Online Mobile Micro-Task Allocation in Spatial Crowdsourcing

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

1 SKLSDE Lab, Beihang University, China
2 The Hong Kong University of Science and Technology, Hong Kong, China
3 Microsoft Research, Redmond, WA, USA
1 {yxtong, lbwang}@cse.ust.hk, 2 {jshe, leichen}@cse.ust.hk, 3 bolind@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
    \[ CR_A = \min_{\forall G(T, W, U)\text{and } v \in V} \frac{\text{MaxSum}(M)}{\text{MaxSum}(OPT)} \]
  - Random Order Model: Average-Case Analysis
    \[ CR_{RO} = \min_{\forall G(T, W, U)} \frac{\mathbb{E}[\text{MaxSum}(M)]}{\text{MaxSum}(OPT)} \]

Extended Greedy-RT Algorithm

<table>
<thead>
<tr>
<th>Arrival Time</th>
<th>8:00</th>
<th>8:01</th>
<th>8:02</th>
<th>8:07</th>
<th>8:08</th>
<th>8:09</th>
<th>8:15</th>
<th>8:18</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st Order</td>
<td>$W_1$</td>
<td>$t_1$</td>
<td>$t_2$</td>
<td>$t_3$</td>
<td>$t_4$</td>
<td>$t_5$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2nd Order</td>
<td>$t_1$</td>
<td>$w_1$</td>
<td>$t_2$</td>
<td>$t_3$</td>
<td>$t_4$</td>
<td>$w_4$</td>
<td>$t_5$</td>
<td></td>
</tr>
</tbody>
</table>

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

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

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

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): $CR_{RO} = \frac{1}{8}$

TGOA-Greedy Algorithm

- Optimize the efficiency using a greedy solution to get the matching instead of the optimal matching in the second phase.

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.