计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.201200066

混合部署数据中心失效负载分析

蒋从锋[1]　殷继亮[1]　胡海周[1]　闫龙川[2]　张纪林[3]　万　健[4]　仇烨亮[5]
1 杭州电子科技大学计算机学院　杭州 310018
2 国家电网有限公司信息通信分公司　北京 100053
3 杭州电子科技大学网络空间安全学院　杭州 310018
4 浙江科技学院信息与电子工程学院　杭州 310023
5 阿里云计算有限公司　杭州 311121

摘　要　数据中心工作负载混合部署在显著提升云数据中心的资源利用率的同时,也增加了调度的复杂性和作业的失效率。以阿里云发布的数据中心日志数据集 cluster-trace-v2018 为例,从离线批处理工作负载角度出发,详细地分析了不同类型工作负载在成功率和资源利用上的特征。主要发现如下:1)少量类型作业的失效会影响集群整体作业成功率并造成集群资源的浪费;2)伏羲分布式调度系统在任务故障切换执行时间上满足高斯分布,在任务调度延迟方面满足齐夫分布;3)通过分析失败实例在集群节点上的分布,发现集群作业发生失败在空间上具有随机性,且失败的实例很容易再次发生失败,而在时间上集群整体失败率则存在不平衡性;4)以任务实例的失效为基准,计算了集群节点的平均无故障时间,大部分节点的平均无故障时间在 1000 s 左右,小部分节点的任务实例失效率低,其平均无故障时间可达 10000 s 以上。

关键词:混合部署;工作负载特征;分布式调度;失效分析
中图法分类号　TN391

Analysis of Workload Failure in Co-located Data Centers

JIANG Cong-feng[1], YIN Ji-liang[1], HU Hai-zhou[1], YAN Long-chuan[2], ZHANG Ji-lin[3], WAN Jian[4] and QIU Ye-liang[5]
1 School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310018, China
2 State Grid Electrical Information Communication Co., Ltd., Beijing 100053, China
3 School of Cyberspace Security, Hangzhou Dianzi University, Hangzhou 310018, China
4 School of Information and Electronic Engineering, Zhejiang University of Science and Technology, Hangzhou 310023, China
5 Alibaba Cloud Computing Co., Ltd., Hangzhou 311121, China

Abstract　Datacenter workload co-location can greatly increase the resource utilization of cloud data centers, while it also increases the scheduling complexity and job failures. In this paper, the cluster trace dataset from Alibaba Cloud is investigated, and the characteristics of batch workload failure rates and cluster resource utilization are studied. The main contributions and findings of this paper are as follows. First, Only a small portion of specific types of jobs account for the overall cluster failure rate and resource waste due to job failures. Second, the execution time of task failover in the Fuxi distributed scheduler can be quantified as Gaussian distribution, and the task scheduling delay can be quantified as Zipf distribution. Third, Based on the failed instances distribution on cluster nodes, it's found that the job failures randomly occur in the cluster spatially, and the failed jobs are prone to fail again after their failovers. Moreover, job failures occur in the cluster temporally but not evenly distributed in the cluster. Fourth, the mean time between failures of the cluster is calculated according to instance failure data, and the results show that most of the cluster nodes have the mean time between failures values as 1000 seconds, while a few of them have the mean time between failures values as 10000 seconds.

Keywords　Co-located cluster, Workload characteristics, Distributed scheduling, Failure analysis

现代数据中心一般由上千台机器组成,用来提供可靠、高效、可伸缩的网络服务,如电子商务和社交媒体。然而,很多云服务提供商的集群只运行单一的在线服务,产生资源利用率低下的问题。如何有效提高集群整体利用率,成为这类数据中心面临的难题[1]。一种较为有效的方法,是在不影响在线服务运行的情况下,将离线批处理工作负载与在线服务混合部署[2-3],即使用一套物理设备运行两种服务,提高集群整体资源和能源的利用率[4]。

基金项目:国家重点研发计划项目(2017YFB101000);国家自然科学基金面上项目(61972118);浙江省重点研发计划项目(2019C01059)
This work was supported by the National Key Research and Development Pragram of China(2017YFB101000), National Natural Science Foundation of China(61972118) and Zhejiang Key Research and Development Program of China(2019C01059).
通信作者:蒋从锋(cjiang@hdu.edu.cn)

在现代分布式调度系统[5-8]中，混合部署技术的应用会大大增加集群工作负载调度的复杂性，使故障成为一种常态。因此，容错性是衡量分布式系统的一项重要指标。优秀的容错机制[9]需要具备用户透明、自动恢复、高可用等条件[10-11]，对集群调度系统的设计提出非常高的要求。而如今对分布式调度系统的研究主要关注如何通过优化调度提高效率[12]，往往忽略了其失效方面的研究。通过分析集群跟踪数据集中失效工作负载的特征，有助于理解失效负载的工作模式[13]，从而根据其失效特征动态地调整调度决策[14-15]，帮助提高集群负载的成功率。

Ge 等[16]和 Wang 等[17]分别从资源利用和负载特征分布角度对阿里混部集群进行了分析，本文则主要从离线批处理工作负载方面对其失败特征进行分析，主要有以下贡献。

(1)将离线批处理工作负载进行了分类分析(类型通过原始数据集标签获取，经发布者脱敏处理，使用数字代替)，不同类型的工作负载在失败率和资源消耗上有不同的特征。其中，类型 10 的离线批处理工作负载失败率最高，成为影响集群整体成功率的主要因素；类型 1 的失败工作负载资源消耗最大，是造成集群中资源浪费的主要因素。

(2)在集群系统中，task failover 是造成工作负载失败的原因之一，failover 的发生会对运行的离线批处理结果造成一定的影响。task failover 的执行在时间上满足高斯分布。

(3)任务调度系统调度效率较低，有 65.3% 的 task 需要在注册 1 s 以后才开始执行 instance；task 执行延时满足 Zipf 分布的特点。

(4)对 Failed instance 在时间和空间两个方面的分布展开研究。空间上，发现 instance 发生失败具有随机性，失败的 instance 更容易再次发生失败。时间上，特定情况下集群 Failed instance 数量存在短时间内剧增，集群失败率也不平衡的现象。

(5)分析集群 CPU 资源和内存资源的使用概况，使用 MTBF(Mean Time Between Failure)作为集群节点的评价指标。

本文第 1 节介绍相关背景和所用术语；第 2 节讨论离线批处理工作负载的成功率和资源利用；第 3 节分析离线批处理任务 failover 和延迟；第 4 节研究离线批处理实例的分布；第 5 节展示集群资源使用概况，并提出 MTBF；第 6 节介绍相关工作；最后总结全文。

1 数据集和调度系统

1.1 阿里巴巴跟踪数据集

cluster-trace-v2018[18]是阿里巴巴开放集群跟踪项目(Alibaba Open Cluster Trace Program)在 2018 年发布的最新版本集群跟踪数据集。与阿里巴巴 2017 年发布的数据集相比，cluster-trace-v2018 将节点数量扩展到 4 034 台，将取样时间从 24 h 延长到 192 h(8 天)，并首次描述了任务的有向无环图(Directed Acyclic Graph，DAG)信息。

1.2 伏羲调度系统

伏羲是阿里云分布式调度系统，其主要功能是为离线批处理工作负载提供资源调度和任务调度。如图 1 所示，伏羲调度系统主要由 5 个部分组成，各部分的主要功能如下。

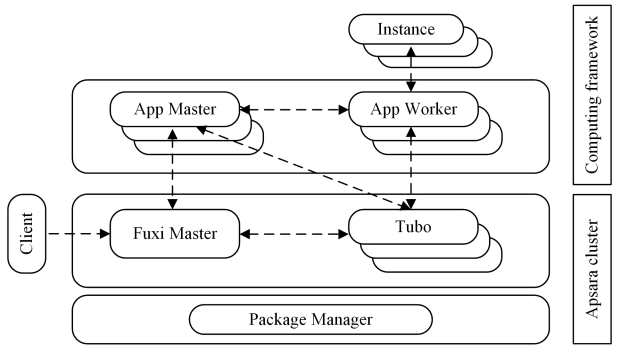


图 1 伏羲系统架构图

Fig. 1 Architecture of fuxi

Fuxi Master：集群主控，用于集群资源的管理与调度；

Tubo：每台机器均部署的一个 Agent，用于管理物理机进程的开启与关闭；

Package Manager：存储包含可执行文件和相关配置的压缩包，压缩包由用户上传；

App Master：由 Fuxi Master 调度并部署在相应机器上，负责一个作业的管理和调度；

App Worker：由 App Master 调度并部署在相应机器上，接收 App Master 发来的 instance，并真正执行用户计算逻辑；同时，需要向 App Master 汇报 instance 执行进度的相关信息。

1.3 符号与术语

本文所用符号与术语及其解释如表 1 所列。

表 1 符号与术语

Table 1 Symbols and terminology

符号与术语	解释
Batch workload	离线工作负载，包含所有 job，task 和 instance，简写为 workload
Batch job	离线作业，组成 workload 的单位，可分为一个或多个 task，简写为 job
Batch task	离线任务，组成 job 的单位，可分为一个或多个 instance，简写为 task
Batch instance	离线实例，组成 task 的单位，不能继续分解，是集群中最小的实际执行单位，简写为 instance

2 离线批处理作业

在阿里集群系统中，离线工作负载由用户提交的作业组成，每个作业即为一个 job，拥有唯一的 ID 标识。

2.1 概况

通过对 job 状态的统计，可获得离线工作负载的总体概况。如表 2 所列，工作负载共有 4 种状态，分别为 Terminated，Failed，Running 和 Waiting 4 种，其中 Terminated 状态占主体。

表 2 离线批处理作业概况统计

Table 2 Offline batch job statistics

Status	*Number*	*Rate*/%
Terminated	4 107 706	97.92
Failed	83 193	1.98
Running	3 975	0.09
Waiting	0	0
Total	4 194 874	100.00

状态为 Terminated 的 job 在整体 job 中所占的比例，可表示整个集群离线工作负载的成功率大小。经统计，集群工作负载整体成功率为 97.92%。

2.2 离线批处理作业失败类型

通过上述分析可知，Running 和 Waiting 状态的 job 占比非常小，所以在后续集群成功率的分析中将不再考虑，而只关注于 Failedjob 的分析。经统计，离线作业中所有 Failed job 的类型分布如表 3 所列。

表 3 Failed job 中各类型的分布

Table 3 Distribution of types in failed job

Type	*Number*	*Rate*/%
1	24113	28.98
3	330	0.40
8	157	0.19
10	58316	70.10
11	273	0.33
12	4	0

虽然集群中 job 种类共有 12 类，但含有 Failed job 的仅有 6 类，而其中占比较高的主要有两类。占比最高的是类型 10，达 70.10%；其次是类型 1，占比 28.98%。

另外，从失败率来看，类型 10 的失败率也明显高于其他类型。类型 10 所有 job 仅占集群总 job 量的 7.7%，而 Failed job 数量却是最多的，导致其失败率高达 18.11%，其余类型失败率均不足 2%。因此，降低类型 10 的失败率能够显著增加系统整体成功率。

2.3 离线批处理作业资源消耗

如果仅从集群工作负载成功率角度进行分析，会忽略系统资源的有效利用。为了进一步探究不同类型 job 的特征，将对含有失败 job 的 6 种作业进行组成结构和持续时间两方面的分析。

2.3.1 作业组成结构

不同类型 job 包含的 instance 数量互不相同，这导致不同 job 运行所产生的开销也不相同。在这里，job 的开销定义为 job 包含的实例数量。考虑到优化集群资源的使用，高开销 job 运行失败的情况需要尽可能地减少。

如图 2 所示，各失败类型 job 的 CDF 图置于子图右上角。其横坐标为 job 所含 instance 数量，单位为个，纵坐标为 job 数量的累积比例。其中，蓝色实线表示该类型 All job（包括成功和失败的总作业）的分布，橙色的虚线表示该类型 Failed job 的分布。为了更清楚地显示两者在 instance 数量上的对比关系，本文将横坐标 instance 数量分成 6 个大小不等的区间，并统计每个区间上 All job 与 Failed job 各自所占比例，作直方图。6 个区间分别为[1,1]，(1,10]，(10,100]，(100,1000]，(1000,10000]，(10000,100000]，…。

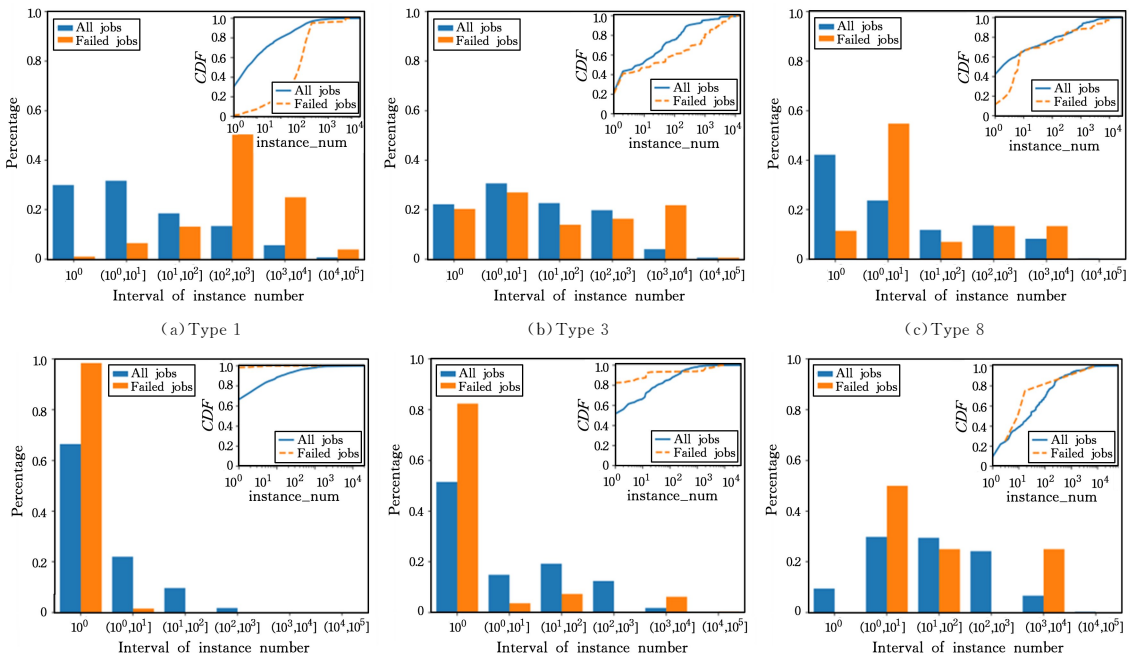


图 2 各类型工作负载中 All job 和 Failed job 各自包含的实例数量对比分布（电子版为彩色）

Fig. 2 Comparison of the number of instances in each type of workload about all job and failed job

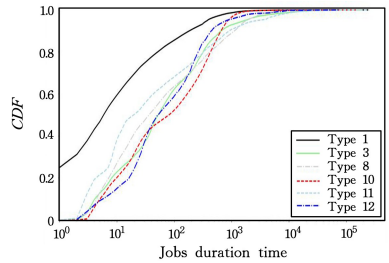
图 2(a)为类型 1 作业的对比分析图，其 All job 主要分布在 instance 数量的前 3 个区间，而 Failed job 主要分布在 instance 数量的后 3 个区间。这说明类型 1 作业在 instance 数量较小时失败率较低，在 instance 数量较大时失败率较高。图 2(d)为类型 10 作业对比分析图，其显示出与图 2(a)截然不同的特征，可以看到其 Failed job 基本都分布在第一个区间，其比例占总 Failed job 数量的 98%以上，说明类型 10 在含有 instance 数量为 1 时，发生失败的概率最大，在 instance 数量较多时反而失败率减小。图 2(e)中的类型 11 也有相似的特征，但其在区间(1000,10000]上失败率也增加。

另外，图 2(b)为类型 3，其 All job 和 Failed job 在多个区间上的分布较为均匀。图 2(c)为类型 8，在(1,10]时，其失败率最高。图 2(f)为类型 12，其 Failed job 仅发生 4 次，其中两次发生在(1,10]上。

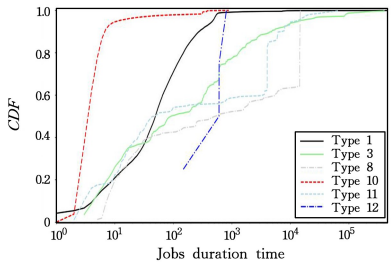
2.3.2 作业持续时间

不同类型 job 在持续时间上进行分析，其结果如图 3 所示，图 3(a)和 3(b)分别代表 Terminated job（成功执行的作业）与 Failed job 随持续时间变化的累积分布图，横坐标为持

续时间，单位为秒。



(a)CDF of terminated job duration time



(b)CDF of failed job duration time

图 3 Terminated job 与 Failed job 持续时间的对比分布
(电子版为彩色)

Fig. 3 Comparison of the duration time of terminated job and failed job

分析 Terminated job 的累计分布图 3(a)，其曲线的变化大部分集中在区间[0,1000]。其中，类型 1 曲线一直处于其余 5 条曲线之上。当持续时间为 1s 时，类型 1 的 job 数量占其总量的 25%左右，而其他类型的数量均不足 0.01%。当持续时间为 10s 时，类型 1 累计到 59%以上，而其余类型均在 38%以下。因此，与其他类型相比，类型 1 中持续时间短的 job 分布较多。

分析 Failed job 的累计分布图 3(b)。类型 10 的曲线至 10s 已累计到 95%左右，说明类型 10 有 95%的 Failed job 持续时间在 10s 及以下。而观察其他曲线，job 的持续时间大部分都分布在 10s 以上。

结合图 3(a)和图 3(b)两图进行分析，观察类型 10 的红(深)色虚线，图 3(a)说明类型 10 有 80%以上的 Terminated job 持续时间大于 10s，图 3(b)说明类型 10 有 95%以上的 Failed job 持续时间不足 10s。由此可知类型 10 中 Failed job 往往会比 Terminated job 更早结束。对比图 3(a)和图 3(b)中类型 1 的黑(深)色实线。在持续时间点 1,10 和 100 时，图 3(a)中类型 1 的累计比例分别为 25%、59%和 86%，而图 3(b)中相应比例为 4%,21%和 71%，图 3(a)中的比例均大于图3(b)。由此可知，与类型 1 的 Terminated job 对比，其 Failed jobs 往往持续时间更长。换句话说，类型 1 的 Failed job 往往会比 Terminated job 更晚结束。

2.4 成功率 vs 资源浪费

下面讨论 Failed job 对集群工作负载成功率和资源浪费两个方面的影响。因上文对成功率有过详细分析，在此不再赘述。而资源浪费方面，可从集群 CPU 和内存两个维度进行考虑。

2.4.1 集群资源浪费

如表 4 所列，CPU 代表各类 Failed job 浪费的集群 CPU 资源情况。类型 1 占到浪费总量的 88.95%，类型 3 和类型 11 其次，依次为 5.66%和 3.53%。Mem 代表各类 Failed job 浪费的集群内存资源情况。类型 1 依然占比最高，达 92.66%，类型 3 占 4.15%，其余 4 种类型浪费内存总和仅 3.19%。因此，由于负载失败而造成的集群资源浪费，其主要因素在于类型 1。

表 4 Failed job 各类型资源浪费的对比

Table 4 Comparison of various types of resource waste in failed job

(单位：%)

Type	CPU	Mem
1	88.95	92.66
3	5.66	4.15
8	1.76	2.68
10	0.02	0
11	3.53	0.45
12	0.09	0.06

2.4.2 集群成功率和资源浪费的评估

结合表 3 和表 4，可以评估出各类 Failed job 在集群失败率、CPU 资源浪费和内存资源浪费 3 个方面的影响程度。为了方便表示，以下根据不同的占比，设置对应的影响等级。其中，0～25%，25%～50%，50%～75%，75%～100%分别对应 low,medium,high 和 very high。

如表 5 所列，类型 1 成为影响集群资源浪费的最大因素，类型 10 成为影响集群成功率的最大因素。而其余类型由于本身 job 数量较小或 Failed 较少，对集群的影响程度均较低。

表 5 Failed job 在集群成功率和资源浪费方面的评估

Table 5 Evaluation of cluster success rate and resource waste for failed job

Job 类型	对集群成功率的影响	对集群 CPU 资源浪费的影响	对集群内存资源浪费的影响
1	Medium	Very high	Very high
3	Low	Low	Low
8	Low	Low	Low
10	High	Low	Low
11	Low	Low	Low
12	Low	Low	Low

3 离线批处理任务

在阿里集群系统中，job 会被分解成一个或多个 task，并通过 task 服务执行 MapReduce 计算架构，实现 DAG 调度。task 服务的 failover 可能会对整个 job 的完成产生影响，故以下分析发生在阿里生产集群中的 taskfailover 特征。

3.1 Taskfailover 对成功率的影响

根据 task 是否发生过 failover，可将 task 分为 Failover task 与 No failover task 两类，它们各自在最终运行状态上的分布情况如表 6 所列。

表 6 发生和未发生 failover 的 task 最终状态对比

Table 6 Final state comparison of the task with failover and without Failover

(单位：%)

Type	Status		
	Terminated	Failed	Running
Failover task	98.20	0.90	0.90
No failover task	98.88	0.53	0.58

结果显示 Failover task 成功率比 No failover task 低 0.68%，这表明在集群中 task failover 也是造成工作负载失

败的原因之一，任务调度服务器 failover 的发生会对正在运行的离线批处理任务运行结果造成一定的影响。

3.2 Failover task 的时间分布特征

发生 Failover 的 task 均会经历再注册的过程，即依靠上层调度系统重新分配资源，并重启 task 服务。从 task 首次执行到完成再注册，这个过程所花费的时间关系到整个 job 完成效率，故对其时间分布进行分析。分析结果显示，Failover task 再注册时间满足高斯分布规律，其平均花费时间在 79.58 s 左右。

如图 4 所示，由于 task 第一次注册执行的时间会被再注册覆盖而无法统计，所以这边的横轴代表的是原 task 中第一个 instance 开始执行到发生 failover 后再注册的时间间隔，单位为秒。纵坐标为属于该时间间隔下的 task 数量在所有 Failovertask 中所占比例。考虑到时间间隔跨度较大，本文仅截取了 0 到 160 s 的区间作图。如红色虚线所示，用高斯函数进行了拟合，其拟合系数 $\mu=79.58$，$\sigma=10.02$，其决定系数高达 0.96。

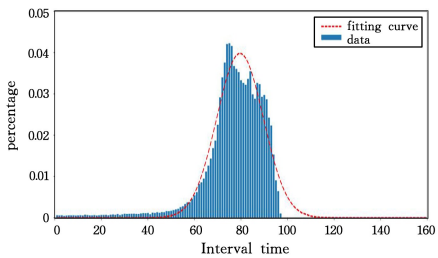


图 4 Failover Task 中第一个 instance 开始时间和 Task 再注册时间之间的间隔时间分布(电子版为彩色)

Fig. 4 Distribution of interval time between first instance start time and Failover Task re-registration time

这里，决定系数(coefficient of determination)作为评判拟合结果与观测值之间拟合程度的度量参数，一般大于 0.8 即可被认为具有较高拟合度。

3.3 任务调度延迟特征

对 Nofailover task，其 task 首次注册到第一个 instance 实际执行，也需要系统进行资源的调度。该过程花费的时间即任务调度延迟，同样关系到整个 job 完成效率，故对该时间分布也进行了分析。

如表 7 所列，Nofailover task 排名前十的延迟时间分布在 0 到 10 s。其中，仅有 44.7% 将延时时间控制在 1 s 以内，有 65.3% 在注册后需等待 1 s 以上才能真正开始 instance 执行。

表 7 根据 task 延迟时间出现频率的前十名

Table 7 Top ten task delayed time ranked by frequency

Rank	Delay time/s	Frequency/%
1	(0,1]	44.7
2	(3,4]	11.4
3	(2,3]	10.1
4	(1,2]	5.4
5	(4,5]	3.1
6	(6,7]	2.0
7	(5,6]	2.0
8	(7,8]	1.4
9	(8,9]	1.1
10	(9,10]	1.0

如图 5 所示，横坐标为延迟区间排名，纵坐标为 task 的数量。在对数坐标系下，每个红点代表对应延迟区间排名下的 task 数量。最终的统计结果显示其满足 Zipf 分布的特点，其拟合系数 $C=6.22\times10^6$，$\sigma=1.65$。图中蓝色虚线为拟合曲线，决定系数高达 0.99。

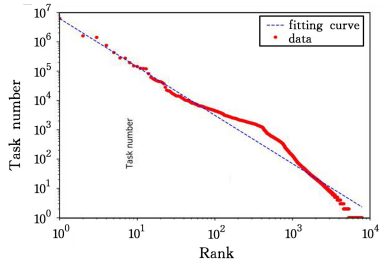


图 5 Task 延迟时间的 Zipf 分布(电子版为彩色)

Fig. 5 Zipf distribution of tasks delay time

Task 延迟时间在一定程度上也反映集群资源的供求关系。该集群有 65.3% 的 task 在部署后无法立即执行，侧面反映负载所需资源量高于集群现有资源，有较多请求发生了排队等待的情况。

4 离线批处理实例

4.1 离线批处理实例

Instance 是数据集中最小粒度的记录信息，也是集群负载的最小执行单位。对于 Failed instance，调度系统将执行 instance 重试机制。

表 8 统计了 instance 执行次数和失败率的关系。其中，Seq 为实例被调度的次数，Total 为每次调度的实例总量，Failed 为每次调度后仍然失败的实例数量，Failed rate 为实例失败率。对比不同 Seq 下的失败率可知，失败过的 instance 重新调度后，更容易再次发生失败。

表 8 Instance 的执行次数和失败率

Table 8 Execution times and failed rate of instance

Seq	Total	Failed	Failed rate/%
1	1 350 395 004	1 739 788	0.13
2	567 063	224 159	39.53
3	200 505	122 384	61.04
4	70 370	28 181	40.05
5	11 244	10 013	89.05
6	9 865	7 851	79.58
7	1 256	470	37.42
8	275	111	40.36
9	111	49	44.14
10	49	34	69.39
11	33	10	30.30

4.2 失败实例的空间分布状况

空间上，Failed instance 的分布具有随机性。如图 6 所示，集群共有节点 4 034 个，Failed instance 的数量分布较为均衡，有 99.97% 的节点均出现过 Failed instance，节点发生失败实例是一种常态。

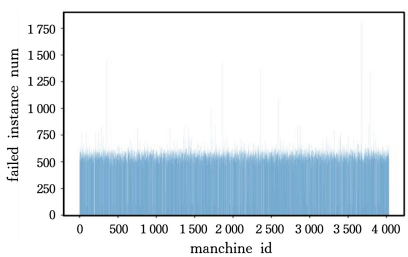


图 6 失败实例在节点上的分布

Fig. 6 Distribution of failed instances on nodes

4.3 失败实例的时间分布状况

时间上，Failed instance 的分布是不平衡的。如图 7 所示，(a)图代表 Failed instance 数量随时间分布的 CDF 图，0～24 h 集群中几乎不存在 Failed instance，但 84～96 h 有明显的增长；(b)图为 Failed instance 数量随时间分布的柱形图，可清楚地看到，在 54h 时 Failed instance 数量最多，而在 84～96 h，Failed instance 数量最密集。

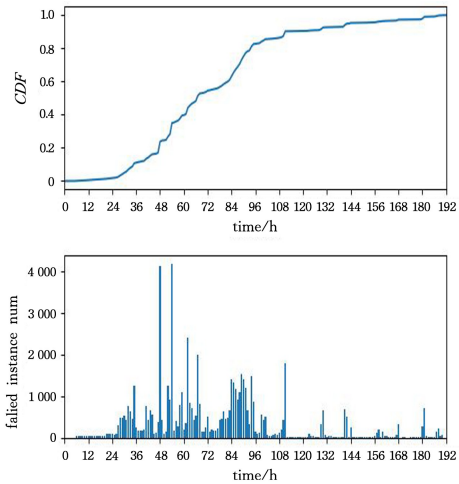


图 7 失败实例数量随时间的分布状况

Fig. 7 The number of failed instances is distributed over time

集群节点 instance 失效时间分布如图 8 所示。其中 x 轴为时间戳，y 轴为节点 id，图中红色区域代表节点发生 Failed instance。0～80 000，400 000～430 000 左右出现的空白，代表了对应时间段集群整体发生 instance 失败的概率显著降低了。

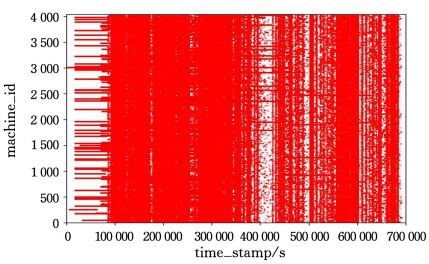


图 8 失效时间分布(电子版为彩色)

Fig. 8 Failure time distribution

5 集群节点分析

5.1 集群资源使用概况

阿里集群同时运行在线和离线两种工作负载。如图 9 所示，集群整体的 CPU 和内存利用率随时间具有不同的变化规律，CPU 利用率呈明显的周期性变化，而内存利用率则相对平稳且远高于 CPU。

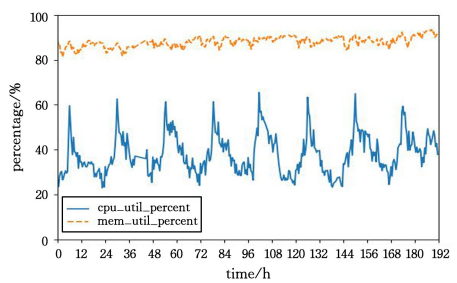


图 9 集群资源使用概况

Fig. 9 Overview of cluster resource usage

5.2 集群节点的评价指标 MTBF

简单调度系统对节点的选择往往只基于节点资源的剩余量。而更优秀的策略应该考虑分析节点的历史记录，获得每个节点关于 instance 运行情况的评价指标。当调度系统需要部署 instance 时，从候选节点中选择该评价指标最高的节点进行部署，从而有效提高 instance 部署和运行的成功率。

为了获得描述集群节点 instance 运行情况的评价指标，这里借鉴了 MTBF(Mean Time Between Failure)的概念。集群节点 MTBF 的计算方式如式(1)所示，其中 downtime 为 instance 结束时间，uptime 为 instance 开始时间，downtime-uptime 即为该 instance 运行总时间。式(1)的分子是节点所有 instance 运行时间的总和，分母为节点运行 Failed instance 的数量。

$$MTBF=\frac{\sum(downtime-uptime)}{Failed\ times} \tag{1}$$

集群节点的 MTBF 值具有较大的波动性，其分布如图 10 所示，大部分节点 MTBF 在 1 000 s 左右，小部分节点 MTBF 高达 10 000 s 以上。在实际调度过程中，若集群资源相对充裕，调度系统应将 instance 调度到这些高 MTBF 节点上，以此来降低部署失败的概率。

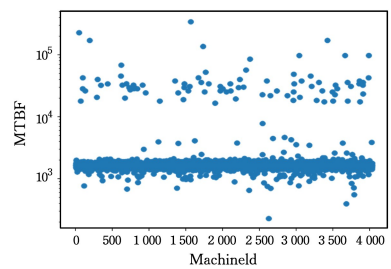


图 10 MTBF 在不同节点上的分布

Fig. 10 Distribution of MTBF on different nodes

6 相关工作

谷歌在 2011 年发布了对 12 500 台异构节点 29 天的跟踪数据集，Reiss 等[19]分析发现异质性和动态是谷歌数据中心两个重要的因素。阿里巴巴在 2017 年发布 1 313 台节点 24 h 的混部跟踪数据集，许多团队对其进行了研究。深圳先进技术研究院(Shenzhen Institutes of Advanced Technology)Lu 等[1,19]，率先发布 4 个不平衡结论，分别是空间不平衡、时间不平衡、工作负载对多维资源的使用不平衡及在线负载和离线负载在资源请求和持续时间上不平衡，认为这将导致严重的资源浪费和低集群利用率。2018 年，Liu 等[1,20]关注资源利用率(IRU)和在线服务质量(QoS)的权衡，将阿里与谷歌数据集进行对比。而 Chen 等人[1,21]使用可视化技术和聚类方法，根据工作负载的资源利用特征进行分组。与此同时，乔治梅森大学(George Mason University)的 Cheng 等[23-24]通过分析工作负载的特征，发现调度系统对在线和离线工作负载的调度会做出看似独立的决策。此外，武汉科技大学 Deng 等[1,24]揭示了有关资源利用的重要特征。2019 年，杭州电子科技大学的 Jiang 等[1,25]分析了生产云的运行特征，并在提高资源利用率和故障恢复设计方面提出重要的见解。

分布式调度系统是实现海量数据高效计算的关键技术，基于 MapReduce[1,26]的核心思想，国内外大量研究者先后提出了不同的系统架构方案[1,27]。Hadoop MR[29-30]提出经典

主从架构,主控服务器负责集群所有节点的调度,但其在规模扩展方面存在重要缺陷。2011 年,YARN[1,30]在 Hadoop MR 的基础上进行发展,将资源调度与工作负载调度作了区分,使之可以支撑起更大的集群规模,但其仅支持内存维度的资源调度。Mesos[1,31]实现资源与工作负载调度的分离,添加了更多样化的计算框架支持,但存在资源请求协议设计不精确和不支持资源抢占的问题。Omega[1,32]则是一个共享状态调度程序,它兼顾分布式和可伸缩两方面需求,极大地提高了系统并发能力。但其模型只处于模拟阶段,尚未在实际生产环境中应用。Sparrow[1,33]也是针对分布式调度设计的程序,其显著提高了短任务的并行作业性能,旨在减少延迟。

在上述研究的基础上,本文分析 2018 年阿里巴巴最新发布的集群跟踪数据集,FuXi[1,34]为阿里云自主研发的离线分布式调度系统。

结束语 在生产云中实现离线和在线工作负载混合部署,极大增加了集群资源利用率,却给工作负载的调度带来了更高复杂性。上文从离线批处理工作负载角度出发,分析不同类型负载失败率和资源利用的特征,找到影响集群整体成功率和资源浪费的主要因素。此外,研究了 FuXi 调度系统在 taskfailover 和 task 调度效率方面的特征,并将失败实例与集群节点相结合,讨论 Failed instance 在时间与空间上的分布。最后,提出 MTBF 作为集群节点失效相关的评价指标。

致谢 阿里云公司调度团队徐国耀、丁海洋、吕齐、李治等工程师我们的研究工作提供了富有启发的讨论和交流,特此致谢!

参 考 文 献

[1] XU G,XU C,JIANG S. Prophet:Scheduling executors with time-varying resource demands on data-parallel computation frameworks [C]// 2016 IEEE International Conference on Autonomic Computing (ICAC). Piscataway, NJ: IEEE, 2016: 45-54.

[2] YAN Y,GAO Y,CHEN Y,et al. Tr-spark: Transient computing for big data analytics [C]// Proceedings of the Seventh ACM Symposium on Cloud Computing. New York,NY: ACM, 2016:484-496.

[3] JYOTHI S A,CURINO C,MENACHE I,et al. Morpheus: Towards automated slos for enterprise clusters [C]// 12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16). Berkeley,CA:USENIX,2016:117-134.

[4] RAJAN K,KAKADIA D,CURINOC,et al. PerfOrator:eloquent performance models for Resource Optimization [C]// Proceedings of the Seventh ACM Symposium on Cloud Computing. New York,NY:ACM,2016:415-427.

[5] CHEN W,RAO J,ZHOU X. Preemptive,low latency datacenter scheduling via lightweight virtualization [C]// 2017 {USENIX} Annual Technical Conference ({USENIX}{ATC} 17). Berkeley,CA:USENIX,2017:251-263.

[6] CORTEZ E,BONDE A,MUZIO A,et al. Resource central: Understanding and predicting workloads for improved resource management in large cloud platforms [C]// Proceedings of the 26th Symposium on Operating Systems Principles. New York, NY:ACM,2017:153-167.

[7] JIANG C,WANG Y,OU D,et al. EASE:Energy efficiency and proportionality aware virtual machine scheduling [C]// 2018 30th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD). Piscataway, NJ: IEEE,2018:65-68.

[8] QIU Y,JIANG C,WANG Y,et al. Energy aware virtual machine scheduling in data centers[J]. Energies,2019,12(4):646.

[9] GARRAGHAN P,YANG R,WEN Z,et al. Emergent Failures: Rethinking Cloud Reliability at Scale[J]. IEEE Cloud Computing,2018,5(5):12-21.

[10] PAN A,WANG X,LI H. Conceptual Modeling on Tencent's Distributed Database Systems [C]// International Conference on Conceptual Modeling. Cham:Springer,2018:12-24.

[11] KAUR H,CHHABRA A. Fault-aware advance reservation scheduling in heterogeneous computing systems[J]. International Journal of Applied Engineering Research,2018,13(11):9636-9645.

[12] CHEN W,PI A,WANG S,et al. Characterizing scheduling delay for low-latency data analytics workloads [C]// 2018 IEEE International Parallel and Distributed Processing Symposium (IPDPS). IEEE,2018:630-639.

[13] CAO X,ZHONG Y,ZHOU Y,et al. Interactive temporal recurrent convolution network for traffic prediction in data centers [J]. IEEE Access,2018,6:5276-5289.

[14] JIANG C,HUANG W,REN Z,et al. Towards building a scalable data analytics system on clouds:An early experience on alicloud [C]// 2018 IEEE 11th International Conference on Cloud Computing (CLOUD). Piscataway,NJ:IEEE,2018:891-895.

[15] MAZUMDAR S,KUMAR A S. Statistical analysis of a data center resource usage patterns:A case study [C]// Proceedings of the International Conference on Computing and Communication Systems. Singapore:Springer,2018:767-779.

[16] GE Z F,W J W,JIANG C F,et al. Analysis of resource utilization of co-located clusters[J]. Chinese Journal of Computers, 2020,43(6):1103-1122.

[17] WANG J W,GE Z F,JIANG C F,et al. Load characteristics and task scheduling optimization analysis of co-located data center [J]. Computer Engineering and Science,2020,42(1):8-17.

[18] GitHub. The Alibaba ClusterData 2 0 1 8 trace data [EB/OL]. (2018-12-13) [2019-04-30]. https://github. com /alibaba/clusterdata.

[19] REISS C,TUMANOV A,GANGER G R,et al. Towards understanding heterogeneous clouds at scale: Google trace analysis [R]. Intel Science and Technology Center for Cloud Computing, 2012.

[20] LU C,YE K,XU G,et al. Imbalance in the cloud:An analysis on alibaba cluster trace [C]// 2017 IEEE International Conference on Big Data (Big Data). Piscataway, NJ: IEEE, 2017: 2884-2892.

[21] LIU Q,YU Z. The elasticity and plasticity in semi-containerized co-locating cloud workload: A view from Alibaba trace [C]// Proceedings of the ACM Symposium on Cloud Computing. New York,NY:ACM,2018:347-360.

[22] CHEN W,YE K,WANG Y,et al. How does the workload look like in production cloud? Analysis and clustering of workloads on Alibaba cluster trace [C]// 2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS). Piscataway,NJ:IEEE,2018:102-109.

(下转第 264 页)

于时间限制，还未能在特征提取分析中分析山体的加速度的运动特征量，所以为了进一步提高蠕变灾害预警准确度和应用范围，我们后续还会进一步研究加速度在山体蠕变运动中的变化规律。

参 考 文 献

[1] ELLIOTT D K, CHRISTOPHER H. Understanding GPS: Principles and Applications, Second Edition[M]. The United States: Artech House Publishers, 2006: 14.

[2] WANG M, CAI H, PAN Z. BDS/GPS relative positioning for long baseline with undifferenced observations[J]. Advances in Space Research, 2015, 55(1): 113-124.

[3] WANG X J, TAO Z P, PENG H L. Introduction of BeiDou high precision location service platform and its application[C]// China Satellite Navigation Conference(CSNC 2017). Academic Exchange Center of China Satellite Navigation System Management Office: Organizing Committee of China Satellite Navigation Annual Conference, 2017: 5.

[4] LIU R L. Research on the Application of DGPS Based on BDS [J]. Telecom Power Technology, 2020, 37(8): 4-6.

[5] ZHANG H Y. A system design of landslide early warning based on BeiDou satellite navigation system[J]. Electronic Test, 2017, (24): 11, 13.

[6] YANG R G. Stability theory of rock and soil structure and Landslide prediction [M]. Geology Press, 2010.

[7] SUN X Y, JIN Y S, CHEN B, et al. Analysis on practices of dual-coupling aging curve and prediction criterion for landslide[J]. Architecture Technology, 2015, 46(10): 886-890.

[8] ZHANG M S, ZHANG X Z, CHEN B P, et al. A new self-adaptive Kalman filtering method for GPS kinematic positioning[J]. Journal of Central South University, 2003, 34(5): 543-546.

LIU Ya-chen, born in 1997, postgraduate. Her main research interests include BeiDou high-accuracy position and navigation, and IoT software and system.

(上接第 231 页)

[23] CHENG Y, CHAI Z, ANWAR A. Characterizing co-located datacenter workloads: An Alibaba case study [C]// 9th ACM SIGOPS Asia-Pacific Workshop on Systems (APSys 2018). New York, NY: ACM, 2018: 12: 1-12: 3.

[24] CHENG Y, ANWAR A, DUAN X. Analyzing Alibaba's co-located datacenter workloads [C]// 2018 IEEE International Conference on Big Data (Big Data). Piscataway, NJ: IEEE, 2018: 292-297.

[25] DENG L, REN Y L, XU F, et al. Resource utilization analysis of Alibaba cloud [C]// International Conference on Intelligent Computing. Berlin, German: Springer, 2018: 183-194.

[26] JIANG C, HAN G, LIN J, et al. Characteristics of Co-allocated Online Services and Batch Jobs in Internet Data Centers: A Case Study from Alibaba Cloud[J]. IEEE Access, 2019, 7: 22495-22508.

[27] DEAN J, GHEMAWATS. MapReduce: simplified data processing on large clusters[J]. Communications of the ACM, 2008, 51(1): 107-113.

[28] MONU M, PALS. A Review on Storage and Large-Scale Processing of Data-Sets Using Map Reduce, YARN, SPARK, AVRO, MongoDB[C]// Proceedings of International Conference on Sustainable Computing in Science, Technology and Management. Jaipur, India: SSRN, 2019: 1-8.

[29] PRATT B, HOWBERT J J, TASMAN N I, et al. MR-tandem: parallel X! tandem using hadoop MapReduce on amazon Web services[J]. Bioinformatics, 2011, 28(1): 136-137.

[30] DEDE E, GOVINDARAJU M, GUNTERD, et al. Performance evaluation of a mongodb and hadoop platform for scientific data analysis [C]// Proceedings of the 4th ACM workshop on Scientific cloud computing. New York, NY: ACM, 2013: 13-20.

[31] VAVILAPALLI V K, MURTHY A C, DOUGLAS C, et al. Apache Hadoop yarn: Yet another resource negotiator [C]// Proceedings of the 4th annual Symposium on Cloud Computing. New York, NY: ACM, 2013: 5.

[32] HINDMAN B, KONWINSKI A, ZAHARIA M, et al. Mesos: A platform for fine-grained resource sharing in the data center [C]// NSDI'11. Berkeley, CA: USENTX, 2011: 295-308.

[33] SCHWARZKOPF M, KONWINSKI A, ABD-EL-MALEK M, et al. Omega: flexible, scalable schedulers for large compute clusters [C]// Proceedings of the 8th ACM European Conference on Computer Systems. New York, NY: ACM, 2013: 351-364.

[34] OUSTERHOUT K, WENDELL P, ZAHARIA M, et al. Sparrow: distributed, low latency scheduling [C]// Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles. New York, NY: ACM, 2013: 69-84.

[35] ZHANG Z, LI C, TAO Y, et al. Fuxi: a Fault-Tolerant Resource Management and Job Scheduling System at Internet Scale[J]. Proceedings of the VLDB Endowment, 2014, 7(13): 1393-1404.

JIANG Cong-feng, born in 1980, Ph.D, professor, Ph.D. supervisor, is a member of China Computer Federation. His main research interests include cloud computing, system optimization and performance evaluation.