2019.10.14

Deadline-Aware Fair Scheduling for Multi-Tenant Crowd-Powered Systems

多租户、群众驱动（众包）系统的截止时间感知公平调度

解决了当一个worker可用时，把哪一个HIT分配给他的问题

优化任务之间的公平性和任务的deadline感知

(ACM Transactions on Social Computing, 2019)

Abstract

问题：

执行时间和按时完成不能得到保证：目前，这些系统对执行时间提供了有限的保证，这对许多应用程序来说是有问题的。由于众包平台特有的因素，如workers可用性和并发任务等，甚至无法保证按时完成。

之前的工作：

重塑了交互机制，push的众包模型（workers接收任务，而不是选择任务）

本次的工作：

1. 设计了一个通用的调度算法，支持公平调度和deadline
2. 补充概念验证实验，通过一组广泛的仿真来分析所提出调度算法的特性

实验结果表明，在考虑人为因素的情况下，微任务调度可以实现best-effort batches的公平性，提高production batches

1 Introduction

众包的背景

pull-based model（workers在batches列表进行浏览和选择）存在的问题：通过对AMT收集的日志进行分析，并讨论使用众包的选择服务的操作模型

动机1：batch的饥饿现象

Human Intelligence Task (HIT)

把batch分为4类：tiny, small, medium, large

即使绝大多数的运行batches都是tiny，但是large batches仍然支配着AMT的吞吐量

large batches首先以更快的速度完成，然后逐渐失去动力，因为最后剩下的任务需要更长的时间才能完成。

batch完成时间取决于它的规模和HIT价格

large batches有着最高的吞吐量，因为有高的reward

动机2：Production和Best-effort Batches

Production对响应时间的要求比较高

Best-effort Batches不一定有时间限制，但是要有稳定的进展。

在一个拉式的众包平台上，这两类执行要求不同的batch并行运行，争夺workers池中的资源。

贡献：

基于push的架构，自动将任务分配给workers。允许我们对任务的执行进行排序，并根据humans资源的不可预测性应用调度策略。

两个问题：

1. 我们能否将调度算法应用于在线人力资源？
2. 成功地在众包平台上调度微任务需要哪些调整？

在我们的经验评估中，我们改变了crowd的大小、顺序、优先级以及batches的规模。还考虑了workers的特性，如上下文切换和工作连续性的影响，设计了我们的群体感知调度算法。

实验设置包括：

1. 固定数量的workers
2. 不同数量的workers和工作负载，来自于一个真实的众包平台
3. 一个具有真实参数的仿真，证明了我们的调度算法能够同时处理的best-effort batches和production batches，并有着提高执行速度的要求

评估结果：

使用微任务众包的调度方法可以最小化batches的总体延迟，优雅地平衡工作负载，并显著提高工人的生产率（采用平均执行时间度量）。

综上所述，本文的主要贡献有:

1. 支持任务调度的众包架构
2. 将未决任务分配给可用工人的调度系统;
3. 一个理论调度框架和算法，优化任务之间的公平性和任务的deadline感知;
4. 一个延迟调度机制，减少了上下文切换对工人的影响，当暴露在一系列异构任务;
5. 以真实的crowd workers为对象进行了一系列实验，研究影响human任务调度的因素;
6. 使用crowd模拟器进行了广泛的经验评估，以分析提出的算法的性质。

5 RELATEDWORK

我们的重点是提高众包效率，重用现有的例如AMT的众包平台，而不是部署专门的众包平台。

两种worker资源：

retainer pools of workers（worker积极地等待任务过来）和on-demand workers（具有一定自主性）

主动学习方法

我们同时考虑batch和worker双方（batch的优先级和worker的人力因素）

我们侧重于确保竞争的batch，都能恰当的完成

我们侧重于将任务分配给单个worker而不是team，以平衡负载

我们注重优化执行时间，以满足deadline

中断任务的连续性（切换到不同类型的任务）会降低crowd workers的效率

我们设计了一个worker-conscious的调度方法，按顺序向crowd worker提供相同类型的任务，以避免上下文切换的负面影响

在我们的工作中，我们的目标是通过有意识地给员工安排任务来减少任务切换。

不是让多租户在云中运行计算作业，而是让多个用户(或系统)在AMT上启动众包微任务。

我们不支持软期限或硬期限，这主要是由于劳动力可用性的不可预测性。

不考虑worker技能和需求的匹配，注重通过考虑batch优先级和平衡负载来优化执行时间。

相关工作小结：

5.2小节：着眼于如何使用实时可用的工作池和主动学习方法等技术来加速任务执行。相反，我们着眼于可以处理不同batch优先级的调度技术。

5.3小节：考虑了worker技能与任务需求的匹配问题。与它们相比，而不是技能/需求匹配，我们关注batch优先级来调度任务并平衡工作负载来优化它们的执行。

5.4节：着眼于影响高效任务完成的人为因素，在不同类型的任务之间切换是不好的。

7 CONCLUSIONS

在这篇文章中，我们扩展了我们之前关于调度众包任务的工作，通过支持有deadline的best-effort batches和production batches。我们探索了众包任务调度的解决方案，既支持公平性，又支持deadline。我们推导出公平调度算法的变种，其中的deadline和未完成的任务优先于production batches。

2 THE ARCHITECTURE OF MULTI-TENANT CROWD-POWERED SYSTEMS

需求分析

1. 可扩展性：动态适应当前crowd的可用性，当工作需求增加时可以通过伸缩扩展进行实时调度
2. 公平性：系统用户之间的公平性，large和small batch受到相同的对待和处理
3. Deadline：优先满足deadline
4. 请求的优先级：在多租户系统中，一些请求的优先级高于其他的请求
5. 考虑worker：考虑人力因素，在最小化上下文切换和调度的公平性之间权衡

基础调度算法都是非抢占的，包括：

1. First-in, First-out (FIFO).
2. Shortest Job First (SJF).
3. Earliest Deadline First (EDF).
4. Round Robin (RR).