集群工作负载的多样性及其对研究成果的影响

摘要

六年前，Google发布了一系列宝贵的调度程序日志，这些日志已在450多个出版物中使用。但是，我们发现其他数据源的匮乏导致研究人员将他们的工作过分适应Google的数据集特征。我们通过引入来自两个私有和两个高性能计算（HPC）集群的四个新轨迹来证明这种过度拟合。我们的分析表明，由数据分析作业组成的私有集群工作负载与Google工作负载的相关性更高，与HPC集群工作负载的相似性更高。该观察结果表明，在评估新研究的普遍性时应考虑其他痕迹。

为了帮助社区向前发展，我们发布了四个已分析的跟踪记录，包括：跨HPC群集生命周期的所有61个月的最长的公开可用跟踪以及来自300,000核心HPC群集的跟踪，这是具有公开跟踪的最大群集。我们对私有和HPC集群跟踪进行了分析，这些跟踪跨越了工作特征，工作负载异构性，资源利用率和故障率。我们将我们的发现与Google的跟踪特征进行对比，并在文献中找出受影响的工作。最后，我们通过使用所有四个跟踪和Google跟踪来评估作业运行时预测变量的性能，证明了数据集多样性和多样性的重要性。

1引言

尽管在云计算和作业调度研究领域活跃，但可公开获得的集群工作负载数据集仍然很少。今天的三个主要数据集来源是：2011年收集的Google集群跟踪[58]，自1993年以来收集的高性能计算（HPC）跟踪的并行工作负载档案[19]和2011年发布的SWIM跟踪[10]。其中，Google跟踪已在450多个出版物中使用，使其成为迄今为止最受欢迎的跟踪。不幸的是，这一为期29天的跟踪记录通常是唯一用于评估新研究的跟踪记录。通过将其特性与来自不同环境的更新轨迹进行对比，我们发现仅Google轨迹不足以准确证明新技术的普遍性。

我们的目标是发现现有工作过分适合Google跟踪的特征。为了实现这一目标，我们的第一项贡献是引入了四个新的踪迹：两个来自对冲基金Two Sigma的私有云，另外两个来自位于洛斯阿拉莫斯国家实验室（LANL）的HPC集群。我们的两条西格码迹线是迄今为止最长的非学术性私人集群迹线，跨越9个月，提供了超过300万个工作岗位。我们介绍的两条HPC轨迹也很独特。第一条迹线跨越了通用HPC群集的整个5年寿命，使其成为迄今为止最长的公共迹线，并且比现有的公共HPC迹线还表现出更短的工作。第二条迹线来自LANL目前拥有300,000核的旗舰超级计算机，据我们所知，它是具有公开迹线的最大群集。我们将在第2节中介绍所有四个跟踪以及它们的收集环境。

我们的第二个贡献是在考虑了四个新跟踪时分析了从Google跟踪得出的工作负载特征的一般性。总体而言，我们发现私有的两个Sigma集群工作负载具有与HPC相似的特征，尽管它们包含的数据分析工作与Google工作负载更为相似。表1总结了我们的所有发现。对于Google工作负载异常的那些特征，我们已经调查了文献并列出了受影响的先前工作。总共，我们调查了450篇参考Google跟踪研究的论文[41]来确定流行的工作负载假设，并将它们与两个西格玛和LANL工作负载进行对比，以检测违规情况。我们将调查结果分为四类：工作特征（第3节），工作负载异构性（第4节），资源利用率（第5节）和故障分析（第6节）。

我们的发现表明，仅使用Google跟踪评估新研究不足以保证通用性。为了帮助社区前进，我们的第三项贡献是公开发布本文介绍和分析的四个痕迹。我们还提供了一个案例研究，用于评估新研究时数据集的多样性和多样性的重要性。对于我们的演示，我们使用JamaisVu调度系统的作业运行时间预测器JVuPredict [51]。最初，仅使用Google跟踪[51]来评估JVuPredict。但是，通过我们的四条新轨迹来评估其性能，可以帮助我们确定可以更容易地检测出具有可预测行为的相关和重复作业的功能。这使我们能够量化单个跟踪字段在运行时预测中的重要性。我们将在第7节中描述我们的发现。最后，在第8节中，我们简要讨论跟踪长度对准确表示集群工作量的重要性。在结束之前，第9节中列出了研究集群跟踪的相关工作。

2数据集信息

我们介绍了四组作业计划程序日志，这些日志是从LANL的通用集群和尖端超级计算机以及对冲基金Two Sigma的两个集群中收集的。以下小节更详细地描述了每个数据集，每个群集的硬件配置如表2所示。用户通常通过提交产生跨群集节点分布的多个过程或任务的命令来执行特定计算，从而与群集调度程序进行交互。 。每个这样的命令都被认为是一项工作，用户经常编写脚本，这些脚本会生成更复杂的多任务计划。在HPC集群中，类似于Emulab [4、16、27、57]，在物理节点的粒度上分配资源的情况下，来自不同作业的任务永远不会调度在同一节点上。在诸如“两个西格玛”之类的私有集群中，不一定是这样。

2.1两个Sigma群集

我们介绍的私人工作负载痕迹来自对冲基金公司Two Sigma的两个数据中心。工作负载包括处理财务数据的数据分析作业。这些工作的一小部分由Spark [49]安装处理，其余的由本地数据分析框架提供服务。从2016年1月开始，数据集涵盖两个数据中心运营的9个月，涵盖了总共1313个相同的计算节点，31512个CPU内核和328TB RAM。日志包含320万个工作和7850万个任务，由运行在Mesos之上的内部开发的工作计划程序收集[28]。由于两个数据中心的工作负荷相同，并且由同类节点组成，因此我们将两个数据源统称为TwoSigma跟踪，并一起进行分析。我们期望这种工作负载比HPC集群更像Google集群，在HPC集群中，长期运行，计算密集和紧密耦合的科学工作是常态。首先，与LANL不同，作业运行时没有严格的预算；对冲基金集群的用户在提交工作时无需指定时间限制。其次，用户可以分配单独的核心，而不是在LANL分配整个物理节点。收集的数据包括：从提交到终止的作业阶段的时间戳记，作业属性（例如大小和所有者）以及作业的返回状态。

2.2 LANL野马集群

从2011年到2016年，Mustang是LANL用于容量计算的HPC集群。诸如Mustang之类的容量集群被设计为具有成本效益的通用资源，可满足大量用户的需求。野马在LANL的科学家，工程师和软件开发人员中得到了广泛的使用，并且以物理节点的粒度将它们分配给了这些用户。该集群由1600个相同的计算节点组成，总共有38400个AMD Opteron 6176 2.3GHz内核和102TB RAM。我们的Mustang数据集涵盖了从2011年10月到2016年11月机器运行的整个61个月，这是迄今为止最长的公开可用群集跟踪。野马跟踪也很独特，因为它的工作比现有的HPC跟踪的工作要短。总体而言，它由565个用户提交并由开源集群资源管理器SLURM [45]收集的210万个多节点作业组成。跟踪中可用的字段类似于TwoSigma跟踪中的字段，每个作业都添加了一个时间预算字段，如果超出该字段，则该作业将被杀死。

2.3 LANL Trinity超级计算机

在2018年，Trinity是LANL上最大的超级计算机，用于能力计算。功能集群是一种大规模的高需求资源，它引入了新颖的硬件技术，可帮助实现至关重要的计算里程碑，例如更高分辨率的气候和天体物理学模型。 Trinity的硬件在两个试生产阶段就已经投入使用，然后才能投入全面生产，并且在第二阶段完成之前就收集了我们的踪迹。在数据收集时，Trinity由9408个相同的计算节点，总共301056个Intel Xeon E5-2698v3 2.3GHz内核和1.2PB RAM组成，这使其成为最大的群集，具有按CPU内核数量公开可用的跟踪信息。我们的Trinity数据集涵盖了从2017年2月到2017年4月的3个月。在此期间，Trinity以OpenScience模式运行，即，该计算机正在进行Beta测试，可供更多用户使用，其数量超出预期的数量。最终的安全分类。我们注意到，OpenScience工作负载代表了功能超级计算机的工作负载，因为在引入新计算机或淘汰旧计算机之前，大约每18个月就会发生一次。数据集（以下我们称为OpenTrinity）由88个用户发出的25237个多节点作业组成，并由开源集群调度系统MOAB [1]收集。跟踪中的可用信息与Mustang跟踪中的信息相同。

2.4 Google集群

2012年，Google发布了在其计算集群之一中运行的作业的痕迹[41]。它是一个为期29天的跟踪，包含672074个工作和4,800万个任务，其中一些任务是通过MapReduce框架发布的，并于2011年5月在12583个异构节点上运行。工作量包括长期运行的服务和批处理工作[55] ]。 Google尚未发布每个群集节点的确切硬件规格。相反，如表2所示，通过匿名平台名称表示节点，这些平台名称表示具有微体系结构和芯片集的不同组合的计算机[58]。请注意，跟踪中每个节点的CPU内核和RAM的数量已标准化为集群中功能最强大的节点。在我们的分析中，我们估计Google集群中的内核总数为106544。我们通过假设最流行的节点类型（带有0.5 CPU内核的Google B）是双插槽服务器（搭载四核AMD Opteron）来得出此数字。 Google当时称其数据中心使用的Barchelona CPU [26]。与以前的工作负载不同，可以为作业分配CPU内核的一部分[46]。

3工作特点

文献中先前工作的许多实例都依赖于重尾分布的假设来描述单个工作的规模和持续时间[2、8、13、14、40、50]。在LANL和TwoSigma工作负载中，这些尾巴看起来轻得多。

观察1：平均而言，TwoSigma和LANL跟踪中的作业所请求的CPU内核比Google跟踪中的作业多3到406倍。 LANL跟踪中的作业大小更均匀地分布。

图1显示了所有迹线中CPU内核的作业请求的累积分布函数（CDF），x轴为对数刻度。我们发现Google跟踪中90％的最小作业需要16个CPU内核或更少。 TwoSigma作业的相同部分在LANL跟踪中要求108个内核和1-16K内核。在Google之外，大型工作也很常见。对于LANL HPC群集而言，这并不奇怪，在该群集中，为一个作业分配数千个CPU内核并不少见，因为群集的主要用途是运行大规模并行科学应用程序。但是，有趣的是，尽管TwoSigma集群的核心数少于我们检查的其他集群（比Google集群少3倍），但其中位数工作却比Google跟踪中的工作大了一个数量级。 。对分配的内存的分析得出类似的趋势。

观察结果2：Google跟踪中的中位作业比LANL或TwoSigma跟踪中的中位作业短4-5倍。但是，Google跟踪中最长的1％作业比LANL和TwoSigma跟踪中相同部分的作业长2-6倍。

图2显示了所有跟踪的作业持续时间的CDF。我们发现在Google跟踪中，每个工作的80％持续时间少于12分钟。在LANL和TwoSigma跟踪中，作业至少要长一个数量级。在TwoSigma中，相同比例的作业最多可持续2个小时，而在LANL中，对于Mustang而言，它们可以长达3个小时，对于OpenTrinity而言，它们可以长达6个小时。令人惊讶的是，LANL群集的分布的尾端比Google和TwoSigma群集的分布稍短。最长的工作是在Mustang上工作16个小时，在Open-Trinity中工作32个小时，在TwoSigma中工作200个小时，在Google中至少工作29天（跟踪的持续时间）。对于LANL，这是由于硬性限制导致作业被任意杀死。对于Google来说，发行版的长尾很可能归因于长期运行的服务。

含义。这些观察结果影响了作业调度方法的即时适用性，该方法的效率取决于以下假设：绝大多数作业的工期在数分钟左右，并且作业规模与集群的规模相比微不足道。例如，Ananthanarayanan等。 [2]建议通过复制较小工作的任务来减轻流浪汉的影响。这是处理Internet服务工作负载的有效方法（本文中代表Microsoft和Facebook），因为绝大多数工作都可以从中受益，而不会显着提高总体群集利用率。例如，对于Google跟踪，90％的作业请求的群集少于0.01％，因此复制它们只会稍微提高群集的利用率。同时，LANL和TwoSigma跟踪中的25-55％的作业每个都要求超过集群核心的0.1％，这降低了该方法的效率，并建议应谨慎使用复制。这并不意味着LANL任务也是紧密耦合的，并且必须重复整个作业。

另一个例子是Delgado等人的工作。 [14]，通过将分散的调度程序分配给集群的一小部分来提高它们的效率。该分区的范围从Yahoo和Facebook跟踪的2％到Google跟踪的17％（工作明显更长），以避免增加工作服务时间。对于TwoSigma和LANL跟踪，我们已经表明作业比Google跟踪更长（图2），因此可能需要更大的分区才能实现类似的效率。同时，在TwoSigma和LANL群集中运行的作业也更大（图1），因此，除非缩小分区，否则长作业的服务时间预计会增加。其他可能受到影响的工作示例包括短期和小型工作的任务迁移[50]和旨在改善短期工作的行头阻塞的混合调度[13]。

4工作负载异构

关于云工作负载的另一个常见假设是，它们的特征是可用于作业的资源以及作业的到达间隔时间各不相同[7，23，31，46，56]。但是，我们研究的私有集群和HPC集群由同类硬件组成（请参见表2），并且用户活动遵循定义明确的昼夜模式，即使调度请求的速率在各个集群之间变化很大。

观察3：日间模式是普遍的。集群在白天收到更多的调度请求和较小的工作，而Google跟踪的偏差较小。

在图3中，我们显示了一天中每小时的作业调度请求数。我们选择显示其他两个四分位数所包围的中位数天的指标，因为天之间的高变化会导致大多数天的平均值不具有代表性（请参见第8节）。总体而言，昼夜模式在每条迹线中都很明显，用户活动集中在白天（上午7点至下午7点），类似于先前的工作[38]。 Google跟踪是一个例外，该跟踪在午夜至凌晨4点最活跃，大概是由于批处理作业利用了可用资源。提交的作业的大小也与一天中的时间相关。

我们发现，LANL跟踪中较长，较大的作业通常是在夜间安排的，而较短，较小的作业通常是在白天安排的。对于Google跟踪而言，情况恰恰相反，这提示了我们先前对夜间批处理作业的假设。尽管具有类似于LANL群集的昼夜模式，但在TwoSigma群集中也计划在白天进行长时间的大型工作。这可能是由于TwoSigma的工​​作量包括财务数据分析而引起的，该工作量依赖于股市的交易时间。

观察4：调度请求率在整个集群中相差最多3个数量级。为了跟上工作量，亚秒级的调度决策似乎是必要的。

从图3中可以删除的另一件事是，在集群之间调度请求的速率可能有很大差异。对于Google和TwoSigma跟踪，每小时要提交数百到数千个作业。另一方面，LANL调度程序在任何给定的小时内都不会收到超过40个请求。这可能与工作负载或系统中的用户数量有关，因为Google集群提供的用户ID是Mustang集群的2倍，而OpenTrinity是9倍。

启示：诸如Omega [46]和ClusterFQ [56]之类的先前工作提出了分布式调度设计，尤其适用于异构集群。对于LANL和TwoSigma等环境而言，这似乎不是问题，它们有意构建同类集群以降低性能优化和管理成本。

随着群集大小的增加，调度请求的速率也会增加，这促使我们重新检查以前的工作。 Quincy [31]将调度表示为任务节点图上的最小成本最大流（MCMF）优化问题，并不断完善任务放置。但是，这种方法的复杂性对于诸如我们研究的大规模集群来说是一个缺点。 Gog等。 [23]发现，昆西平均需要66秒才能收敛到10,000个节点的集群中的放置决策。我们研究的Google和LANL集群已经达到了这种规模（表2）。我们在图3中显示，LANL跟踪中作业提交的平均频率是每90秒执行一次作业，这意味着此调度延迟可能有效，但这种情况不会持续很长时间。 Trinity目前在19,000个节点上运行，根据美国能源部的百亿亿次计算项目[39]，计划在未来5年内将计算机规模扩大25倍。请注意，到目前为止，在讨论调度时，我们指的是作业，因为HPC作业具有帮派调度要求。但是，诸如Quincy之类的布局算法专注于任务布局。

对Quincy的改进是Firmament [23]，这是一种集中式调度程序，采用基于MCMF优化技术组合的通用方法来平均实现亚秒级任务放置延迟。如图4所示，亚秒级延迟至关重要，因为Google和TwoSigma跟踪中的任务放置请求的速率可以高达每小时10万个请求，即每36ms执行一次任务。但是，随着群集利用率的提高，固件的放置延迟会增加到几秒钟。对于TwoSigma和Google跟踪，这可能会出现问题。

5资源利用

云的一个众所周知的动机是资源整合，目的是降低设备拥有成本。但是，云的一个同样众所周知的特性是其资源仍未得到充分利用[6、15、35、36、41]。这主要是由于用户资源请求与实际资源使用之间的差异，最近的研究工作试图通过工作负载特征化和积极整合来缓解这种差异[15、33、34]。我们的分析发现，与Google跟踪相比，LANL和TwoSigma跟踪中的用户资源请求具有更高的可变性。我们还研究了工作到达时间以及在评估新研究时如何估算它们。

观察5：与Google集群不同，我们没有研究其他集群过度使用资源。

总体而言，我们发现在我们研究的所有集群中，分配给作业的CPU内核比例在一段时间内是稳定的。对于Google而言，CPU内核的超额配置量为10％，而对于其他群集，未分配的内核的范围为2-12％，即使它们的调度程序支持资源超额配置。内存分配数量也遵循类似的趋势。不幸的是，LANL和TwoSigma跟踪不包含有关实际资源利用率的信息。结果，我们既不能确认也不能与早期关于资源分配和利用之间的不平衡的研究结果相矛盾。组织之间的不同之处在于保持资源被利用或可用的动机。对于Google [41]，Facebook [10]和Twitter [15]而言，在仅维持必要硬件以保持较低运营成本的财务动机与满足高峰需求的需求之间存在紧张关系，这导致总体需求较低利用率。对于LANL，群集被设计为在预定时间段内容纳一组预定义的应用程序，并且计划将高利用率作为有效利用联邦资金的一部分。对于TwoSigma群集，即使高峰使用导致了较低的总体利用率，但由于业务收入与分析工作的响应时间密切相关，因此为高峰需求进行配置更为重要。

观察6：大多数的工作间隔时间是亚秒级以下。

间隔时间是实验设置的关键参数，因为间隔时间决定了被测系统的负载。两种常见的配置是第二粒度[15]或泊松分布的到达间隔[29]，我们发现它们都不能准确地描述到达间隔。在图5中，我们显示了作业到达间隔长度的CDF。我们观察到44-62％的间隔到达时间是亚秒级，这意味着工作到达的速度比以前的假设要快。此外，由于Kolmogorov-Smirnov检验[37]拒绝了p值<2：210􀀀16的原假设，我们在此数据上拟合Poisson分布的尝试仍未成功。该结果不考虑存在速率参数随时间变化的基础泊松过程的情况，但建议在假定泊松分布时应格外小心。

另一个普遍的假设是，工作很少是大型的，即由多个任务组成[29，56]。在图6中，我们显示了跨组织每个工作的任务数量的CDF。我们观察到，有77％的Google作业是单任务作业，但是其余集群则承载着更多的多任务作业。我们注意到，TwoSigma发行版仅适用于较大的职位。这表明在Google之外，任务放置可能是一个更棘手的问题，因为在Google那里单任务工作很常见，这加剧了我们在第4节中概述的现有任务放置算法的评估问题。

观察7：在LANL和TwoSigma跟踪中，用户资源请求比在Google跟踪中具有更多的可变性。

通过工作负载合并可以缓解资源利用不足。为了确保最小的干扰，通常根据历史数据对应用程序进行概要分析和分类[15，33]。我们的分析表明，这种方法在Internet服务领域之外可能不太成功。为了量化用户行为的变异性，我们检查了各个用户所有请求的变异系数1（CoV）。对于Google跟踪，我们发现大多数用户在CPU内核中发出的作业是其平均请求的2倍之内。另一方面，对于LANL和TwoSigma跟踪，60-80％的用户可以偏离其平均请求的2-10倍。

启示：对Google [41]，Twitter [15]和Facebook [10]数据的许多早期研究都强调了资源分配和利用之间的不平衡。 Google通过过量使用资源来解决此问题，但LANL和TwoSigma并非如此。提出的另一种解决方案是Quasar [15]，该系统在保证预定的QoS水平的同时合并工作负载。这是通过在提交时对作业进行概要分析并将其归类为先前遇到的工作负荷之一来实现的；通过在正在运行的应用程序中插入探针来检测错误分类。对于LANL，这种方法是不可行的。首先，无法按比例缩小作业的概要，因为通常会针对请求的分配大小精心配置提交的代码。其次，提交的代码太复杂而无法在几秒钟内准确地进行概要分析，并且在运行时对其进行探测以检测错误分类会引入性能抖动，而这种性能抖动在紧密耦合的HPC应用程序中是无法实现的。第三，在LANL跟踪中，我们经常发现用户在重新提交作业之前先对其进行调整，因为他们会重新校准模拟参数以实现成功运行，这很可能会影响分类的准确性。第四，为工作负载精心预留了资源，并且利用率很高，这使得为配置提供资源变得困难。对于TwoSigma和Google跟踪，Quasar可能更适合，但是，以每秒2.7个作业的速度（图3），在提交时进行15秒的分析[15]会导致将6个作业的预期负载组合在一起。由于Quasar需要进行4次并行且隔离的运行才能收集足够的性能分析数据，因此我们需要资源来同时运行至少360个VM，并确保两者之间的性能隔离以跟上平均负载。这进一步假设分析时间不需要增加超过15秒。最后，Quasar [15]使用多秒的到达间隔进行评估，因此有必要进行测试以确保可以处理一个数量级以上的负载（图5），并且不会进一步增加配置成本。

TSF [56]提供了另一种与工作负载合并相关的方法，这是一种调度算法，它试图最大化分配给每个作业的任务插槽的数量，而不偏爱更大的作业。这可以确保算法保持无饥饿状态，但是会导致具有100多个任务的作业的运行时间显着减慢，这被作者定义为很大。对于LANL来说，这是禁止的，因为LANL中必须将作业作为一个整体进行调度，而这样的“大”作业更为普遍并且持续时间更长。其他用于调度和放置的方法假设我们在这里研究的集群中可能没有可用的资源的可用性，并且在高效利用的集群中，它们的性能已降低[25，29]。

6故障分析

经常对作业调度程序日志进行分析，以了解不同环境中的作业失败特征[9、17、21、22、43]。这些知识可以构建更强大的系统，这一点在我们过渡到每隔几分钟就会发生故障的百亿美元计算系统以及建立在会增加故障率的复杂软件堆栈上的云计算环境时特别重要[9，44]。

定义。任何故障分析的重要起点是定义故障事件的构成。在我们考虑的所有跟踪中，我们将所有由于事件非用户或系统管理员的意图而结束的工作都定义为失败的工作。我们无法通过失败的根本原因来区分失败的工作，例如软件和硬件问题，因为无法可靠地获得此信息。除了成功和失败之外，跟踪中还有其他作业终止状态。对于Google跟踪，作业可以被用户杀死，可以将任务逐出以安排更高优先级的任务，或者退出状态未知。对于LANL跟踪，可以有意取消作业。我们将所有这些工作成果归类为中止的工作，并将失败和中止的工作统称为不成功的工作。

还有另一个工作成果类别。在LANL，要求用户为每个作业指定运行时估计。与SLO相似，此估计被视为时间限制，如果超过该限制，调度程序将终止该作业。我们将这些杀死事件称为超时作业，因为它们可以在以下三种情况下产生有用的工作，所以将它们分别列出：（a）HPC作业使用时间限制作为停止标准；（b）作业状态定期检查到磁盘时；以及（c）当工作在时限之前完成工作但未能彻底终止时。

观察结果8：Google跟踪中未成功终止的工作比其他跟踪中高1.4-6.8倍。 LANL上不成功的作业使用的CPU时间减少了34-80％。

在图7中，我们按作业结果细分了作业总数（左）以及所有作业消耗的CPU总时间（右）。首先，我们观察到，对于Google跟踪，不成功的工作所占的比例明显高于其他跟踪（1.4-6.8x）。这种比较忽略了野马超时的作业，因为如上所述，它们不太可能浪费资源。我们还注意到，Google跟踪中几乎所有不成功的作业都被中止了。根据跟踪文档[58]，这些作业可能已被用户或调度程序中止，或者被失败的相关作业中止。结果，我们不能排除这些工作与故障有关的可能性。因此，先前的工作会将所有不成功的工作归为“失败”标签[17]，为清楚起见，我们选择避免​​该标签。另一个事实进一步突出了失败和中止的作业之间的界限有多模糊，这是TwoTwo跟踪中所有未成功的作业都被分配了失败状态。简而言之，我们将工作分类为“不成功”可能看起来很广泛，但这与文献中对“失败”一词的宽泛使用是一致的。

我们还发现，不成功的工作并不会同样不利于所有集群的整体效率。尽管TwoSigma跟踪的不成功工作率与OpenTrinity跟踪中的不成功工作率相似，但是每个不成功的工作持续时间更长。特别是，与Google和TwoSigma跟踪相比，LANL跟踪中的不成功作业浪费的CPU时间减少了34-80％。值得注意的是，在LANL上有49-55％的CPU时间分配给了超时的作业，这表明通过使用更好的检查点策略，至少有一部分时间可以使用。

观察9：对于Google跟踪，不成功的工作往往比成功的工作需要更多的资源。这对于所有其他跟踪都是不正确的。

在图8中，我们显示了单个作业的作业大小的CDF（在CPU内核中）。对于每条迹线，我们分别显示不成功和成功工作的CDF。通过根据结果将工作分开，我们发现，在Google跟踪中，成功的工作总体上比不成功的工作需要更少的资源。在早期的工作中也曾观察到这种现象[17，21]，但它并不适用于我们的其他痕迹。在TwoSigma和LANL跟踪中，对成功作业的CPU请求类似于未成功作业的CPU请求。这种趋势与旧的HPC作业日志中看到的趋势相反[59]，并且由于这些跟踪也是通过SLURM和MOAB收集的，因此我们不希望这种差异归因于跨跟踪定义故障的方式上的语义差异。

观察10：对于Google和TwoSigma跟踪，成功率下降的原因是占用更多CPU时间的作业。 LANL跟踪则相反。

对于我们分析的痕迹，未能可靠记录结果背后的根本原因。没有这些信息，就很难解释和验证结果。例如，我们预计硬件故障是随机事件，其发生频率大致取决于组件的平均故障间隔时间等级。结果，更大和/或更长时间的工作更有可能失败。在图9中，我们根据作业消耗的CPU时间（大小和长度的度量）对作业进行了分组，并显示了每个组的成功率。突出的趋势是，在Google和TwoSigma跟踪中，占用更多CPU时间的作业的成功率会降低，但对于两个LANL群集，成功率都有所提高，而且仍然很高。这可能归因于对LANL更大，更长的工作进行了更仔细的计划和测试，但也可能是由于在跨迹定义成功和失败的方式上存在语义差异。

含义。大多数分析Google跟踪构建失败预测模型中工作失败特征的论文都假设存在我们已经显示的成功率和失败工作资源消耗的趋势。 Chen等。 [9]强调了成功与失败之间资源消耗的差异，El-Sayed等人。 [17]请注意，这是其故障预测模型的第二大影响力预测器（仅次于早期任务故障）。如图9所示，不成功的作业未与其他跟踪中的资源消耗链接。两项研究中都强调的另一个预测因素是重新提交工作，成功提交的工作被重新提交的次数更少。我们确认，即使大多数工作（83-93％）仅提交了一次，该趋势在所有跟踪中都保持一致。对于LANL而言，最后的观察结果并不正确，那就是失败的作业的CPU时间会随着作业的运行时间而增加[17，22]。

7关于多元化和多样性的案例研究

针对多条痕迹进行评估的系统使研究人员能够确定新研究的实际敏感性并证明其普遍性。我们通过对JamaisVu [51]集群调度程序的job runtime2预测程序模块JVuPredict进行案例研究来证明这一点。我们对JVuPredict以及所有已引入的跟踪的评估揭示了逻辑工作名称的预测能力以及工作负载跟踪中一致的用户行为。相反，我们发现很难在提供的信息不足以标识作业重新运行的系统中获得准确的运行时间预测。本节简要介绍了JVuPredict的体系结构（第7.1节）和我们的评估结果（第7.2节）。

7.1 JVuPredict背景

最近的调度程序[12、24、32、51、52]使用有关作业运行时的信息来做出更好的调度决策。对作业运行时的准确了解使调度程序可以在群集中更积极地打包作业[12、18、54]，或延迟高优先级的批处理作业来调度对延迟敏感的作业，而不会超过批处理作业的截止日期。在异构集群中，还可以使用作业运行时的知识来决定是立即在次优的硬件上启动作业还是让其等待，直到首选硬件可用，还是干脆抢占其他作业让它更好它运行[3，52]。这样的调度程序假定提供的大多数运行时信息都是准确的。提供的运行时的准确性很重要，因为这些调度程序仅在合理的误差范围内才很健壮[52]。

获取运行时知识的传统方法通常不如期望用户提供估计那样简单，这是在HPC环境（如LANL）中使用的方法。正如我们在第6节中所看到的那样，用户经常将这些估计用作停止标准（超过它们的工作会被杀死），指定一个太高的值或只是将其固定为默认值。另一种选择是检测具有易于识别的已知结构的作业，以确保进行准确的预测，此方法由Dryad [30]，Jockey [20]和ARIA [53]等系统采用。对于定期工作，基于历史记录的简单预测也可以很好地发挥作用[12，32]。但是，对于没有已知结构或历史记录的合并集群，这些方法仍然不足。

JamaisVu [51]的运行时预测模块JVuPredict旨在使用有关过去作业特征和运行时的历史数据来预测作业提交时的运行时。它与传统方法不同，它尝试检测重复的作业，即使连续运行未声明为重复也是如此。由于仅使用与新提交的工作相关的历史记录的一部分来生成估计值，因此更有效。为此，它使用提交的作业的功能（例如用户ID和作业名称）来构建多个独立的预测变量。然后，根据在历史数据上获得的准确性对这些预测变量进行评估，然后选择最准确的预测变量进行将来的预测。做出预测后，会将新作业添加到历史记录中，并重新计算每个模型的准确性得分。基于更新的分数，选择新的预测变量并重复该过程。

7.2评估结果

最初仅使用Google跟踪评估了JVuPredict。尽管预测并不理想，但Google跟踪下的性能相当不错，其中86％的预测处于实际运行时间的两倍之内。这种准确性水平对于JamaisVu调度程序而言已足够，该调度程序进一步应用了一些技术来减轻此类错误预测的影响。最后，带有Google跟踪的JamaisVu的性能足以与具有理想作业运行时信息的假想调度程序紧密匹配，并且胜过不知道运行时的调度[51]。本节使用新的TwoSigma和LANL迹线重复对JVuPredict的评估。我们成功的标准是达到或超过Google跟踪所达到的预测精度。

可以有效预测工作重复的功能是工作的名称。通常，通过对程序的名称和参数进行哈希处理，或者仅对提供给调度程序的用户定义的可读作业名称进行哈希处理，可以使该字段匿名。对于Google跟踪，由于准确性高，JVuPredict最常选择使用逻辑作业名称字段的预测变量。

图10显示了我们的评估结果。在x轴上，我们绘制了JVuPredict的运行时估算值的预测误差，以作业实际运行时的百分比表示。图中的每个数据点都是一个桶，其值表示最近的十分位数的5％以内。 y轴显示其预测属于每个存储区的作业百分比。高估作业的运行时间比低估更容易容忍，因为它们会使调度程序在调度作业时更加保守。因此，图形右端的上升并不令人担忧。对于Google跟踪，其运行时被低估的作业总数为32％，其中11.7％的低估低于实际运行时的一半。我们将这些数字标记为可接受，因为事实证明，在Google跟踪中JVuPredict的性能非常出色。

尽管逻辑工作名称是一项对Google跟踪效果很好的功能，但我们发现它要么不可用，要么在其他跟踪中不可用。这是因为在生成匿名版本时，同时保持足够的信息以区分重复作业时固有的困难。而是为每个作业分配一个唯一的值，或者从跟踪中完全忽略该字段。我们在本文中介绍的所有痕迹都受到此限制。但是，缺少该字段似乎不会显着影响JVuPredict的性能。 JVuPredict选择作为Mustang和TwoSigma跟踪的作业运行时间的最有效预测器的字段是：提交作业的用户的ID，作业请求的CPU内核数或两者的组合。我们发现，TwoSigma工作负载可实现与Google相同的性能：31％的作业运行时被低估了，而15％的预计运行时数少于实际运行时的50％。但是，Mustang的工作负载更具可预测性，其中38％的预测落在实际运行时的5％之内。尽管如此，仍有16％的作业运行时被低估了一半以上的实际运行时。 TwoSigma和Mustang结果之间的相似性表明，JamaisVu在这些工作负载下也将表现良好。请注意，当省略作业名称时，这些结果将扩展到Google跟踪。

OpenTrinity的性能比其他所有跟踪都差。即使首选的预测变量仍然是用户ID和作业中的CPU内核数，但55％的预测仍被低估了。更糟糕的是，有24％的预测被实际运行时的95％以上低估了。此结果的可能原因是迹线中存在可变性。我们不确定此可变性是由于跟踪的持续时间短，还是由于OpenScience配置期间的工作负载更加不一致。

综上所述，通过对JVuPredict进行多次跟踪评估，可以获得两个见解。首先，我们发现尽管逻辑工作名称在Google跟踪中很好用，但是很难以匿名形式为其他跟踪生成它们，因此它们通常不可用。其次，我们发现，在没有工作名称的情况下，除了OpenTrinity跟踪以外，还有其他字段可以替代它们并提供可比的准确性。具体来说，对于TwoSigma和Mustang跟踪，每个作业的用户ID和CPU核心数似乎表现最佳。

8关于走线长度的重要性

处理痕迹通常会迫使研究人员在解释数据时做出关键假设，以应对丢失的信息。使用或分析轨迹时的一个常见（未写成的）假设是，它足以代表收集轨迹的环境的工作量。同时，Google跟踪仅跨越29天，而我们在本文中研究的其他跟踪则要长3-60倍，就野马而言，甚至覆盖了集群的整个生命周期。由于不确定29天是否足以准确描述集群的工作量，因此我们决定检查TwoSigma和Mustang跟踪中各个29天的代表性如何代表总体工作量。

我们的实验包括将痕迹划分为29天。然后，对于每个这样的月份，我们将单个指标的分布与整个跟踪的总体分布进行比较。我们考虑的指标是：工作规模，持续时间和到达间隔时间。总体而言，对于所有这些指标，我们发现连续几个月的分布差异很大。但是，一个明显的趋势是，在第三年中，野马集群主要由短暂涌入的短期工作主导。

图11通过将每个月不同指标的平均值与整个轨迹的总体平均值进行比较来总结我们的结果。箱形图显示了每月平均作业间隔（左）和工期（右）的分布，并通过跟踪的总体平均值进行了归一化。框线图是标准的Tukey框线图，其中框由第25个百分位数和第75个百分位数框起来，黑线表示中位数，并且晶须定义为分布四分位间距（IQR）的1.5倍，或者是距离最远的数据点不存在异常值（在此处以圆圈显示）。我们看到，野马迹线的各个月份差异很大，而TwoSigma迹线的月份之间的差异较小。更具体地说，给定月份的平均工作间隔可以是TwoSigma跟踪中总体平均值的0.7-2.0x，或者是Mustang跟踪中总体平均值的0.2-24x。在TwoSigma轨迹中，平均作业时间可以在平均作业时间的0.7-1.9倍之间，而在Mustang轨迹中的平均作业时间可以在0.1-6.9倍之间波动。总体而言，我们的结果最终表明，我们的群集工作负载每个月都显示出显着差异。

9相关工作

并行工作负载档案（PWA）[19]包含最大数量的公共HPC跟踪集合。在撰写本文时，1993年至2015年间已收集了38条HPC迹线。野马跟踪在许多方面都是独一无二的：其持续时间是最长的公开跟踪的持续时间长将近两倍，包含的工作量是后者的四倍，并且覆盖了集群的整个生命周期，从而可以进行纵向分析。它的大小也与PWA中最大的群集相似，并且其作业持续时间分布的分布比所有其他HPC迹线都短。就CPU核心数量而言，OpenTrinity跟踪也是对现有跟踪的补充，因为它是在一台机器上收集的，该计算机几乎是具有公开跟踪（Argonne National Lab's Intrepid）的最大超级计算机的两倍。

先前的研究已经研究了专用集群的痕迹，特别是为了表征MapReduce工作负载。任等人。 [42]研究了来自学术Hadoop集群的三条痕迹，以试图确定流行的应用程序样式并表征输入/输出文件的大小，持续时间以及各个MapReduce阶段的频率。与我们研究的Google和TwoSigma群集相比，这些群集处理的流量明显更少。有趣的是，单个作业的间隔时间相当长，超过100秒，这类似于我们的HPC工作负载。同时，大多数作业的持续时间少于8分钟，这近似于Google跟踪中的行为。 Chen等。 [10]研究了Cloudera客户的私有集群和Facebook的Internet服务集群。一方面，他们的私人跟踪不到两个月，而另一方面，他们的Facebook跟踪比Google跟踪更长。尽管如此，以每小时提交的工作量来衡量，流量还是有相似之处。具体来说，Cloudera客户的私有集群每小时处理数百个作业提交，这类似于两个西格玛集群的流量模式，而Facebook每小时处理多达一千个提交，这与Google集群的流量更多相关。这些工作负载之间的多样性进一步强调了研究人员有必要集中精力使用各种痕迹来评估新研究。

其他有关私有群集的研究都集中在虚拟机工作负载上。沉等。 [47]分析来自两个私有集群中的单个VM的监视数据的数据集。他们报告说，虚拟机之间的资源消耗差异很大，但总体群集利用率较低。 Cano等。 [5]研究了来自Nutanix客户2000个集群的遥测数据。遥测收集的频率从几分钟到几天不等，包括存储，CPU测量和维护事件。作者报告说，这些系统中的硬件故障少于以前的文献报道。科尔特斯等。 [11]表征了Microsoft云计算平台Azure上的VM工作负载。他们还报告说集群利用率低，租户工作规模的可变性低。

结论

我们已经引入并分析了来自两个私有群集和两个HPC群集的作业调度程序跟踪。我们公开发布了所有四种踪迹，由于它们的独特特征，我们希望它们引起研究人员的兴趣，其中包括：迄今为止，最长的公共踪迹跨越了一个群集的整个5年寿命，其中一个代表了最大的具有公共踪迹的群集迄今为止，两个最长的私人非学术类集群跟踪已公开。

我们的分析表明，私有集群类似于所研究的HPC工作负载，而不是流行的Google跟踪工作负载，这令人惊讶。这种观察涵盖了工作负载的许多方面：作业大小和持续时间，资源分配，用户行为可变性以及不成功的作业特征。我们还列出了以前的工作，这些工作过分依赖Google跟踪的特征并且可能会受到影响。

最后，我们证明了数据集的多样性和多样性在评估新研究中的重要性。对于作业运行时预测，我们表明使用多个跟踪可以使我们通过预测能力对数据特征进行可靠的排名。我们希望通过发布我们的踪迹，我们将使研究人员能够更好地理解新研究对不同工作负载特征的敏感性。

数据集可用性

可以从ATLAS存储库下载LANL野马，LANL OpenTrinity和两个两个Sigma调度程序日志，可以通过以下网站进行公开访问：www.pdl.cmu.edu/ATLAS

致谢

我们要感谢John Wilkes，Steve Senator，我们的牧羊人Konstantinos Karanasos和匿名审阅者的宝贵意见。 我们还感谢PDL联盟的成员和公司的见解，反馈和支持：阿里巴巴集团，博通，戴尔EMC，Facebook，谷歌，惠普企业，日立，IBM Research，英特尔，美光，微软研究院，MongoDB，NetApp ，甲骨文，Salesforce，三星，希捷科技，两个西格玛，东芝，Veritas和Western Digital。 这项工作的一部分是在美国能源部合同AC52-06NA25396的支持下，在洛斯阿拉莫斯国家实验室的超大规模系统研究中心（USRC）进行的。 该出版物已分配了LANL标识符LA-UR-18-24488。