★CACM 中国版★

计算机协会通讯

CACM.ACM.ORG

2014 年 12 月第 57 卷第 12 期

构建人类情感的计算模型

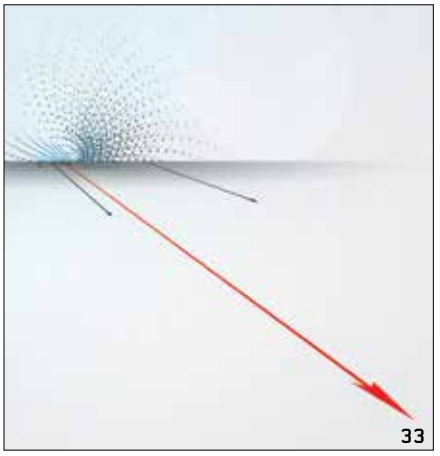
改善信息整合 • HACs

快速应变的企业•通过调度管理缓存

图灵奖激动人心的改变

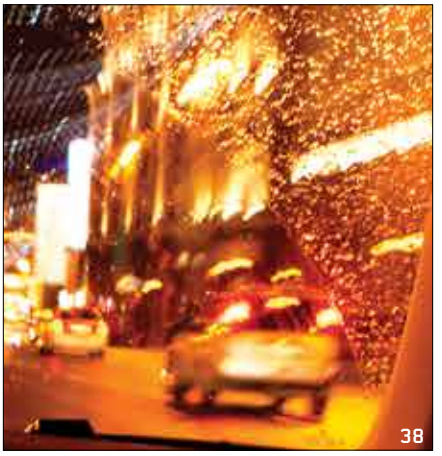
Association
for Computing Machinery

新闻



观点

实践



投稿文章

评论文章

研究亮点

关于封面：

在计算科学中，对人类行为的计算建模正起着越来越重要的作用。Stacy Marsella 与 Jonathan 所著的本期 Gratch 封面故事（第 56 页）探讨了人类情绪的建模方法，以进一步促进人机交互。封面图片由 Ollyy 提供

中间图片SERGEY MIRONOV提供；右边图片由ANDRIJ BORYS ASSOCIATES提供

Association for Computing Machinery
Advancing Computing as a Science & Profession

ACM 通讯

精选文章中文译文

ACM中国理事会办公室

中国北京清华大学
东主楼 11-236 室
邮编：100084
电话：+86-10-62785025
电子邮件：acmchina@acm.org
联系人：辛爽

ACM通讯

(ISSN 0001-0782) 由计算机协会（2 Penn Plaza, Suite 701, New York, NY 10121-0701）按月发行。

Association for Computing Machinery

新闻 | DOI:10.1145/2685372

ACM图灵奖奖金增至100万美元

2014 年 11 月 13 日，ACM 宣布其“A.M. 图灵奖”的奖金额增至 100 万美元。图灵奖是计算机科学领域公认的最高荣誉，有“计算机界诺贝尔奖”之称，谷歌公司将全额资助这一奖项。

新的奖金将达到之前奖金的四倍。将于明年年初揭晓的 2014 年 ACM 图灵奖获奖者将获得这笔现金奖励，这也体现了计算行业通过创新及技术对人们日常生活产生的日益重大的影响。此外，新的奖金额旨在进一步提高图灵奖作为计算机界至高荣誉的地位，该奖项表彰那些为计算领域作出持久而重大技术贡献的计算机科学家和工程师。

ACM 主席 Alexander L. Wolf 表示：“图灵奖目前在奖金水平上已经与全球最负盛名的文化和科学奖项相当。在谷歌的慷慨支持下，我们得以颂扬计算行业在改造世界、改变人类沟通、联系、购物和社交方式方面所发挥的中流砥柱作用，并庆贺 ACM 图灵奖获奖者在推动计算机科学和专业方面所做的具有开创性的重大贡献。”Wolf 是伦敦帝国理工学院计算系教授。

（左）谷歌公司工程部副总裁Stuart Feldman和ACM总裁Alexander L. Wolf在11月13日的纽约市新闻发布会上宣布谷歌耗资百万美元资助ACM A.M.图灵奖。

谷歌公司工程部副总裁 Stuart Feldman 表示：“能够资助 ACM 图灵奖，谷歌倍感自豪。我们认为，表彰那些为计算机科学做出杰出贡献的人具有重要意义，我们希望与 ACM 合作，让更多人认识这些创新者和他们对世界所做的贡献。”

自 1966 年设立以来，ACM 图灵奖已表彰了多位计算机科学家和工程师，他们所创造的系统以及相关理论基础推动了信息技术行业的发展。该奖项是以英国数学家艾伦·图灵 (Alan Turing) 命名的，他奠定了当今永远在线的互联世界的基础。图灵提出了会思考的机器的设想，从而催生出改变了世界的各种创新成果，例如可编程计算机、移动设备、密码学、人工智能、机器人、基因组学和科学哲学等等。

ALEXANDER L. WOLF
ACM总裁

“图灵奖目前在奖金水平上已经与全球最负盛名的文化和科学奖项相当。”

2014 年图灵奖颁奖仪式将于 2015 年春季在 ACM 颁奖晚会上举行。有关该奖项和往届获奖者的详细信息，请访问。C

译文责任编辑：陈文光

照片由 JASON GARDNER 提供

DOI:10.1145/2629458

作者：Martin Naedele、Rick Kazman 和 Yuanfang Cai

观点
打造适用于软件开发的“制造执行系统”

尝试改善制造流程中的信息整合。

如今，软件开发流程得到了众多工具的支持，如集成开发环境、配置管理系统、问题跟踪数据库、需求管理系统、估算工具、时间核算应用程序以及人力资源和财务数据的企业级资源规划系统。但其中缺少了用于支持软件开发管理的*信息整合工具*，它可以打破不同工具信息渠道的限制，消除各利益相关者的顾虑，确保管理人员制定明智的决策。毫无疑问，当今的软件经理如果不具备信息整合工具，就无法就工作效率、质量和资源配置做出最佳决策。

我们愿景的灵感来自制造业广泛使用的制造执行系统 (MES) 模型[6]（参见图 2）。[1,5]MES 系统的成功之处在于它可以将制造流程中的各方面数据整合起来。

MES 可以不断收集和整合制造流程的数据，并以此提供一系列优势，如协助确保规划和估算的准确性，利用指标来支持流程改进，更妥善地管理员工和人才，降低生产成本，改善合规性管理和供应商管理，

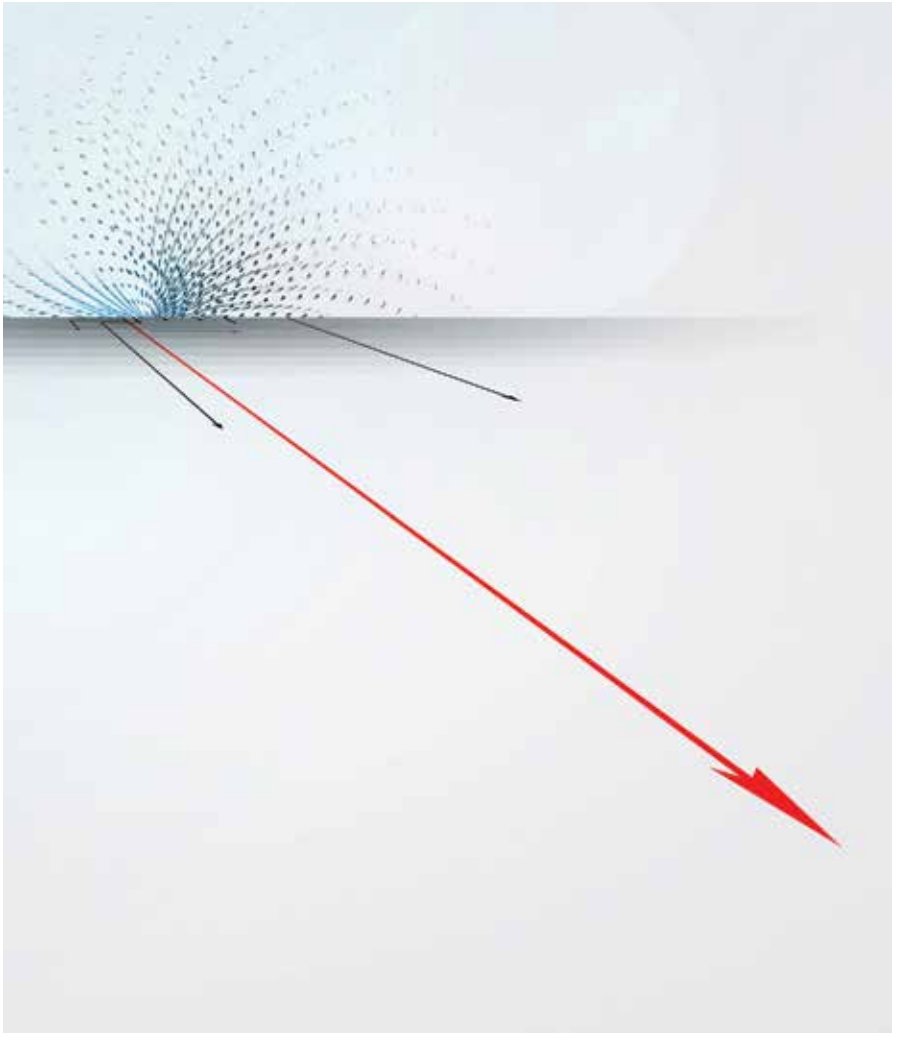


图1.MES 功能区及数据相关性

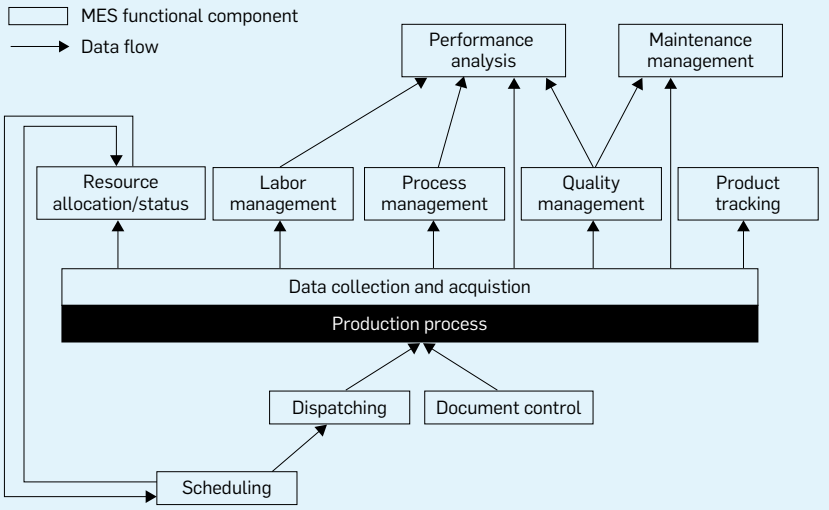


图 2.通过整合不同来源的信息来确定一项功能的所有权总成本

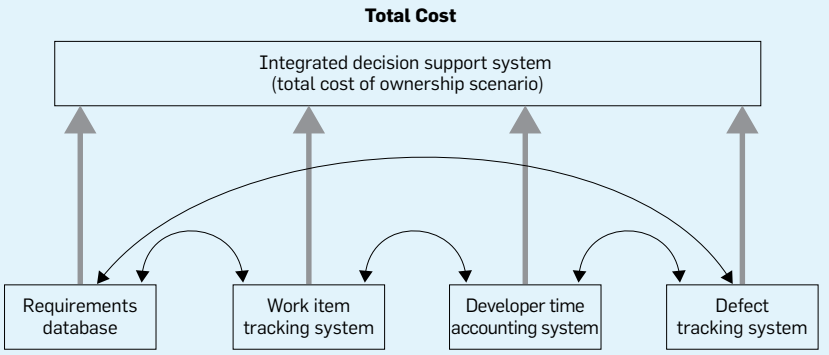
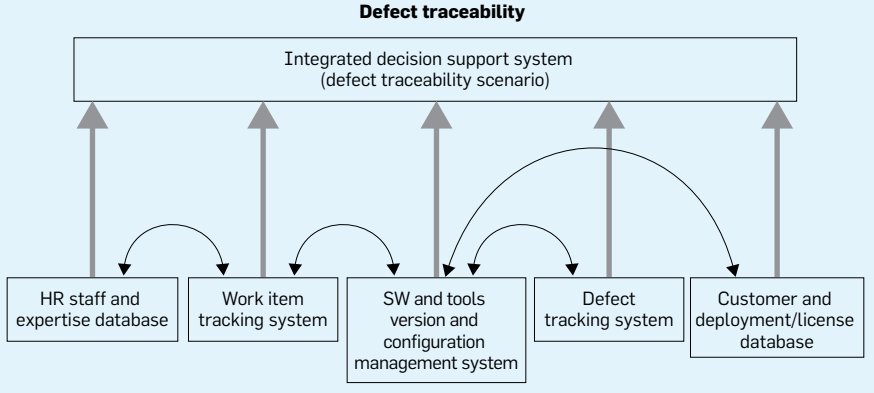


图 3.通过整合不同来源的信息来跟踪缺陷



缩短产品上市时间，协调局部生产目标与企业全局目标，以及减少人工数据录入。图1展示了MES可提供支持的10个功能区。数据从“生产流程”（此处即指软件开发）中收集而来，然后用于各类分析、监管和执行控制子系统。生产流程的推进同样依赖于MES中的三项决策：资源配置（如，Fred、Wilma和Barney负责数据库架构再归一化项目）、任务计划（如，UI刷新任务必须在中间件升级启动前完成）以及任务调度（如，Fred自1月17日周一起开始负责工作项目1763）。而参与生产流程的人员则从相应文件中获得指导。

从软件管理工具的近期发展趋势中可以发现，*软件 MES* 所需的大多数信息实际上都可以获取。为阐明信息整合工具可为软件开发管理带来哪些好处，我们描述了五种可使用“软件 MES”提供支持的情景。每种情景都需要结合使用来自不同工具的数据，其中蕴含了一些前所未有的观点。

情景 1：一项功能的总成本

推出一项功能的总成本不仅包括设计、实现、测试和部署成本，还包括在生命周期内修复缺陷或执行其他维护工作的成本。此外，还可能存在与之相关的营销、标准化和认证成本，或因招聘和留用某些专家而产生了成本。

为计算一项功能的总成本，必须将该功能在需求管理系统中的需求与相应的工作项关联起来。工作项又必须与开发人员核算和计费系统中的账目挂钩。我们需要在时间核算软件的不同成本目录下分别记录设计、实现和测试等活动。初步开发完成后，需要将问题跟踪系统中的缺陷关联到相关功能以及活动类工作和计费信息（参见图2）。为便于建立此类关联，软件 MES 将使用通用信息和标识符模型。

了解推出一项功能的总工作量和相应成本有助于对产品进行管理，确定开发优先事项，设定该功能的客户售价，并利用研发管理来改善未来工作量的估算，比如，将情景点映射到工作量和成本价值。比较不同功能在整个生命周期内的开发和维护成本是一件极其有趣的事情。

情景 2：按照开发人员的专业知识和个人成长计划来分配工作

对开发人员进行在职培训并组建交叉培训团队的必要前提是：开发人员有机会从而有助于他们掌握新的专门知识的工作。这一点通常在年度目标制定文件和相关人力资源系统中阐明，但在分配开发任务团队时往往会被忽略，这是因为负责人力资源配置的工作人员并不是开发部门的经理，他们并不了解人力资源的工作职能和目标管理系统。由于缺少可以直接参考的历史数据，交叉培训的成本无法预估。

完善的系统应结合项目资源配置计划应用程序与人力资源系统，以便自动提供关于员工能力、专业知识水平和工作效率的数据。此外，也可以为不具备相关专业知识的开发人员安排有助于个人成长的工作计划。

目前，主流的软件管理流程都将人力资源信息隔离在外，软件管理工具并不记录开发人员的专业知识和工作效率变化情况。

情景 3：
合规性文件

如今，基于软件的系统必须符合众多法规，合规性将在未来更加重要。合规性涉及到知识产权所有权和授权许可、安全性以及其他法律与合约义务，如出口和武器控制。必要的合规性证据可以关联到开发人员、流程以及项目。

为提供合规性信息，配置管理系统中的任何变更集都必须与出处文件、安全信息或适用于项目版本的其他认证信息关联起来。此外，针对项目的任何改动都必须关联到参与版本创建、测试、审查及发布的一切工作人员。必须能够从人力资源系统提取出这些工作人员的相关信息，如培训、认证、授权记录等信息以及国籍或安全许可级别等属性。

我们可能无法提前知道会在产品生命周期后期需要哪类合规性文件。因此，具备灵活的查询接口是十分必要的，这样便可以从所有交叉连接数据源创建合规性报告。主流软件管理工具侧重于跟踪软件项目，但大多缺乏关于人员和相关合规项的完整信息。

情景 4：
缺陷的可跟踪性

在注重质量的制造业（如医药或食品制造业），我们可以通过跟

踪质量问题来确定涉事人员、机器、原料、流程以及可能遭遇相同问题的其他人员。关键软件产品最好也能具备这种能力（参见图 3），即通过跟踪含有缺陷的项目来确定开发工作的所有参与者和使用工具（如编译器），并找出由相同工具和 / 或人员创建的所有其他开发项目，以检查这些项目是否存在相似缺陷。

此外，软件配置管理系统和记录软件部署的授权管理数据库最好能够建立连接，以便在必要时向使用某问题版本或配置的所有客户 / 用户告知问题情况和可能的修复方法。

情景 5：管理技术债务

记录基本子系统和组件的健康状况对任何长期和复杂的系统都至关重要，对制造工厂和软件系统也不例外。尽管软件与有形资产不同，客观上不会随着时间的推移而贬值，但会因预期和实际质量之间的差距日益扩大而逐渐“衰老”。软件的可维护性会随时间而下降，造成这一现象的主要原因是，Bug 修复和功能添加等短期活动已被视为一种“技术债务”。[2]

“我们应该将时间用于重构，还是不断添加新功能？”软件经理反复陷入这一进退两难的困境。要想从战略上解决上述问题，必须就如何使用有限的软件维护预算制定最优投资决策。为制定最优决策，我们首先需要识别故障点并统计数量，因为故障点是产生技术债务的地方。这项工作可以通过计算工作量、流程参数以及项目存储库的结构复杂参数来完成。比如，我们可以对每个文件进行监控：记录修复一项 Bug 或做出一项改动所需要的平均时间，了解某个文件与其他文件的耦合方式，确定该文件在历史上是否与很多高严重性 Bug 存在关联。[7] 就像之前的情景中那样，我们可以从软件 MES 的生产流程中提取此类信息（具体而言就是源代码库、版本控制系统以及问题跟踪系统）。

目前，要将所有必要信息都关联起来并非毫无办法，但成本较为高昂，且工作量十分繁重。

仅识别潜在的故障点还不够。为尽快作出最优决策，管理人员不但需要权衡多个因素，还应考虑以下内容：产品的未来质量、即将出现的需求、相关截止日期、开发人员具备的技能，以及涉及软件 MES 多个功能区的开发、维护和运营成本。最后，关于软件项目的每项决策几乎都是经济决策，我们应依靠全面的软件 MES 来制定此类决策。

结论

这篇《观点》所描述的情景中的信息记录方法可能并不是目前所有软件公司的标准做法，但这些方法都是切实可行的。我们所面临的难题在于：如何对互不兼容的工具生成的异构数据进行信息整合，并实现整个开发流程的可追溯性。

目前，要将所有必要信息都关联起来并非毫无办法，但成本较为高昂，且工作量十分繁重。我们已经开始利用研发决策支持系统来实现以上情景之一，该系统可以辅助识别并测算模块化债务 [3] 并为重构决策提供支持。Microsoft 启动了一项名为 CODEMINE 的类似的信息整合项目。[4]

显然，要将这些设想变为现实仍需要开展大量工作。希望本文可以唤起软件行业的行动。 C

参考资料

1. Bajric, A., Mertins, A.K., Rabe, M., and Jaekel, F. A success story:Manufacturing execution system implementation.In *Enterprise Interoperability IV* (Springer 2010), 357–366.
2. Brown, N., Cai, Y., Guo, Y., Kazman, R. et al.Managing Technical Debt in Software-Reliant Systems.In *Proceedings of the FSE/SDP Workshop on the Future of Software Engineering Research at ACM SIGSOFT FSE-18*, (Santa Fe, NM, Nov. 2010).
3. Cai, Y., Kazman, R., Silva, C.A., Xiao, L., Chen, H-M.A decision-support system approach to economics-driven modularity evaluation.In *Economics-Driven Software Architecture*, Elsevier, 2013.
4. Czerwonka, J., Nagappan, N., Schulte, W., and Murphy, B. CODEMINE:Building a software development data analytics platform at Microsoft.*IEEE Software 30*, 4 (July–Aug. 2013), 64–71.
5. Manufacturing Execution System (MES) Market—Global Forecast and Analysis (2011–2016) (Feb. 2012); http://www.marketsandmarkets.com/Market-Reports/manufacturing-execution-systems-mes-market-536.html.
6. MESA White Paper 6.*MES Explained:A High Level Vision*, 1997; MESA Organisation; http://www.mesa.org/pdf/pap6.pdf.
7. Schwanke, R., Xiao, L., and Cai, Y. Measuring architecture quality by structure plus history analysis. In *Proceedings of the 34th International Conference on Software Engineering*, May 2013.

Martin Naedele (martin.naedele@ch.abb.com) 是 ABB Corporate Research 的信息技术部工作人员，该机构位于瑞士巴登。

Rick Kazman (kazman@hawaii.edu) 是檀香山夏威夷大学信息管理技术系的信息技术管理教授。

Yuanfang Cai (yfcai@cs.drexel.edu) 是德雷塞尔大学计算机科学系的计算机科学副教授，该学校位于宾夕法尼亚州费城。

译文责任编辑：谢涛

DOI:10.1145/2677032

文章编写主导者 acmqueue
queue.acm.org

不久之后，每家公司都会成为软件公司。

ERIK MEIJER，VIKRAM KAPOOR

快速应变的企业：信奉黑客之道

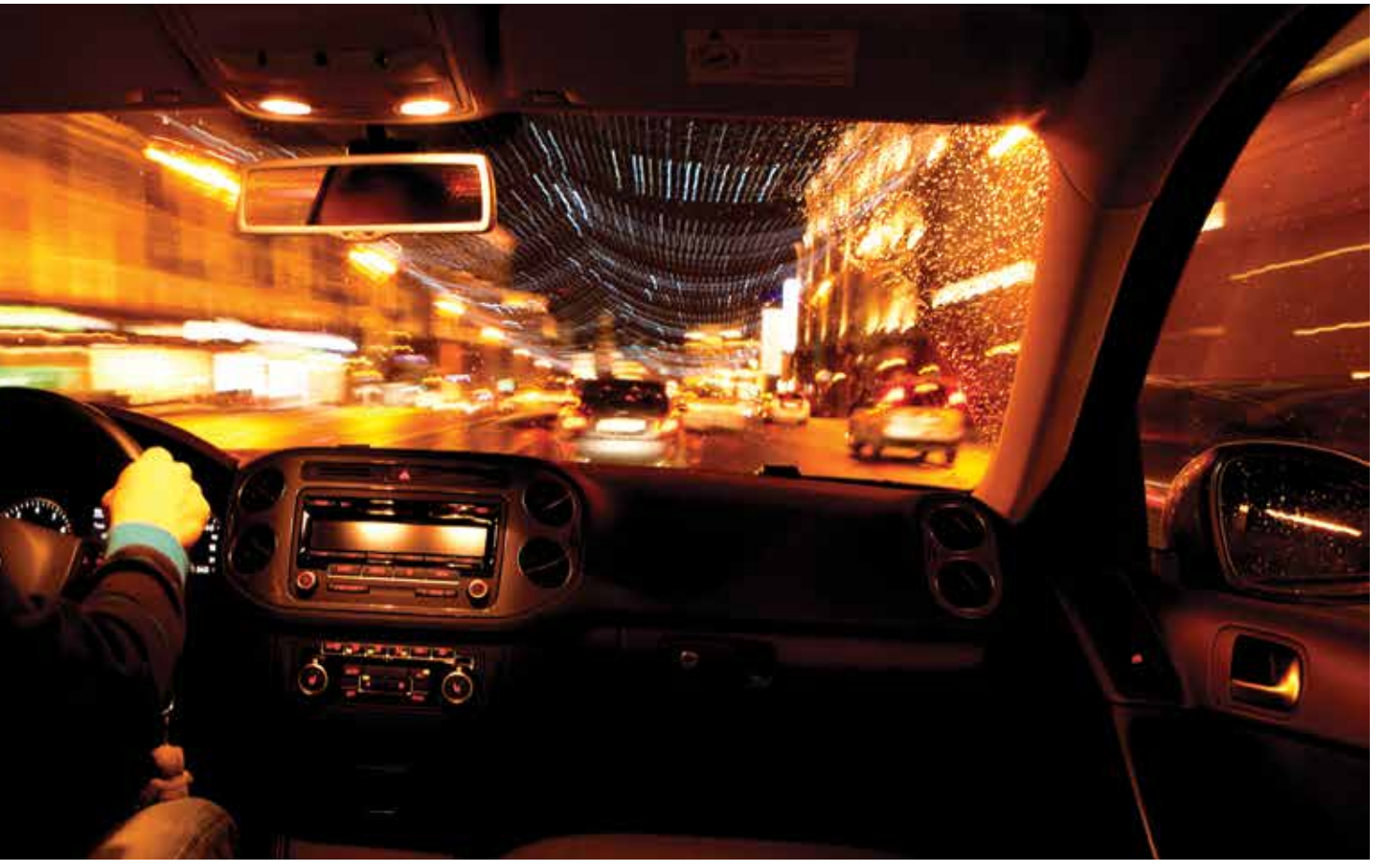
脸谱（Facebook）成立于 2004 年，但在 2014 年 7 月便已跻身标普 500 最有价值公司前 20 强。短短 10 年间，这家软件公司已与 IBM、Oracle 和可口可乐比肩而立。[3] 按 2014 年的市值（表 1）衡量，增长最快的前 5 家公司中，有三家是软件公司：苹果、谷歌和微软（实际上，人们还可以说英特尔也是受软件驱动的公司。这样，5 家中有 4 家都是如此）。

热门和新兴的公司（比如 Uber、Tesla 和 Airbnb）本质上也是软件公司。这些公司正在分别颠覆传统的出租车、汽车和酒店市场。

与此相反，排名倒数的 5 家公司中，大部分是在行业经营数十年之久的传统公司。这些公司 2014 年上半年的市值均出现了下跌（表 2）。

鉴于这一信息，人们不禁会问，为什么基于软件的公司能够接管世界？[2] 答案很简单，软件赋予了公司一种能力，使得它们能够快速应变，依赖数据驱动，从而能够快速应对变化。为了解释这一点，让我们从通俗的视角看下控制理论。

在控制理论中，*开环*（无反馈）控制系统仅使用外部输入来计算系统的控制输入，而不考虑系统的当前输出（图 1）。在完全了解静态世界的情况下，开环控制系统运行良好；不过，当环境演化时，或不存在受控系统的完美模型时，该类型的控制系统面临崩溃。举例来说，在开环系统中，某位出租车司机每次送完旅客后均返回相同的酒店，与作为同行的出租车司机抽烟聊



图片由SERGEY MIRONOV提供

天，他碰上新旅客的几率不大，而且他也没考虑市中心的剧场表演刚刚结束，所以需求在其他的地方。

*闭环控制系统*考虑了系统的当前输出，并把这种信息反馈到控制器中，它把这些测量值与当前的外部输入结合起来，共同持续纠正和优化受控系统（图 2）。例如，Uber 了解当前的交通状况，任意时刻的确切交通供求情况以及顾客的历史价格弹性；因此，它能实时优化搭乘出租车的价格，引导司机前往交通需求最高的地点。

在 Office 2007 之前，微软总是通过大量的用户研究进行引入和测试 Office 中的新特性。在 Office2007 时，已有的用户界面被“带状功能区（ribbon）”取代，让很多用户感到沮丧。当 Windows 8 中的开始按钮消失时，世界上的所有 Windows 用户都丧失了他们积累的，控制 Windows 操作系统的肌肉记忆。当公司变老，充耳不闻反馈和外部输入时，这种失误会经常发生。忽视反馈的公司变成*无控制系统*，仅根据其内部的回音室生成输出（图 3）。

内部的流程以及控制风险的愿望开始否决真正的创新。投资者所施加的优化季度结果和盈亏底线的压力，进一步鼓励了这种充耳不闻的现象。最后同样重要的是，因为这种递归反馈系统会逐渐排除那种接受外部刺激的员工，转而偏爱那种崇拜遗产和流程的应声虫，整个员工队伍僵化了。

快速应变的公司是软件公司

在接下来的 10 年里，每种业务都会数字化，并有效地转变成软件公司。利用软件后，从更广的角度上说，利用计算机思维后，可使用闭环反馈系统让业务快速应对变化。这点会成为公司能否在这个新世界存续的关键因素。在这个新世界中，业务 = 数据 + 算法。下文列举了一些快速应变的公司。

与传统的出租车公司不同，Uber 没有自己的车队。Uber 的价值在于实时数据和把实时数据转变成决策的算法。未来几年内，使用自动驾驶的汽车替代司机后，Uber 可能会用自动化赶走这种平衡关系中所有的人类，不过自动驾驶的汽车本身只能依赖软件技术的进步实现。

Netflix 没有运营自己的数据中心，而是在公共云基础设施上运行所有的操作。简言之，云计算把硬件虚拟化成软件。[5] 只要数行软件代码，在一眨眼的时间内，公司便能够添加计算和存储容量，或者释放多余的设施。云计算实现了这种效果，让公司不需要在物理的计算机和存储设备上缓慢前行。

通过大大加快输出和输入间的反馈速度，软件提升了敏捷性。过

表1增长最快的五家公司

Facebook	118.80%	2004
苹果	55.90%	1976
英特尔	48.50%	1968
吉利德科学公司（Gilead Sciences）	47.60%	1987
微软	41.80%	1975

表2市值下跌的倒数五家公司

葛兰素史克	6.30%	1900（并购）
花旗集团	5.00%	1812
菲利普莫里斯（Philip Morris）	4.90%	1900
赛诺菲	4.30%	1973（并购）
沃尔玛	2.60%	1962

去，公司能按季度度量绩效，但它很难快速调整，应对环境中的变化。与此相反，Facebook 每天发布几个新的产品版本，根据网站实际使用情况的实时反馈确定功能的优化和补丁。像亚马逊和缤客（Booking.com）之类的公司会连续地对用户进行 A/B 测试或多臂赌博机实验，优化其网站上的购买率。

亚马逊甚至向第三方开发人员免费提供其 A/B 测试技术（https://developer.amazon.com/public/apis/manage/ab-testing）。像 Facebook 和 Netflix 之类的其他公司也采取了同样的做法，开源了大多数内部软件。

连续的交互和 A/B 或多臂赌博机实验不仅需要快速推出新产品，而且还意味着错误的决策可以迅速撤回。接受决策可能是错的并予以承认需要勇气，而不愿承担风险的传统企业倾向于引入流程来打消这种勇气。

快速应变的公司接受故障总会发生这一事实，并通过故意引发故障来保护自己免受连锁故障。故意制造故障来让系统变得更健壮，更能防范故障。这一想法本身就会立即让很多传统的中层管理人员焦躁不安，难以承受。然而，在 Netflix，在数据中心中“把一群猴子放出来”，让它们拔出虚拟的电缆并造成重大破坏，这已成为常见的实践。很多敏捷开发方法提出在编写代码前编写测试用例。然而，代码部署在运行环境中，人们无法在测试环境中如实地构建其中各种复杂情况的模型。相反，软件应用故障的处理方式应该与任何其他故障一样：立即把代码部署在生产环境中，当问题发生时，进行回滚。

基础心理学教导我们，为了让人类学习（换句话说，进步），他们行为必须立即得到反馈。吸烟者继续吸烟，因为吸烟可以给他们立刻带来满足感，但是肺部疾病的反馈效果多年以后才会显现。开发者也是人，通过对他们的代码质量立即给予物理反馈，可以激励他们编写更好的软件。通过让开发人员佩戴寻呼机，发生问题时半夜把他们叫醒，以及让他们对“整个堆栈（又称为 DevOps 模型）”负责，公司可以实施这种反馈环。断开开发过程与运营侧的联系后，取消了系统中价值颇高的（递归）反馈环，因此减少了系统整体的应变性。

观察闭环系统的另一种方式是（非确定）Mealy 机。[7] Mealy 机是一种有限状态机，其下一个输出值和下一个状态由当前状态和当前输入决定（图 4）。Mealy 机的观点强调了组织的“企业记忆”，且假定基于当前输入的决策可能会受到在过去多次输入和输出迭代中累积形成的状态的影响。然而，在元层面，企业运营所处的环境也会随时间改变；因此，企业记忆需要偶尔“重置”，因为它不再跟踪相关的输入。

具体来说，以微软为例，自 1975 年以来，在发行盒装产品方面，它积累了引人注目的企业记忆。在过去，当世界渴求盒装软件时，“微软机器”能够优化输出，创造丰厚的利润。然而，环境变了，开始要求云服务。突然间，盒装软件的优化知识成了增长的障碍。微软最近不得不解雇数千人，以“忘记”过去，转至“云优先”的方向。

每家公司都应该有“第 10 人”（“当 9 个人都同意一件事情时，第 10 人有责任反对，而不管其反对的理由有多么荒谬”）或在流程中放入一定量的随机和混乱的[6]魔鬼代言人（吹毛求疵的人）。这样可以防止公司陷入局部最优的圈套，或做出错误的决策，由于顾客偏好的变化被忽视以及无人胆敢指出皇帝没穿衣服那样。

快速应变企业中的人员层面

从人的角度来看，管理一家由软件驱动的公司与管理传统的公司截然不同。一旦公司意识到，要想成功，必须接受快速应变的企业由软件驱动这一观点，它就必须应用开环反馈控制，使发动机运行顺畅。更重要的是，它必须深深信奉开发人员是增长引擎这一观点。

软件开发人员完全不像身着正装的传统企业员工，而无能的老板往往最难理解这点。或许这种区别可用最简要的方式总结如下：“你不能让绵羊赛跑，也不能放牧赛马”[8]以及“管理开发人员如同养蜂人养蜂。”[1]

在 Netflix 的幻灯片介绍中，描述了“我们文化的七大方面”（价值观源于我们珍视的事物；高绩效；自由与责任；情景管理而非控制；高度一致，松散耦合；给予市场最高薪酬；晋升与发展），它们清楚地勾勒了以软件为中心的企业中人员管理的准则。[4] 在下文中，我们从 Netflix 的七大核心价值观中选取了四个进行说明：

*情景管理而非控制。*根据经验，即兴表演过爵士乐或戏剧的人知道，创造力只能在严格的约束下盛放。让开发人员在由所有计算函数组成的软件世界中漫无清晰目的地瞎逛是项目失败的主要因素之一，而不是因为缺少流程或计划。如果我们给予开发人员严格的准则和命令，指出需要优化的方面，并对他朝那些目标做出的决策结果立即给予实时反馈。极客们爱竞争的天性会被激发出来，系统会迅速地朝最优解决方案迭代。中层管理的唯一目标是提供并加强这种情景，使开发人员在这些定义清晰的边界内保持快乐，富有生产力。我们不关心开发人员对公司长期战略的看法。

*给予市场最高薪酬。*在职业运动中，为了组建处于顶级联盟的球队，花几百上千万美元吸引顶级球员并支付薪酬相当正常。为了吸引和挽留最好的开发人员（并诱惑年轻人从事高技术工作，而不是成为律师或股票经纪人），需要根据他们创造的价值和他们有限的职业寿命给他们丰厚的薪金和一块大蛋糕，而不是根据他们打卡的小时数。

*高度一致，松散耦合。*下文节选自《*舰队海军陆战队手册（Fleet Marine Force Manual）1——作战（Warfighting）*》中的章节。在文中，用*软件*替换*战争*后，我们可以立即感受到软件开发的日常情况：[9]

“软件是一项复杂的工作。它受到人类意志的操纵。它具有冲突、不确定、易变、危险和混乱的特点。虽然软件的本质不会变，但是它仍然无法预测，且受到各种物理、精神和心理因素的综合影响。虽然软件同时具有艺术和科学的特点，但它主要由人类经验的决定。”

图1.开环系统

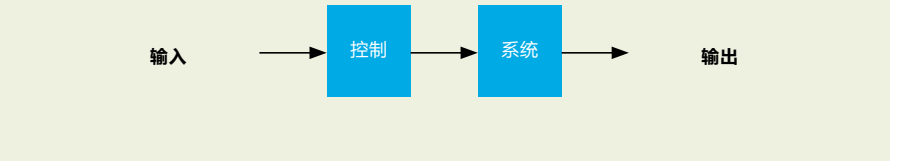


图 2.闭环系统

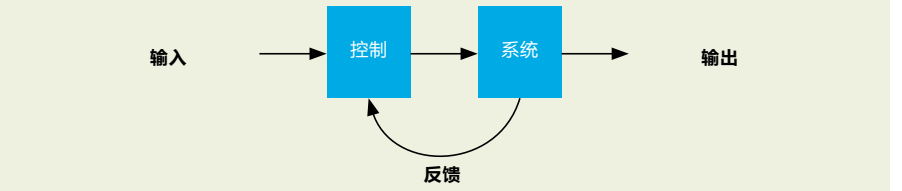
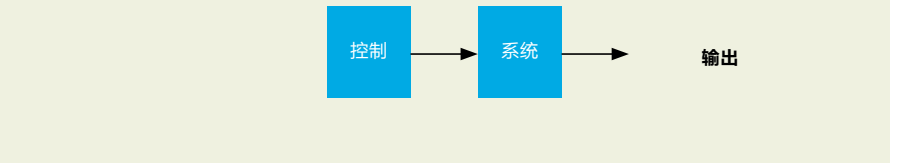


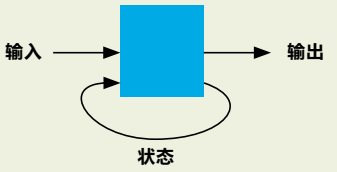
图 3.无控制系统



为了处理此类情况，军队的分区组织结构和行动已经磨练了多个世纪。因此，从军事中可以学到很多东西。具体来说，软件开发应该遵循指挥的哲学（Philosophy of Command）：“为了顺应战场的易变和混乱本质，指挥权必须下放。下级指挥员必须自己主动采取行动完成任务，实现上级的目的。”

*自由与责任。*维基百科把传统定义为，“群体或社会内部沿传而来的观念或行为，带有象征意义或特殊意义，起源于过去。”在快速应变的企业中，传统没有任何作用。在传统的企业中，传统体现在流程中。流程是引发组织性失聪的因素，导致无反馈系统的失败。盲目坚持流程会赶走富有创造力的人，嘉奖无生产力、但精打细算的管理人员。在快餐业，需要流程让资质平平的员工得出足够好的结果。与此相反，高技术公司由一流的黑客组成，不需要流程。

图 4.Mealy机



没有其他公司像 Facebook 那样鲜明地体现了黑客文化。在 Facebook 申请上市 IPO 时，该公司的 CEO 和创始人马克 - 扎克伯格（Mark Zuckerberg）在致信投资者时强有力地阐释了这一观点：“黑客文化还相当开放，任人唯才。黑客们认为，最好的观念和实现应该一直能赢——而不是最擅长游说想法的人，或是管理的人数最多的人赢。”[10]

转变成快速应变的企业

软件是无法阻挡的力量。为了生存，企业必须顺应潮流，而不是阻挡变化，最后与铁匠和敏捷导师等行业一样消失。下文列出了一些需要铭记在心的导则。

1. 欣然接受软件是您所有工作的核心。在十九世纪和二十世纪，机器和物流是大多数组织的枢纽。如果您拥有业界最好的机器和供应链，您就能赢。在二十一世纪，拥有最好的软件则成为实现业务繁荣的大道——也就是说，您的软件会一直直接由市场的实际需要和需求决定（闭环和数据驱动的设计决策驱动具体实现）。

软件解除了创造和改变服务时在速度方面存在的物理限制。交付渠道（如移动设备和广泛使用的 Wi-Fi 技术）消除了过去必须克服的物理交付边界。结果产生了一种新的现实，即软件驱动的企业，它的 CEO 必须深入了解软件（以及创造软件的黑客），洞悉把软件融入业务模型的方法，以便管理的组织取得成功。一些最成功的 CEO（马克 - 扎克伯格（Mark Zuckerberg）、拉里 • 佩奇（Larry Page）、比尔 • 盖茨（Bill Gates）和拉里 • 埃里森（Larry Allison）均具有开发人员的背景，这一点也不奇怪。

2. 管理您的组织机构时，应在组织的每个层级设置机制相同的闭环实时反馈系统，其中各层之间均有双向的反馈环。组织倾向于坚持那种在过去为他们创造成功的，“经过验证的”业务模型。但在由软件驱动的公司组成的新时代，过去的多数业务模型无效了。在快速应变的企业背后，演进在持续的反馈环中不断发生，其与创造力、残酷的实验和数据驱动的决策结合后，激活了企业实时学习和应对环境的能力。组织需要施行软件驱动的业务模型，使用关于市场机会的战略思考替代静态的商业方案。然后，再把这些假设转换成 A/B 实验，在市场上的实际客户上进行测试，并开展实时分析，检验假设是否有效。

没有其他公司像Facebook那样鲜明地体现了黑客文化。

传统的商业方案基于（未检验的）历史假设进行了前瞻，而软件驱动的业务模型则基于对源于实验和反馈中的数据进行持续的诠释。

3. 不管人们是否喜欢，高效的组织是具有层级的。由于某个原因，各公司似乎一直在忽略军事上使用层级结构所取得的成功，这一结构则基于“适者生存”的思维方式演化而来。与此相反，即便坚守事业部的组织结构发挥的作用可能更大，他们还是不停地尝试新的组织结构，比如“功能性的”或“扁平化的”结构。

用本文介绍的概念来讲，公司应该反复地分解为互相通讯的闭环控制系统，其中每个内环的情景（目标）由直接包含它的上一级环设定，但这些目标的实现过程则由组成内环的团队决定。由于每个内环会向包含它的环提供反馈，组织整体会成为一个单一的闭环控制系统。注意，在编程时，分而治之也是解决复杂问题的最有效方式之一。所以，在使用同样的方法去构建组织架构的时候，分治也具有重要意义。

4. 开发人员是增长的引擎，负责运营的战术层面。由于快速应变的公司会反复分解，它的最底层会变成由 10 名开发人员组成的小队，或者用亚马逊的话讲，“双比萨团队”。这些黑客小队在甲板下铲煤，让船一直航行。他们是实际创造软件的人，而这些软件支撑了组织的运转。因此，在快速应变的公司中，让黑客开心是人事最重要的方面。人员管理变成了黑客管理。开发软件已经堪比耍鸡蛋——换句话说，为了保持高产出，除了正在处理的代码之外，开发人员的心中应该空无一物。管理层应该为开发人员

提供他们工作所需的所有软件和硬件，而不应抱有任何疑问。

开发人员的薪酬应该堪比体育明星。这不仅因为他们是高效地创造公司价值的人，还因为编码的强度会损害开发人员的身心健康。与求胜心切的运动员一样，到了 35 岁左右时，开发人员基本筋疲力尽了。

5. 中层管理人员只要提供联结战略与战术的运营场景。在很多传统公司中，管理通常与（精细）控制含义相同——控制（假定的）风险、控制行为、控制不可控的因素。中层管理人员不应促使事情发生，而应设定清晰的边界和目标，容许事情发生，然后只要放手。管理者需要从牧羊人变成养蜂人。这意味着，没有站立会议、没有燃尽图、没有每周进展报告，特别是，没有计划的扑克游戏。

适合成为养蜂人的人不多。他们承担了责任重大的工作，需要找到确切的正确约束，使得开发人员在此类约束下产能最大。如果开发人员的约束设置过多，他们就会不高兴，因此徒劳无益；如果约束过少，又会让他们失去重心，因此还会徒劳无益。当然，对开发人员的约束应该与公司的总体战略完美地协调一致，而且状态信息必须妥善地沿链条向上反馈，进而给中层管理人员施加更大的压力，促使他们提高判断能力，提升技能。

倘若您本身不是黑客，您无法理解和欣赏黑客的思维。只有具有黑客背景的人才能管理黑客，正如最佳的体育教练曾经是顶尖的运动员；而且，与成功的教练类似，成功的中层管理人员应该得到丰厚的薪酬；但是，如果事实证明他们无法让团队获胜，那么应该迅速撤换他们。

6. 决策主要由数据驱动，而非地位和年资。当开发人员编写软件时，他们使用调试器、分析器和静态检查程序等工具改进代码的质量，提高代码的性能。面临与代码有关的决策时，只使用这些工具生成的数值和数据。对软件驱动的公司而言，同样如此。流向链底部的控制信息和流向链顶部的反馈信息必须得到真实数据和测量值的支撑，而且还必须一丝不苟，避免掺杂污染原始数据的任何主观解读——直到需要进行人工解读，提供创造力的时刻。组织中的每个层级必须设置数据科学家 / 数据驾驭者（实质上只是特定的开发角色），帮助挖掘、解释和可视化数据，进而助力决策。

7. 资深的领导团队设定长期的战略（宏观）方向。快速应变的企业是一种分部门的组织，其中战略方向仅源于资深管理团队。高级管理人员不仅负责确保反馈引擎执行顺畅，更重要的是，还应实施一种元反馈环。这种环确保公司正在从一直变化的外部世界中接收正确的输入信号，且公司生成的输出仍为预想的输出；最后，他们接受的反馈保持与其决策相关。因此，资深管理团队还应负责总体的人员管理，比如获取新的产能来应对外部变化，驱逐组织中的过时产能，刷新企业记忆。

结语

每家公司都将变成软件公司，这一过程可能会比你预想的要早。管理软件公司最平常的方法是把公司当成元软件应用，反复地把公司架构成由多个互相通信的闭环反馈系统组成的层次，然后依照久经考验的军队等级结构构建严格的层级架构，并应用软件激发的分析和调试技术来优化公司的盈利能力。在运营方面，公司不应谈论代码，而应该遵循“黑客之道”，使用持续改进和迭代手段专注于编写代码，并雇佣公司能够负担的最佳开发人员。 C

queue.acm.org 上的相关文章

All Your Database Are Belong to Us
Erik Meijer
http://queue.acm.org/detail.cfm?id=2338507

The Theft of Business Innovation
http://queue.acm.org/detail.cfm?id=1874536

The Software Industry IS the Problem
Poul-Henning Kamp
http://queue.acm.org/detail.cfm?id=2030258

参考资料

1. Card, O.S.How software companies die.Windows Sources:2008; http://www.netjeff.com/humor/item.cgi?file=DeveloperBees.
2. Denning, S. Why software is eating the world.Forbes (April 11, 2014); http://www.forbes.com/sites/stevedenning/2014/04/11/why-software-is-eating-the-world/.
3. Dogs of the Dow.Largest companies by market cap, 2014; http://www.dogsofthedow.com/largest-companies-by-market-cap.htm.
4. Hastings, R. Netflix culture:Freedom and responsibility, 2008; http://www.slideshare.net/reed2001/culture-1798664.
5. Helland, P. Condos and clouds.*ACM Queue 10*, 11 (Nov. 2012); http://queue.acm.org/detail.cfm?id=2398392.
6. Legge, A. What *World War Z* can teach you about critical thinking.*Evidence Mag*, 2014; http://evidencemag.com/world-war-z/.
7. Mealy, G.H.A method to synthesizing sequential circuits.*Bell System Technical Journal 34*, 5 (1955), 1045–1079.
8. Thomas, D. Developing expertise:Herding racehorses, racing sheep.*InfoQueue*, 2008; http://www.infoq.com/presentations/Developing-Expertise-Dave-Thomas.
9. U.S.Marine Corps.*Warfighting*.U.S.Government Printing Office, 1997; http://www.marines.mil/Portals/59/Publications/MCDP%201%20Warfighting.pdf.
10. Zuckerberg, M. Mark Zuckerberg's letter to investors:The hacker way.Reprinted in *Wired* (2012); http://www.wired.com/2012/02/zuck-letter/.

Erik Meijer（emeijer@applied-duality.com）是Applied Duality 的创始人，代尔夫特理工大学大数据工程教授。他因其对编程语言（如Haskell、C#、Visual Basic、Hack和Dart）的贡献以及大数据技术领域的工作（如LINQ和Rx 框架）而闻名。

Vikram Kapoor（v.kapoor@prowareness.nl）是Prowareness 和iSense的创始人。2013年，他被同行推举为荷兰“年度IT CEO（IT CEO of the Year）”。

译文责任编辑：崔斌

DOI:10.1145/2631912

情感计算模型丰富了人工智能理论并推动了人本应用程序的发展。

作者：STACY MARSELLA 和 JONATHAN GRATCH

构建人类情感的计算模型

情感在人类行为中扮演何种角色，这一争论由来已久，并且与计算机科学之间的联系日益密切。两千五百年以前，亚里士多德指出，适度的情感（如愤怒）是有益的，尤其有助于人际交往。这一观点与当代心理学理论存在惊人的相似之处。在有些场合表达愤怒情绪是值得提倡的，而在另一些场合缺少愤怒表现会被视为不妥。四百年后，斯多葛学派提出了相反的观点，塞内加认为情感（如愤怒）会对理智构成威胁，“不受情感干扰的理智才是强大的”。在 18 世纪，戴维·休谟彻底驳斥了斯多葛学派的观点。他强调情感是关键的刺激因素，并指出“理智是激情的附庸，并且应该如此”。

在人工智能 (AI) 和代理人的研究史上也出现过类似的对立观点。哈伯特·西蒙 [35] 在其早期著作中指出，情感对智力行为具有关键作用，情感作为一项中断机能，可以让有机体在相互冲突的目标中作出切换，使反应过程和深思过程达到均衡。马文·明斯基则提出这样的问题：没有情感的机器人是智能的吗？然而，在 20 世纪后期，斯多葛学派的观点在 AI 研究领域占据上风，情感与智力被视为相互对立。

当代心理学和神经科学研究让这场争论愈演愈烈。情感评估理论 [21] 侧重于研究理智对情感的诱发作用，认为情感是由人们评估自身与环境之间的关系所产生的，对适应性反应具有指导作用。近期的研究确定了情感对决策的关键作用，主要依据为情感过程的神经功能缺陷会导致决策错误。[3] 这与西蒙的情感中断观点一脉相承，情感是感知和反应模式的先决要素。比如，发怒的人更易于察觉到威胁 [11]，并且通常会以激进方式应对。[20]

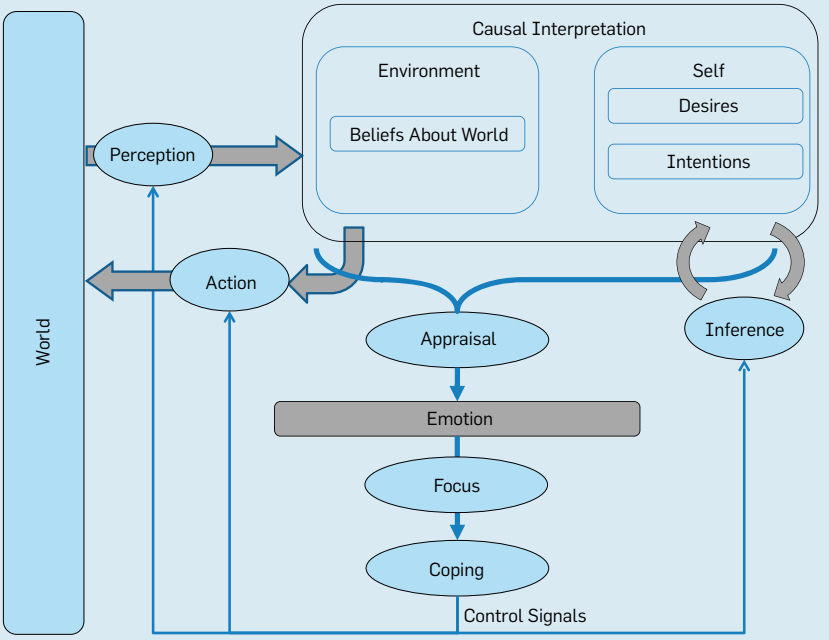
相关研究表明，情感和表达在社会交际中发挥强大的适应性作用，这呼应了亚里士多德的观点。比如，情感的表露可以传达个人的

» 重要见解

- **任何智能实体在面对动态、不确定和社会环境时都必须具备类似情感的过程。**
- **情感心理学理论（如评估理论）可用作旨在识别、建模并模拟人类情感的机器人的结构规范。**
- **将心理学理论转为工作计算模型可以提供具体化的形式，揭示隐含假设并建立用于实证研究的动态构建，从而推动计算机科学的发展。**

图 1. 情绪反应的动态过程的图示。

图 2. EMA 模型中的评估、应对和再评估。



信念、愿望和意图等信息，因此有助于人们相互了解并施加影响。愤怒和内疚通过最大限度减少社会冲突来改善集体效应，[15] 而表达悲痛会得到社会的同情。[13] 休谟认为，这些信号能够产生巨大影响，是因为情感的刺激作用是人所共知的。如果有人对某情境感到不安，我们可能会有所察觉，但促使我们采取行动的，往往是个人行为中透露出的情感信息，比如父母的愤怒表情，或者孩童的痛哭。

上述发现重新唤起了 AI、机器人和代理人研究界对情感建模的兴趣。值得一提的是，在软件代理人和机器人研究领域，情感计算模型已成为研究对象，并被视为一种解决控制和决策权衡的方法，即利用模型将认知资源用于解决有机体的适应值问题。[5,7,33,36]

自主代理人和多代理人系统以及人机交互领域（如 Conati 和 MacLaren[8]）的研究则探索如何利用情感的社会功能来促进计算机系统与人类用户之间的交互。虚拟人借助情感和情感表现来利用情感表达的社会功能，以激发并建立共鸣 [22] 和纽带。[27]

计算模型研究也改变了人类情感理论的构建和评估方式，对这些理论造成一定影响。将理论转为计算模型的必要条件是，从形式上详述各过程及其相互作用，从而揭露隐含假设和隐含复杂性。将模型导入大型模拟中会进一步揭示隐性问题并拓展理论的范围。例如，研究人员通过将情感导入更全面的人类行为模拟中，可以研究关于评估过程与其他认知过程、感知和行为的关系等根本性问题。[4,6,12,25]

在概念层面上，计算模型丰富了语言与情感理论。例如，一些情感理论因计算模型的应用而改写，并借鉴了来自 AI 领域的很多概念，包括知识表示（如 Gratch 和 Marsella[17]）、计划（如 Dias 和 Paiva[12]）以及神经网络（如 Armony 等人 [1]）。

在实证层面上，计算模型与传统理论相比，扩大了人类的预测范围。计算机模拟是一种探索情感过程的时间动态的方法，可以用来预测这些动态的来源和时间进程。先使用计算模型可以更全面地探索实验条件的操作，因为在不利条件下切除功能或测试响应可能成本高昂，存在风险或者引起关于活体的伦理问题。[1] 模拟可以揭示意外的模型属性，提示相关人员开展深入研究。此外，虚拟人 [37] 和软件构件已经融入情感和情感表达模型，它们拥有像人类一样的外观和行为模式，能够与同样生存在虚拟世界中的人物进行交互。这些系统从根本上确保了情感研究可以在虚拟生态环境中进行。

本文将探讨我们在情感计算模型方面的研究工作，详细阐述促进研究及其在认知架构中的实现的设计原则。按照我们的设想，情感计算模型必须解决一项根本性难题，

即情感是如何在诱发情境中产生和演变的，诱发情境包括物理刺激和复杂社交情境，同时还要掌握情感在个人行为和社会交际中发挥的作用。情感反应时而激进且被动，时而看似较为慎重，会在几分钟、几天或几周内表现出来。比如，你对同事的行为产生了怨恨，但你可能不会立即表露出来，而是先揣测他究竟为什么会这样做。总而言之，情感在本质上是动态的，它与外部世界的动态性以及个人心理、认知和行为过程的动态性密切相关。

我们借鉴了关于情感的著名心理学理论，探讨了如何利用情感计算模型来解决情感时间进程和诱发条件范围这两大问题。我们在展示情感计算模型以后，阐述了该模型如何转变了我们对情感的看法，并改变了有关认知与情感关系的古老争论。此后，我们讨论了该模型的一项 AI 应用案例，并验证其有效性。无论使用情感模型来改善某些应用程序，还是将其用作人类情感研究中的方法论工具，都必须解决如何验证该模型在相关用途中的有效性的问题。然而，我们先利用一个显而易见的例子来展示情感对行为的作用。

示例

为了支撑我们对评估理论及其建模计算方法的讨论，我们先描述了真实世界中的一例情感诱发情境。我们所记录的这一偶发事件足以说明情感过程以及情感诱发条件范围充满动态性。作为虚拟人建模工作的一部分，我们要与几位演员进行即兴对话，并记录下他们使用的非语言行为。我们在洛杉矶的某个盛夏夜晚进行了排演，期间一只鸽子从敞开的窗户飞了进来；图 1 展示了其中某位演员的一系列反应。尽管这一突发事件使严谨的反应分析增添了不确定性，但我们建议按照以下方式来解读：

鸽子从窗户飞进来，然后试图飞回窗外，但一头撞在窗沿上。演员受到声音的刺激，并转过身去。她的第一反应显然是惊讶，与查尔斯·达尔文[9]所论述惊讶特征相吻合，即眉毛上扬。该表情可以扩大视野范围，并警示他人可能发生了意外事件。

随后，她的眉毛降低，嘴巴张开。嘴巴张开具有功能意义，可以用作视觉或声觉信号（如大喊），或者用于吸进空气，给血液充氧，以准备战斗或逃跑。眉头低下代表消极的情感反应（恐惧）。

这时，演员远离了鸽子的威胁（逃跑反应），同时开始呈现更具攻击性和防御性的姿态。她用双手抓起一把伞作为武器，可能准备击打那只鸽子（战斗反应）。

她继续快速离开（大约七英尺）。在这一过程中，她放低了雨伞，并用一只手捂住嘴，她的反应像是房间内的其他人陷入了困境。她快速作出了代表停止或小心的经典手势，同时转身向鸽子走去。她的关注点从自身转移到鸽子，因为鸽子被另一名演员抓住了，并且可能会受到伤害。在图 1 中的最后一幅画面中，演员拿着一条手帕，想要救助那只鸽子，她用手帕把鸽子包裹起来，然后把它毫发无伤地放出窗外。然而，当鸽子被安全地放出窗外后，该演员的情绪仍处于高度戒备状态，令排演无法继续进行。

我们从该事件的解读中发现，情感的很多功能作用都值得 AI、自主代理人和多代理人系统领域的研究人员关注。

这也详细地解释了情感在中断和重新确定认知优先次序方面的功能，西蒙也对此进行了论述。[35] 我们观察到，当前关注中断、认知过程以及物理反应之间发生了切换，以便为解决新威胁情境补充资源。具体而言，就是收集与个人相关的更多信息，中断与即兴会话相关的当前目标，将目标切换到应对潜在威胁，采取行动并准备以物理（如战斗和逃跑）和生理反应来应对（如吸进氧气给血液充氧）威胁。演员的上述一切活动都是为了适应自身与环境之间的动态关系。情境会发生变化，原因来自两个方面：演员以外的事件（比如，鸽子从窗户飞进并冲向演员）和演员自身的反应（比如“武装自己”并远离该事件）。演员对情境的解读似乎也发生了演变，她先将鸽子视为威胁，然后又将鸽子视为受害者。这表明感知和推理过程拥有内在动态性，得出推论、重估情境以及重新计划都需要时间，因为新知识的吸收也需要时间。更为持久的动态性也是存在的。在事件过后，演员仍保持着变化后的情绪，该情绪具有功能作用。高度兴奋状态会让个人对未来可能发生的相似事件做出快速反应。

随着事件的发展，情感在改善多代理人协调和集体效应[15]方面的作用也显露无遗。情感及其表达可以传递关于心理状态的信息，包括可影响交际的情感和意图；比如，表达愤怒可能阻挡敌对代理人。情感的表达也有助于调整和协调集体反应；表达恐惧可以向代理人的集体内的成员传递通用的威胁信号，表达同情可以影响人际态度和同感，因此有利于建立共同目标。

从这些反应和行为的快速演变中，我们还发现了情感作为刺激因素的强大作用，这与休谟所提出的情感刺激作用遥相呼应。

最后，排练室事件的发展轨迹也揭示了情感具有快速变化的性质。从整体来看，演员的各种反应也体现了以下变化过程：先是对意外事件感到惊讶，然后关心如何保护自己，最后关心他人的安危，也包括最初被视为威胁来源的鸽子的安危。这一转变的发生速度极快。在大约 2.6 秒钟内，演员的目标从逃跑变为战斗，

再变为帮助鸽子。眉毛上扬的表情（通常代表惊讶）持续了约 60 毫秒，眉毛和下巴降低的表情（通常代表愤怒和对威胁的反应）持续了约 300 毫秒。演员的关注点从自身安危转向他人安危，其情感经历了从恐惧 / 愤怒到同情 / 安心的动态变化，与之密切相关的应对反应也出现了从防御 / 进攻到帮助的相应转变。

如上所述，该示例了反映了情感过程模型的根本性难题。首先，模型必须能够解决这一问题：情感是如何在诱发条件范围中产生并演变的，这些诱发条件包括简单的物理事件和复杂的社交情境。其次，情感反应快则在数毫秒内发生，慢则在数分钟或数天之内展现，并会涉及一些思考或推理过程。情感的动态性与外部世界的动态性以及个人心理、认知和行为过程的动态性息息相关。

从理论到模型

情感计算模型通常以情感心理学理论作为计算模型的依据。理论至少可以定义情感是什么，愤怒等情感状态由什么构成。此外，理论也能阐述情感的前因后果，这些因素对构建情感过程计算模型具有关键作用。

由于某些情感心理学理论存在较大差别，构建情感计算模型变得更加复杂。理论上的差别主要体现在以下几个方面：情感固有的组成部分（如认知能力、身体过程、行为倾向还是反应能力）；各组成部分之间的关系（如认知能力先于还是后于身体过程）；表征性的区别（如愤怒是语言现象还是自然现象）。

本文不会讨论哪些理论才有助于模型开发，如欲了解更全面的相关论述，请参见 Marsella 等人的著述。[26] 不过，我们可以将众多当代理论大致划分为三类：

情感离散理论。[14] 这些理论认为，有限数量的核心情感是由生物本能决定的，它们是与生俱来的自然情感（比如愤怒和忧伤）。而且，这些情感的表达是各民族和文化所共通的。某相关观点认为，情感可以概念化为独特的神经回路；[29] 请参见 Armony[1] 的论文，来了解条件性恐惧反应和相关神经网络模型理论。

情感维度理论。这些理论认为，情感和其他情感现象应概念化为连续空间（通常 2D 或 3D）上的点，而不是离散的类别。[2,28] 维度理论还认为，离散的情感类别（如愤怒）不具备特定的生物学基础，也就是说，任何情感类型都没有特定的大脑区域或回路。相反，它们是人类分配给结合松散的心理状态和物理状态集合的常识心理学标签。[2] 建立在维度理论上的计算模型通常会使用 Mehrabian 和 Russell 的 PAD 理论，[28] 三个维度分别对应愉悦度（衡量情感正负状态）、激活度（表示感情激活水平）以及优势度（衡量影响力或控制力）。很多计算模型都依赖维度理论来控制行为选择，为映射到情感类型的显性感觉状态建模，以及建立情绪模型。[4,7,16]

情感评估理论。这些理论起源于人们对诱发情感的详细心理过程的探索。[21,32] 评估理论认为，情感是在比较个人需求与外部要求这一过程中诱发的，就像示例中的演员要根据快速变化的环境来调整自己的目标一样。也就是说，情感的解读不能单靠环境，也不能单靠个人。相反，情感反映的是“个人与环境的关系”。评估理论进一步假设，这种个人与环境的关系可以按照一套标准来描述或评估。这套标准叫做评估变量、检查项或者维度；比如，事件是否符合个人的目标或关切？事件是由谁引起的？事件的发生是否在预期中？我在事件的发展中起到什么作用？这些评估检查项的结果会依次映射到情感。一些评估理论的变体进一步详述了产生的情感会如何影响个人的认知和行为反应，例如，Lazarus[21] 详细探讨了情感如何引起尝试改变外部世界的应对反应（问题导向应对）或个人的情感反应（情感导向应对）。

评估理论不但对心理学造成深远影响，还成为构建情感计算模型的主要框架；请参阅 Reisenzein 等

人[31]的著述来深入了解使用评估理论的模型。评估理论的主导地位也反映了其在情感心理学领域的重要性。然而，从根本上说，评估概念（涉及愿望、预期和原因等判断）完美地映射到传统 AI 概念（如代理人的信念 - 愿望 - 意图模型），我们会在描述基于评估的模型时强调这点。

评估理论也并非十全十美。评估理论侧重于推理过程的作用，这一特点对计算建模者最具吸引力，也是心理学中最具争议之处。评估理论注重对个人与环境关系的某些外显表征的判断，传统上认为评估涉及大量的认知推理，是情感反应出现的前兆；也就是说，个人必须先进行推理，然后才能在情感上对一些刺激作出反应。这一特征似乎完全不符合我们所观察到的现象，演员在应对鸽子时出现的情感反应是极快的，而且像是不由自主的。在二十世纪，评估理论的主要反对者[40]认为，情感的被动性是固有的，评估宜被视为情感反应的结果而不是前兆。

然而，如果从计算角度来审视评估理论，这些批判大多毫不相关。我们在建模工作中采用以下方式来解决针对评估理论的批判，我们将评估和推理视为两个不同过程，它们处理相同的个人与环境关系的心理表征。我们将表征的构造和评估区分开来。表征的构造可能涉及缓慢而慎重的推理或快速而被动的推理。无论如何，评价表征的评估过程都是快速、并行、自动发生的。

EMA 模型

按照我们的定义，评估计算模型的组成部分包括：评估 - 导出过程（解读个人与环境关系表征，并导出一套评估变量），情感 - 导出模型（利用评估变量来产出情感反应）；一套行为结果过程或应对策略（由情感引起，随后操控个人与环境关系）。情感模型的构建取决于这三个过程的具体表现形式，并且会因此而有所不同（对应评估理论的不同变种）；可以参阅 Marsella 等人的著述。[26] 我们目前使用的是我们独有的 EMA（情感和适应）模型[17,24]（参见图 2）。

情感计算模型应用于虚拟人的设计中，虚拟人是可以自主表达情感并通过语言和非语言行为进行面对面交互的角色。

个人与环境的关系。EMA 模型使用“代理人与环境关系”的外显表征作为各种评估过程的输入和输出。该外显表征是代理人对自身与环境关系的看法，由一套信念、愿望、意图、计划、实效性及可能性组成，这些概念从决策理论和 AI 计划的传统观念中借鉴而来。我们将外显表征叫做“因果解释”以强调因果推理和评估过程的解释性（主观性）特征的重要作用。因果解释（对应认知架构术语中的代理人工作记忆）对推理过程的输入、中间结果以及输出进行编码，这三项要素调节着代理人的目标与物理及社会环境（如感知、计划、解说以及自然语言过程）之间的关系。推理过程可以是快速的，就像演员立刻意识到飞向自己的鸽子构成了威胁；推理过程也可以更加慎重，就像演员要制定计划来救助鸽子。

因果解释是代理人对其与环境关系的最新认识的快照。这种认识会根据观察或推理随时发生变化；比如，如果代理人或其他社会角色的行为改变了环境，一旦代理人的感官对此有所察觉，相关影响就会反映在因果解释中。此外，单纯的思考动作可以改变感知到的代理人与环境关系；比如，当代理人制定计划或形成意图时，这些意图也会表现为因果解释中的变化。

因果解释的表征支持评估 - 导出过程进行快速且基本被动的评价。因此，快速评估要求推理过程使用因果解释。

评估 - 导出过程。评估理论采用一套特定的评估变量来描述情感诱发事件，但大多都未明确阐述这

些判断背后的过程。我们假定评估过程是快速、并行且自动发生的。我们实现这些特征的方法是，将评估过程建模为一套连续活动的特征探测器，利用探测器将因果解释的特征映射到评估变量。因果解释中的所有重要特征都会在同一时间自动地分别得到评估；比如，假如因果解释对具有好坏两种后果的行为进行编码，那么每个结果都会并行地评估，而且，任何会影响到这些结果的合意性或可能性的因素，一旦被记录在因果解释中，都会自动地反映在评估里。从这个意义上来看，评估虽然不会改变因果解释，但会就其内容提供持续更新的“感情总结”；请参阅 Gratch 和 Marsella[17] 的著述来进一步了解区分评估与维护因果解释的认知操作的架构原则。

EMA 模型中的评估过程将数据结构（或评估结构）与每项命题关联起来。评估结构针对每项命题都使用一套持续更新的评估值。评估值包括：

视角。判断命题的观点；EMA 模型结合独有的视角和其他代理人的视角来评估各项事件。

相关性。如果命题对一些代理人具有非零效用，那么 EMA 模型判断该命题是相关的。

合意性。命题对于代理人的价值可以为正，也可以为负。例如，命题会促进还是抑制效用？

可能性。命题的可能性。

预期性。在多大程度上可以从因果解释中预测出某状态。

因果归因。谁应该受到奖励 / 责备？

可控性。结果会因代理人的行为而改变吗？

可变性。结果会因其他代理人而改变吗？

每个被评估的事件都会映射到某些类型或强度的情感实例，该话题会在下文予以讨论。

融入情感和情感表达的虚拟人可以利用和回应情感的社会功能，仿佛具有生命一般。

情感导出。在 EMA 模型中，评估会指导代理人的应对反应，但会因整体情绪状态而存在偏差。某些子符号（大脑或身体）过程 [40] 对于协调评估模型和经验观察（如情绪对判断力 [34] 和核心感情 [2] 的影响）具有重要作用，情绪为这些流程充当了代理 (proxy)。我们曾在鸽子案例中讨论过，情绪影响可以作为适应性功能。

EMA 模型在评估层面使用多个评估结构，分别适用于因果解释中的每项命题。同一个事件中也可能存在多个评估结构，因为事件会以不同方式影响不同目标。各评估结构（和相关强度）聚合成情绪，即一种随时间推移而发展的评估事件的平均值，并与原始诱发事件脱离关联；也就是说，情绪并不是故意的。EMA 模型将情绪调节应用于各评估结构。EMA 的实时应对反应是由基于激活作用的简单关注焦点自动选择的，并从评估结构中找到最近访问的具有最高情绪调节强度的结构作为应对。

感情结果。最后，EMA 还包括了评估过程中的应对计算模型。应对决定了代理人如何回应事件的评估值，目的是改变个人对其与环境关系的主观性解释。EMA 使用应对策略来维护理想的焦点事件或评估实例，或扭转不受欢迎的事件或实例。这些策略的根本工作方向与激励它们的评估相反，确定可生成评估并且应该维护或改变的因果解释的特征（如信念、愿望、意图和预期）。

在 EMA 模型中，应对策略是启用或禁止认知过程的控制信号，认知过程会根据因果解释工作，比如察觉和避免威胁，改进计划，或添加 / 删除目标和意图。应对支持行为符合当前的评估模式。EMA 模型从应对策略如何影响代理人的关注、信念、愿望或意图的角度出发，定义了应对策略，

从而正式实现了临床心理学文献探讨的应对策略。

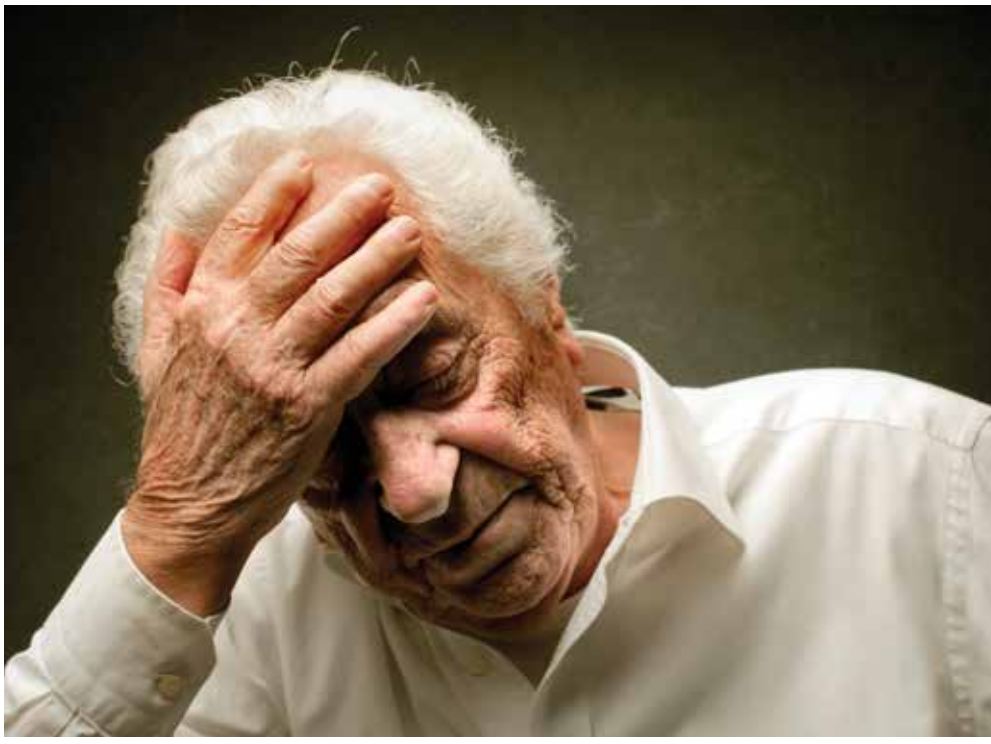
与关注相关的应对。某些应对策略调整了代理人对环境特征的关注，通过改变其工作记忆（因果解释）来改变其情感状态；比如，在鸽子的示例中，演员转向刺激物的行为可以视为“寻找关于意外事件的信息”。或者，应对策略可以“限制”信息；比如，关心课程项目的学生可能避免查看到期日期。

与信念相关的应对。很多应对策略会建议通过改变信念状态来调整情感状态；比如，学生如果在重大考试中失意，可能会向老师推卸责任，从而减轻内疚感。学生在参加一场困难的考试前，可能会利用“美好幻想”来缓解压力，他们会幻想自己能够取得好成绩。

与愿望相关的应对。通过改变目标的优先级别来调整情感状态。学生在考试前可能会觉得无所谓，认为考得好不好并不重要。学生也可能“乐观地看待此事”或者“找到一线希望”。如果学生自觉无法通过考试，他会在考试前自由玩乐。

与意图相关的应对。通过改变意图或采取行动来调整情感状态。比如，学生在考试前可能会有学习的打算。有趣的是，意图的形成会改变 EMA 模型中的当前情感状态，即便计划并未付诸实施。或者，学生可能会向学友寻求有利支持，以此缓解压力。此外，如果学生考试成绩很差，他会向父母道歉，并承诺以后会努力学习，以此减轻内疚感。学生也会无奈地顺从，通过放弃某个意图来达到理想状态（比如，认为医学院预科班是可望不可及的）。

EMA 模型并行地提出策略，但按顺序采用策略，并且包含一套偏好解决关系；例如，在 EMA 模型中，如果可控性这一评估值较高，那么问题导向的策略更受青睐（如采取措施，制定计划），如果可控性和



可变性较低，那么注重情感的策略（如轻视、顺从和幻想）将优先考虑。

图 1 概括了因果解释、评估、关注以及应对如何相互影响，并与代理人的感知和推理过程发生作用。请回想鸽子情景中，模型的动态性涉及多个来源。事件在外部世界中自然发生，通过感知能力来改变因果解释。代理人也可以执行动作，以此影响外部世界和相应的因果解释。代理人自身的推理过程（此处包括维持信念、选择行为和制定计划）可以改变因果解释。因果解释的变化引起了评估的变化。评估和应对对情感过程极为重要，二者在一个循环中发挥作用，以此对某情境进行评估，从而引起情感和应对反应，并对可改变个人与环境关系的代理人推理过程造成影响。这种影响进而左右了针对该情境的后续重新评估。因此，在 EMA 模型中，情感意义会随着个人与物理和社会环境的相互作用而演变。

在理论形成和测试中的作用

如上所述，情感与理性思维之间的关系是一个由来已久的争论话题。计算模型的一项好处在于，它们是潜在的强大研究工具，并且会对这场争论带来深远影响。计算机模型将促使研究人员明确阐述心理过程如何实现，相互关联，并随时间推移而显现。

值得一提的是，EMA 模型明确阐述了情感与代理人认知过程之间的关系。评估在本质上被视为针对心理表征内容的被动的自发性评价。诱发条件、感知过程和推理过程（包括深思和反应过程和）使用个人与环境关系的表征，三者的时间进程存在着差别，因此情感动态性的时间进程存在不同。这使得该模型能以统一的方式来解释快速且自发的情感反应以及较慢且慎重的情感反应。相比之下，一些更为复杂的模型和理论需要假定多个过程。[12,32]

EMA 假定认知和感知能力对个人与事件的相关性进行编码，使评估变得简单、快速且普适，并且会随着认知过程更新代理人与环境关系而不断演变。因此，评估不是一个过程，它是对认知和感知的输出要求。由认知和感知过程生成的值

构成了评估值。同样，我们提出了应对策略的逆要求，应对输出的是对认知和评估所依赖的关注、信念、愿望和意图的调整。应对可以被视为逆向的评估过程，旨在调整因果解释来改变后续评估。

我们在解决任何情感模型都存在的关键难题时得出了这些设计说明。情感模型的一大难题在于解释情感过程的快速而看似被动的动态性，这也是评估理论亟需攻克的一项难题，我们在鸽子示例中进行了概述。[40] 另一大难题在于情感是如何在一系列诱发情境（包括物理事件和复杂社交情境）中产生并演变的。

在 EMA 模型中，评估通常可以解决复杂的社会交际以及物理威胁需求，主要因为评估独立于一系列由它利用的感知和认知过程。EMA 模型也概括了情感在代理人整体架构中的作用。在 EMA 中，评估和应对通常对调解代理人的反应发挥重要作用。这延续了西蒙的情感中断机制观点 [35] 和情感决策作用研究。

EMA 模型使用评估和应对形状，但不决定代理人的反应，比如，代理人是否通过形成意图来做出应对，取决于计划和问题解决过程能否找到适当的意图或计划。请回想鸽子情景中女演员拿着的雨伞。在 EMA 模型中，准备打击鸽子的反应取决于面对威胁的情感反应，提前拿着雨伞可能启动（或准备）了意图的认知形成。

与此同时，EMA 模型也具有明显的缺陷，就像计算模型和基本理论研究都存在难题。更重要的是，EMA 模型的评估过程侧重于导出评估值，并将情感类别大致视为附带现象。推理过程通过使用因果解释来支持评估，这是 EMA 模型的另一处局限性。主要原因在于，因果解释在为其他代理人的信念建立模型方面的能力有限，并且没有为更加复杂的社会情感（如尴尬）建立模型。推理过程还需要更妥善地限制应对策略，因为信念更正缺少限制；整体模型允许幻想和顺从改变信念和目标，但忽视了对相关信念或目标的潜在影响；请参阅 Ito 等人的著述 [19] 来了解基于效用的方法，以解决该局限性。

然而，EMA 等模型的优点和缺点都在于它们基于以下观点：情感计算模型是一种强大的工具，可以用来解决两方面的问题，即情感背后的过程以及情感与认知的关系。构建 EMA 模型促使我们就以下内容作出具体说明：个人与环境关系的表征，基于这些表征的评估的计算，感知、记忆、解释和推理在评估中的作用，应对的建模，以及评估、情感和应对之间的关系。此外，一旦在计算层面实现模拟，就可以系统地探索并操纵模型，从而生成预测结果并根据受试者反应加以验证。我们在开发模型时也发现了这些必须同时在模型和理论中予以解决的关键缺点。

在虚拟人和 HCI 中的作用

如上所述，情感计算模型应用于虚拟人的设计中，虚拟人是可以自主表达情感并通过语言和非语言行为进行面对面交互的角色。融入情感和情感表达的虚拟人可以利用和回应情感的社会功能，仿佛具有生命一般。

在 EMA 模型中，我们的目标不单单是添加情感。我们已经研究了虚拟人的认知过程是如何围绕评估和应对来组织的，以及应对可如何促进多个认知能力的设计和集成，这些认知能力是创建类人行为所必备的，包括感知、计划、对话过程和非语言交流。评估理论建议使用一套通用的标准和控制策略来指导并协调多种认知和社交功能行为。

无论处理感知输入，还是考察替代计划，认知过程必须做出类似决定。处理中的情境 / 输入是合意并且期望的吗？该模块有资源来处理所涉问题吗？这些同质特征往往是可能的，即使各个组成部分明显不同。使用相同的一般术语来描述每个模块的状态，就可以加工出适用于各模块的一般控制策略，从而使全局行为更加一致。

请考虑 Swartout 等人 [37] 解决自然语言模糊性的案例。人类参与者出现在虚拟环境中的事故现场（参见图 3）；一群虚拟人聚集起来；一个受伤的男孩躺在地上；一名虚拟士兵和一支部队已经集合完毕；现场还有一辆损坏的车辆。人类参与者问士兵：“这里发生了什么？”这个问题是模糊不清的，因

图 3.虚拟人事故现场（南加州大学创新技术研究所）。

为这里发生过很多事情：参与者刚刚到来，军队集合完毕，一场事故发生，一群人聚集起来。在某种意义上，所有这些事件都可以是正确回答，但如果士兵回答“你刚开车来到这里”，你就会觉得他很愚蠢。预期的回答应该是对事故的描述。

要想给出正确回答，就必须确定谈话中的语义重点。常见的启发式算法使用“时效性信息”，或找出最近讨论或发生的事情。在这种情况下，时效性信息并不适用于事故情景，因为事故之后又发生了多个事件。

然而，人们往往最关注那些影响他们情感的事情，因此，需要使用基于情感的启发式算法来确定关注点。由于虚拟士兵融入了 EMA 模型，语言例程访问了他对事故产生的情感，并且使用该信息来确定语言关注点，因此士兵可以给出最适当的回答，即详细描述事故情况以及发生方式。

此外，在 EMA 模型中，决策制定是由对替代计划的评估而驱动的，并且使用应对策略来驱动谈判中的替代反应 [38]（比如，尝试避免谈判，寻找与综合解决方案相反的分配解决方案）。EMA 模型还可以用于影响代理人的信念；比如，假如虚拟人因受到谴责而承受巨大压力，它可以通过改变对谴责的看法来减轻内疚感，或减少对遭受报复的担忧。

验证

必须判断情感计算模型能否实现最终目的。接近自然人行为的模型与驱动合成电脑角色的模型可能明显不同。我们开展研究的根本目的是建立能准确预测人类情感的模型：人们所面临的情境如何诱发情感产生，情感如何影响人们的在各时段的信念和行为，情感的表露如何影响其他社会参与者的信念和行为。本文回顾了指导我们进行模型验证实证研究的原则；请同时参阅 Staller 和 Petta 的著述。[36]

接近自然人行为的模型与驱动合成电影角色的模型可能明显不同。

*了解情感与不断发展的情境之间的关系。*尽管众多情感理论假定情感起源于个人与环境之间的关系，但大多数实验工作都回避了对改变、评估和再评估的动态循环进行直接操作（我们在鸽子案例中的观察）。更常见的情绪诱发研究则认为，参与者的情绪脱离情境而独立形成（比如，倾听快乐或忧伤的音乐 [34]）或者由“单稳”情境所产生（比如，观察轮盘旋转的反应），因为它们提供了大量的实验控制。

受到情感诱发情景案例（比如鸽子情景）的启发，我们开始尝试开发一种可以系统地操作个人与环境关系的时间动态的技术。我们已经探索了将实验室参与者置入情感诱发任务的技术，并使个人与环境关系的各个方面都在随时间推移而发展并改变的情境中得到系统地控制，从而测量个人评估、情感和应对倾向中的实时变化。[35] 我们随后将这些反应与 EMA 模型的预测结果进行比较。

*关联个人内部情感和人际间情感。*正如鸽子情景中那样，情感既有助于协调个人行为，也有助于协调社会行为。大多数实证工作都采纳以上观点之一，但我们认为，认知和社会视角是相互制约理论和模型发展的两个信息来源。因此，我们尝试证明，用来预测个人内部情感前因后果的计算模型也可以用来驱动类人代理人的社会行为并诱发与人际情感交流相似的社会反应。

逆向评估理论是一个例子，[10] 它阐述了评估理论如何提供解释性框架来预测个人对他人的情感信号做出反应。该理论认为，人们使用情感表达来了解他人的想法。当人们观察到他人对环境事件做出情感上的反应时，就会推断或逆向推断经历者如何是评估情境的，并使用推断结果来重新了解什么目标会

引起这种评估。我们让参与者与由 EMA 模型驱动的计算机代理进行博弈游戏，为以上观点提供实证支持。

*以模型为导向的实验。*最后，我们尝试在计算方法和心理学方法之间建立真正的协同以了解情感。我们并不满足于简单地证明我们的模型“适合”人类数据，而是要展示该模型可以生成全新的见解和预测，让我们更好地了解情感。从这个角度来看，虽然计算模型更加具体形象，能够进行更深层的假设和实验，令传统理论难以企及，但计算模型只是理论而已。

关于此类以模型为导向的实验案例，可以参阅我们的因果归因评估建模著述。我们的一位学生为因果归因评估背后的因素（如因果关系、意图、预知以及胁迫）创建了模型。该模型可以生成假设为不同的情境，而且这些情境应产生不同的评估。我们向人类参与者展示了这些情境，该模型的预测结果与人类参与者的反应是一致的。[23] 另一位学生使用评估理论导出了关于问题描述或设计方式如何影响决策制定的模型。[18] 该模型使用评估变量，并突破了得与失的一维处理框架。[39] 该模型进而用于生成展示给受试者的替代决策情景，并且模型的预测结果被证明是正确的。

在不断发展的任务中审视情感，将情感的认知功能和社会功能关联起来，采用以模型为导向的实验，这三项原则将继续指导我们的研究。然而，哪种研究方法才更加适用，这取决于研究情感的根本目的；比如，为智能系统设计的模型可能会避免使用一些看似不合理的人类应对应用程序。

结论

在过去的半个世纪中，采用计算方法来理解人类行为并促进人机交互的跨学科研究经历了快速发展，情感计算模型研究是其中的重要组成部分。当代人类情感研究有时会被认知科学和 AI 领域所忽视，但它确定了情感在人类行为中发挥着重要作用。因此，研究人员开始利用情感计算模型这一工具来研究人类情感，并将该模型运用到应用程序中。我们在本文中阐述了此类模型的各种用途，并从科研界的一般视角出发，提供了关于 EMA 模型研究成果的详细信息。

研究人员开始利用情感计算模型这一工具来研究人类情感，并将其运用到应用程序中。

通过在代理人上建立评估理论模型，我们获得了关于情感与理智关系的有趣观点。评估理论认为，情感用来概括刺激反应，它以更一般化的方式通过生命机体反应级别来描述刺激类型。就代理人而言，评估维度是一般化而统一的价值函数，确立了事件的个人值与社会值。评价标准（如合意性、应对潜能、意外性以及因果归因）与任何社会代理人存在明显关系，无论该代理人是否被视为具有情感。评估结果采用统一方式来描述，以协调系统内的应对反应。应对反应用于指导代理人对诱发事件作出特定反应，从根本上帮助代理人找到自己的生态位。因此，情感与代理人（人类或人工）的反应方式和应对外部世界的方式有着密不可分的关系。

鸣谢

本文提及的研究工作得到了美国陆军和空军实验室的支持。 文中的论述和观点并不代表美国政府的立场或政策，也不暗含任何官方认可。

C

参考资料

1. Armony, J.L., Servan-Schreiber, D., Cohen, J.D., and LeDoux, J.E.Computational modeling of emotion:Explorations through the anatomy and physiology of fear conditioning.*Trends in Cognitive Science 1*, 1 (Apr. 1997), 28–34.
2. Barrett, L.F.Are emotions natural kinds?*Perspectives on Psychological Science 1*, 1 (Apr. 2006), 28–58.
3. Bechara, A., Damásio, H., Damásio, A., and Lee, G. Different contributions of the human amygdala and ventromedial prefrontal cortex to decision making. *Journal of Neuroscience 19*, 13 (July 1, 1999), 5473–5481.
4. Becker-Asano, C. and Wachsmuth, I. Affect simulation with primary and secondary emotions.In *Proceedings of the Eighth International Conference on Intelligent Virtual Agents* (Tokyo).Springer, 2008, 15–28.
5. Blanchard, A. and Cañamero, L. Developing affect-

modulated behaviors:Stability, exploration, exploitation, or imitation?In *Proceedings of the Sixth International Workshop on Epigenetic Robotics* (Paris, 2006).
6. Broekens, J. Modeling the experience of emotion. *International Journal of Synthetic Emotions 1*, 1 (Jan. 2010), 1–17.
7. Broekens, J., Kosters, W.A., and Verbeek, F.J.On affect and self-adaptation:Potential benefits of valence-controlled action-selection.In *Proceedings of the Second International Conference on Bio-inspired Modeling of Cognitive Tasks* (Manga del Mar Menor, Spain).Springer, 2007, 357–366.
8. Conati, C. and MacLaren, H. Evaluating a probabilistic model of student affect.In *Proceedings of the Seventh International Conference on Intelligent Tutoring Systems* (Maceio, Brazil, 2004).
9. Darwin, C. *The Expression of the Emotions in Man and Animals, Third Edition.* Oxford University Press, New York, 1998.
10. de Melo, C., Gratch, J., and Carnevale, P.J.Reverse appraisal:Inferring from emotion displays who is the cooperator and the competitor in a social dilemma. In *Proceedings of the Cognitive Science Conference* (Boston, July 2011).
11. DeSteno, D., Dasgupta, N., Bartlett, M.Y., and Cajdric, A. Prejudice from thin air.*Psychological Science 15*, 5 (May 2004), 319–324.
12. Dias, J. and Paiva, A. Feeling and reasoning:A computational model for emotional agents.In *Proceedings of the 12th Portuguese Conference on Artificial Intelligence* (Covilhã, Portugal).Springer, 2005, 127–140.
13. Eisenberg, N., Fabes, R.A., Schaller, M., and Miller, P.A.Sympathy and personal distress:Development, gender differences, and interrelations of indexes. In *Empathy and Related Emotional Responses New Directions in Child Development*, N. Eisenberg, Ed. Jossey-Bass, San Francisco, CA, 1989, 107–126.
14. Ekman, P. An argument for basic emotions.*Cognition and Emotion 6*, 3–4 (1992), 169–200.
15. Frank, R. *Passions With Reason:The Strategic Role of the Emotions.* W.W.Norton, New York, 1988.
16. Gebhard, P. ALMA:A Layered Model of Affect. In *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems* (Utrecht, the Netherlands), 2005.
17. Gratch, J. and Marsella, S. A domain-independent framework for modeling emotion.*Journal of Cognitive Systems Research 5*, 4 (Dec. 2004), 269–306.
18. Ito, J.Y. and Marsella, S.C. Contextually based utility:An appraisal-based approach at modeling framing and decisions.In *Proceedings of the 25th AAAI Conference on Artificial Intelligence* (San Francisco, Aug. 7–11). AAAI Press, 2011, 1442–1448.
19. Ito, J.Y., Pynadath, D.V., and Marsella, S.C. Modeling self-deception within a decision-theoretic framework. *Autonomous Agents and Multi-Agent Systems 20*, 1 (May 2010), 3–13.
20. Keltner, D., and Haidt, J. Social functions of emotions at four levels of analysis.*Cognition and Emotion 13*, 5 (1999), 505–521.
21. Lazarus, R.S.*Emotion and Adaptation.* Oxford University Press, New York, 1991.
22. Lester, J.C., Towns, S.G., Callaway, C.B., Voerman, J.L., and FitzGerald, P.J.Deictic and emotive communication in animated pedagogical agents.In *Embodied Conversational Agents*, J. Cassell, S. Prevost, J. Sullivan, and E. Churchill, Eds.MIT Press, Cambridge, MA, 2000, 123–154.
23. Mao, W. and Gratch, J. Evaluating a computational model of social causality and responsibility.In *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems* (Hakodate, Japan), 2006.
24. Marsella, S. and Gratch, J. EMA:A process model of appraisal dynamics.*Journal of Cognitive Systems Research 10*, 1 (Mar. 2009), 70–90.
25. Marsella, S. and Gratch, J. Modeling the interplay of plans and emotions in multi-agent simulations. In *Proceedings of the 23rd Annual Conference of the Cognitive Science Society* (Edinburgh, Scotland, 2001).
26. Marsella, S., Gratch, J., and Petta, P. Computational models of emotion.In *A Blueprint for Affective Computing:A Sourcebook and Manual*, K.R.Scherer, T. Bänziger, and E. Roesch, Eds.Oxford University Press, New York, 2010, 21–46.
27. Marsella, S., Johnson, W.L., and LaBore, C. Interactive pedagogical drama.In *Proceedings of the Fourth International Conference on Autonomous Agents* (Montréal, Canada, 2000), 301–308.
28. Mehrabian, A. and Russell, J.A.*An Approach to Environmental Psychology.*MIT Press, Cambridge, MA, 1974.
29. Panskepp, J. *Affective Neuroscience:The Foundations of Human and Animal Emotions.* Oxford University Press, New York, 1998.
30. Reisenzein, R. Exploring the strength of association between the components of emotion syndromes: The case of surprise.*Cognition & Emotion 14*, 1 (2000), 1–38.
31. Reisenzein, R., Hudlicka, E., Dastani, M., Gratch, J., Hindriks, K., Lorini, E., and Meyer, J.-J.C. Computational modeling of emotion:Toward improving the inter- and intradisciplinary exchange.*IEEE Transactions on Affective Computing 4*, 3 (July 2013), 242–245.
32. Scherer, K.R.Appraisal considered as a process of multilevel sequential checking.In *Appraisal Processes in Emotion:Theory, Methods, Research*, K.R.Scherer, A. Schorr, and T. Johnstone, Eds.Oxford University Press, New York, 2001, 92–120.
33. Scheutz, M. and Sloman, A. Affect and agent control:Experiments with simple affective states.In *Proceedings of the Second Asia-Pacific Conference on Intelligent Agent Technology* (Maebashi City, Japan). World Scientific Publishing, 2001, 200–209.
34. Schwarz, N. and Clore, G.L.Mood, misattribution, and judgments of well-being:Informative and directive functions of affective states.*Journal of Personality and Social Psychology 45*, 3 (Sept. 1983), 513–523.
35. Simon, H.A.Motivational and emotional controls of cognition.*Psychological Review 74* (Jan. 1967), 29–39.
36. Staller, A. and Petta, P. Introducing emotions into the computational study of social norms:A first evaluation. *Journal of Artificial Societies and Social Simulation 4*, 1 (Jan. 2001), 27–60.
37. Swartout, W., Gratch, J., Hill, R., Hovy, E., Marsella, S., Rickel, J., and Traum, D. Toward virtual humans.*AI Magazine 27*, 1 (2006).
38. Traum, D., Swartout, W., Marsella, S., and Gratch, J. Fight, flight, or negotiate.In *Proceedings of the Intelligent Virtual Agents Conference* (Kos, Greece). Springer, 2005.
39. Tversky, A. and Kahneman, D. The framing of decisions and the psychology of choice.*Science 211*, 4481 (Jan. 30, 1981), 453–458.
40. Zajonc, R.B.Feeling and thinking:Preferences need no inferences.*American Psychologist 35* (Feb. 1980), 151–175.

Stacy Marsella (stacymarsella@gmail.com) 是马萨诸塞州波士顿东北大学的计算机科学和心理学教授。

Jonathan Gratch (gratch@ict.usc.edu) 是加利福尼亚州洛杉矶南加州大学的计算机科学和心理学教授。

译文责任编辑：陶霖密

DOI:10.1145/2629559

HAC提出了一种新的科学来探索社会的计算学与人类学方面。

N.R.JENNINGS, L. MOREAU, D. NICHOLSON, S. RAMCHURN, S. ROBERTS, T. RODDEN, A. ROGERS

人-代理集体

计算机最初只是实验室里的科学工具，但现在已经得到了长足的发展。我们生活的世界中遍布了大量的计算机系统，它们分散在各个物理和信息环境中，越来越多得影响我们的日常行为。计算机技术影响了我们生活的方方面面，而我们与数字的关系业已发生了根本性的改变，因为计算机已经冲破了工作场所，脱离了桌面的限制。随着周围世界中内置的数字设备种类越来越多，联网的计算机、平板电脑和个人设备现已司空见惯。数据和信息按前所未有的速度和容量生成，信息源的范围越来越宽，收集信息的传感器类型种类也越来越多。随后，它被以无法预见的方式整合，整合的方式仅受限于人类的想象力。与以往相比，人类的活动和协作更加依赖于这种无处不在的信息，并与其交织在一起。

随着这些趋势继续急速扩张，下列情况会变得愈加明显：很多工作会涉及人类和计算机的交织共生。不仅如此，出现这些密切关联的伙伴关系后，促成了深远的变化。计算机系统感知和响应现实世界中我们不间断活动的能力正在改变我们的生活，正在为二十一世纪造就新的数字社会。具体而言，我们现在不是向被动的机器发出指令，而是开始与高度互联的计算组件步调一致地工作。被动的机器在实施任何工作前会一直等待，直到得到请求，而互联计算组件的行为是自治的和智能的（亦称代理[42]）。为了应对可用信息和可用服务的容量、种类和节奏，需要这种转变。

若期待个体能了解全局的潜在相关的可能性的并具备相应的人工整合能力，这显然不可行。计算机需要承担更多的工作，根据用户的喜好和约束情况主动引导用户互动。在这个过程中，必须更为关注人类和机器之间控制的平衡。在很多情况下，人类是负责人，代理主要发挥支撑作用，提供建议并提供选项。然而，在其他情况下，代理起控制作用，人类发挥支撑作用（例如，汽车的自动驻车系统和证券市场的算法交易）。不仅如此，在某项活动的过程中，这种关系可能会

» 重要见解

- **HAC 是一种新的社会 - 技术系统类型，其中人类和智能软件（代理）会构筑灵活的关系，以便实现他们个人的和集体的目标。有时候由人类领导，有时候由计算机领导，且这种关系会动态变化。**
- **这些挑战是重大的科学挑战。开发在大型、动态和不确定的环境（在这些环境中，可能会引起隐私和道德顾虑）中与人类互动并激励人类与代理协同工作的系统时，这些挑战必须得到处理。**
- **HAC 中的关键研究挑战包括：实现人类和软件之间的灵活自治，构建敏捷团队统筹和协调他们的行为。**

拼贴画由ANDRIJ BORYS ASSOCIATES/SHUTTERSTOCK提供

改变（例如，人类可能会被更紧急的要求打断，采取较不需要亲身参与的方式处理手头工作，或者，某代理可能会碰到未预见的情况，不得不请求人类协助处理它原来计划自主完成的任务）。

我们把这种新兴的系统类别称为人 - *代理的集体*（HAC），以反映人类和计算机之间的紧密伙伴关系和灵活的社会性互动。除了呈现更强的自治特性外，此类系统在本质上是开放的、社会的。这种开放性意味着参与者需要持续不断地、灵活地建立和管理各种各样的社会关系。因此，根据手头的任务，人、资源和信息的不同组合必须聚集在一起，协同一致地工作，然后再分散开来。由于开放性及存在很多特点各异的干系人，且每个干系人都有自己的资源和目标，这也就就是说，他们的参与的驱动力来源广泛——有外部（例如，金钱或税收优惠）、社会或形象方面的动机（例如，公众认可或排行榜的排名），或内部的（例如，个人对社会事业的兴趣、利他主义或爱好[2]），而不是政府命令。不仅如此，一旦获取此类激励措施后，便需要对这些干系人进行评估和奖励，且这种评估和奖励须确保他们的持续行为有益于他们参与形成的系统。[33]

在未来的 HAC 系统中，人们定期地协同地与自治软件互动和协作。这种系统的例子现正在萌发。例如，在我们旅行的过程中，互联程度日渐增强的交通管理系统彼此合作，让我们的旅行更顺利。Waze（http://www.waze.com/）等系统整合了市民和（电子）传感器信息，生成相关内容为用户提供帮助。此外，软件代理可以主动发起互动行为，根据旅行者的偏好和当前情况安排游览地点和就餐地点。然而，尽管 AI（人工智能）、HCI（人机交互）、CSCW（计算机支持协同工作）和 UbiComp（普适计算）社区正在对此问题部分开展相关研究，开发用于 HAC 的综合性和原理性理论方法仍是一项重大的研究

挑战，这点毋庸置疑。同理，设计和构建此类系统的流程以及在广袤世界里让人类接受和部署 HAC 的方法也是如此。

人 - 代理的集体有何不同?

HAC 系统展现了诸多独特的特性，使得规划和预测其行为变得异常棘手。它们的开放本质意味着控制和信息广泛地散布在大量潜在的自利人群和代理之间，而他们有着不同的目的和目标。多样化的系统元素造就了多种可用性；有些是持久的，有些是瞬间的。独立的参与者需要灵活地与人类和代理协调一致，它们会因势利导地改变自身的行为和行动，用最佳的方式达到目标。现实世界的情境意味着有不确定性、模糊性和无处不在的偏见，所以代理需要处理质量、可信度和来源各异的信息。因此，需要各种技术来提供可追踪的信息轨迹，覆盖从信息捕捉点（传感器或人类参与者）经融合和决策过程到行动点过程的整个轨迹，而且代理必须推断各种协作者的可信度，以便采取最佳的方案。最后，在很多情况下，自愿参与的参与者的集体行为必须被公众社会所接受（比如公平、效率或稳定），这点相当重要。通盘考虑后，HAC 的特性要求我们：

- 理解如何提供灵活的自治，允许代理有时候按完全自治的方式采取行动，而不需要询问人类；而有时候则需要人类更紧密的参与予以指导。
- 发现有效的方式方法，允许代理和人类组成的团体构建敏捷团队合作，随时携手完成联合任务，然后在成功完成合作行为后可以分散。
- 详细阐述激励原则，在设计参与者的奖励时，鼓励参与者采取的行动需合乎社会需求。
- 设计负责任的信息基础设施，允许对无缝结合的人类和代理的决策、传感器数据以及大众生成的内容进行真实性和准确性方面的确认和审计。

在若干研究领域中，研究人员已经着手探索这种宏观构想的多个方面。然而，其中没有任何研究领域处理这一整体，也缺乏对相关系统级挑战的探索。例如，互动的智能代理日渐成为设计和构建拥有自主干系人的系统的常用手段，其中每个干系人拥有自己的目标和资源。[9] 迄今为止，这方面的很多工作关注于由软件或硬件代理组成的系统【例如，机器人或无人自治系统（UAS）】。然而，研究人员越来越深刻地认识到，在此类系统内部，让人类参与进来作为主动的信息收集者和信息处理者，与自治的软件代理步调一致，不仅有必要，还有帮助。[11,38] 例如，研究人员展示了多个系统，其中人类收集现实世界的信息，然后把信息传给执行一些基本聚集的自治代理，再把信息发布到网上。[30] 此类方法往往被称为参与式感知 [26] 或公民感知。[15] 与此相似，研究人员还展示了另外的多个系统，其中自治代理把信息处理任务传给人类参与者，然后收集和归总结果。[40] 不过，此类范围广泛的各种工作通常会假定人类与代理之间的优先关系是固定的，且存在大量的，固定的熟练人类参与者集合，他们在自愿的基础上参与。[a] 这与 HAC 中代理在动态环境中运行的观点相左。在动态的环境中，灵活的自治会按基于情境的方式变换人 - 代理的优先关系，而其中的参与者会基于他们的偏好和其属主的属性做出个体的决策。

在 HCI 和 CSCW 的领域中，研究越来越多地转向大众，研究如何利用计算机系统处理和协调人类的工作。[41] 本质上，这一任务已经变成对指挥人们和协调人们响应的手段进行管理，让他们的行为具有意义。这种大尺度和联网的协作通常使用软件系统实现，以便协调和分析人们的工作。不仅如此，在观察和应对人类行为时，软件代理已经成为一项关键技术。[17] 在混合式主动系统 [8] 中以及开发普适的情境感知计算方法时，这一方法的流行程度也越来越高。[1] 然而，在大多数此项工作中，软件代理是一种工具，用于帮助理解和管理用户互动。用户位于前台，代理居于后台。HAC 的挑战正在从支配关系的假设转向探索用户和代理如何在共同的基础上共存的方法，以及如何考虑用户和代理才能让他们之间出现灵活关系的方法。

在 HAC 内部，人类的作用也把参与者的动机摆上了台面。大多数当前的系统有利他和慈善行为的假定，但未理会需要向潜在自利的参与者提供激励的需求，也未明确地以一致的方式对参与内容的内在不确定性进行处理（有关此类行为的样例，参见 Rahwan 等人 [28] 的论文；有关应对该问题的激励结构的设计，参见 Naroditskiy 等人 [23] 的论文）。与此相似，用于负责任的信息基础设施的现有方法已经把重点放在了利用跟踪信息源的能力增强特定系统上，比如数据库 [3] 或计算工作流 [7]。现在，新兴的工作（比如 W3CPROV 建议 [21] ）正在开始允许跨系统追踪来源，并支持追踪到需要维持数据机密性的系统。[14] 但是，没有任何工作处理因群体内部的人类以及长期在线操作而带来的各种并存的挑战。

实践中的人 - 代理集体

让我们考虑下重大自然灾害发生后的情况：若干组织都处于区域内，包括现场急救员（FR）、人道救援机构、新闻记者和本地居民。这些参与者的重要目的之一是评估情

a 亚马逊土耳其机器人 （AMT）和其他类似的系统是一种此类中的例外，它允许软件系统自动生成人工智能任务，然后向完成任务的大量人类参与者支付报酬。

况，确定未来几天和数周内重点关注的区域。为了协助他们完成这项工作，很多机构配备了 UAS（无人机），可用于空中探索，一些当地居民也安装了监控环境的传感器（例如，福岛事故发生两周后，当地居民建造和设置了 500 多个盖革计数传感器，并一直上传它们的读数；见 http://jncm.ecs.soton.ac.uk/）；另外，很多当地居民使用社交媒体平台【如 Ushahidi 或谷歌危机响应（Google Crisis Response）】记录求助请求，完成受灾地区的地图。本文附图说明了这一 HAC 的代表性系统结构，相关的视频参见 http://vimeo.com/76205207。

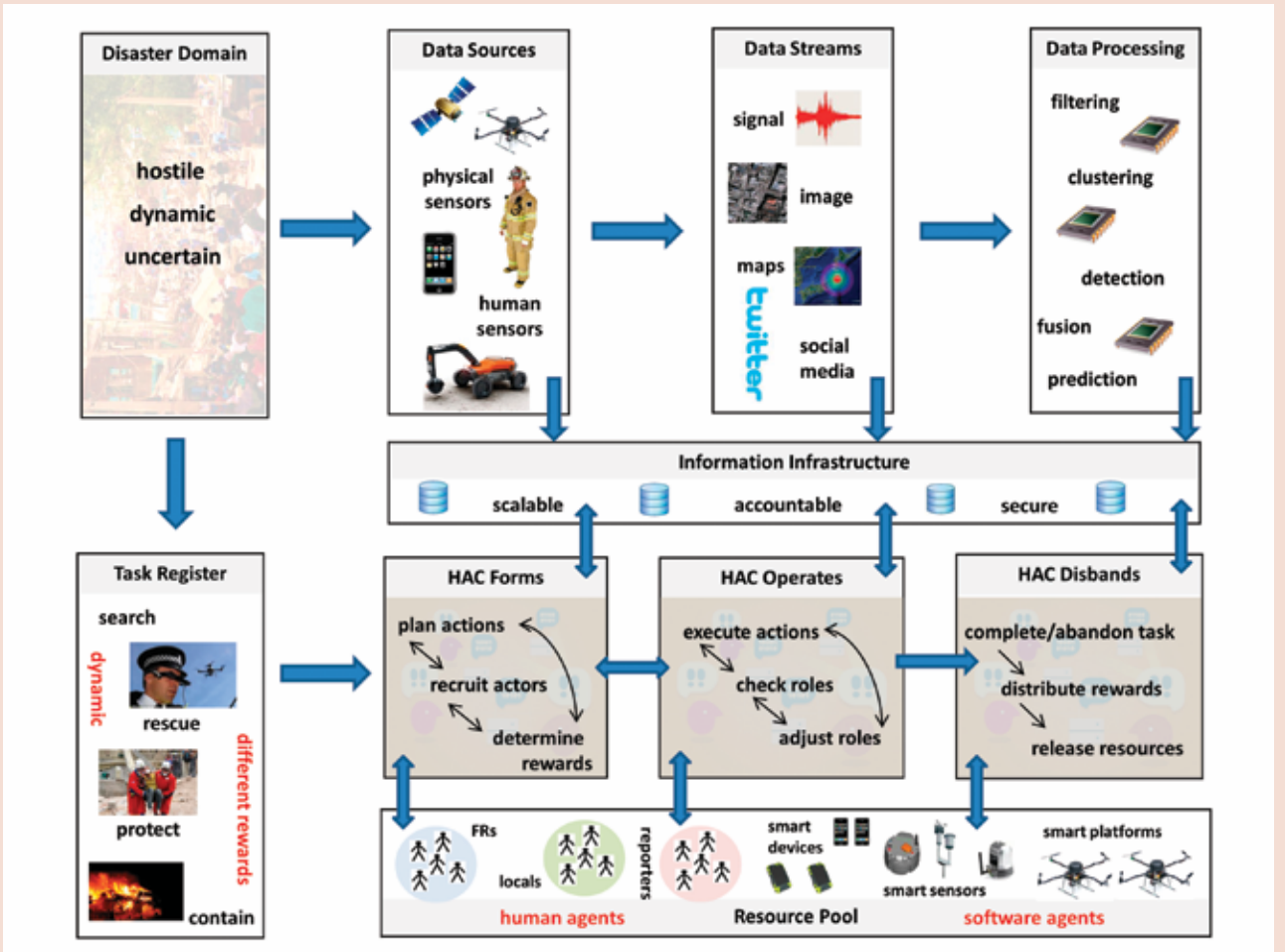
如图中所示，信息基础设施包含了从许多源头产生的，种类繁多的内容（例如，道路和关键便利设施的地图、天气预报、受影响区域内当地居民的社交媒体报告）。其中的某些信息源比其他源头提供的信息质量更高、更值得信任（例如，国际救助机构对比当地建设的环境传感器读数）。为了帮助解释和论证做出的决策，信息源应该在尽可能多的地方保存。不仅如此，需要跟踪急救者和自治代理（包括无人机（UAS））两者做出的决策，确保 HAC 中的所有成员都能对其行动负责，在后续阶段评审此类数据时，也可以更好的理解救援工作的成败情况。

在一天开始时，各种参与者（例如，现场急救员或当地的志愿者）登记他们的空闲情况和相关资源（例如，无人机、地面运输车辆或医疗用品），并说明他们希望开展的特定任务（例如，搜索学校附近的区域或确定特定地区内是否有自来水）。这些任务以后会包含他们对当前情况的评估的汇报，而且也可能受到当地居民特定求助请求的影响。

第一步，某位参与者会编制方案，以达到一项或多项任务目标[b]（换言之，HAC 形成）。该方案可能涉及组建由人、代理和资源组成的团队，共同开展各种子任务，因为很多活动可能会超出单一团队成员的能力。当他们联合以后，各急救员可能会接受本来的方案，并准

b 最初的计划可能由人类提出，也可能由软件代理提出。不仅如此，多个参与者可能会尝试同时构建方案，而其中一些可能不会取得成果。

灾害应急的HAC系统



备好开始实施方案。然而，他们也可能希望做一些小调整（例如，在前往选定区域的道路上放置路标，以最大化的利用所获得的信息的价值，或求助志愿者完成方案中的某些部分）。一些人甚至可能希望做出更大的调整（例如，说明某个开始未包含的特定子任务比建议的任务之一更重要，或者需要相当多的额外资源才能促成方案成功）。这种方案迭代的过程会反复多次，直到达成一致，在其中的多个循环中，人会亲自参与，而代理则根据急救员的偏好重新编制方案。

由于问题和环境的本质，HAC的计划执行（操作）阶段可能不会顺利展开。更高优先级的新任务可能出现，方案中的任务可能会变得不必要，可能可以得到新的参与者和资源，或者承诺的资源可能无法兑现（例如，由于现场急救员筋疲力尽，或是无人机能量耗尽）。所有这些都涉及对代理和人进行持续不断的监控以及重新制定方案，也潜在地涉及解散现有团队，然后组建能力组合不同的、更适于特定类型的救援行动的新团队。不仅如此，在方案实施的过程中，人与代理之间的自治关系也可能会变化。例如，无人机团队最初得到的指示可能是：以完全自治的方式收集特定区域的图像，在整个任务完成前不打扰现场急救员。然而，在执行该任务的过程中，无人机可能会碰到困难，需要人类协助完成复杂的行动（例如，在狭小的空间内机动，获取建筑物的特定视图），或者UAS可能发现了重要的东西，它们认为值得打断救援人员，或值得请求在线的操作人员帮助分析收集的图像。如果发生了无法预见的事件，工作的复杂度增大，需要涉及数百名急救员、志愿者和无人机，那么应变协调员可能更为依赖自治代理来计算方案，为个人和无人机安排支援，部署行动。不仅如此，如果协调员发现代理不足以判断人的能力（例如，由于疲劳人类的工作表现降低了）或任务的性质（例如，复杂的挖掘操作或监控）时，协调员可选择介入，改变方案或输入更多的信息。

> **人员、资源和信息的不同组合必须聚在一起，协同一致地工作，然后解散。**

过了数天后，急救员可能会注意到，当地居民支持救援行动的意愿降低，而且一些人只开展自己区域内的任务相关工作。这意味着某些区域得到的关注度会不够。为了弥补这一缺口，可以安排一些额外的激励方案。首先，如果志愿者发动朋友和家庭成员协助实施救援工作，可以对其进行奖励，这种奖励可以是经济上的（例如，按工作的小时数支付钱款或燃料票），或不是经济上的（例如，承诺更快地归还他们的生活用品）。其次，对于完成了所在当地区域以外的任务以及高优先级任务的个体，可以给予额外的补助，比如社区服务奖或双倍工资。最后，为了鼓励人们准确地报告任务的重要程度和紧急程度，可以引入一种激励机制，增加所做评估与职业救援者的评估相符的个人的信誉，而且也可以向那些招募到顶级任务执行者的招聘人员颁发奖金。

研究的关键挑战

除了其他的事务外，在如何控制用户和代理（灵活自治）的平衡方面，动态组建和解散集体方面（敏捷组队），激励参与者（激励工程）方面以及如何提供支撑这些工作的信息基础设施方面，HAC带来了各种研究挑战。虽然上述领域都不是全新领域，但是HAC系统的情境引入了其他的复杂性，让很多新的元素崭露头角。

灵活的自治。HAC引发了与人和数字系统（这些系统呈现了某种形式的自治）间关系相关联的基础问题。具体来说，人们不能再把计算机系统当成完全的附庸，HAC的

出现突显了人们对这方面关注程度的不断增加。我们通常会毫无疑问地遵循导航系统的指示，或遵循通过电话向我们提供的，由计算机生成的指示。随着此类自治系统越来越多地指导我们，各种新形式的关系正在诞生。这种转变不仅带来了设计与这些系统互动方式有关的问题，还让人们关注了更大范围的社会和道德问题，即责任和问责。

HAC 本质上是社会和技术相融合的系统。用户和自治软件系统之间的关系将受下列两方面同等推动：专注于用户的问题（比如责任、信任和社会接受程度）以及技术问题（比如规划或协调算法）。因此，我们需要揭示灵活自治的互动原则，因为这种自治塑造了此类系统。此处的关键问题是如何实现参与集合的代理和用户之间的控制力的平衡。具体而言，何时应由用户控制，何时应由软件系统接管？

鉴于这种情况，关键的挑战之一是如何确保 HAC 内部的控制具有积极的意义。该问题的核心是日常活动中内在的社会问责和责任的意义。我们通常会有专业的和个人的行为，通过这些行为影响别人，我们也必须对其负责。事实上，也可以争辩说，这塑造了许多范围更广的社会关系和理解。但是，当我们把我们的世界分享给计算单元，使得它们对环境的控制程度与我们一样时，我们会有什么感觉呢？这种关系是轻松的，亦或是紧张的？我们又怎么管理这种关系呢？当控制力的平衡在 HAC 中的软件代理和人之间切换时，需要什么样的制约和平衡才能使它们结成成熟的、富有成效的关系？

处理这些问题时，需要我们思索软件代理可能会采取何种方式为用户工作。现在，此类软件往往“躲在幕后”运行，用户的可见度有限。他们可能会代表我们提出建议，或安排活动并把工作结果呈现给用户。然而，几乎没有任何信息传达了造成这种结果的原理。与此相反，HAC 中的计算代理需要透露自己的行为和原理，这样他们才能承担社会责任。

向用户揭示软件代理的作用和行为后，将会突显一大堆问题，并需要我们考虑更广的社会和道德问题，可能还会促使人们反思这些系统运行所处的法律和政策框架。例如，鉴于工作的集体性质，谁应该为特定的结果最终负责，这又会对这种方法的应用产生何种影响。确定这些相当重要。用更通俗的话说，人们可能会给予其他训练有素的资深专业人士信任和自主权，但对于这些集合中的软件代理，人们会给多大的信任和自主权呢？作为集体的一员，如果软件代理在学习如何执行工作的过程中犯错，这可以接受吗？

此处的关键问题是，在 HAC 中，如何在多层尺度和集合层面上表征人类和代理的工作。在提升社会责任感的同时，还需要具备识别和理解他人活动以及灵活应对这些行为的能力，以支持合作并协调各方行为，使之成为更广泛的社会工作的一部分。因此，提供各种机制让用户了解他人的行为是很多合作系统的核心设计要素。具体而言，有下列几个中心问题：需要什么机制支持我们据此处理用户以及类似的自治软件代理？我们如何感知人类行为，识别用户参与的各种活动？以及如何把这些传递给软件代理？向用户呈现代理行为和进展的最佳技术是什么？

敏捷组队。在 HAC 中，人类和代理会组成短期的团队。在团队解散前，他们会协调他们的行为来完成系统中出现的各种单独的和联合的目标。由于新的目标、机遇和参与者接踵而来，并且这会是个持续不断的过程。迄今为止，在多代理系统社区的研究中，已经产生了大量的算法用于组建和协调团队。具体来说，研究人员在联盟形成和分布式协调领域发现了这些算法。[27,32] 然而，其中很多方法只关注了软件代理之间的互动，而不考虑敏捷组队的时间方面。[36]

在 HAC 的场景中，这些假设受到了挑战。集中控制基本上无法用于大规模动态 HAC。不仅如此，开发的方法不仅必须考虑最优联盟的形式，还要在各自具有自己有限的通讯和计算资源的情况下，考虑人类个体与代理之间如何互相商议组成联盟，且不需要具备系统中所有其他参与者的功用和约束的显性知识。

处理这种分散化问题可能会涉及局部消息传递方法。这些方法利用了概率推理、图模型和博弈论领域的理论和方法。通过利用代理之间的典型稀疏交互（换言之，每个代理彼此之间不一定存在直接的互动），这些方法允许使用图的方法，有效地展现协调和联盟形成问题。然而，迄今为止，这些方法仅处理了成员几十名的团队，而 HAC 的规模将扩大到几百，甚至几千名成员。与此类似，在开发现有的方法时，明确假设协作中所有的参与者均拥有相似的计算和通讯资源，在大多数 HAC 内部，这种假设几乎必然是错的。这些方法的规模扩大时会面临挑战。应对此类挑战时，它们须能够处理大量具有不均等计算和通讯资源的参与者，这可能要求做出原理性近似（此领域中具有发展潜力的成果见 Rahwan 等人的论文 [29]，其使用了论证充分的网络流优化算法来处理大规模联盟形成问题）。

不仅如此，之前的组队和参与者协调方法通常假定，所有人均能获得与系统的公共工具和约束有关的完整、准确的知识。虽然在迄今所研究的小规模系统中，此类假设可能是有效的，但是它们却无法应

用于运行于动态环境中的、更大规模的 HAC，其中参与者的感知和通讯能力未知。在这一领域中，之前的成果已经通过马尔可夫决策过程（MDP）和部分可观测马尔可夫决策过程（MDP）的框架处理了这种不确定性。现在，虽然这些方法中隐含的贝叶斯框架论证充分，原理性强，但是现在它的规模还是不够大，不能在那些必须确保决策时间的大型系统中使用。与之前的情况相同，这需要设计新的计算和近似方法。

最重要的是，HAC 形成和运行的新方法必须处理系统中人类的需求。用户必须与软件代理商议，就他们集体形成的短期联盟结构达成一致，且应协调他们在所形成的联盟中的各种活动。相比主要聚焦于计算实体而基本不考虑用户组队方式，也不考虑用户与群组（如团队）关系的机制和技术，这是重要的革新。因此，计算学方面的探索必须通过聚焦于集体内人类参与者的说服力和参与度而达到平衡。例如，由于人类天性喜欢稳定和信任，我们如何理解和管理在解散和重组团队的需要中存在的冲突？用户对同时在多个团队工作的可能性有何看法？接收软件代理发出的指令时，他们的感觉如何？在混合现实游戏 [20] 和社会机器人 [6] 的场景中，研究人员已经开始探讨这些问题。然而，为了全面地了解这些动态因素如何影响对底层的组队和协调算法的要求，我们需要在真实的场景中构建、部署和评估原型 HAC。

激励工程。什么会促成 HAC 的形成，什么又会激励他们一起有效地工作？我们如何让多个参与者（单独或群体）的激励措施目标一致，朝向系统设计者的目标生成特定的结果？当参与者的行为受到个人和潜在冲突的动机引导时，上述两项工作都相当艰巨。[4] 现在，虽然人们承认，此类参与者可能受到很多不同类型的激励措施的影响，但迄今为止，大多数研究利用了微观经济学，聚焦于金钱方面的激烈措施，并假设参与者的效用函数定义清晰且是线性的。不仅如此，人们通常认为参与者是理智的，因为他们会进行复杂的计算，以推断自己在形式中的最佳行为。不幸的是，这些假设通常会引发集中化的激励机制。在 HAC 等开放系统的场景下，这种机制不稳定。

HAC 要求我们重新考虑很多假设，而这些假设却是激励工程的最新方法中的核心假设。他们会涉及有限理性的参与者 [10]，且这些参与者的行为有时无法控制。例如，在灾害应急的场景中，（责任心不同的）当地志愿者可能会（通过他们的社交网络）受到他们的家庭成员和朋友的影响，需要调整他们与不同机构（能力不同）的紧急救护人员一起工作。不仅如此，人和软件代理有时可能也不乐意接受金钱的报酬，社会或内在的激励措施的反响可能更好。例如，在 DARPA Red Balloon Challenge（国防高等研究计划署红气球挑战赛）中，获胜团队的激励机制成功了，因为它把个人找到气球的经济动机与从他们的社交网络及其他方面招募成员的动机保持一致。[28] 与此相反，在 MyHeartMap Challenge（http://www.med.upenn.edu/myheartmap/）中，基本相同的方案却几乎没有吸引到志愿者，因为那些人主要受到愿意拯救心脏病发作病人的利他动机的影响。最近，Scekic 等人 [33] 调查和归类了社会化计算平台中使用的激励机制的此类早期例子。他们注意到，绝大多数当前系统在工作人员之间举行了简单的竞赛，并通过主观评价得出“获胜者”。然后，根据该获胜者完成的工作，他通常会得到经济方面的奖赏。相比之下，很少有系统使用了非货币激励。使用非货币激励的系统通常聚焦于声望问题。例如，他们举出了 avvo.com，该网站吸引了美国大量的律师向访问网站的人提供免费回复和建议，并根据他们提供的回复的质量和及时性生成这些律师的声望等级。现在，网上排名明显会影响客户受吸引并寻求他们私人服务的几率，但是这种效果是间接的，而且与直接支付货币相比，这容纳了更大范围的个人动机。然而，迄今为止，比较而言，几乎没有研究尝试从形式上定义这种类型的激励机制。这是个重大的遗漏，因为我们相信，把行为经济学的理论方法和行为变化的“助推”方法 [10,38] 带入博弈论提供的机制设计的形式化描述中后，会得到一个富有潜力的出发点，可用于策划 HAC 需要的激励类型。

如今，即便可以确定正确的激励措施，系统中的参与者也有可能达不到最佳的绩效，这或是因为他们的能力有限，或是因为他们本质上错综复杂。例如，在 AMT（亚马逊土耳其机器人）等系统中，成千上万名工作人员会尝试完成报酬几美分的微任务，而战略型的工作人员则尝试用最小的努力尽可能多地完成任务。与此类似，在宇宙动物园（Zooniverse）等公民科学项目中，业余科学家通常会选择他们最享受的任务，而不是项目最需要的任务。[34] 不仅如此，在长期互动中，人类参与者可能会受到疲劳的困扰，他们的表现可能会随时间退步。然而，如果未加思索地排除所有不能执行某些任务的参与者，可能也意味着错失了他们正确执行其他任务的能力。因此，关键是设计各种机制，用以确保为执行所派任务的参与者而设置的激励措施与他们的能力和责任心相符。举例而言，在这种背景下，众包和公民科学中最初的工作已经说明了如何设定微任务所需支付的价格，或者应该为每位参与者分配多少特定类型的任

务来激励他们表现良好。[39] 在激励人类参与者花费数小时从事通常被视为无聊的任务方面，游戏化方法也已经取得了成功。[41] 然而，需要投入更多的工作来让这些方法变得更通用，并证明他们在不同应用领域的效用。

虽然这些挑战与激励措施的选择和呈现方式有关，但是计算 HAC 场景中的这些激励措施也是一项重大的挑战。实际上，HAC 涉及大量参与者，这一事实意味着需要设计计算效率高的算法来枚举和优化激励措施的组合（为 HAC 内部的群体、联盟或个人设置的激励措施）。当此类 HAC 的行动展开相当长的时间后，可能会涉及很多反复的互动和协商，获取拟设置的激励措施的详单会是一种更大的计算挑战。在大多数情况下，无法达到最优，因此目标应该是找出近似解。有一些相关的例子，比如激励大量的电动车所有人安排不同的时间给车充电，以避免本地变压器过载 [31] 以及为大团队中各成员粗略算出公平奖励的算法。[19]

负责任的信息基础设施。HAC 将会大大影响我们对支撑 HAC 的数字基础设施的认识方式。具体来说，我们需要考虑如何才能共享数据基础。信息的源头至关重要。此处，来源描述了数据的信息来源（什么）、对其负责的人或代理（谁）以及获得信息的方法（怎样）。反过来，基础设施处理信息源头来评估信息质量，允许用户理解和审计 HAC 的历史行为，并帮助人类确定 HAC 的决策是否可以信赖。

HAC 运行的方式要求我们重新考虑某些关于源头工作方面的流行假设。通常情况下，人们认为源头是细粒度的、确定的，并且准确和完整地描述了执行情况。[21] 在 HAC 中，这一假设无效，因为人类活动不仅很难捕捉，而且也不可靠。不仅如此，异步通信可能会让源头变得不完整。最后，源头的细粒度本质让它很难被人类理解。处理这些挑战十分重要，且需要各种类型的技术。例如，基于源头构建的概率模型或许能帮助人们捕捉与所发生事件相关的不确定性，而抽象技术或许能允许归总常见的模式，继而让大图变得更容易管理。在这些前景广阔的方向中，需要探索此类源头描述的意义以及它们支持的推理类型。给定 HAC 在代理和人类方面以及执行时间方面的潜在规模后，推理算法的可扩展性也成了一个重要的问题，需要进一步的研究。

“关键的挑战之一是如何确保HAC内部的控制具有积极的意义。该问题的核心是日常活动中内在的社会问责和责任的意义。”

负责任的信息基础设施的愿景是帮助人和代理理解所作出的决策并确定它们是否可以信任。事实上，源头能帮助产生信任，评估质量，这已是默认的常识，但是目前尚无原理性的方法可直接应用于 HAC。在这种背景下，从源头学习的能力就相当重要，因为它有潜力把源头变成丰富的信息来源，用以建立可信度并引导 HAC 中的决策过程。具体来说，假定源头信息通常以图的形式呈现，一些为图开发的通用方法或许是可定制的，并且有可能会被有效地执行。网络指标就是此类解决方案的一个例子，它用便捷和简凑的方式汇总了复杂的场景和行为。具体而言，网络指标可以被具体化成源图，以与应用无关的方式帮助刻画 HAC 过去的行为。[5] 然后，通过在面向源的网络指标上面应用机器学习技术，我们可以标记图和节点，获取有关代理或数据质量的可信度。

迄今为止，现存的基础设施机制倾向于体现一种“中间件”的观点，对数据模型进行形式化，开发算法并使用应用构思各部分（如来源）的集成。然而，HAC 需要理解和响应人类的行为，但人类行为的捕捉、处理和管理方式会引起重大的道德和隐私顾虑。通常，这些顾虑的核心是人类与被收集的人类数

据的隔离方式。具体而言，在当前的基础设施中，人们通常不知道他们流出的数字信息，这些信息的处理方式以及从这些数据获取的分析推理所产生的影响。因此，在管理与基础设施的关系时，人们在道德上处于劣势，因为他们在很大程度上不知道其行为的数字结果，也缺乏有效的控制或撤销手段。HAC 基础设施需要对人类负责，允许他们与其数据发展更丰富、双向性更强的关系。

开发负责任的基础设施还响应了隐私研究人员（如 Nissenbaum[24]）的要求。他们要求理解和支持用户及其数据之间的关系。事实上，她的语境完整性（Contextual Integrity）理论把隐私设想为不同社会代理之间的辩证过程。基于这一点，其他人提出，在服务的设计中需要内置双向的关系，以便人们认可它们在本质上是社会性的。[35] 这表明，用户应该能够明确地了解并控制向他人进行的数据披露 [25] 以及软件代理对这种数据的使用。建立这种双向关系还要求我们重新构建治理和管理人类数据的现有方法。

这方面最关键的问题或许与在信息系统中谋求使用个人数据的权限有关。当前的方法使用了传统的模型，只询问用户一次他是否同意各种条款，而这些条款往往相当复杂。在生物伦理学领域，这一传统模型已经受到了质疑，Manson 和 O'Neill[18] 争辩说，需要从比当前合约概念更广的角度思考同意。我们建议，与此类似，HAC 也需要重新考虑同意的设计原则，并重新处理代理与用户之间的平衡。[16] C

参考资料

1. Abowd, G.D., Ebling, M., Hung, G., Lei, H., and Gellersen, H.W.Context-aware computing.*IEEE Pervasive Computing 1*, 3 (2002) 22–23.
2. Ariely, D., Bracha, A. and Meier, S. Doing good or doing well?Image motivation and monetary incentives in behaving prosocially.*American Economic Review 99*, 1 (2007), 544–55.
3. Buneman, P., Cheney, J. and Vansummeren, S. On the expressiveness of implicit provenance in query and update languages.*ACM Trans.Database Systems 33*, 4 (2008), 1–47.
4. Dash, R.K., Parkes, D.C. and Jennings, N.R.Computational mechanism design:A call to arms. *IEEE Intelligent Systems 18*, 6 (2003) 40–47.
5. Ebden, M., Huynh, T.D., Moreau, L., Ramchurn, S.D. and Roberts, S.J.Network analysis on provenance graphs from a crowdsourcing application.In *Proc.4th Int.Conf. on Provenance and Annotation of Data and Processes*. (Santa Barbara, CA, 2012), 168–182.
6. Fong, T., Nourbaksha, I. and Dautenhahn, K. A survey of socially interactive robots.*Robots and Autonomous Systems 42* (2003), 143–166.
7. Gil, Y., Deelman, E., Ellisman, M., Fahringer, T., Fox, G., Gannon, D., Goble, C., Livny, M., Moreau, L. and Myers, J. Examining the challenges of scientific workflows.*IEEE Computer 40*, 12 (2007), 26–34.
8. Horvitz, E. Principles of mixed-initiative user interfaces. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems* (New York, NY, 1999), 159–166.
9. Jennings, N.R.An agent-based approach for building complex software systems.*Commun.ACM 44*, 4 (Apr. 2001) 35–41.
10. Kahneman, D. Maps of bounded rationality:Psychology for behavioral economics.*American Economic Review 93*, 5 (2003) 1449–1475.
11. Kamar, E., Gal, Y. and Grosz, B. Modeling information exchange opportunities for effective human-computer teamwork.*Artificial Intelligence Journal 195*, 1 (2013) 528–555.
14. Kifor, T. et al.Provenance in agent-mediated healthcare systems.*IEEE Intelligent Systems 21*, 6, (2006) 38–46.
15. Krause, A., Horvitz, E., Kansal, A. and Zhao, F. Toward community sensing.In *Proc.Int.Conf. on Information Processing in Sensor Networks* (St Louis, MO, 2008), 481–492.
16. Luger, E. and Rodden, T. An informed view on consent for UbiComp.In *Proc.Int.Joint Conf. on Pervasive and Ubiquitous Computing* (2013), 529–538.
17. Maes, P. Agents that reduce work and information overload.*Commun.ACM 37*, 7 (1994), 31–40.
18. Manson, N.C. and O'Neill, O. Rethinking informed consent in bioethics.*CUP*, 2007.
19. Michalak , T., Aaditha, K.V., Szczepanski, P., Ravindran, B. and Jennings, N.R.Efficient computation of the Shapley value for game-theoretic network centrality.*J. AI Research 46* (2013), 607–650.
20. Moran, S., Pantidi, N., Bachour, K., Fischer, J.E., Flintham, M. and Rodden, T. Team reactions to voiced agent instructions in a pervasive game.In *Proc.Int. Conf. on Intelligent User Interfaces*, (Santa Monica, CA, 2013), 371–382.
21. Moreau, L. The foundations for provenance on the Web. *Foundations and Trends in Web Science 2*, 2–3 (2010) 99–241.
22. Moreau, L. et al.Prov-dm:The prov data model.W3C Recommendation REC-prov-dm-20130430, World Wide Web Consortium, 2013.
23. Naroditskiy, V., Rahwan, I., Cebrian, M. and Jennings, N.R.Verification in referral-based crowdsourcing.*PLoS ONE 7*, 10 (2012) e45924.
24. Nissenbaum, H. Privacy as contextual integrity. *Washington Law Review 79*, 1 (2004), 119–158.
25. Palen, L. and Dourish, P. Unpacking privacy for a networked world.In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2003), 129–136.
26. Paxton, M. and Benford, S. Experiences of participatory sensing in the wild.In *Proc.11th Int.Conf. on Ubiquitous Computing*, (Orlando, FL, 2009), 265–274.
27. Rahwan, T., Ramchurn, S.D., Jennings, N.R. and Giovannucci, A. An anytime algorithm for optimal coalition structure generation.*J. Artificial Intelligence Research 34* (2009), 521–567.
28. Rahwan, I. et al.Global manhunt pushes the limits of social mobilization.*IEEE Computer 46*, 4 (2013a) 68–75.
29. Rahwan, T., Nguyen, T-D, Michalak, T., Polukarov, M., Croitoru, M., Jennings, N.R.Coalitional games via network flows.In *Proc.23rd Int.Joint Conf. on Artificial Intelligence*, (Beijing, China, 2013b), 324–331.
30. Reddy, S., Parker, A., Hyman, J., Burke, J., Estrin, D. and Hansen, M. Image browsing, processing, and clustering for participatory sensing.In *Proc.4th Workshop on Embedded Networked Sensors*, (Cork, Ireland, 2007), 13–17.
31. Robu, V., Gerding, E.H., Stein, S., Parkes, D.C., Rogers, A. and Jennings, N.R.An online mechanism for multi-unit demand and its application to plug-in hybrid electric vehicle charging.*J. Artificial Intelligence Research* (2013).
32. Rogers, A., Farinelli, A., Stranders, R. and Jennings, N.R.Bounded approximate decentralised coordination via the max-sum algorithm.*Artificial Intelligence 175*, 2 (2011), 730–759.
33. Scekic, O., Truong, H.-L. and Dustdar, S. Incentives and rewarding in social computing.*Commun.ACM 56*, 6 (2013), 72–82.
34. Simpson, E., Roberts, S.J., Smith, A. and Lintott, C. Bayesian combination of multiple, imperfect classifiers. In *Proc.25th Conf. on Neural Information Processing Systems*, (Granada, Spain, 2011).
35. Steeves, V. Reclaiming the social value of privacy. *Lessons from the Identity Trail*.I. Kerr, V. Steeves and C. Lucock, eds.Oxford University Press, 2009, 193–208.
36. Stone, P., Kaminka, G.A., Kraus, S. and Rosenschein, J.S.Ad hoc autonomous agent teams:Collaboration without pre-coordination.In *Proc.24th Conference on Artificial Intelligence* (2010).
37. Tambe, M. et al.Conflicts in teamwork:Hybrids to the rescue" In *Proc.4th Int.Joint Conf. on Autonomous Agents and Multiagent Systems*, (Utrecht, The Netherlands, 2005), 3-10.
38. Thaler, R.H. and Cass, R.S.*Nudge:Improving Decisions About Health, Wealth, and Happiness*.Yale University Press, 2008.
39. Tran-Thanh, L., Venanzi, M., Rogers, A. and Jennings, N.R.Efficient budget allocation with accuracy guarantees for crowdsourcing classification tasks.In *Proc.12th Int.Conf. on Autonomous Agents and Multi-Agent Systems* (St. Paul, MN, 2013), 901–908.
40. von Ahn, L. et al. recaptcha:Human-based character recognition via web security measures.*Science 321*, 5895 (2008), 1465–1468.
41. von Ahn, L. and Dabbish, L. Labeling images with a computer game.In *Proc.SIGCHI Conf. on Human Factors in Computing Systems*, (Vienna, Austria, 2004), 319–326.
42. Wooldridge, M.J. and Jennings, N.R.Intelligent agents:Theory and practice.*Knowledge Engineering Review 10*, 2 (1995) 115–152.

N.R.Jennings （nrj@ecs.soton.ac.uk）是英国南安普顿大学电子与计算机科学学院计算机科学钦定教授（Regius Professor）和英国政府首席科学顾问。

L.Moreau （lavm@ecs.soton.ac.uk）是英国南安普顿大学电子与计算机科学学院计算机科学教授， 网络和互联网科学小组（WAIS）组长以及该学院的副院长（研究与企业）。

D. Nicholson （dn4@ecs.soton.ac.uk）是BAE的高级工业科学家，EPSRC资助的人-代理集体（Human Agent Collectives）项目的知识传递官。

S.Ramchurn （sdr@ecs.soton.ac.uk）是英国南安普顿大学电子与计算机科学学院代理、互动和复杂性小组（AIC）的讲师。

S.Roberts （sjrob@robots.ox.ac.uk）是英国牛津大学机器学习研究小组的领军人物。他还是萨默维尔学院的资深研究教授和牛津人研究所的教职人员。

T.Rodden （tar@cs.nott.ac.uk）是诺丁汉大学的计算学教授和混合现实实验室的联席主任。

A.Rogers （acr@ecs.soton.ac.uk）是英国南安普顿大学电子与计算机科学学院代理、互动和复杂性小组（AIC）的计算机科学教授。

译文责任编辑：田丰

DOI:10.1145/2682585

若要查阅相关论文，
请访问 doi.acm.org/10.1145/2682583

技术视角
重新思索吞吐量型处理器的缓存

作者：Stephen W. Keckler

缓存成为计算机设计的中流砥柱已有将近 50 年，它也是学术和行业研究中经久不衰的话题。缓存的目的是为了利用程序指令或程序数据中的局部性，让计算机系统能够提供拥有大而快内存的假象，而底层硬件实际上提供的是小而快（缓存）或大而慢（主存）的物理结构。当今的 CPU 系统具有多达四种级别的缓存，涵盖片上 SRAM 到片外嵌入式 DRAM 等范围。

CPU 系统通常将缓存用作掩盖内存延迟的一种方式。若无缓存，单线程程序需要将大量时间花在等待数据从片外 DRAM 内存返回。然而，向量处理器和 GPU 等以吞吐量为导向的计算系统能够使用并行性来容忍内存延迟，降低了对缓存的延迟缩减效应的需求。尤其是 GPU，它使用大量的多线程处理来容忍延迟；当一个线程执行访问主内存的加载指令时，其他线程可以同时执行，让处理器保持繁忙状态。以吞吐量为导向的系统对内存延迟不敏感，而倾向于对内存带宽敏感。因此，它们的内存层次结构通常设计为运用缓存来减少对 DRAM 带宽的需求，而不是缩减延迟性。

这些对缓存的不同追求导致了现代 GPU 和 CPU 系统之间的不同权衡取舍。例如，当代 NVIDIA Kepler GPU 上 15 个流式多处理器中每个都有高达 2,000 个线程，它们共享 256KB 寄存器文件和总共 96KB 的 1 级数据缓存。15 个流式多处理器共享 1.5MB 的片上 2 级缓存。因此，在执行最大数量的线程时，每个线程可以专享访问 128 字节寄存器文件，平均每个线程可以用到 48 字节的 1 级缓存和 50 字节的 2 级缓存。相比之下，12 核 IBM Power8 处理器的缓存容量要多三个数量级，每个线程的 L1 和 L2 缓存分别有 8KB 和 64KB。GPU 缓存专注于保存在许多线程之间共享的数据，而 CPU 缓存则寻求在利用一个线程内的时间局部性。

不过，随着 GPU 成为主流的并行处理引擎，许多面向 GPU 的应用现在拥有更加适合传统缓存考验的数据局部性。随之而来的挑战是设计的缓存系统要能够同时利用空间和时间局部性，而不必将每线程的缓存容量扩展几个数量级，以达到和 CPU 体系结构相匹配的水平。

在过去，数据局部性优化的重点完全放在缓存上，在分配策略、回收策略和基本缓存组织上费尽心思。在下面这篇论文中，Rogers 等人的研究采取了不同的方法，通过注重多线程处理器的线程调度来改善缓存行为并提高计算机系统性能。基本理念非常简单：当线程拥有数据局部性时，增加它的调度频次以提高其数据保留在缓存中的可能性。这种方法具有减少一个时间窗口中执行的线程数量的效果，可以提高吞吐量和总体性能（这或许在一定程度上违反直觉）。因此，在某些情形中“少即是多”。

在下面这篇论文中，Rogers 等人的研究采取了不同的方法，通过注重多线程处理器的线程调度来改善缓存行为并提高计算机系统性能。

论文中所述的体系结构具有诸多优点。首先，它不需要程序员的输入，也不需要提示工作集合中保留多少线程数。其次，它自动适应程序的局部性行为，最终在线程级数据局部性不足时返回到基准的“最多线程数”方法。除了研究中所呈现的性能益处外，这种改进局部性的方法也降低了带宽需求，以及跨芯片数据传输所耗的功率。

本论文展示了当代计算机体系结构研究的两个方面。其一，重新思索了当新的体系结构模型、技术或应用领域涌现时所需的传统智慧。其二，本论文展示了在传统计算机体系结构界限之间进行联合优化时可以利用的机会。尤其而言，线程调度不仅有改善 L1 缓存的潜能（本论文中所述），而且也能改进二级缓存、DRAM 和互连体系结构。

Stephen W. Keckler 是 NVIDIA 公司体系结构研究资深总监，也在德克萨斯大学（奥斯汀）计算机科学系担任兼职教授。

译文责任编辑：陈文光

DOI:10.1145/2682583

了解你的极限：通过调度管理大规模多线程缓存

作者：Timothy G. Rogers、Mike O'Connor 及 Tor M. Aamodt

摘要

处理器和内存性能之间的巨大差距已经成为过去三十年中微处理器研究和开发的焦点。现代体系结构利用两种正交的方法来帮助缓减这一问题：(1) 几乎每一个微处理器都包含某种形式的片上存储，通常是以缓存的形式，来缩减内存延迟并提高有限内存带宽的使用效率。(2) 图形处理单元 (GPU) 等大规模多线程体系结构试图通过直接在硬件内快速切换许多线程来隐藏内存的高延迟。本文探索的是这两种方法的交汇部分。我们研究了如何在大规模多线程 GPU 上加速具有显著局部性的大规模并行工作负载。我们发现，片上缓存所见的内存访问流是硬件线程调度器所做决定的直接结果。我们的研究提出了一种硬件调度技巧，它可以对来自内存系统的反馈进行反应，从而创造对缓存更加友好的访问流。我们通过模拟评估我们的技巧，结果表明，性能相对于以前提出的调度机制有了明显提升。我们将使用我们的调度器和 LRU 替换策略时的缓存命中率与使用最优缓存替换策略的其他调度技巧相比较，从而展示了作为缓存管理技巧进行调度的效果。

1. 引言

你是否有过这种状况，想要努力完成许多事但一事无成？这是一种称为不可分资源理论的典型心理学原则，即人类投入到同时进行的任务的注意力在容量上受到限制。[16] 试图在同时进行的过多任务之间分散注意力可能会对你的表现产生不利影响。我们在高度多线程硬件环境中探索了类似的问题，并将人类多任务处理作为类比，用于帮助解释我们的发现。本文所研究的体系结构能够以时钟周期为单位高效切换数十项任务。我们的论文提出了以下问题：虽然大规模多线程处理器*能够*在这些任务之间快速切换，但它们*应当* 在何时切换呢？

在过去 30 年中，大量的研究和开发投入到了更加高效地利用片上缓存上。在硬件级别上，通常通过改进层次结构、[3] 替换 / 插入策略、[14] 一致性协议 [19] 或以上所述的某种组合来优化缓存管理。过去对硬件缓存的研究假设：内存系统所见的访问流是固定的。然而，大规模多线程系统为这一问题带来了另一个维度。每个时钟周期内，硬件线程调度器必须选择接下来要发射处理器核心的哪一个活动线程。这一决定对一级缓存所见的访问数据流有着显著的影响。在大规模多线程系统中，每一周期通常都有许多已做好调度准备的线程。本论文利用这一发现，并使用线程调度器来显式管理内存系统所见的访问流，从而最大化吞吐量。

本研究的主要贡献是*缓存感知型波面调度*（Cache-Conscious Wavefront Scheduling，CCWS）系统，它利用来自内存系统的局部性信息*重整* 通过硬件线程调度进行的后续内存访问。与优化缓存替换和插入策略的传统做法相似，CCWS 也尝试预测将被重新利用的缓存行。然而，缓存路径管理策略的决定是在一小组数据块之间做出的。线程调度器从比缓存相联性大得多的潜在内存访问池中高效地选择要插入到缓存中的数据块。与缓存替换策略有效*预测*各行的重新引用间隔的方式类似，[14] 我们提出的调度器尝试*更改* 重新引用间隔，以减少对高局部性数据的重复访问之间的干扰引用数。程序员或编译器 [25] 在软件中使用缓存排列技术通过重组代码来减小内层循环的缓存足迹，与之类似，CCWS 主要通过在硬件中动态限制共享缓存的线程数量来聚合缓存足迹。

本论文的最初版本以“缓存感知型波面调度”为题，发表于 2012 年《*第 45 届 IEEE/ACM 微体系结构国际研讨会论文集*》。本论文的另一版本也以“大规模多线程处理器的缓存感知型线程调度”为题，发表于 2013 年的《*IEEE Micro 特刊：2012 计算机体系结构大会 Micro 精选*》。

进行本研究时，Mike O'Connor 在 Advanced Micro Devices (AMD) Research 任职。

任何由多个硬件线程共享同一缓存的体系结构都应考虑细粒度缓存调度对缓存管理的重要性。一些例子包括 Intel 的 Knights Corner、[11] Oracle 的 SPARC T4、[24] IBM 的 Blue Gene/Q，[9] 以及大规模多线程*图形处理单元* (GPU)，如 AMD 和 NVIDIA 出品的 GPU。在本研究中，我们探索了线程调度对 GPU 的影响，因为它们代表了多线程体系结构的最极端示例。不过，我们的技术可以扩展并应用到由许多线程共享一个缓存的任何设计上。

1.1. 提高容量的作用

提高多任务处理能力的一种办法是简单地增加你一次可以关注的信息量。增大 GPU 缓存的大小是一种选择，我们的完整版论文 [22] 对此进行了更加详细的阐述。然而，无论使用的缓存大小怎样，它始终存在限制，而我们的研究则侧重于在这一限制基础上最大化性能。图 1 突出了增加 GPU 缓存容量的潜在好处。图 1 显示了在表 1 中所列的具有经济重要性的服务器应用程序上，通过将一级*数据* (L1D) 缓存大小增大 256 倍（从 32KB 提升到 8MB）而获得的速度增长。我们完整版论文 [22] 对这些应用程序进行了更为详细的介绍。所有这些应用程序在使用更大的 L1 数据缓存时性能提升了 3 倍或更多，表明这些程序中存在可观的局部性，并且可以从缓存更多信息中获益。

既然我们的工作负载有显著的局部性，那么下一个问题就是：它是从哪里来的？要理解这一问题，首先我们必须解释波面（或线程束，Warp）的概念。波面是 GPU 在锁步中执行其指令的一组线程。每一静态汇编指令在运行时跨越多条路径隐式执行。每台机器的一个波面中的线程数量是该机器的一个固有参数。我们基准硬件体系结构的波面大小为 32。因此，一个加载 / 存储指令在执行时可以访问多达 32 个不同的缓存行。我们研究了高缓存敏感性基准测试（图 2）的局部性，它显示了使用无限大 L1D 缓存时*每千条指令* (PKI) 的平均命中数和缺失数。该图将命中分为两个类别。我们将数据从同一波面被初始引用和重新引用时出现的局部性归类为*波面内局部性*。数据最初由一个波面引用、后来由另一波面重新引用而导致的局部性则归类为*波面间局部性*。图 2 表明我们的高缓存敏感性基准测试中观察到的大部分数据重用来自于波面内局部性。如果我们在人类尝试多任务处理的情景中思考这一结果，它意味着你的每项任务都需要大量与其他任务不共享的信息。

1.2.CCWS 的主要目标

根据任务不共享大量数据的观察结果，我们以获得更多波面内局部性为主要目标来设计 CCWS。为此，CCWS 充当一种动态线程节流机制，在预测访问会干扰缓存中已经存在的波面内局部性时防止波面发射内存指令。CCWS 通过来自“局部性损失探测器”的内存系统反馈来检测线程是否需要更为独占地访问缓存（Ⓖ图 6 中提供了我们基准体系结构的概貌，第 2 节中将进行更加详细的阐述）。局部性损失探测器通过在波面专有部分中存储 L1D 的替换受害者标签，观察 L1D 在什么时候过载。它使用这些标记来判断，是否能够给波面提供更多独占缓存访问来避免 L1D 中的缺失。局部性损失探测器提供的反馈类似于：你记得应该要做某件小事，但忘了这件事究竟是什么。你可以利用此信息来减少你要同时应付的任务数量，因为你认识到你已经超过了注意力容量。对这一反馈的解读由 CCWS 的局部性评分系统（Ⓗ）来执行。局部性评分系统决定应当允许哪些波面发射内存指令，限制那些被预测为会导致干扰的波面。更改线程调度器来提

图 1.高度缓存敏感的工作负载在各种 L1D 缓存大小上使用轮询调度器时的性能，已归一化为 32K 缓存大小。所有缓存都是 8 路组相联缓存，具有 128B 缓存行。

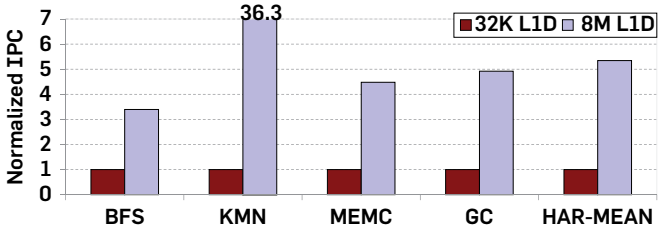
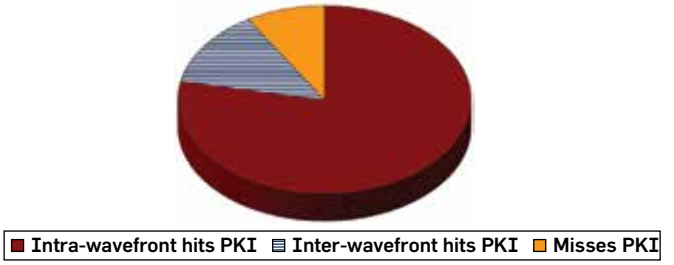


表 1.GPU 计算基准测试（CUDA 和 OpenCL）

高度缓存敏感			
名称	缩写	名称	缩写
BFS 图遍历[5]	BFS	Kmeans[5]	KMN
Memcached[10]	MEMC	垃圾收集[2]	GC

图 2.对高缓存敏感性基准测试使用无界限 L1 缓存（128B 缓存行）时每千条指令 (PKI) 的平均命中数和错失数。



高缓存效率会带来传统缓存技术未曾遇到过的新挑战。评分系统的目标是最大化吞吐量，而不仅仅是缓存命中率。通常而言，大规模多线程体系结构通过从更多线程发射指令来隐藏长延迟运算。CCWS 评分系统平衡了增加延迟容忍度和减少延迟隐藏需求之间的权衡取舍。增加主动交错的线程数量可以提高延迟容忍度，而减少交错的线程数则可降低长延迟事件的发生频率。这类似于努力找到正确的平衡，既不要一次性做太多事情，同时又要一次做多件事情。

1.3. 聪明地管理你的注意力有何作用

聪明地管理你的注意力分配方式可以增强你的多任务处理效果。用我们的问题来说，这相当于改进缓存的替换策略。为了对比硬件线程调度和替换策略的效果，你可以思考图 3 和图 4。它们从访问的缓存行数的角度显示了两种不同的线程调度器所产生的内存访问。在本例中，我们假设每一指令生成四个内存请求，使用一个有四个缓存行的全相联缓存，其采用*最近最少使用* (LRU) 的替换策略。图 3 演示了调度器尝试通过执行另一波面来填充空周期时发生的情况。在图 3 中，没有哪一波面可以在缓存中找到其数据，因为在波面被再次调度前该数据已被其他波面的请求逐出。事实上，即使使用了 Belady 最佳替换策略 [4]，缓存中也只可能有 4 次命中。图 4 中的调度器清楚各个波面访问流中存在的数据重用，而且已重新排列了波面的指令发射顺序，使其对缓存更加友好。图 4 中的调度器选择将波面 0 的内存访问一起调度，即使这意味着在波面 0 的请求返回期间让处理器保持闲置。这将导致 12 次缓存命中，捕获了波面所进行的每一个重复访问。

图 3.面向吞吐量的轮询调度器导致的访问模型示例（表示为被访问的缓存行）。字母（A、B、C...）表示所访问的缓存行。W_i 表示这一组访问生成的波面。例如，对缓存行的前四个访问 A、B、C 和 D 由波面 0 中的一个指令生成。

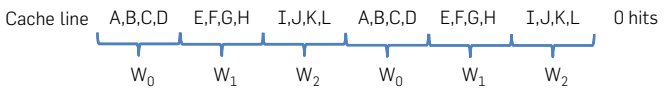


图 4.由了解其对缓存系统影响的硬件调度器产生的访问模式示例。红框突出了调度通过改变内存访问次序而改进局部性的位置。

1.4. 始终做较少事情的效果

有一种缓减任务之间干扰的简单方法，那就是对一次性执行的任务数量施加硬性限制。我们的论文提出了一种效果类似的简单机制，称为*静态波面限制*（Static Wavefront Limiting，SWL）。SWL 通过对波面调度逻辑的稍加扩展来实现，对程序启动时主动调度的波面数量施加一个最大值限制。图 5 显示了限制核心上主动调度的波面数量对高缓存敏感性应用程序的缓存性能和系统吞吐量的影响。

图 5 显示：峰值吞吐量出现在线程数小于最大并发数、但大于最优缓存缺失率（将并发数限制为一个波面）之时。在 SWL 中，编程器必须指定启动内核时波面数量的限制。如果在启动内核前用户知道或可以轻松计算最佳波面数量，这一技术就很有用。

我们的完整论文 [22] 中表明不同的基准测试程序具有不同的最佳波面数。我们另外也发现，在每一基准测试中此数字会随着输入数据的变化而改变。这种对基准测试和输入数据的依赖性使得能自适应线程运行时局部性的 CCWS 系统更具有吸引力。

2.GPU 体系结构

我们研究了对图 6 中所示的类 GPU 加速器体系结构的改动。我们研究的工作负载采用 OpenCL 或 CUDA 编写。最初，应用程序开始在主机 CPU 上执行，启动包含大量 GPU 线程的内核。我们的基准系统使用 GPGPU-Sim 3.x。[1]

我们的研究关注*波面指令发射仲裁器*（wavefront issue arbiter，WIA）（Ⓐ图 6）所做的决定。一个按序计分板 (in-order scoreboard) (Ⓑ] 和解码单元 (Ⓒ] 控制指令缓冲 (I-Buffer Ⓓ] 中的每条指令何时发射就绪。WIA 决定接下来发射这些已就绪指令中的哪一条。

图 5.高缓存敏感性基准测试在使用不同级别的多线程时的平均*每千条指令错失数* (MPKI) 和*调和平均* (HMEAN) 性能改进。*每周期指令数* (IPC) 已归一化为 32 个波面。

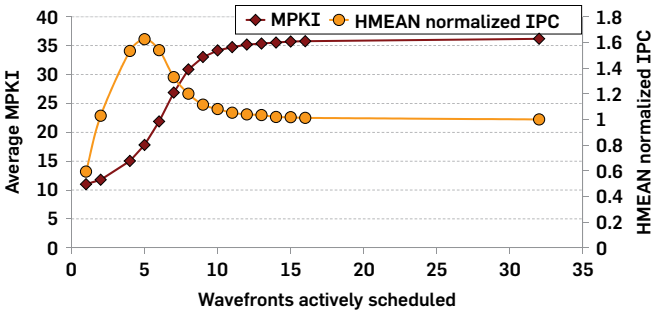
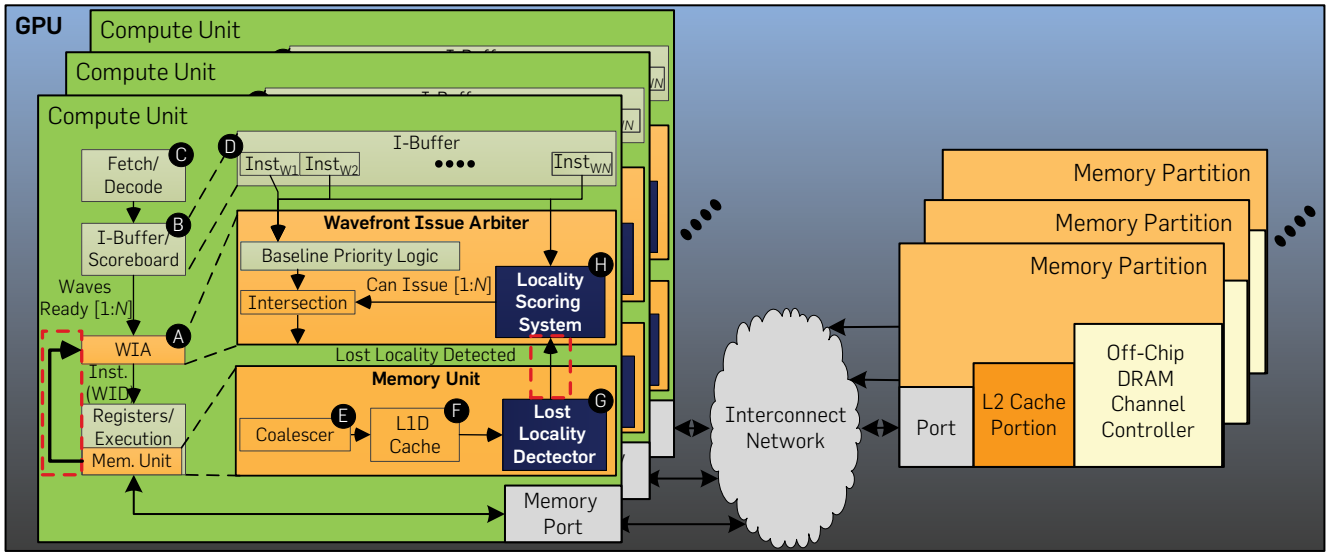


图 6.我们的类 GPU 基准加速器体系结构的概貌。$Inst_{Wi}$ 表示已准备好为波面 *i* 发布的下一指令。*N* 是核心上存储的波面上下文的最大数量。I-Buffer：*指令缓冲区*。



如前文所述，每一内存指令可以生成多个内存访问。现代 GPU 试图利用访问聚结器 (E) 来减少从每个波面生成的内存访问数量，访问聚结器在波面范围内存在空间局部性时将该路径的内存请求分组为缓存行大小的块。具有高度规则访问模式的应用程序最少可通过生成一两个内存请求来服务所有 32 个缓存行。我们的基准 GPU 包含 32K L1 数据缓存 (F)，它可以从聚结器接收内存请求。

3. 缓存感知型波面调度 (CCWS)

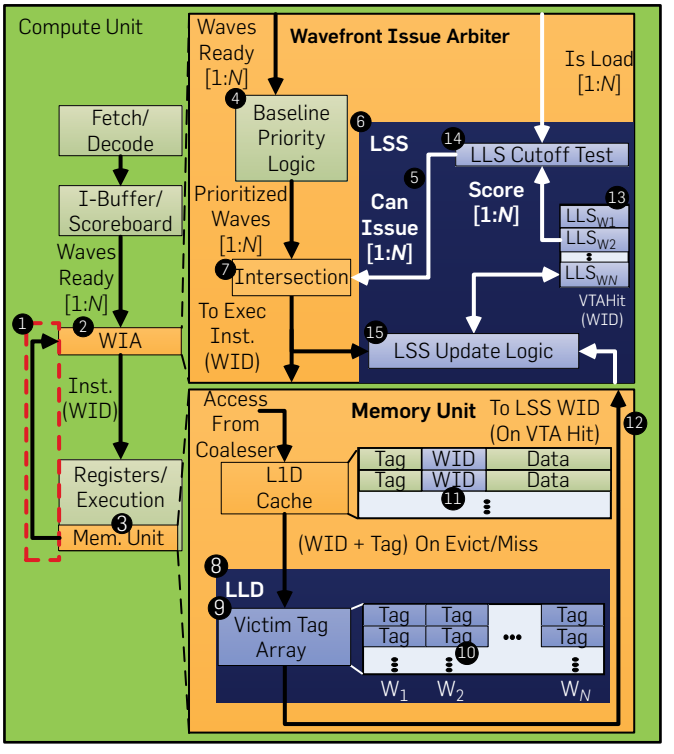
CCWS 的目标是动态判断被允许访问内存系统的波面数量，以及这些波面应该是哪些。图 7 显示了更详细的 CCWS 微体系结构视图。总体来说，CCWS 是一个波面调度器，对来自 L1D 缓存的访问级反馈（1图 7）和内存层面上*受害者标记数组* (VTA) 作出反应。

CCWS 的调度决定由局部性评分系统 (6) 做出。图 8 直观地展示了评分系统工作原理背后的情况。每一波面按照其丧失的*波面内局部性*有多少被评定一个分数。这些分数随着时间推移而变化。分数最高的波面掉到一个有序小堆栈的底部（如 T_1 时的 W_2}，使得分数较低的波面被上推超过阈值（T_1 时的 W_3），以防止它们访问 L1D。事实上，局部性评分系统移除其他波面的访问来减少来自同一波面的数据重新引用之间的访问数量。

3.1. 对基准指令发射逻辑的影响

图 7 显示对 CWS 所需的基准 WIA (2) 和内存单元 (3) 的修改。CCWS 通过扩展系统的基准波面优先级逻辑 (4] 得到实现。优先级可以贪心、轮询或两级方式进行。CCWS 通过由局部性评分系统 (6] 输出的 *Can Issue* 位向量 (5) 来防止被预测为干扰波面内局部性的负载发布指令，从而实现其运行

图 7.模拟的 GPU 核心微体系结构。*N* 是核心上存储的波面上下文的最大数量。LSS：*局部性评分系统*；LLD：*波面内局部性损失探测器*；WID：*波面 ID*；LLS：局部性损失分数；VTA：*受害者标签数组*；I-Buffer：*指令缓冲区*。



目的。交叉点逻辑块 (7) 从具有指令发布许可的就绪波面中选择优先级最高的波面。

3.2. 波面内局部性损失探测器 (LLD)

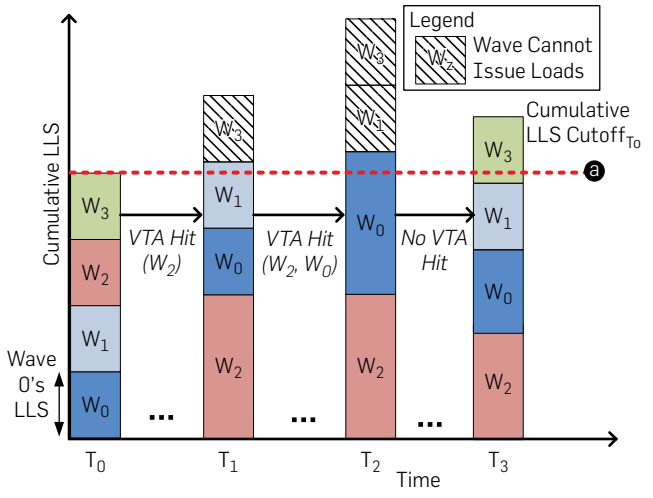
要评估哪些波面正在损失波面内局部性，我们引入了 LLD 单元 (8)，它利用*受害者标签数组* (VTA) (9)。VTA 是受害者缓存的高度修改变体。[15]VTA 的条目在此核心上支持的所有波面上下文中进行细分。这赋予每个波面其专属的小型 VTA (10)。VTA 仅存储缓存标签，不会存储缓存行数据。当出现缺失并且 L1D 缓存中预留了缓存行时，写入预留该行的波面的*波面 ID* (WID)，以及标记 (11)。当该行从缓存中逐出时，其标记信息写入到该波面在 VTA 中的对应部分。只要 L1D 缓存中出现错失，就要到 VTA 中探测。如果 VTA 中该波面的部分找到此标签，LLD 会向局部性评分系统 (12) 发送 VTA 命中信号。这些信号通知评分系统，缓存行上的一个波面发生了一次缓存缺失，如果该波面对 L1D 缓存有更加独占的访问，那么这原本可能就是一次命中。

3.3. 局部性评分系统运算

我们回到图 8 中的示例，计算单元最初被分配了四个波面。时间 T_0 与这些波面最初分配到此核心的时间对应。堆叠柱形的每一段表示每个波面获得的分数，以量化它所丧失的波面内局部性（与它所需要的缓存量相关）。我们把这些值称为*局部性损失分数* (LLS)。在 T_0 时，我们为每个波面分配一个常量基准局部性分数。LLS 分数存储在局部性评分系统的最大值堆 (13) 内。当 LLD 发送波面的 VTA 命中信号时，此波面的 LLS 就会提高。每个周期分数会掉一分，直至达到基准局部性分数。局部性评分系统通过防止 LLS 最低的波面发布加载指令，让波面内局部性丧失最多的波面拥有对 L1D 缓存更为独占的访问权限。LLS 上升到超过有序堆中的累积 LLS 阈值（ a 图 8] 的波面被禁止发射加载指令。

LLS 阈值测试块 (14] 从指令缓冲区提取一个位向量，以此指明哪些波面正在尝试发布加载指令。它还提取一个 LLS 有序列表，执行前置求和，再清除 LLS 已超出该阈值、但又尝试发布加载指令的波面的 *Can Issue* 位。在图 8 所示的示例中，在 T_0 和 T_1 之间，W_2 收到了 VTA 命中并提高了分数。W_2 较高的分数将 W_3 推到累积 LLS 阈值上方，在 W_3 试图发射加载指令时清除它的 *Can Issue* 位。从微体系结构的角度上看，LLS 由分数更新逻辑 (15) 进行修改。更新逻辑块从 LLD 接收 VTA 命中信号（带有 WID），触发对该波面的 LLS 的更改。如图 8 所示，在 T_1 和 T_2 之间，W_2 和 W_0 都收到了 VTA 命中，将 W_3 和 W_1 推到了阈值上方。在 T_2 和 T_3 之间，没有出现 VTA 命中，W_2 和 W_0 的掉落足以允许 W_1 和 W_3 再次发射加载指令。这演示了该系统如何随时间推移而自然地弱化线程截流。我们的论文对该评分系统进行了更为详细的阐述。

图 8.局部性评分系统运算示例。LLS：*局部性损失分数*。

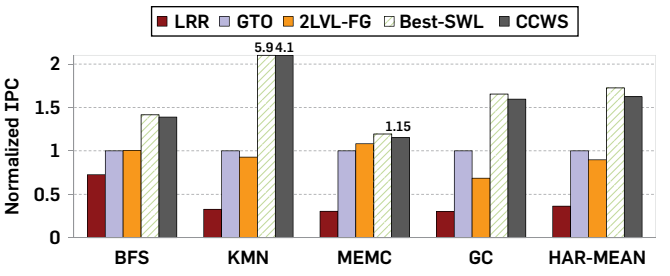


4. 评估

我们在 GPGPU-Sim[1]（3.1.0 版）中模拟了 CCWS。模拟器配置为模拟 NVIDIA Quadro FX5800，使用与 NVIDIA Fermi 类似的 L1 数据缓存和 L2 统一缓存进行扩展。我们也使用 Belady 最优替换策略 [4] 评估了 L1 数据缓存命中率，该策略选择逐出未来被重新引用时间离现在最远的缓存行。Belady 替换通过利用基于踪迹的缓存模拟器进行评估，该模拟器获取 GPGPU-Sim 缓存访问踪迹作为其输入。我们对表 1 中所列的高缓存敏感性应用程序进行了实验。完整版的论文 [22] 中还提供了与一组适度缓存敏感和对缓存不敏感的工作负载相关的其他数据，也更加详细地介绍了我们的实验设置。

图 9 和 10 中的数据利用 GPGPU-Sim 收集，用于下列机制：

图 9.高缓存敏感性基准测试中各种调度器的性能。归一化为 GTO 调度器。



LRR： 松散轮询调度。波面设定优先顺序，以轮询次序进行调度。但是，如果某一波面在其轮次上不能发射指令，轮询次序中的下一波面将有机会发射指令。

GTO： “先贪心再最旧”的调度器。GTO 运行一个波面直到其停顿，然后选取最旧的已就绪波面。

2LVL-GTO： 一种与 Narasiman 等人描述的调度器相似的两级调度器。[20] 他们的方案将核心上等待调度的波面细分为多个*获取组* (FG)，并且仅从一个获取组中执行，直到该组中的所有波面都已停止。FG 内部和 FG 之间的仲裁以 GTO 方式进行。

Best-SWL： 一种神谕式的 解决方案，即它在内核开始执行之前就已清楚可同步调度的最佳波面数量。

CCWS： 具有 GTO 波面优先级设置逻辑的缓存感知型波面调度。

图 10 中所示的 Belady 最优替换*每千指令错失数* (MPKI) 的数据通过我们基于踪迹的缓存模拟器生成：

(scheduler)-BEL： 在使用 Belady 最优替换策略时我们的缓存模拟所报告的缺失率。其使用的是通过运行具有指定 (scheduler) 的 GPGPU-Sim 而生成的访问流。

4.1. 性能

图 9 表明，在高缓存敏感性基准测试中 CCWS 比使用简单贪心型波面调度器时性能的调和平均值提升 63%，比使用 2LVL-GTO 调度器时提升 72%。GTO 调度器的表现比较好，因为给予较旧的波面更高的优先级可以使它们能够获得对 L1 数据缓存更为独占的访问，从而获取波面内局部性。

CCWS 和 SWL 具备比 GTO 调度器更进一步的优势，因为这些程序具有相当多的非合并读，在相对较少的内存指令中触及许多缓存行。因此，即使限制为仅执行最旧的波面，依然会访问过多数据，无法在 L1D 中容纳。

图 10 显示的是使用 LRU 替换策略和 Belady 最优替换策略 (BEL) 时，我们所有高缓存敏感性应用程序的平均 MPKI。它展示了波面限制方案具备的性能优势在于急剧减少了 L1D 缺失数量。此外，它也说明调度器选择不善可以减弱任何替换策略的效用。

4.2. 波面局部性的详细划分

图 11 将我们评估的调度器的 L1D 访问划分为访问缺失、波面间命中和波面内命中。此外，它也量化了因线程内局部性而造成的波面内命中数的比例。它说明，使用 CCWS 和 Best-SWL 时缓存缺失数的降低主要来自于波面内命中数的提高。此外，这些命中的大部分来自于线程内局部性。此规则的例外是*广度优先搜索* (BFS) 图形遍历，其中仅 30% 的波面内命中来自线程间局部性，而且我们也在波面间命中上看到了 23% 的提升。对代码检查后发现，当应用程序的输入图中的节点共享邻居时，线程间共享就会出现（它将自身显现为波面内局部性和波面间局部性）。限制主动调度的波面数量将提高这些访问的命中率，因为它限制缓存中的非共享数据量，提高这些共享访问的命中率。

5. 相关研究

其他论文中也提到了限制系统中线程数量可以改善性能的观察结果。Bakhoda 等人 [1] 观察到，若是在没有 L1 缓存的 GPU 核心上启动较少的工作组（或 CTA），可以通过缓减内存系统的争用来改善性能。Guz 等人 [8] 推出了一种分析模型，它可以量化共享某一缓存的线程数上升时存在的*性能低谷*。它们显示，提高线程数将提升性能，直到缓存中不再能够装下聚合工作集合。超过此点后再增加线程数会降低性能，直到有充足的线程能够掩藏系统内存延迟为止。实际上，CCWS 可以动态检测工作负载进入机器*性能低谷*的时间，并通过减少共享相关缓存的线程数量来进行补偿。Cheng 等人 [6] 引入了一种线程节流方案，以缓减多线程 CPU 系统中的内存延迟。他们提出了一种分

图 10.高缓存敏感性基准测试中各种调度器和替换策略的 MPKI。

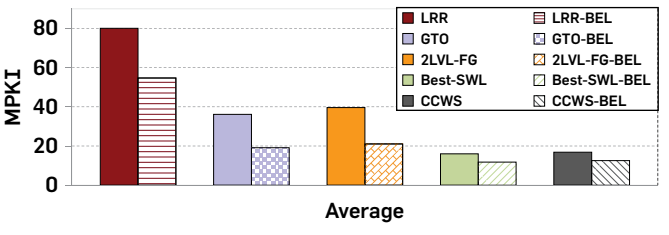
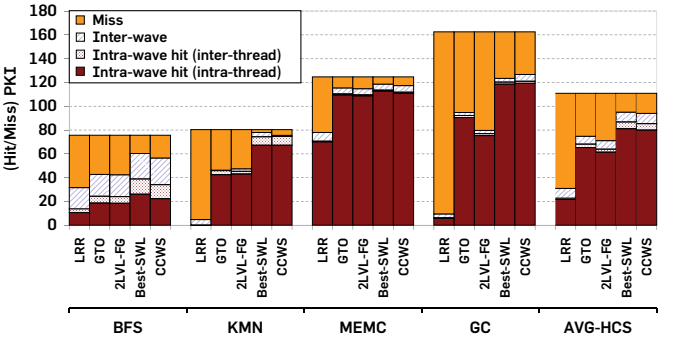


图 11.高缓存敏感性基准测试中每千条指令的 L1D 缺失数、波面内局部命中数（分为线程内和线程间）和波面间局部性命中数的细分情况。



析模型和内存任务限制节流机制，来限制内存层面上的线程干扰。

有大量的研究试图通过改进替换策略（如 Jaleel 等人[14]的研究）来提高缓存命中率。它们都试图利用各种程序行为启发法，以尽可能接近地预测数据块的重新引用间隔并对 Belady 最优[4]策略进行镜像。虽然 CCWS 也努力最大化缓存效率，但它是通过缩短重新引用间隔而不是进行预测来达成此目的。CCWS 必须限制合格波面的数量，同时保持足够的多线程处理能力以涵盖大多数的内存和运算延迟，从而对重新引用间隔的缩短加以平衡。其他方案试图管理异构工作负载之间的干扰，[12, 21]但我们的工作负载中的每个线程都有大致相似的特征。近期的研究已探索了 GPU 上对预取的利用。[18]然而，应用程序带宽受限时预取就无法改善性能，而 CCWS 可以通过减少片外通信来帮助改善。Jaleel 等人[13]的研究与我们同步，他们提出了 CRUISE 方案，利用 LLC 使用信息在多编程芯片多处理器 (CMP) 中执行高级调度决定。我们的研究关注的是大规模多线程环境中的一级缓存，并且在更加精细的粒度上加以应用。CRUISE 所做的调度决定将程序与核心绑定，而 CCWS 则在指令发布级别上决定接下来哪些线程应当进入执行流水线。

其他人也研究了 GPU 上的指令发射级调度算法。Lakshminarayana 和 Kim[17]在 GPU 无硬件管理缓存的背景下探索了大量的线程束调度策略，他们的研究表明，对于线程束执行对称（平衡）动态指令计数的应用程序而言，基于公平度的线程束和 DRAM 访问调度策略可以改善性能。多项研究探索了两级调度对 GPU 的作用。[7, 20]两级调度器利用波面间局部性，同时通过将波面分组汇总调度来确保波面在不同的时间到达长延迟的运算。然而，图 2 中显示，我们研究的高缓存敏感性基准测试从利用波面内局部性中获得的益处要多于波面间局部性。过去的调度器没有考虑发射更多波面对于之前调度的那些波面的波面内局部性产生的影响。面对 L1D 的考验时，他们的技术的轮询性质将导致较旧波面的波面内局部性被摧毁。我们在 CCWS 中的洞察基础上，进行了*分歧感知型线程束调度*（Divergence-Aware Warp Scheduling，DAWS）[23]相关研究。DAWS 试图积极地预测各个扭曲的缓存足迹。它通过考虑内存的影响来实现这一目的，同时考虑计算内核的循环结构以控制分歧。我们研究中的示例表明，DAWS 可以让未经程序员进行局部性优化 GPU 代码与优化代码的性能差距在 4% 以内。

6. 结论

当前 GPU 的体系结构在加速具有丰富并行度的应用程序上表现优秀，其内存访问是规则的并且静态可预测的。现代 CPU 具有深度缓存层次结构，每线程有数量相对较多的缓存可用；这使得它们更加适合局部性不规则的工作负载。然而，CPU 在线程数量和可用内存带宽上的局限，阻碍了它们对普遍的并行机制的利用。在运行非常重要的一类高度并行且不规则的应用程序（在这些应用程序中，线程从离散内存区域访问数据）时，每种设计都存在问题。未来的体系结构如何加速这些应用程序，这一问题具有重要意义。

许多高度并行且不规则的应用程序从商业角度而言非常重要，常常运用于现代数据中心。我们论文中的高缓存敏感性基准测试包含了多个此类工作负载，例如 Memcached[10]、并行垃圾收集器[2]，以及广度优先搜索图遍历[5]程序。我们的研究表明，在使用本机线程调度器时，这些应用程序对 L1 缓存容量高度敏感。我们的研究也表明，相对小的 L1 缓存可以获取其局部性并提高性能，前提是使用一种智能的指令发射级线程调度方案。

尽管我们的研究主要关注性能，但 CCWS 以及由此而带来的精细线程调度对功耗的影响也很重要。随着现代芯片在功耗上的限制日益加剧，维持缓存中的局部性可以成为降低功耗的一种有效方式。CCWS 可以经过调节以进一步减少数据缓存错失数，但代价是牺牲一些性能。

在我们看来，这一研究为细粒度内存系统管理提供了一个全新角度。我们认为，将缓存系统与指令发射级线程调度器进行整合以改变内存系统所见的访问流，为缓存管理的研究开辟了一个新的方向。CCWS 表明，相对简单、动态适应的反馈驱动型调度系统可以大大改进一类重要应用程序的性能。

鸣谢

衷心感谢 Norm Jouppi、Yale N. Patt、Aamer Jaleel、Wilson Fung、Hadi Jooybar、Inderpreet Singh、Tayler Hetherington、Ali Bakhoda 和匿名审阅人，感谢他们提供的宝贵反馈。同时也对 Rimon Tadros 在垃圾回收器基准测试上进行的研究致以诚挚谢意。此研究的部分资金来自 Advanced Micro Devices Inc. 的资助。

参考资料

1. Bakhoda, A., Yuan, G., Fung, W., Wong, H., Aamodt, T. Analyzing CUDA workloads using a detailed GPU simulator.In *Proceedings of International Symposium on Performance Analysis of Systems and Software (ISPASS 2009)*, 163–174.
2. Barabash, K., Petrank, E. Tracing garbage collection on highly parallel platforms.In *Proceedings of International Symposium on Memory Management (ISMM 2010)*, 1–10.
3. Beckmann, B.M., Marty, M.R., Wood, D.A.ASR:Adaptive selective replication for CMP caches.In *Proceedings of International Symposium on Microarchitecture (MICRO 39)* (2006), 443–454.
4. Belady, L.A.A study of replacement algorithms for a virtual-storage computer.*IBM Syst.J.5*, 2 (1966), 78–101.
5. Che, S., Boyer, M., Meng, J., Tarjan, D., Sheaffer, J., Lee, S.H., Skadron, K. Rodinia:A benchmark suite for heterogeneous computing.

In *Proceedings of International Symposium on Workload Characterization (IISWC 2009)*, 44–54.
6. Cheng, H.Y., Lin, C.H., Li, J., Yang, C.L. Memory latency reduction via thread throttling.In *Proceedings of International Symposium on Microarchitecture (MICRO 43)* (2010), 53–64.
7. Fung, W., Aamodt, T. Thread block compaction for efficient SIMT control flow.In *Proceedings of International Symposium on High Performance Computer Architecture (HPCA 2011)*, 25–36.
8. Guz, Z., Bolotin, E., Keidar, I., Kolodny, A., Mendelson, A., Weiser, U. Many-core vs. many-thread machines:Stay away from the valley.*Comput.Architect.Lett.8*, 1 (Jan. 2009), 25–28.
9. Haring, R., Ohmacht, M., Fox, T., Gschwind, M., Satterfield, D., Sugavanam, K., Coteus, P., Heidelberger, P., Blumrich, M., Wisniewski, R., Gara, A., Chiu, G.T., Boyle, P., Chist, N., Kim, C. The IBM Blue Gene/Q compute chip.*Micro IEEE 32*, 2 (Mar.–Apr. 2012), 48–60.
10. Hetherington, T.H., Rogers, T.G., Hsu, L., O'Connor, M., Aamodt, T.M. Characterizing and evaluating a key-value store application on heterogeneous CPU-GPU systems. In *Proceedings of International Symposium on Performance Analysis of Systems and Software (ISPASS 2012)*, 88–98.
11. Intel Xeon Phi Coprocessor Brief. http://www.intel.com/content/www/us/en/high-performance-computing/high-performance-xeon-phi-coprocessor-brief.html.
12. Jaleel, A., Hasenplaugh, W., Qureshi, M., Sebot, J., Steely, S., Jr., Emer, J. Adaptive insertion policies for managing shared caches.In *Proceedings of International Conference on Parallel Architecture and Compiler Techniques (PACT 2008)*, 208–219.
13. Jaleel, A., Najaf-abadi, H.H., Subramaniam, S., Steely, S.C., Emer, J. CRUISE:Cache replacement and utility-aware scheduling. In *Proceedings of International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2012)*, 249–260.
14. Jaleel, A., Theobald, K.B., Steely, S.C., Jr., Emer, J. High performance cache replacement using re-reference interval prediction (RRIP).In *Proceedings of International Symposium on Computer Architecture (ISCA 2010)*, 60–71.
15. Jouppi, N.P. Improving direct-mapped cache performance by the addition of a small fully-associative cache and prefetch buffers.In *Proceedings of International Symposium on Computer Architecture (ISCA 1990)*, 364–373.
16. Kahneman D. Attention and Effort Prentice-Hall, 1973.
17. Lakshminarayana, N.B., Kim, H. Effect of instruction fetch and memory scheduling on GPU performance.In *Workshop on Language, Compiler, and Architecture Support for GPGPU* (2010).
18. Lee, J., Lakshminarayana, N.B., Kim, H., Vuduc, R. Many-thread aware prefetching mechanisms for GPGPU applications.In *Proceedings of International Symposium on Microarchitecture (MICRO 43)* (2010), 213–224.
19. Marty, M.R., Hill, M.D. Coherence ordering for ring-based chip multiprocessors.In *Proceedings of International Symposium on Microarchitecture (MICRO 39)* (2006), 309–320.
20. Narasiman, V., Shebanow, M., Lee, C.J., Miftakhutdinov, R., Mutlu, O., Patt, Y.N.Improving GPU performance via large warps and two-level warp scheduling.In *Proceedings of International Symposium on Microarchitecture (MICRO 44)* (2011), 308–317.
21. Qureshi, M.K., Patt, Y.N.Utility based cache partitioning: A low-overhead, high-performance, runtime mechanism to partition shared caches.In *Proceedings of International Symposium on Microarchitecture (MICRO 39)* (2006), 423–432.
22. Rogers, T.G., O'Connor, M., Aamodt, T.M. Cache-Conscious Wavefront Scheduling. In *Proceedings of IEEE/ACM International Symposiumon Microarchitecture (MICRO-45)* (2012).
23. Rogers, T.G., O'Connor, M., Aamodt, T.M. Divergence-Aware Warp Scheduling.In *Proceedings of IEEE/ACM International Symposium on Microarchitecture (MICRO-46)* (2013).
24. Shah, M., Golla, R., Grohoski, G., Jordan, P., Barreh, J., Brooks, J., Greenberg, M., Levinsky, G., Luttrell, M., Olson, C., Samoail, Z., Smittle, M., Ziaja, T. Sparc T4:A dynamically threaded server-on-a-chip.*Micro IEEE 32*, 2 (Mar.–Apr. 2012), 8–19.
25. Wolf, M.E., Lam, M.S.A data locality optimizing algorithm.In *Proceedings of ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI '91)* (1991), 30–44.

Timothy G. Rogers 和 Tor M. Aamodt ({tgrogers, aamodt}@ece.ubc.ca) 就职于加拿大温哥华英属哥伦比亚大学电子与计算机工程系。

Mike O'Connor (moconnor@nvidia.com) 就职于德克萨斯州奥斯汀的 NVIDIA 研究中心。

译文责任编辑：陈文光