Fairness Issues in Crowdsourcing

Jiwei Huang, Songyuan Li
State Key Laboratory of Networking and Switching Technology
huangjw@bupt.edu.cn

Download PPT File

Cite a Historical Allusion !!!

- Chess-Playing Machine Turk (1770)
  - Appeared to beat its human opponents with confident ease
  - Actually, it's A HOAX !!!
    - A rolling roster of human chess masters hided inside the machine, controlling chess moves and making it look like the 'machine' was outsmarting humans.

Cite a Recent Allusion !!!

- AlphaGo(WINNER) V.S. Lee Sedol (2016)
  - Human professional Go player was beat by machines for the first time.
  - We should acknowledge the improvement of AI technology.
  - Nevertheless, success of AlphaGo is due to
    - **Deep Learning Algorithm**, imitating the structure of ***human*** *brain neurons*
    - Large Volumes of Tagged Data, completed by *substantial* ***human*** *workers*

Def. Crowdsourcing

- The act of a company/institution taking a function once performed by employees and outsourcing it to an *undefined* (and generally *large*) network of people in the form of a *flexible open* call

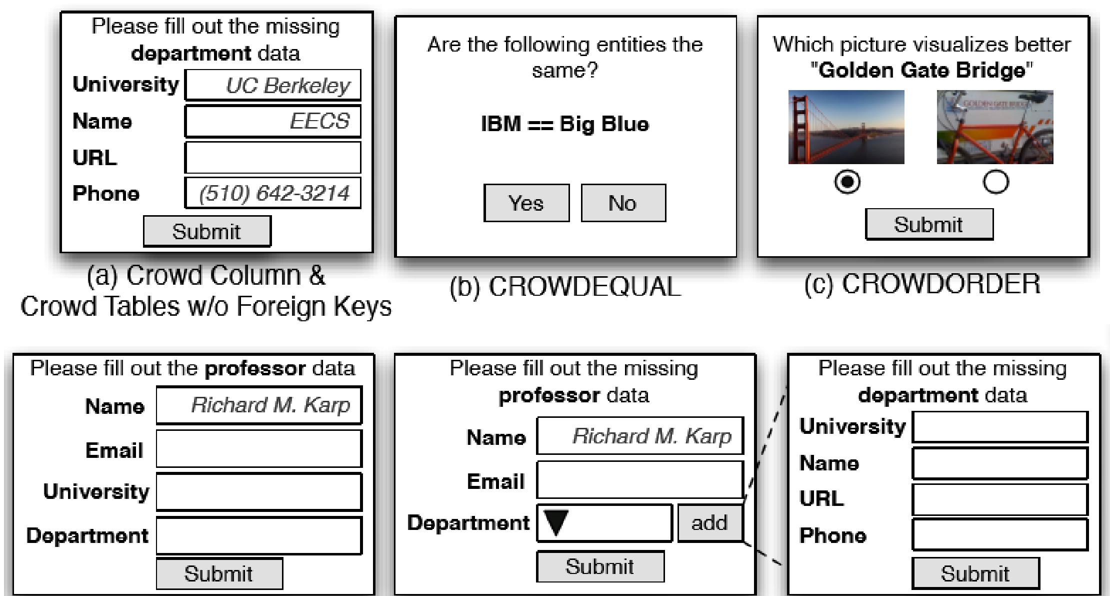
Role. Crowdsourcing

- Worker Agent (WA)
  - Gain *economic* rewards, social recognition, self-esteem, or the development of skills.
- Crowdsourcer
  - Obtain and use what the user has brought to the venture, whose form will depend on the type of activity undertaken

Mutual Benefit !!!

Feature. Crowdsourcing Tasks

- Difficult to be precisely learned for *machine*
- *BUT:* Easy to understand for *human beings*



(d) Foreign Key(normalized) (e) Foreign Key (denormalized)

Application Examples

- Amazon Mechanical Turk https://www.mturk.com/
- CrowdDB https://amplab.cs.berkeley.edu/projects/crowddb-answering-queries-with-crowdsourcing/
- Microsoft's Universal Human Relevance System Clickworker， Lionbridge， Appen， ISoftStone
- Google Ewok Project https://github.com/jacobhall/Project-Ewok-2.0 (To Be Verified)

Our Research Proposal

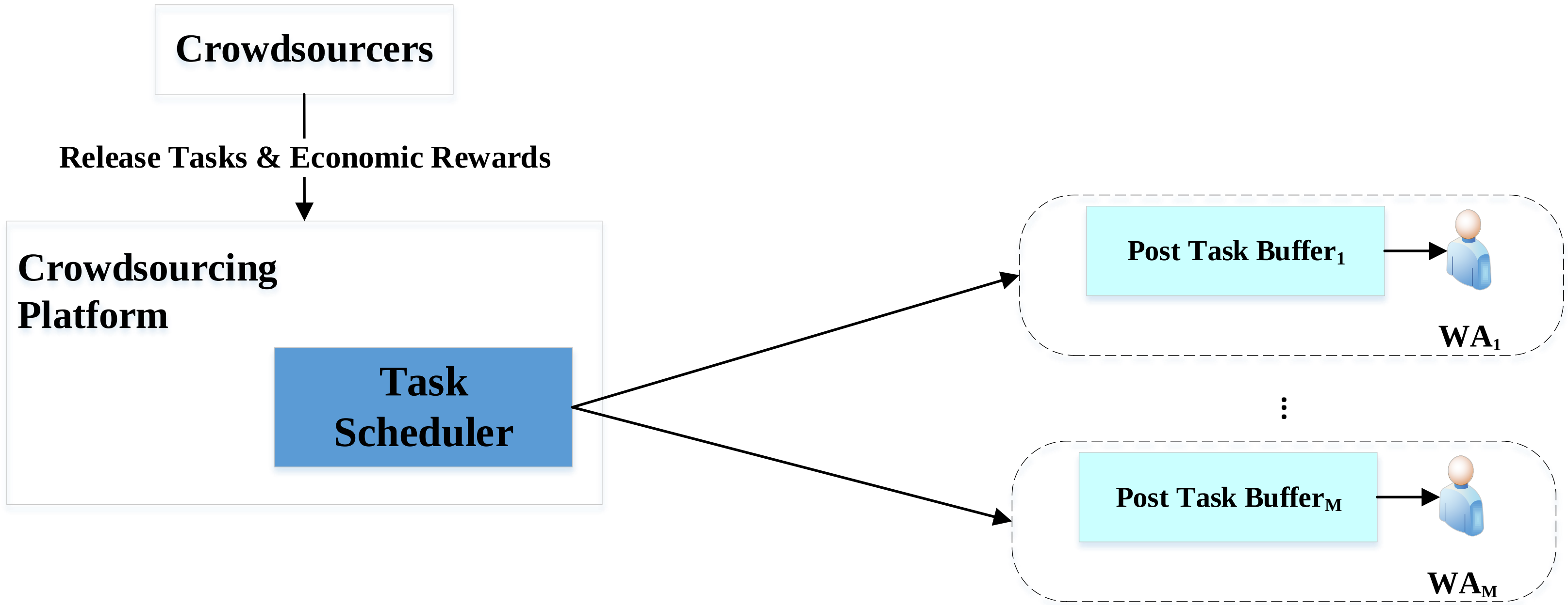
Dataset: Angile Manager Game

- Link:
  - http://agilemanager.algorithmic-crowdsourcing.com/

Angile Readme.pdf

Angile Dataset.zip

High-Level Offline Task Scheduling



Crowdsourcing Model Settings

- 任务发布者对众包任务发放固定的总奖金
- 任务发布者关心任务的完成质量和效率
- 参与众包任务的工人能力水平和参与规模未知、对任务发布者透明
- 参与众包任务的工人瓜分总奖金

Research Challenge I

- 任务完成的质量、效率的矛盾
  - 最优化任务质量
    - 将所有任务交给能力最强的工人
    - 工作效率较低
  - 最优化任务效率
    - 将任务分发给尽可能多的的工人，分布式完成
    - 工人能力参差不齐，任务质量下降
- Solution: ***选择性地分发给多个特定工人***

Research Challenge II

- 公平的任务奖金分发策略
  - 从工人角度出发
    - 希望和付出努力、完成质量正相关的报酬
  - 从任务发布者角度出发
    - 奖金的分发，不仅要看任务完成质量，还要看任务效率
    - 任务效率不仅取决于工人的工作水平，还和众包平台的任务调度相关
  - 不对称的理想奖金分发策略
- Solution: ***降低任务调度对任务效率的影响***

Research Challenge III

- 总奖金划分策略
  - 从工人角度出发
    - 每个参与众包的工人希望获得更多的奖金
    - 参与同一项任务的人数越**少**越好
  - 从任务发布者角度出发
    - 希望参与同一项任务的人数越**多**越好，以保证任务质量和效率任务效率
- *Solution:* ***在满足任务发布者的任务质量和效率需求的前提下，最小化参与同一项任务的人数***

Research Challenge IV

- 公平的总奖金划分策略
  - 目标：参与众包的工人尽可能获得相对平等的报酬
  - 解决方案：能力较弱的工人优先选择任务；任务难度和工人的能力匹配
- *Solution:* **引入** *Max-Min Fairness*

Metrics I

- 工人 j 高质量完成任务 i 的概率

$$p_{i,j}=\left(1\text{-}\frac{d_i}{D+1}\right)\cdot\rho_j \quad \forall i\in\{1,2,\ldots,N\} \quad \forall j\in\{1,2,\ldots,M\}$$

- 工人 j瓜分任务 i 奖金比例

$$r_{i,j}=\left(1-\frac{d_i}{D+1}\right)\cdot\rho_j\cdot\left(1-\left[\max\left(0,\frac{pos_{i,j}-dll_i}{T_i^{(\max)}}\right)\right]^{\alpha_i}\right)\cdot effort_i$$

- 工人 j 执行任务 i 获得的奖金

$$R_{i,j}=r_{i,j}\Big/\sum_{k=1}^{M}r_{i,k}$$

Metrics II

- 工人 j 执行任务获得的奖金

$$R_j^{(total)} = \sum_{i=1}^{N} R_{i,j}$$

Scheduling Framework

Part I. 基于任务质量和效率要求，Task Scheduler 决策参与任务工作的具体工人

提高每个参与任务的工人薪酬水平

Part II. 确定参与工人后，MMF公平划分任务总奖金

保证工人个体间瓜分奖金的公平性

Algorithm. Part I

❖ 最小化参与任务的工人，提高人均奖金

$$\min \sum_{i=1}^{N} \sum_{j=1}^{M} x_{i,j}$$

s.t. $$\sum_{i=1}^{N} x_{i,j} p_{i,j} \geq P_j \qquad \forall j \in \{1, 2, \ldots, M\}$$ 质量要求

$$\frac{\sum_{i=1}^{N} x_{i,j}}{W_j^{(\max)}} \leq dll^r \qquad \forall j \in \{1, 2, \ldots, M\}$$ 效率要求

$$x_{i,j} \in \{0,1\}$$

Algorithm. Part II

❖确定参与任务的工人后，公平划分总奖金s

$$\text{lexmax}\ \ g=\left(R_{k_1}^{(total)},R_{k_2}^{(total)},\ldots\ldots,R_{k_M}^{(total)}\right)$$

$$R_j^{(total)}=\sum_{i=1}^{N}R_{i,j}$$

$$R_{i,j}=r_{i,j}\Big/\sum_{k=1}^{M}r_{i,k}$$

$$\forall i\in\{1,2,\ldots,N\}$$
$$\forall j\in\{k_1,k_2,\ldots,k_M\}$$

$$r_{i,j}=x_{i.j}\cdot\left(1-\frac{d_i}{D+1}\right)\cdot\rho_j\cdot\left(1-\left[\max\left(0,\frac{pos_{i,j}-dll_i}{T_i^{(\max)}}\right)\right]^{\alpha_i}\right)\cdot effort_i$$

$$\langle pos\rangle_j=\left(\langle pos\rangle_{i_1,j_1}^{1},\langle pos\rangle_{i_2,j_2}^{2},\ldots\ldots,\langle pos\rangle_{i_{K_j},j_{K_j}}^{K_j}\right)$$

$$K_j=\sum_{k=1}^{N}x_{k,j}$$

$$\langle pos\rangle_{i_k,j_k}^{k}-\langle pos\rangle_{i_{k-1},j_{k-1}}^{k-1}=1$$

$$k=\{2,\ldots,K_j\}$$

$$pos_{i,j}\in\{1,\ldots,K_j\}$$

Scheduling Framework

Part I. 基于任务质量和效率要求，Task Scheduler 决策参与任务工作的具体工人

$$\min \sum_{i=1}^{N} \sum_{j=1}^{M} x_{i,j}$$

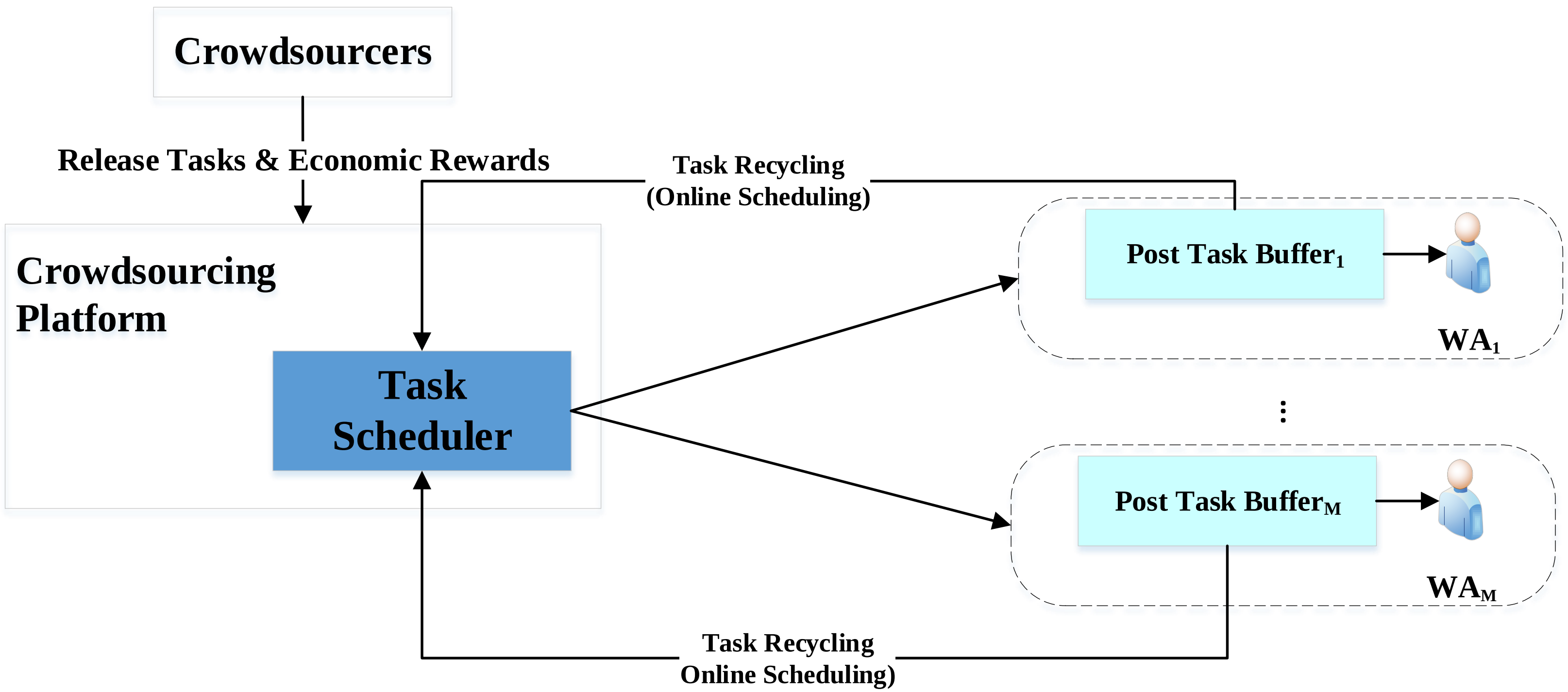
Part II. 确定参与工人后，MMF公平划分任务总奖金

$$\text{lexmax} \quad g = \begin{pmatrix} R_{k_1}^{(total)}, \\ \dots, R_{k_M}^{(total)} \end{pmatrix}$$

Blue Sky Thinking

- 关于奖金需求的MMF总奖金划分策略
  - 每个工人对奖金回报的期望不同
    - 工人A期待赚得较多的钱
    - 工人B期待赚得较少的钱

High-Level Online Scheduling



Online V.S. Offline

- Offline Task Scheduling
  - 服务工人固定，并且无休止服务
  - 不考虑工作能力 (高质量完成能力、单位内最多完成任务数) 的波动性
  - 算法复杂度高 – 整数规划，NP-hard
- Online Task Scheduling
  - 工人会休息、停止工作
  - 新的工人加入众包平台
  - 工人的工作能力具有波动性
  - 要求算法复杂度低，执行效率高

Online Task Scheduling

- 改进方向
  - 设计滑动窗口、时间序列分析，应对动态变化
    - 工人退出/新加入众包系统
    - 工人工作能力的波动性特者
  - 设计Offline Task Scheduling的等效或近似的低复杂度算法，提高调度执行效率

Thank you!

Q&A

Jiwei Huang
Beijing University of Posts and Telecommunications
Beijing, China
huangjw@gmail.com