Online Mobile Micro-Task Allocation in Spatial Crowdsourcing

Yongxin Tong 1, Jieying She 2, Bolin Ding 3, Libin Wang 1, Lei Chen 2

1 SKLSDE Lab and IRI, Beihang University, China
2 The Hong Kong University of Science and Technology, Hong Kong, China
3 Microsoft Research, Redmond, WA, USA

1{yxtong, lbwang}@buaa.edu.cn, 2{jshe, leichen}@cse.ust.hk, 3bolind@microsoft.com

Introduction
- Spatial Crowdsourcing (a.k.a Mobile Crowdsourcing)
  - Online platforms that facilitate spatial tasks to be assigned and performed by crowd workers, e.g. O2O applications.

  - Motivation
    - Dynamic micro-task assignment is absent.
    - Most O2O applications need to be addressed in real-time:
      - Fast Food Delivery.
      - Real-Time Taxi-Calling Service.
      - Product Placement Checking of Supermarkets.

The GOMA Problem
- Given
  - A set of spatial tasks $T$
    - Each $t \in T$: location $l_t$, arriving time $a_t$, deadline $d_t$ and payoff $p_t$.
  - A set of crowd workers $W$
    - Each $w \in W$: location $l_w$, arriving time $a_w$, deadline $d_w$, range radius $r_w$, capacity $c_w$ and success ratio $\delta_w$.
  - Utility Function: $U(t, w) = p_t \times \delta_w$.
- Find an online allocation $M$ to maximize the total utility $\text{MaxSum}(M) = \sum_{t \in T, w \in W} U(t, w)$ s.t.
  - Deadline Constraint.
  - Capacity Constraint.
  - Range Constraint.
  - Invariable Constraint: Once a task $t$ is assigned to a worker $w$, the allocation of $(t, w)$ cannot be changed.

Online Algorithm Evaluation: Competitive Ratio (CR)
- Adversarial Model: Worst-Case Analysis
  - $CR_A = \min_{G(T, W, U)} \frac{\text{MaxSum}(M)}{\text{MaxSum}(\text{OPT})}$
- Random Order Model: Average-Case Analysis
  - $CR_{RD} = \min_{G(T, W, U)} \frac{\text{E}[\text{MaxSum}(M)]}{\text{MaxSum}(\text{OPT})}$

Extended Greedy-RT Algorithm
- Arrival Time 8:00 8:01 8:02 8:07 8:08 8:09 8:15 8:18
- 1st Order $w_1, w_2, w_4, w_3, w_5, w_4, w_5$
- 2nd Order $w_1, w_2, w_3, w_4, w_5, w_4, w_5$

The arrival orders in the all examples use the 1st order.

Steps
- 1. Choose an integer $k$ from 1 to $\lceil \ln(U_{\text{max}} + 1) \rceil$ randomly.
- 2. Filter the edges with weights greater than $e^k$.
- 3. Use a greedy strategy on the remaining edges.

Competitive Ratio (Adversarial Model): $CR_A = \frac{1}{2e\ln(U_{\text{max}}+1)}$

Two-Phase-based Framework (TGOA Algorithm)

The first half of objects are filtered and disposed greedily.

Steps
- 1. Take a fixed fraction of arriving objects as samples and dispose the samples in a greedy way.
- 2. When a new object arrives, compute the optimal matching on the revealed part of the graph.
- 3. Match the new object to its adjacent node in the optimal matching if possible.

Competitive Ratio (Random order Model): $CR_{RD} = \frac{1}{4}$

TGOA-Greedy Algorithm
- Optimize the efficiency using a greedy solution to get the matching instead of the optimal matching in the second phase.
- Competitive Ratio (Random order Model): $CR_{RD} = \frac{1}{8}$

Experimental Evaluation

(a) Utility of varying $|W|$  
(b) Utility of varying $|T|$  
(c) Run time of varying $|W|$  
(d) Utility of scalability test  
(e) Memory of varying $|W|$  
(f) Utility of EverySender