Distributed Game-Theoretical Route Navigation for Vehicular Crowdsensing

En Wang[1], Dongming Luan[1], Yongjian Yang[1], Zihe Wang[2], Pengmin Dong[1], Dawei Li[3], Wenbin Liu[1], and Jie Wu[4]

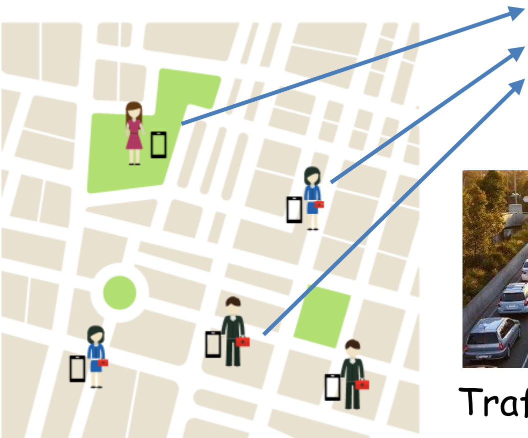
[1]Jinlin University, [2]Renmin University of China
[3]Montclair State University, [4]Temple University

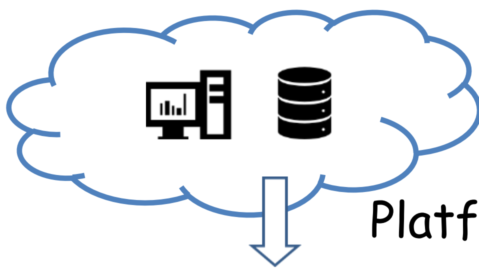
Outline

Motivation

Mobile Crowdsensing (MCS)

- Vehicular crowdsensing
- The existing task allocation strategies:
  - A heavy computation complexity
  - Fail to satisfy the preferences of users and the system.





Platform

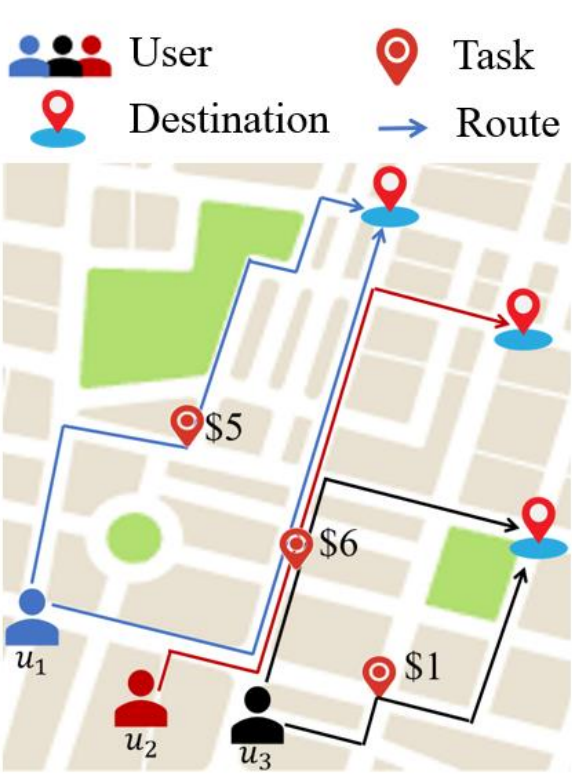
Traffic monitoring

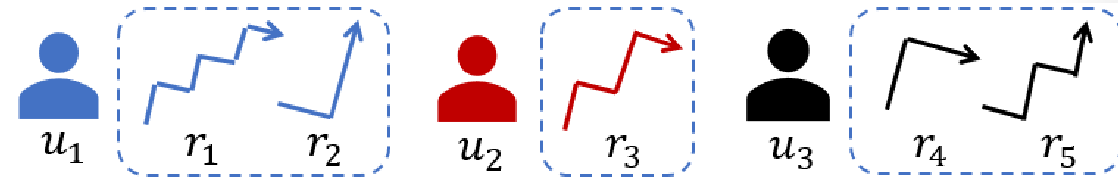
Noise monitoring

• • •

Distributed task allocation with the route navigation

Problem





Approach	Solution	Profit	Equilibrium
Maximum profit	$u_1: r_2$ $u_2: r_3$ $u_3: r_4$	u_1: 6/3=2 u_2: 6/3=2 u_3: 6/3=2 6	No
Distributed equilibrium	$u_1: r_1$ $u_2: r_3$ $u_3: r_4$	u_1: 5 u_2: 6/2=3 u_3: 6/2=3 11	Yes
Centralized optimal	$u_1: r_1$ $u_2: r_3$ $u_3: r_5$	u_1: 5 u_2: 6 u_3: 1 12	No

How to find an equilibrium state?

u_3 can select r_4 to get more profit.

Challenges

- How to construct a distributed model to achieve the equilibrium while guaranteeing the profit performance?
- How to design a unified distributed algorithm such that it could take the requirements of both the platform and users into consideration?
- How to guarantee a lower performance bound with respect to the centralized optimal solution?

System model

Profit of user i under strategy profile $\mathbf{s}$: $\mathbf{s} = (s_i, s_{-i})$

$$P_i(\mathbf{s}) = \alpha_i \cdot \boxed{\sum_{k \in \mathcal{L}_{s_i}} w_k(n_k(\mathbf{s}))/n_k(\mathbf{s})} - \beta_i \cdot d(s_i) - \gamma_i \cdot b(s_i)$$

the cost incurred by traveling the detour distance $\quad d(s_i) = \varphi \cdot h(s_i)$

the cost incurred by the congestion $\quad b(s_i) = \theta \cdot c(s_i)$

User parameters:
$\alpha_i, \beta_i, \gamma_i$

System parameters:
φ, θ

An illustrative example of the influence of ϕ and θ

User Destination Task Routes r_1, r_2

Profit function for u_i: $P_i(r_j) = \dfrac{w(r_j)}{n(r_j)} + \varphi \cdot h(r_j) + \theta \cdot c(r_j)$

	r_1	r_2
$h(r_i)$	0	2
$c(r_i)$	3	1

Achieve different purposes by adjusting the values of φ and θ.

Maximize task # | Minimize detour | Minimize congestion

φ	θ	Solution	Task #	Detour	Congestion
0.1	0.1	u_1: r_1 u_2: r_2	$\boxed{2}$	0+2=2	3+1=4
1	0.1	u_1: r_1 u_2: r_1	1	$\boxed{0+0=0}$	3+3=6
0.1	1	u_1: r_2 u_2: r_2	1	2+2=4	$\boxed{1+1=2}$

Theoretical Analysis

- NP-hardness of The Centralized Problem

Theorem 1. The problem of finding the solution with the maximum total profit in a centralized manner is NP-hard.

- Nash equilibrium

No user can improve the profit by altering the strategy unilaterally in a Nash equilibrium

- Potential game

✓ Nash equilibrium existence

✓ Finite improvement property

- Potential game proof

Theorem 2. The multi-user route navigation game is a weighted potential game and has a Nash equilibrium and finite improvement property.

Strategies

For user

Algorithm 1 Distributed Game-Theoretical Route Navigation Algorithm for user $i \in \mathcal{U}$.

Initialization Phase

Input α_i, β_i, λ_i, the initial location and the destination.
Receive the recommended routes R_i.
Initialize $s_i(0) = r$ by randomly selecting a route $r \in R_i$.
Report $s_i(0)$ to the platform.
Receive n_k for each task k that is covered by $s_i(0)$.
Calculate the profit P_i.
Receive $d(r)$ and $b(r)$ for each route r in R_i.
repeat for each decision slot t
    Obtain n_k for each task k that is covered by R_i.
    Compute the best route set $\Delta_i(t)$.
    if $\Delta_i(t) \neq \varnothing$ **then**
        Send the request to contend the opportunity for updating decision.
        if Win the opportunity **then**
            Update the route selection decision $s_i(t)$ by selecting a route $r \in \Delta_i(t)$.
            Report $s_i(t)$ to the platform.
        else
            Choose the original decision $s_i(t) = s_i(t-1)$.
until The termination message is received.

Update strategy

For platform

Algorithm 2 Information Update Algorithm for the platform.

Send the recommended route set R_i to the user $i \in \mathcal{U}$.
Receive $s_i(0)$ from each user $i \in \mathcal{U}$.
Calculate n_k for each task $k \in \mathcal{L}$.
Send n_k, $d(r)$ and $b(r)$ to the corresponding user.
repeat for each decision slot t
    Receive the request from the users and let $\mathcal{U}'$ denote the set of users that send the request.
    if $\mathcal{U}' \neq \varnothing$ **then**
        Select a set of users μ by SUU or PUU algorithm.
        Inform the users in μ to update the decisions.
        Receive $s_i(t)$ from user $i \in \mu$ and update n_k for each task $k \in \mathcal{L}$.
until No request is received from the user.
Send the termination message to all users.

Terminate the algorithm

Send the information to users

Select a set of users to update the strategy

Performance Evaluation

- Convergence for Nash equilibrium

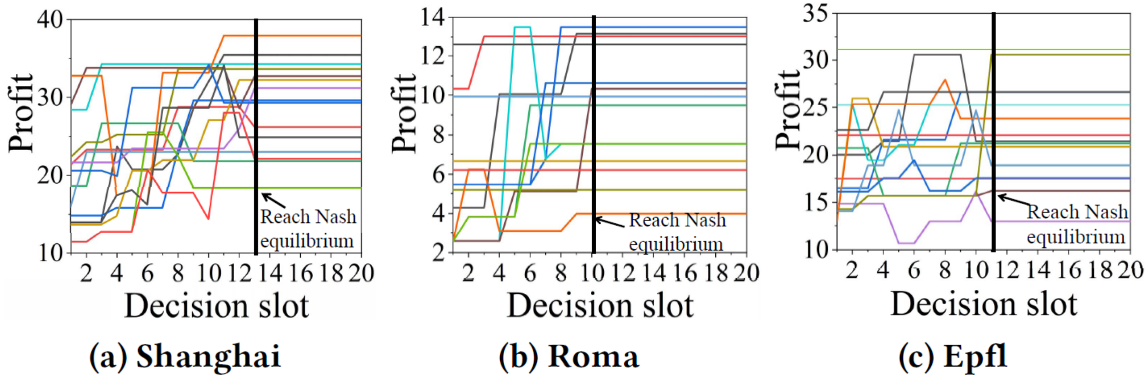


Figure 3: User profit vs. decision slot.

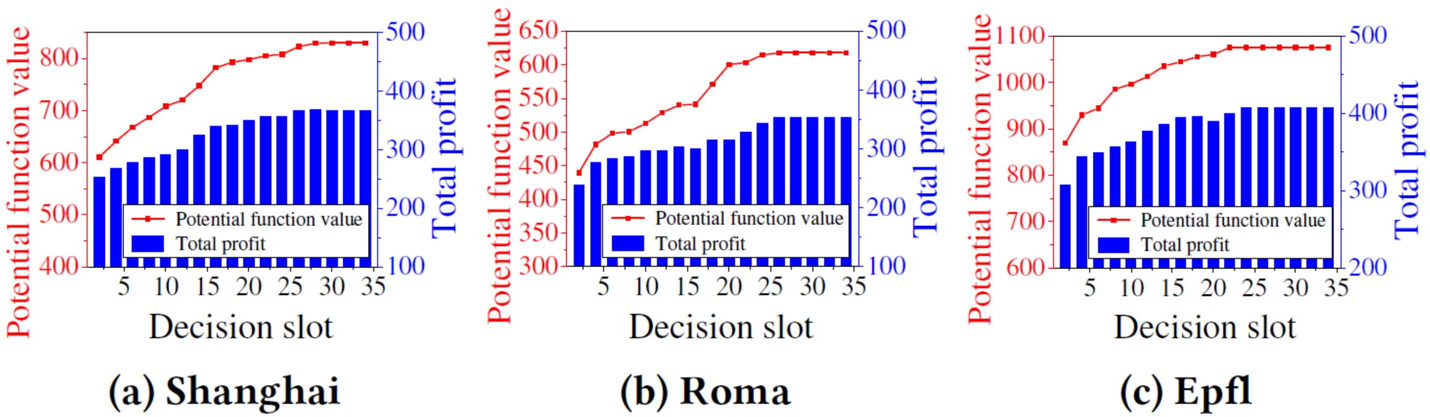
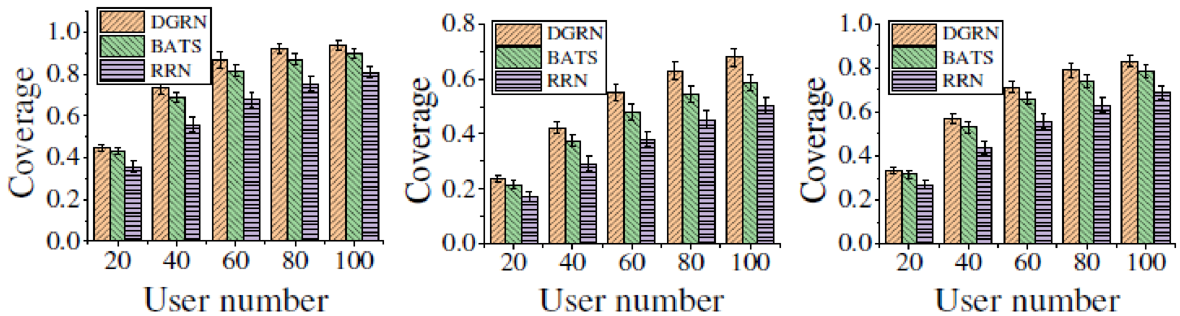


Figure 6: Potential function and total profit vs. decision slot.

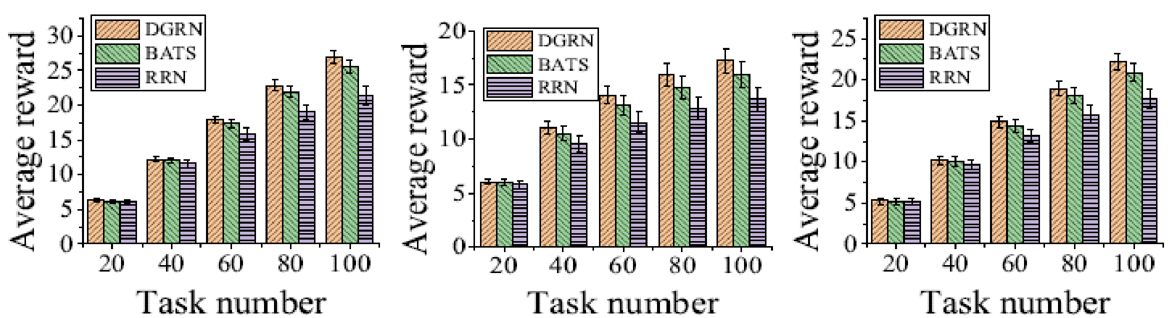
Performance Evaluation

- Coverage and reward



(a) Shanghai (b) Roma (c) Epfl

Figure 8: Coverage vs. user number.

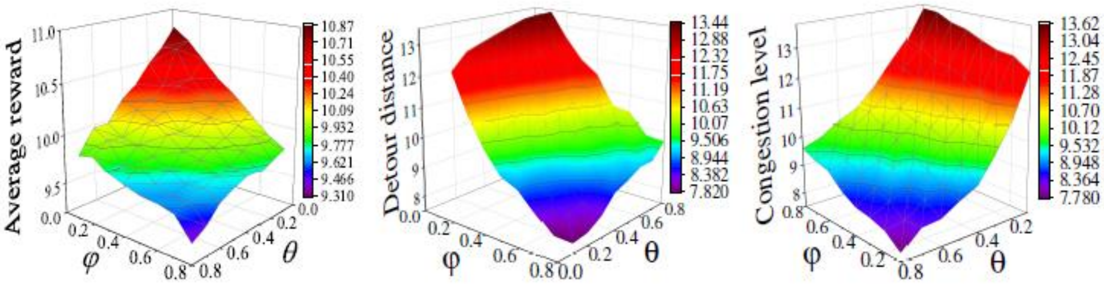


(a) Shanghai (b) Roma (c) Epfl

Figure 9: Average reward vs. task number.

Performance Evaluation

- The influence of user and system parameters



(a) Average reward (b) Detour distance (c) Congestion level

Figure 12: The influence of system parameters.

Table 5: The influence of the user parameters.

α_i	reward	β_i	detour	γ_i	congestion
0.1	7.74	0.1	12.24	0.1	12.03
0.2	7.85	0.2	10.97	0.2	10.48
0.3	7.94	0.3	9.88	0.3	9.52
0.4	7.96	0.4	9.38	0.4	8.75
0.5	7.98	0.5	8.84	0.5	8.48
0.6	8.08	0.6	8.38	0.6	8.20
0.7	8.10	0.7	8.07	0.7	8.05
0.8	8.16	0.8	7.99	0.8	7.97

Thanks for listening

Q&A