Trichromatic Online Matching in Real-time Spatial Crowdsourcing

Tianshu Song [1], Yongxin Tong [1], Libin Wang [1], Jieying She [2], Bin Yao [3], Lei Chen [2], Ke Xu [1]

[1] SKLSDE Lab and IRI, Beihang University, China
[2] The Hong Kong University of Science and Technology, Hong Kong, China
[3] Shanghai Jiao Tong University, China
[1] {yxtong, songts, lbwang, kexu}@buaa.edu.cn, [2] {jshe, leichen}@cse.ust.hk, [3] yaobin@cs.sjtu.edu.cn

Introduction

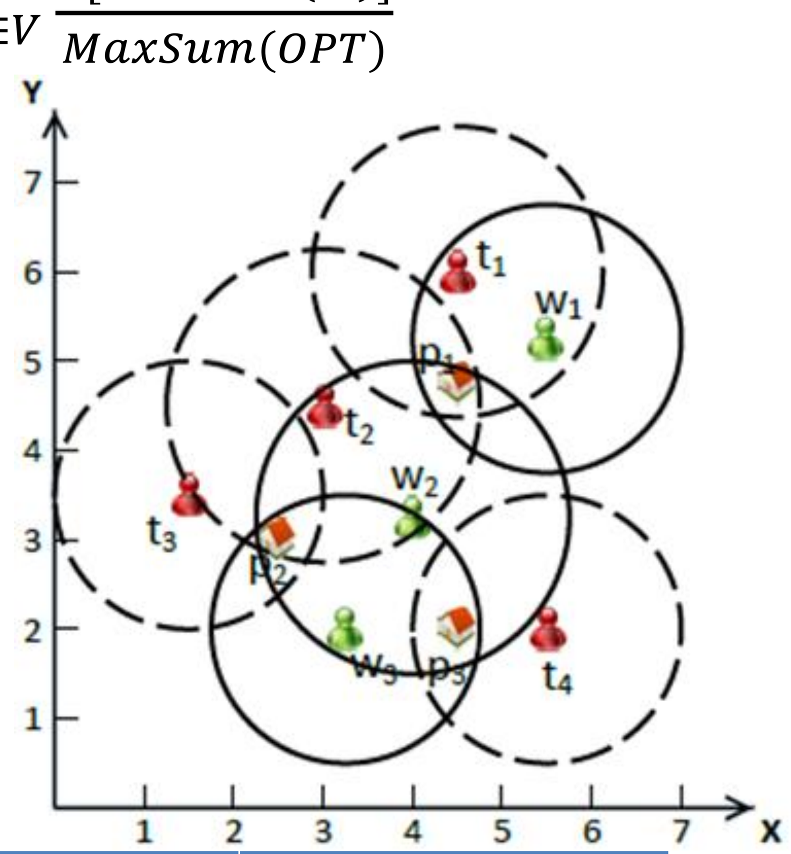
- Spatial Crowdsourcing (a.k.a Mobile Crowdsourcing)
  - Online platforms that facilitate spatial tasks to be assigned and performed by crowd workers, e.g. O2O applications.

- Motivation
  - Most O2O platforms work on real-time scenarios.
  - Some emerging O2O applications need to assign three types of objects :
    - Sports trainers, sports facilities and users.
    - Hairstylists, salon and customers.

The GOMA Problem

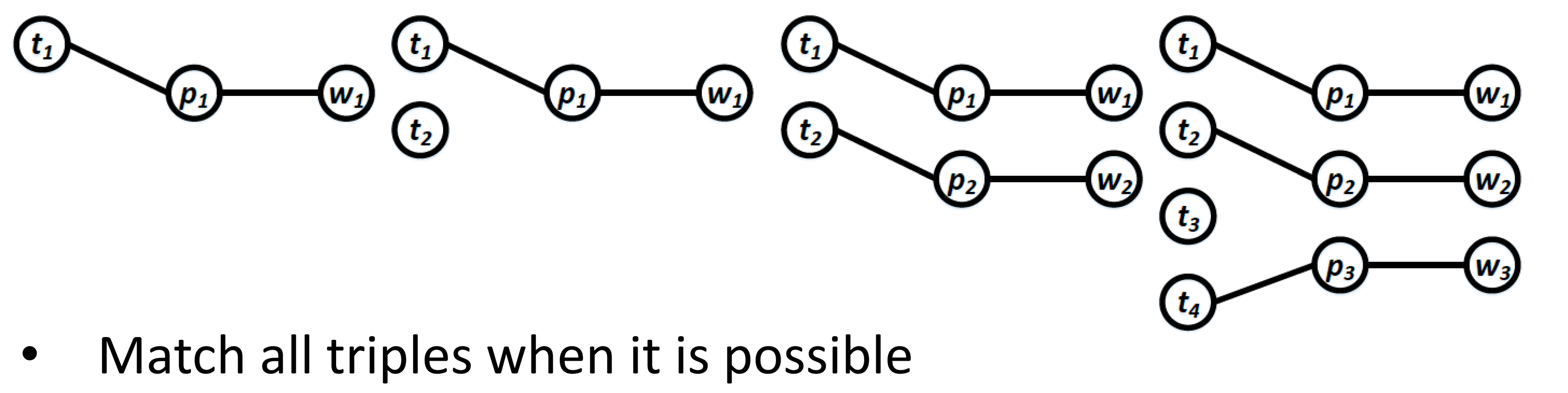
- Given
  - A set of tasks requester T
    - Each $t \in T$: location $\boldsymbol{l}_t$, arriving time b_t, leaving time e_t and range radius r_t.
  - A set of crowd workers W
    - Each $w \in W$: location $\boldsymbol{l}_w$, arriving time b_w, leaving time e_w, range radius r_w.
  - A set of crowd workplaces P
    - Each $p \in P$: location $\boldsymbol{l}_p$, arriving time b_p, leaving time e_p.
  - Utility Function: $U(t, p, w)$.
- Find a matching M to maximize the total utility $MaxSum(M) = \sum_{t \in T, p \in P, w \in W} U(t, p, w)$ s.t.
  - Deadline Constraint.
  - Range Constraint.
  - Invariable Constraint: *Once a task t is assigned to a worker w, the allocation of (t, p, w) cannot be changed.*
- Online Algorithm Evaluation: Competitive Ratio (CR)
  - Randomized Algorithm
    - $\text{CR} = min_{\forall G(T,W,P,U) \text{ and } \forall v \in V} \frac{\mathbb{E}[MaxSum(M)]}{MaxSum(OPT)}$

Object	Location	Arrival Time	Leaving Time
t_1	(4.50,6.00)	8:00	8:10
p_1	(4.50,4.75)	8:02	8:12
w_1	(5.50,5.00)	8:05	8:15
t_2	(3.00,4.50)	8:08	8:18
p_2	(2.50,3.00)	8:10	8:20
w_2	(4.00,3.25)	8:11	8:21
w_3	(3.25,2.00)	8:13	8:23
t_3	(1.50,3.50)	8:15	8:25
t_4	(5.50,2.00)	8:17	8:27
p_3	(4.50,2.00)	8:19	8:29



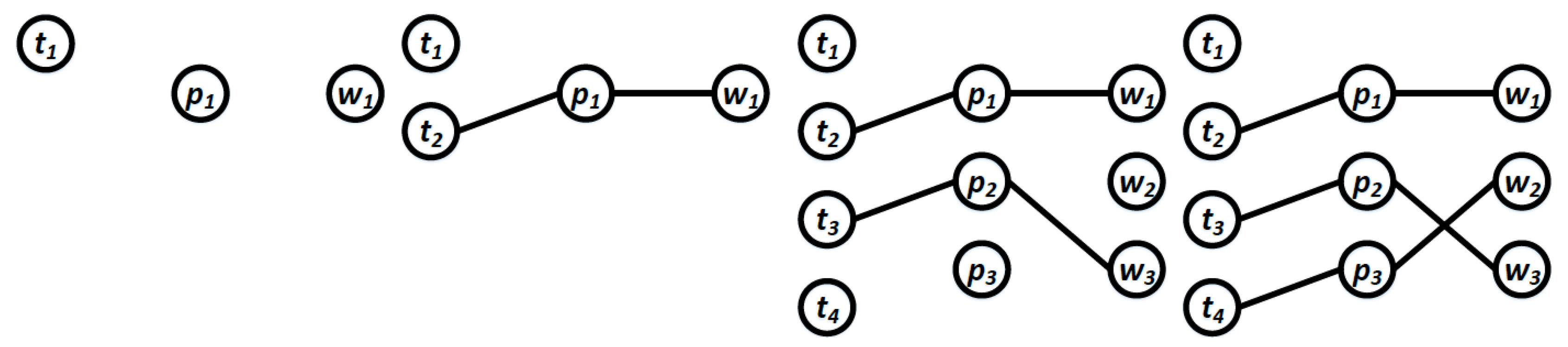
Match	Utility Score	Match	Utility Score
(t_1, p_1, w_1)	18	(t_2, p_2, w_3)	20
(t_1, p_1, w_2)	10	(t_3, p_2, w_2)	12
(t_2, p_1, w_1)	90	(t_3, p_2, w_3)	48
(t_2, p_1, w_2)	20	(t_4, p_3, w_2)	72
(t_2, p_2, w_2)	20	(t_4, p_3, w_3)	12

Greedy Algorithm



- Match all triples when it is possible
- Competitive Ratio: $CR = \frac{1}{3U_{max}}$

Basic-Threshold Algorithm



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

Adaptive-Threshold Algorithm

Threshold	t_1	w_1	t_2	w_2	t_3	p_3
e^0	1	1.02	1.02	1.04	1.04	1.05
e^1	1	1.02	1.02	1.04	1.04	1.05
e^2	1	1.02	1.02	1.04	1.04	1.05
e^3	1	1	1.09	1.09	1.14	1.22
e^4	1	1	1.09	1.09	1.09	1.17

- Adaptively adjust the probability distribution of choosing different thresholds.
- When an object appear, choose a new threshold according to the learned probability distribution
- $MaxSum \geq (1-\varepsilon)MaxSum(OPT_{\text{Basic-Threshold}}) - \frac{\varepsilon(1-\varepsilon)}{2D}\sum_v (g_v^*)^2 - \frac{D\,(1-\varepsilon)}{\varepsilon}\ln(\theta)$

Experimental Evaluation