于本领域，这样才能对一些问题有更深刻的认识，或发现一些其他领域的解决方法并应用在自己的研究问题上。计算机科学有很多领域，每个领域的研究方法有各自的特点。也许有些问题是类似的甚至是等价的，如果能够广泛的阅读，就可以让自己站得更高、看得更远。此外，衡量一个领域的好坏，往往可以根据有多少重要原创方法出自该领域来判断。读者可以在广泛的阅读后给出自己心中认为重要的领域。

最后，做研究一定要选择自己感兴趣的课题，这样工作才能持久并容易做出成果。

论文写作和投稿

高水平的体系结构研究并非一定要以论文发表的形式来体现，如果能够将新的思想加以实现并建成真实的系统，也同样是高水平研究，比如国内的龙芯团队、国外的加州大学伯克利分校。不过这种机会不是人人都有，所以撰写论文仍然是让自己的研究成果被同行接受和理解的重要方式。

在计算机领域，通常认为会议比期刊更重要，这是因为计算机技术发展迅速，会议的审稿周期较短。因此，对于会议投稿，一般会有更多的审稿人和更高的标准。计算机体系结构中最好的会议有 ISCA, MICRO, HPCA 和 ASPLOS，其中 ASPLOS 和系统、编程语言的关系密切。为了让审稿人在有限的时间内将精力更多地放在质量较好的论文上，这些会议现在一般采取两轮的审稿方式。第一轮通常有三位审稿人，如果有至少一个正面评价，就进入第二轮。第二轮一般会有两三个额外的审稿人。在开会讨论前通常会给作者看评审意见，为其提供反驳的机会，而且程序委员会主席也会鼓励大家在网上进行充分的讨论，尽可能地让大家的意见收拢，避免开会时出现相持不下的情况，从而节省时间。在开会讨论时，一篇论文通常有一个支持者，他也是给分较高的同时对论文内容比较了解的审稿人。他先用几分钟对论文进行概括，说明接收的理由。其它审稿人可对其提出质疑，没有参与评审的评审委员会成员也可以发表意见。一部分论文会在之前或者讨论中达成共识，直接被接收。如果仍然存在分歧，则会进行审稿人之间的投票；如果票数过于接近或者相等，则需要整个程序委员会投票。如果一篇论文在被接收后仍然存在分歧，则主席会分配一个指导者 (shepherd) 进行最后的质量控制。体系结构会议的评审方式和系统会议有些相似，可参考文献 [1]。

从评审过程可以看到，一篇论文若想被接收，就需要尽可能地打动至少一位审稿人，同时不能有严重的漏洞。满足这两个条件就必须要有好的想法及严谨的表达和实验。

计算机体系结构论文的结构相对固定，包括摘要、引言、研究动机、新想法、具体实现、试验评估和结论几个部分。

摘要 用最直接的语言指出

本文的贡献，简明扼要地说明文章的主要想法，概括实验结果。由于审稿人的时间有限，一些看法在读完摘要后就可能形成，因此摘要需要突出问题的重要性和好的结果。

引言 需要力求完备(self-contained),这部分也是整个论文的缩影。引言部分像给别人讲一个完整的故事，需要有动机、现有的方案、论文的想法以及它比以前方案好的原因，最后概括实验结果。只有吸引人的引言才能使读者产生继续阅读后面章节的动力。一般来说，引言比较难写和费时间，需要反复修改。对于大部分论文，审稿人在读完引言后就已经做出拒收的决定，后面的内容只是用于寻找拒收的理由。

研究动机 尽可能从本质上讲清楚现有方案存在的问题，这就需要反复阅读别人的论文，深入理解，而不能仅从字面上比较。我们可以多想想，别人为什么这么做？是否因为以前工作的假设？这个假设现在还是否有必要？这样的问题可以不断地问下去，直到对现有工作有了本质的认识为止。

新想法和具体实现 可以专注于描述自己的成果。这里需要注意两个问题，一是新想法部分尽可能少地添加和实现相关的讨论，因为那样容易误导读者，可能使其陷入对细节的理解和困惑中；二是描述一定要流畅，即每句话之间的因果关系要自然且正确。概念一定要先定义再使用，同时只能定义一次。论文虽然不是在证明定理，但是如果我们可以用类似数理逻辑的方法认真推敲每句话之间的联系，是可以发现很多问题的。流畅地表述想法是论文成功的关键。和新想法部分相反，实现部分的描述要确保内容严谨，对细节尽可能考虑全面。

评估 是体系结构论文中很重要的部分。有一些作者可能觉得这个比较简单，给出结果就行了，其实不然，一定要仔细检查结果的合理性。容易出现的问题是，如果结果很好，就不再推敲；如果结果不好，才会寻找原因。但实际上，无论结果好坏，都应该对其进行认真的分析和确认，做到尽可能全面地解释所有结果。

反驳环节 论文作者可以在此环节回答审稿人提出的问题，一般有字数限制且不能添加额外内容。虽然审稿人在反驳环节后改分的可能性不大，但是也有例外。在反驳中，不仅需要用简明的语言，正面回答提出的问题，同时还要有选择性地回答问题。一般来说,需要多回答“专家”(专业水平或者信任度高的审稿人)的问题。

一些见闻

我于2007~2013年在美国伊利诺伊大学师从荷西普·托雷利亚斯(Josep Torrellas)教授攻读博士学位，目前在加州大学伯克利分校从事博士后研究。

托雷利亚斯是计算机体系结构领域的著名学者，为人谦和，虽然已发表过很多论文，但是每次投稿时仍然精益求精。他的最大特点是在研究上给予学生很大的自由度，能够虚心听取学生的想法。即使这个想法很不成熟，他也会和学生一起思考分析，让学生自己找出问题。如果发现一个想法是好的，那么他会以最大的热情和学生一起进行研究和撰写论文。学生往往被他的言传身教所感染，也会努力地工作和积极地思考。他一般不会给学生具体的方法，而是尽量让其自己提出方案。也许正是这种宽松和积极的环境，使得他的学生大多数都希望从事学术研究，因为他们在组里获得了信心，认为研究并不是高不可攀，自己的想法是有可能转化成顶级论文的。每次投稿时，只要有时间，他都会尽最大可能地修改甚至重新写论文，这也大大提高了论文的质量。只有真正和他一同做过研究的人才会了解他对研究的严谨和热情。他非常善于发现细小的问题，也擅长将问题一般化(generalize)。这使得论文不但提出了新的思想，还让读者学到很多背景知识。他的严谨还体现在如果发现自己已发表的论文有一个小错误，就会羞愧难当。2013年图灵奖得主莱斯利·兰伯特(Leslie Lamport)在文献[2]中曾提到，大多数计算机科学家不是因为别人发现了自己论文的问题而感到惭愧，而是对问题没被发现而感到“庆

幸”，因为这样可能又多了一篇论文。显然，托雷利亚斯教授在这方面是少数。

在伊利诺伊大学学习，不得不提萨里塔·阿德芙 (Sarita Adve) 教授。她和卡洛什·加若科罗 (Kourosh Gharachorloo) 关于共享内存的形式化奠定了存储一致性研究的基础，影响相当深远。阿德芙教授对学生的指导十分细致周到，从选题到具体的研究，有时甚至具体到模拟器的参数和配置。这可能与她注重研究的系统性有关，因此其学生的博士论文一般都自成体系。她不仅口才很好，也是一位有个性的教授。在一次演讲中，她提到当年申请读研究生时，最初并没有拿到威斯康星大学的录取通知书，为此非常郁闷，但不甘心就此放弃。于是，她开车到学校找老师面谈，最终被录取了。这种不屈不挠的精神也体现在她的研究中。

由于研究题目的关系，我和阿德芙教授的导师马克·希尔 (Mark Hill) 也有过几次接触。在一次交谈中，我发觉他非常注重生活本质。参加 2008 年在北京举办的 ISCA 会议时，他避开了官方组织或推荐的一些热门景点，而是来到北京南城的胡同，亲眼看普通北京人的生活。在研究上，他同样非常注重探求问题的本质，除了培养出阿德芙教授这样的领军人物之外，还提出了缓存替换冲突理论，开创了现代多处理器执行重放的研究，同时他领导开发的 GEMS 模拟器也被众多研究小组采用。

加州大学伯克利分校体系结构的研究风格和大多数学校不同，也许是经费来源的原因，这里的研究非常注重实现，有很全的工具链，并定期流片进行试验证。他们和工业界的关系十分密切，每年两次的 retreat 活动只邀请工业界，为期 1~2 天，由一系列的 20 分钟报告组成，主要是让公司“检阅”实验室的阶段成果。每次结束前，他们都要求参会者提出反馈意见。对于反馈的内容，他们不能反驳，同时还特别说明会认真对待反馈意见，也许还会成为接下来的研究内容。我认为这种交流模式十分值得借鉴，retreat 已经成为伯克利分校计算机系各个实验室的传统。在这里还能经常看到戴夫·帕特森 (Dave Patterson) 教授的身影，帕特森教授是希尔教授的导师，他最大的特点是拥有很强的自信心。这种信心也许来自他取得的成功，也许更多的是对本质问题认识透彻后的自信。帕特森教授研究兴趣非常广泛，而且紧跟潮流和需求来转换研究题目。他 30 年前成名的贡献是精简指令集计算机 (reduced instruction set computer, RISC) 的设计理念，后来又领导了冗余磁盘阵列、工作站网络和面向恢复的计算等有影响力的项目。近几年，他花了大量时间和福克斯 (Fox) 教授一起讲授基于服务的软件工程课程，并基于该课程写了一本新书。讲求实用的理念和找准未来的需求，应该是帕特森的研究成果最终具有影响力的重要因素。从这个角度讲，帕特森教授对未来的预见力是令人钦佩的。有趣的是，他对龙芯也很了解，在讲授的体系结构课程上曾提到，龙芯是中国的处理器。他最近还写了一篇如何管理好实验室的文章 [3]。

近年来，中科院计算所成立了体系结构国家重点实验室，陈云霁、陈天石、韩银和、包云岗等年轻的研究员屡屡在顶级会议上发表论文，无论论文的质量和数量都可以和美国顶尖大学相比甚至有所超越，我衷心希望他们带领更多的年轻学生和学者继续努力，走在体系结构研究的前沿。■

致谢：

感谢胡伟武、唐志敏研究员指导我走上体系结构研究道路，感谢龙芯团队对我的帮助。

钱学海

美国加州大学伯克利分校博士后研究员。主要研究方向为计算机体系结构与高性能计算。

xuehaiq@berkeley.edu

参考文献

[1] 陈海波. 计算机系统会议论文使如何评审的. 中国计算机学会通讯. 2013, 9(10).

[2] Leslie Lamport. How to Write a Proof. American Mathematical Monthly 102, 7 (August-September 1995).

[3] David Patterson. How to Build a Bad Research Center. *Communication of ACM*. 2014, 57(3).