# Dynamic Public Resource Allocation Based on Human Mobility Prediction

SIJIE RUAN[*], Xidian University, China
JIE BAO[†‡], JD Intelligent Cities Research, China
YUXUAN LIANG, National University of Singapore, Singapore
RUIYUAN LI[*], Xidian University, China
TIANFU HE, Harbin Institue of Technology, China
CHUISHI MENG[†], JD Intelligent Cities Research, China
YANHUA LI, Worcester Polytechnic Institute, USA
YINGCAI WU, Zhejiang University, China
YU ZHENG[*‡], Xidian University, China

The objective of public resource allocation, e.g., the deployment of billboards, surveillance cameras, base stations, trash bins, is to serve more people. However, due to the dynamics of human mobility patterns, people are distributed unevenly on the spatial and temporal domains. As a result, in many cases, redundant resources have to be deployed to meet the crowd coverage requirements, which leads to high deployment costs and low usage. Fortunately, with the development of unmanned vehicles, the dynamic allocation of those public resources becomes possible. To this end, we provide the first attempt to design an effective and efficient scheduling algorithm for the dynamic public resource allocation. We formulate the problem as a novel multi-agent long-term maximal coverage scheduling (MALMCS) problem, which considers the crowd coverage and the energy limitation during a whole day. Two main components are employed in the system: 1) *multi-step crowd flow prediction*, which makes multi-step crowd flow prediction given the current crowd flows and external factors; and 2) *energy adaptive scheduling*, which employs a two-step heuristic algorithm, i.e., energy adaptive scheduling (EADS), to generate a scheduling plan that maximizes the crowd coverage within the service time for agents. Extensive experiments based on real crowd flow data in Happy Valley (a popular theme park in Beijing) demonstrate the effectiveness and efficiency of our approach.

CCS Concepts: • **Information systems → Spatial-temporal systems**;

Additional Key Words and Phrases: Dynamic Resource Allocation, Mobility Data Mining, Urban Computing

[*]S. Ruan, R. Li and Y. Zheng are also with JD Intelligent Cities Research, China and JD Intelligent Cities Business Unit, JD Digits, China.
[†]J. Bao and C. Meng are also with JD Intelligent Cities Business Unit, JD Digits, China.
[‡]Y. Zheng and J. Bao are corresponding authors.

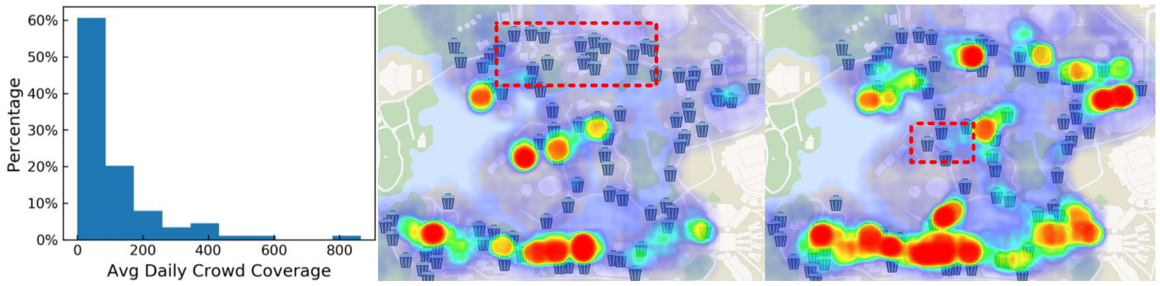
Authors' addresses: Sijie Ruan, sjruan@stu.xidian.edu.cn, Xidian University, Xi'an, Shaanxi, China; Jie Bao, baojie@jd.com, JD Intelligent Cities Research, Beijing, China; Yuxuan Liang, yuxliang@outlook.com, National University of Singapore, Singapore; Ruiyuan Li, ruiyuan.li@jd.com, Xidian University, Xi'an, Shaanxi, China; Tianfu He, Tianfu.D.He@outlook.com, Harbin Institue of Technology, Harbin, Heilongjiang, China; Chuishi Meng, meng.chuishi@jd.com, JD Intelligent Cities Research, Beijing, China; Yanhua Li, yli15@wpi.edu, Worcester Polytechnic Institute, Worcester, Massachusetts, USA; Yingcai Wu, ycwu@zju.edu.cn, Zhejiang University, Hangzhou, Zhejiang, China; Yu Zheng, msyuzheng@outlook.com, Xidian University, Xi'an, Shaanxi, China.

2474-9567/2020/3-ART25 $15.00
https://doi.org/10.1145/3380986

**ACM Reference Format:**
Sijie Ruan, Jie Bao, Yuxuan Liang, Ruiyuan Li, Tianfu He, Chuishi Meng, Yanhua Li, Yingcai Wu, and Yu Zheng. 2020. Dynamic Public Resource Allocation Based on Human Mobility Prediction. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 1, Article 25 (March 2020), 22 pages. https://doi.org/10.1145/3380986

## 1 INTRODUCTION

The objective of public resource allocation, e.g., the deployment of billboards, surveillance cameras, base stations, trash bins, aims to serve or cover more people. However, due to the nature of human mobility, the crowd distribution is highly uneven on both spatial and temporal domains. As a result, redundant resources have to be deployed to guarantee the desired crowd coverage, which in many cases leads to high deployment cost and low usage. Figure 1(a) shows the utilization of 95 trash bins during the whole day in Happy Valley (a popular theme park in Beijing), where 70%∼80% trash bins are not fully utilized. Apart from the outdated deployment due to the development of the park, we further find the utilization varies from time to time. The crowd distributions at 11:00 and 18:00 are visualized in Figure 1(b) and 1(c). It is clear that the trash bins marked with rectangles are not always in service. Non-fully utilized deployment is more undesirable if the resource is expensive, e.g., the billboards. Therefore, there are some works [22, 49] studying how to select the best locations for billboard placement, but those locations are still static.



(a) Trash bin utilization. (b) The crowd flows at 11:00. (c) The crowd flows at 18:00.

Fig. 1. Motivating Example.

Based on these observations, a natural question arises: is it possible to serve more people with fewer resources, if these public resources are mobile? Fortunately, with the rapid development of the robotic systems, various unmanned vehicles emerge for logistic or transportation purposes, as shown in Figure 2(a). Furthermore, some companies, like JD.com, are developing the common mobile bases [1] (shown in Figure 2(b)) that can carry any payload, which can make the resource mobile. However, it is still a non-trivial task to design effective scheduling for these dynamic public resources (or mobile agents) for the following reasons:

- **Maximal Crowd Coverage & Limited Resource.** We want to serve as many people as possible with limited resources. The static public resource allocation itself is essentially a well-known maximum $k$ coverage problem [16] which is NP-hard. The mobile/dynamic resource setting makes it more difficult, as the status in different time steps needs to be considered dynamically.
- **Energy Limitation.** The energy capacity of mobile agents is limited. If we move them very aggressively according to the current crowd distribution, their energy can drain up very quickly. Hence, the system requires an accurate long-term prediction and an energy-aware scheduling method.

[1]https://bit.ly/2WxKpdj

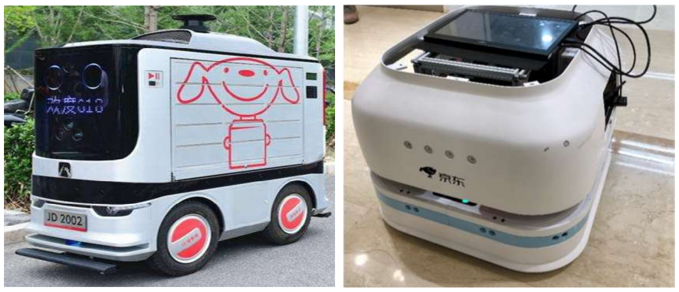

(a) Unmanned Vehicles. (b) Common Mobile Base.

Fig. 2. Opportunity of Mobile Agents.

- **Large Action Space.** Mobile agents can move to any place within an area, which leads to a huge action space. Thus, finding the global optimal solution is extremely difficult.

To this end, the system first employs a *Multi-step Crowd Flow Prediction* module, which uses a spatio-temporal crowd flow model to predict a long-term crowd flow in each grid region based on the current crowd flow observation and other external factors. With the knowledge of crowd flow distribution in the long-term, we design the *Energy Adaptive Scheduling* algorithm. The energy adaptive scheduling assigns the agents to next locations according to the predicted crowd flows and their current status (i.e., remaining energy and current location) in order to maximize the whole day crowd coverage. Because the accuracy deteriorates significantly in the long-term prediction, we borrow the idea from Model Predictive Control (MPC) [8], which is also known as a model-based reinforcement learning approach [15, 29], where we optimize the schedule over a fixed future time horizon using the pre-built system model and current observations. However, only actions in the upcoming time step are executed. In the next time step, the system re-calculates the schedule based on the current status and the newly predicted crowd flows to mitigate the prediction and scheduling errors. The main contributions are summarized as follows:

- We provide the first attempt to study the dynamic public resource allocation problem and present a solution framework incorporating long-term crowd flow prediction and multi-agent scheduling.
- We define the scheduling task as a multi-agent long-term maximal coverage scheduling (MALMCS) problem, and prove its NP-hardness.
- We propose a two-step efficient and effective heuristic algorithm Energy Adaptive Scheduling (EADS) to handle MALMCS, where the allocation strategy is adaptive according to the remaining energy of each agent.
- Experimental results demonstrated that EADS achieves 80.0% and 56.8% resource reduction compared with the current deployment and the static maximal coverage deployment strategies, respectively. We also have released the code and data for public use[2].

The rest of the paper is organized as follows: Section 2 gives the problem statement and system overview. Section 3 presents the multi-step crowd flow prediction. Section 4 describes energy adaptive scheduling. Experiments and case studies are conducted in Section 5. Related works are summarized in Section 6. Section 7 concludes the paper.

[2]https://github.com/sjruan/malmcs

## 2 OVERVIEW

In this section, we first give some preliminaries and used notations, then we define the multi-agent long-term maximal coverage scheduling problem (MALMCS) and outline our solution framework.

### 2.1 Preliminaries

*Definition 2.1. (Location)* The area is divided into a set of disjoint uniform grid locations $G = \{g_i\}$.

*Definition 2.2. (Agent)* A mobile public resource is defined as an agent, which has an energy limitation $E$, and a service radius $r$. The service radius $r$ determines the area that can be served by the agent. The energy cost $c_{ij}$ is incurred when an agent moves from $g_i$ to $g_j$. An agent $a_k$ is associated with a status $s_k^t$ at time interval $T_t$, which is a tuple, i.e., $[p_k^t, e_k^t]$, indicating its location $p_k^t \in G$ and the remaining energy $e_k^t$.

The service area is the property of a certain resource, and we assume each public resource is sufficient to serve the crowds in its service area. For example, the service area of the surveillance camera is the area it can monitor. If $r = 1$, the service area only contains the grid that the agent stands, while if $r = 2$, the service area contains the grid where it stands and its surrounding 8 grids, etc. The energy cost from $g_i$ to $g_j$ is related to many factors, such as the distance of the routes and roads inclination. Modeling the energy consumption between different locations is not the main focus of the paper, and there are several existing studies [4, 30, 39]. The road network data can be obtained from managers of the area of the interest, or even inferred from the mobility data [36]. Given the geographical and the road network data, the energy consumption between different locations can be computed in advance.

*Definition 2.3. (Crowd Coverage)* The crowd coverage in a time interval $T_t$ is defined as the total number of people in the designated area covered by at least one agent during $T_t$. We define $g_i \in G$ is covered by an agent, if and only if $g_i$ is in the service area of the agent. We use indicator $y_i^t$ to denote whether $g_i \in G$ is covered by at least one agent during $T_t$, and $\lambda_i^t$ to represent the number of people in $g_i$ during $T_t$. So, the crowd coverage $cov_{T_t}$ is defined as $cov_{T_t} = \sum_{g_i \in G} y_i^t \lambda_i^t$.

Note that, the human mobility data in each grid during a certain period of time can be obtained from base stations [19], check-ins of mobile Apps [38], or the processing of crowd sourced trajectories [20, 21].

*Definition 2.4. (Charge Station)* There is an immobilized charge station $g_\omega \in G$ in the study area. All agents must start and terminate their trips at $g_\omega$.

### 2.2 Problem Statement

We formulate the energy-constrained dynamic resource allocation task as the multi-agent long-term maximal coverage scheduling problem. The formal definition is given as follows.

Given a set of mobile agents $\mathcal{A} = \{a_k | k = 1, ..., K\}$, service time intervals $\mathcal{T} = \{T_t | t = 1, ..., n\}$, a service radius $r$, the energy limitation $E$, and the location of a charge station $g_\omega$, we want to find a schedule that covers people as much as possible without using up the energy of each individual agent before the agents return the charge station.

Under the perfect knowledge of future crowd flows, the optimization problem can be formulated as an Integer Linear Programming (ILP) problem with the following decision variables: $x_{ijk}^t = 1$, if $a_k$ travels from $g_i$ to $g_j$ at the beginning of $T_t$, $x_{ijk}^t = 0$, otherwise; $u_{ik}^t = 1$ if $a_k$ is in $g_i$ during $T_t$, $u_{ik}^t = 0$, otherwise.

The mathematical formulation is given in Equation 1, with following three groups of constraints: structure constraints, coverage constraints, and energy constraints.

**Structure Constraints.** The constraint (1b) makes sure there is no route from locations except for the charge station $g_\omega$ at the beginning of the scheduling. Constraint (1c) guarantees that all agents start and end their routes at the charge station $g_\omega$, and each agent will be at exactly one location at $T_1$ and $T_n$. The constraint (1d) determines the connectivity of each route. The constraint (1e) ensures that during each time interval $T_t \in \mathcal{T}$, each location $g_i$ is visited by at most one agent.
**Coverage Constraints.** Let $C(g_i, r)$ denotes the location set whose service area with radius $r$ contains $g_i$. The constraint (1f) means, if $y_i^t = 1$, then at least one $g_j \in C(g_i, r)$ has an agent during $T_t$.
**Energy Constraints.** Constraint (1g) guarantees the energy cost of each agent back to the charge station after the service time doesn't exceed the energy limitation $E$.

$$\begin{aligned}
\underset{\{u_{ik}^t\}_{ikt}}{\text{maximize}} \quad & \sum_{t\in\mathcal{T}} \sum_{g_i\in G} y_i^t \lambda_i^t && && \text{(1a)}\\
\text{subject to} \quad & \sum_{a_k\in\mathcal{A}} \sum_{g_i\in G\setminus\{g_\omega\}, g_j\in G} x_{ijk}^1 = 0, && && \text{(1b)}\\
& \sum_{a_k\in\mathcal{A}} \sum_{g_j\in G} x_{\omega jk}^1 = \sum_{a_k\in\mathcal{A}} \sum_{g_i\in G} x_{i\omega k}^{n+1} = K, && && \text{(1c)}\\
& \sum_{g_h\in G} x_{hik}^t = \sum_{g_j\in G} x_{ijk}^{t+1} = u_{ik}^t, && \forall g_i \in G, T_t \in \mathcal{T}, a_k \in \mathcal{A}, && \text{(1d)}\\
& \sum_{a_k\in\mathcal{A}} u_{ik}^t \le 1, && \forall g_i \in G, T_t \in \mathcal{T}, && \text{(1e)}\\
& \sum_{g_j\in C(g_i,r)} \sum_{a_k\in\mathcal{A}} u_{jk}^t \ge y_i^t, && \forall g_i \in G, T_t \in \mathcal{T}, && \text{(1f)}\\
& \sum_{t\in\mathcal{T}\cup\{t_{n+1}\}} \sum_{g_i,g_j\in G} c_{ij} x_{ijk}^t \le E, && \forall a_k \in \mathcal{A} && \text{(1g)}
\end{aligned}$$

Note that in this setting, we omit the moving time of agents among different locations due to following two reasons: 1) The usage scenarios of mobile agents usually are in a small region due to management convenience and energy limitation, so that mobile agents can move to different locations in a short time; 2) The decision interval usually is long. Public resources like trash bins, billboards can only at service when they are fixed. Frequent moving leads to bad user experiences and low utilization. Therefore, in a majority of real-world cases, the decision interval (e.g., 1 hour) is much longer than the moving time (e.g., 1 min).

Such a problem of the multi-agent long-term maximal coverage scheduling problem is NP-hard as proven in Theorem 1 below.

**Theorem 1** (NP-difficulty). The multi-agent long-term maximal coverage scheduling problem (MALMCS) is NP-hard.

Proof. We reduce our problem of multi-agent long-term maximal coverage scheduling problem (MALMCS) to a team orienteering problem with time window (TOPTW) [43], which is known to be NP-hard. The goal of the team orienteering problem (TOP) is to determine $k$ paths with fixed starting and ending vertices, where each path should be within the time limitation (or distance limitation), that maximizes the total collected score at each vertex. Furthermore, if the score at each vertex can only be collected at a certain time window, this problem is called TOPTW. In MALMCS, we can view the energy limitation as the distance budget, and the energy cost between two locations as the distance cost. If $r = 1$, we can view the crowd flows $\lambda_i^t$ as a score that can be collected at $g_i$ at the beginning of time interval $T_t$. Then, our problem boils down to TOPTW. Thus, for any instance of TOPTW, we can find an instance of MALMCS by setting $r = 1$, and the answers are the same. Thus, TOPTW is reducible to MALMCS, which completes the proof of NP-difficulty. □

The ILP model is proposed to provide a global optimum solution as our baseline. However, due to its NP-hardness, with the increasing number of locations and agents, the optimal solution becomes computationally intractable. Because of this, an effective and efficient heuristic algorithm, i.e., energy adaptive scheduling (EADS) is proposed in Section 4 to provide an approximate solution.

As described, the MALMCS problem requires the perfect knowledge of future crowd flows. However, given the dynamic nature of crowd flows, the long-term predictions are inaccurate in a real-world application. Therefore, a Model Predictive Control (MPC) approach is employed to solve the challenge. At the end of time step $T_{cur} \in [T_0, T_{n-1}]$, based on the real-time crowd flow observations and external factors, the crowd flows of $[T_{cur+1}, T_n]$ are predicted. Then, MALMCS problem is solved again with the optimization horizon $\mathcal{T} = \{T_t | t = cur + 1, ..., n\}$, based on the predictions and the current status of the agents. Based on MPC principle, only the first action is executed.

## 2.3 System Framework

Figure 3 gives an overview of our system, which consists of two main components:

**Multi-step Crowd Flow Prediction.** In this component, a multi-step crowd flow prediction model is trained based on historical crowd flow observations and other external factors. Then this component gives crowd flow predictions until the end of the service time during the online scheduling phase.

**Energy Adaptive Scheduling.** This component decides the actions of the agents at the next time step, based on the multi-step crowd flow prediction results and their current status (i.e., location and energy). There are two main steps: 1) *multi-level max cover scheduling*, which tries to assign agents to maximal coverage locations in future time steps if the energy constraint is not violated. And 2) *energy-aware scheduling*, which is called to find a schedule that obtains as much crowd coverage as possible when the maximal coverage location assignment is not feasible.

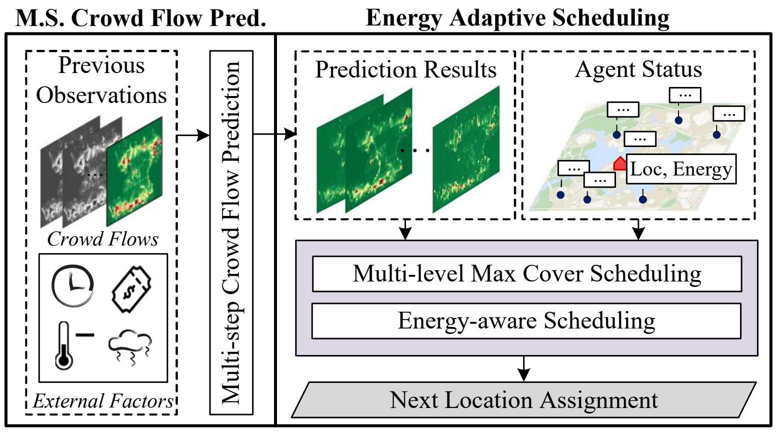


Fig. 3. An Overview of System.

## 3 MULTI-STEP CROWD FLOW PREDICTION

The MALMCS schedules agents based on the estimation of future crowd flows so that more daily crowd coverage can be achieved within the energy limitation. Therefore, an accurate crowd flow prediction model is required.

This component serves as a long-term crowd flow predictor, which predicts the value of crowd flows jointly in all grid locations of the interested region in future several time steps based on historical crowd flows and various external factors. Formally, the long-term crowd flow prediction task can be formulated as:

$$\hat{\Lambda}^{t+1,\ldots,t+l_{pred}} = f(\Lambda^{t-l_{hist}+1,\ldots,t}, E^{t-l_{hist}+1,\ldots,t}) \tag{2}$$

where $\Lambda \in \mathbb{R}_{+}^{l_{hist} \times I}$ is the crowd flows of historical $l_{hist}$ time steps in all $I$ grid locations, $E \in \mathbb{R}^{l_{hist} \times F}$ is $F$ types of external factors (e.g., weather, event, time of day, and day of week) varying with time, and $\hat{\Lambda} \in \mathbb{R}_{+}^{l_{pred} \times I}$ is the predicted crowd flows of future $l_{pred}$ steps in all $I$ grids. Note that $l_{pred}$ is equal to the number of decision intervals $n$ so that even at the first decision time of the day, we can estimate the hourly crowd flow changes until the end of the service time.

The crowd flow prediction is a task that has been widely studied in the machine learning field [9, 32, 33, 45–48], where Deep Learning shows its power to capture the spatio-temporal correlation of crowd flows. Note that existing works of crowd flow time-series prediction usually reshape stacked $I$ grids into $H \times W$ matrix representation, so that the spatial correlation can be easily captured by the convolution operation.

Due to the crowd flow prediction is not the main focus of this paper, we use the state-of-the-art crowd flow prediction model, i.e., Matrix Factorization-based STResNet [33] as our predictor. But any other grid-based crowd flow prediction model can be used as an alternative method as long as it can achieve the goal defined in Equation 2. Though the model gives crowd flow predictions with a fixed length, we only leverage the time steps that we actually needed. For example, at the end of time interval $T_{cur}$, the predictor predicts $\hat{\Lambda}^{cur+1,\ldots,cur+l_{pred}}$, but only $\hat{\Lambda}^{cur+1,\ldots,n}$ is actually used.

## 4 ENERGY ADAPTIVE SCHEDULING

### 4.1 Overview

This component decides the movement of each agent to serve more crowd in the future time steps. A scheduling strategy that covers all remaining time of the day is generated based on the multi-step predictions of the crowd flows and the current status of the agents (i.e., their locations and remaining energy). However, only the schedule of the next one time step is executed, following the principle of MPC, as the predictions at later time steps are less accurate.

Since our multi-agent long-term maximal coverage scheduling problem (i.e., MALMCS) is NP-hard, it is impossible to derive the optimal solution efficiently. To this end, we propose a two-step heuristic solution, i.e., $\underline{\text{E}}$nergy $\underline{\text{Ad}}$aptive $\underline{\text{S}}$cheduling (EADS) to approximately solve the problem. The main idea here is to first find a schedule to visit the maximal $k$ coverage locations at each step. If there is not enough energy for the agents to execute the plan, we then use the energy limits as the search bound to find a schedule that is as close as possible to the optimal location assignment. Algorithm 1 demonstrates the main framework with two steps: the first step *Multi-level Max Cover Scheduling*, which calculates an assignment of the agents to move to the $k$ maximal coverage locations in each future time step, i.e., Line 1-2. If the energy constraint is not violated, the generated schedule is returned; Otherwise, the second step *Energy-aware Scheduling*, i.e., Line 6, is called. *Energy-aware Scheduling* finds a schedule with a feasible schedule using a heuristic derived from the data analytics of crowd flows, and the resulting schedule is returned.

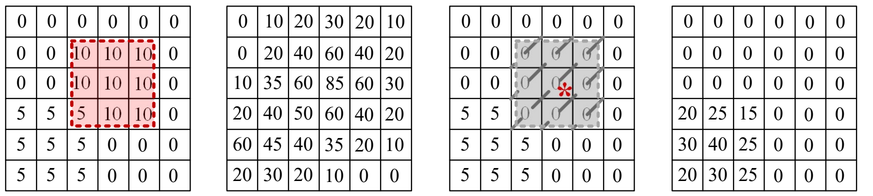
**Algorithm 1** Energy Adaptive Scheduling.

```
Input: The predicted multi-step crowd flows $\hat{\Lambda}^{cur+1,n}$; the current status of agents $S^{cur}$;
Output: The location of each agent at the next time step $T_{cur+1}$;
//Multi-level Max Cover Scheduling
1: $L^{cur+1,n} \leftarrow MultiLevelMaxCover(\hat{\Lambda}^{cur+1,n}, K, r)$;
2: $\{p_k^{cur+1,n}\}_1^K \leftarrow MaxMinAssign(S^{cur}, L^{cur+1,n}, g_\omega)$;
3: if $\min_k remain(s_k^{cur}, p_k^{cur+1,t_n}, g_\omega) \geq 0$ then
4:     return $\{p_k^{cur+1}\}_1^K$;
5: else
   //Energy-aware Scheduling
6:     $\{p_k^{cur+1,n}\}_1^K \leftarrow EnergyAwareScheduling(S^{cur}, \hat{\Lambda}^{cur+1,n})$;
7:     return $\{p_k^{cur+1}\}_1^K$;
```

## 4.2 Multi-level Max Cover Scheduling

**Main Idea.** The intuition of multi-level max cover scheduling is that if it is possible to assign all agents to the maximal coverage locations at each future time step, an optimal coverage schedule can be obtained. Then, we just need to check whether there exists a feasible multi-level location assignment where the energy limitation is not violated. If the minimum remaining energy among all agents after the scheduling is not a negative number, then a solution is feasible. As a result, the procedure contains two main steps: 1) finding the maximal $k$ coverage locations in each future time step; and 2) assigning agents to multi-level locations, which maximizes the minimum individual remaining energy among all agents, and the feasibility is checked.

**Step 1. Max Coverage Location Selection.** Given the prediction of crowd flows, maximal coverage locations can be calculated in each future time step. If the service area of an agent only contains the grid cell that it stands, i.e., $r = 1$, the optimal solution just selects locations with the top-$k$ crowd flow volumes. Otherwise, when the service area contains multiple grids, e.g., all nine nearby grids, max coverage location selection is equivalent to the budgeted maximal $k$ coverage [16], which is a well-known NP-hard problem. A greedy solution is used widely to tackle this problem, with a theoretical bound to be optimal of $1 - \frac{1}{e}$ [31]. It chooses a set that contains the maximum weight of uncovered elements at each stage.



| | | | | | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 10 | 10 | 10 | 0 |
| 0 | 0 | 10 | 10 | 10 | 0 |
| 5 | 5 | 5 | 10 | 10 | 0 |
| 5 | 5 | 5 | 0 | 0 | 0 |
| 5 | 5 | 5 | 0 | 0 | 0 |

(a) Crowd Flows.

| | | | | | |
|---|---|---|---|---|---|
| 0 | 10 | 20 | 30 | 20 | 10 |
| 0 | 20 | 40 | 60 | 40 | 20 |
| 10 | 35 | 60 | 85 | 60 | 30 |
| 20 | 40 | 50 | 60 | 40 | 20 |
| 60 | 45 | 40 | 35 | 20 | 10 |
| 20 | 30 | 20 | 10 | 0 | 0 |

(b) Service Score.

| | | | | | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 * | 0 | 0 |
| 5 | 5 | 0 | 0 | 0 | 0 |
| 5 | 5 | 5 | 0 | 0 | 0 |
| 5 | 5 | 5 | 0 | 0 | 0 |

(c) Uncovered Crowd Flows.

| | | | | | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 20 | 25 | 15 | 0 | 0 | 0 |
| 30 | 40 | 25 | 0 | 0 | 0 |
| 20 | 30 | 25 | 0 | 0 | 0 |

(d) Updated Service Score.

Fig. 4. An Example of Max-$k$-Cover Location Selection.

**Example.** Figure 4 gives an example of calculating the max $k$ cover location at a time step. The service area of an agent contains the location where it stands and its eight nearby grids. Figure 4(a) is the predicted crowd flows at the next time step. The service score of each grid can be calculated to sum the crowd flows in its service area, as shown in Figure 4(b). For example, the location with score 85 in Figure 4(b) is the sum of crowd flows in the red rectangle in Figure 4(a). Among all the service scores, the location with maximum value, e.g., 85, is selected, and the crowd flow is updated by removing crowd flows in the service area, as shown in Figure 4(c). After that, the service score is updated based on the uncovered crowd flow, as shown in Figure 4(d), to select the next location. This process is terminated until $k$ locations are selected. The maximal $k$ coverage locations in each future time step are selected using this method.

**Step 2. Max Min Energy Assignment.** In this step, we want to find a multi-level location assignment to maximize the minimum remaining energy among all agents after the scheduling. Thus, if the agent with the minimum remaining energy after the scheduling still has positive remaining energy, the schedule is executable.

The agent with the minimum remaining energy is also the agent with the maximum energy cost. So the max min energy assignment is equivalent to the min max cost assignment. As a result, the problem can be formulated as a classic multi-level bottleneck assignment problem (MBAP), which is known as NP-hard [6]. An effective heuristic algorithm [2] is widely used based on the iterative solution of the single-level linear bottleneck assignment problem (LBAP). Note that, LBAP can be solved in polynomial time using the threshold algorithm [1]. The main idea contains two steps. In the first step, an initial assignment is obtained by solving LBAP in each adjacent layer. In the second step, we iteratively solve LBAP in each layer by fixing arc assignments in other layers until no further improvement.

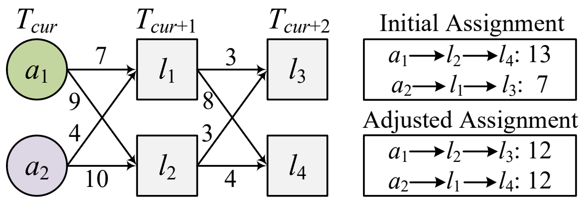


Fig. 5. Multi-level Bottleneck Assignment Example.

**Example.** An example is given in Figure 5. Assume there are 2 agents $a_1$ and $a_2$, and two future time steps. The maximal locations are selected as $l_1$, $l_2$ and $l_3$, $l_4$, respectively. The energy costs between locations are displayed on the edges. It should be noted that due to the agents at $T_{cur}$ already have some energy cost, and all agents have to go back to the charge station at $T_{cur+3}$, the spent energy is added to the first layer edges, and the energy costs going back to $g_\omega$ are also added to the last layer edges, so that we can find an assignment that the maximal individual total energy cost is minimized. In the first step, we solve LBAP in $[T_{cur}, T_{cur+1}]$ and $[T_{cur+1}, T_{cur+2}]$ independently. So that an initial assignment is obtained with the maximal cost 13 (shown in the initial assignment box). In the second step, we iteratively solve LBAP over each adjacent layer while the assignment in other layers is fixed and their costs are taken into consideration. For example, we fix the layer $[T_{cur}, T_{cur+1}]$, and adjust layer $[T_{cur+1}, T_{cur+2}]$, then a better assignment can be achieved with the maximal energy cost decreasing to 12, shown as the adjusted assignment box. This process is terminated when there is no further improvement. After we obtain the min max cost assignment, if the maximum cost among all agents is smaller than the energy limitation, then a feasible schedule is obtained and returned. Otherwise, the *energy-aware scheduling* in section 4.3 is employed.

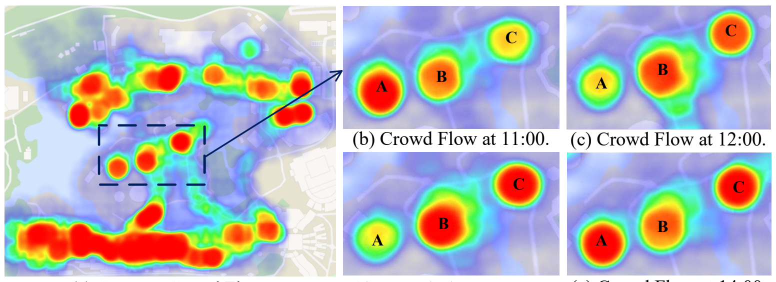
## 4.3 Energy-Aware Scheduling

**Overview.** If the remaining energy of the agents is not enough to execute the multi-level max cover scheduling, a heuristic energy-aware scheduling method is employed, which allocates agents to approximate the multi-level max cover scheduling with the energy constraint.

We first perform the data analysis over the historical crowd flow dataset, where we find two unique properties:

- *Spatial hot spots.* The crowd flow is not uniformly distributed. For example, Figure 6(a) gives an overview of the daily averaged crowd flows, where several spatial hot spots can be identified.

- *Spatio-temporal correlation.* In consecutive time intervals, people trend to move to nearby locations. For example, Figures 6(b)-(e) show the changes of crowd flows from 11:00 to 14:00, where A, B, and C are all relatively hot but with minor differences.



(a) Average Crowd Flow. (b) Crowd Flow at 11:00. (c) Crowd Flow at 12:00. (d) Crowd Flow at 13:00. (e) Crowd Flow at 14:00.

Fig. 6. Insights from Crowd Flow Data.

Above observations give us two insights: 1) the crowd flows are highly skewed, so a relatively good crowd coverage can be obtained even if we let the agents stay in those spatial hot spots over the day; 2) the difference of the crowd flow spatial distributions in adjacent time steps is minor, so the location adjustment is not necessary for all agents in each time step.

To realize the insights, we propose a hill climbing [27] based heuristic scheduling algorithm *energy-aware scheduling*, which consists two steps: 1) *policy initialization*, and 2) *greedy adjustment*. The pseudo-code is given in Algorithm 2. In *policy initialization* step, an allocation strategy that fulfills the energy limitation is obtained (Line 1). The strategy assigns the agents to maximal coverage locations of future averaged crowd flows inspired by the first insight. In *greedy adjustment* step, a greedy location adjustment is performed to iterate over all agents and all the possible adjustments at the remaining time steps to find the best location adjustment with maximal crowd coverage gain (Line 2-4). When we adjust the location for an agent at a time step, the locations of the other agents remain unchanged, inspired by the second insight.

**Algorithm 2** Energy-aware Scheduling.

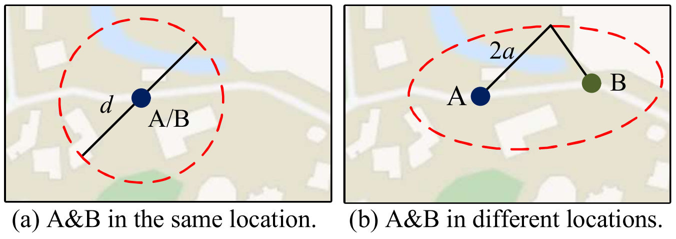
```
   Input: Predicted crowd flows $\hat{\Lambda}^{cur+1,n}$; status of agents $S^{cur}$;
   Output: Location of agents in future time steps $\{p_k^{cur+1,n}\}_1^K$;
   //Step 1. Policy Initialization
1: $\{p_k^{cur+1,n}\}_1^K \leftarrow PolicyInit(S^{cur}, \hat{\Lambda}^{cur+1,n})$;
   //Step 2. Greedy Adjustment
2: while there is still coverage gain do
3:     $a, T, g, gain \leftarrow SelectBest(S^{cur}, \{p_k^{cur+1,n}\}_1^K, \hat{\Lambda}^{cur+1,n})$;
4:     Update the assignment for $a$ to $g$ at $T$;
5: return $\{p_k^{cur+1,n}\}_1^K$
```

**Step 1. Policy Initialization.** In this step, a feasible allocation strategy is obtained by assigning agents to locations of the maximal future average crowd flow within energy limitation. The intuition is that even if the agents do not move in any future time steps, we can still get relatively good crowd coverage.

However, due to the energy limitation, not all of the locations are feasible for an agent to stay so that they will still have energy going back to the charge station. So the candidate location set should be obtained firstly.

Individually checking whether a location is feasible is time-consuming, thus a spatial pruning technique is proposed. We calculate the candidate set for each agent to reduce the action space according to its current location, remaining energy, and the destination.

We first discuss an ideal case, where the energy consumption is proportional to the Euclidean distance the agent travels between target locations. We denote the current location as $A$, and the destination (charge station) as $B$, there are two possible cases to calculate the candidate location set:



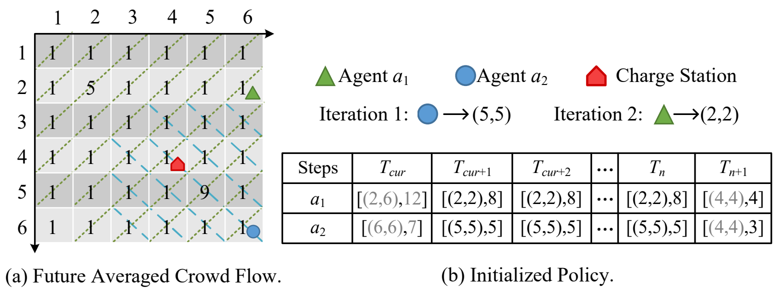
d
A/B
2a
A
B
(a) A&B in the same location.
(b) A&B in different locations.

Fig. 7. Candidate Location Set.

(1) *A and B are in the same location.* As demonstrated in Figure 7(a), the candidate location set is a circle with $A$ or $B$ as the center and the diameter $d$ is the distance can be traveled by the remaining energy. In another word, the agent cannot go to locations which cost more than $1/2$ remaining energy.
(2) *A and B are in different locations.* The candidate location set is an ellipse with $A$ and $B$ as the foci (in Figure 7(b)), and the distance $2a$ is distance can be traveled by the remaining energy.

In a more realistic case, as we discussed in Section 2.1, the energy consumption is related to the road network distance and other geographical factors between locations. The solution is that we can first derive a largest possible traveling distance from the remaining energy (e.g., going straight on the plain ground), and then use that distance to obtain an upper bound of the candidate location set. After such a spatial bound is calculated, we can obtain the actual candidate location set based on pre-computed energy cost through a refinement process.

After the candidate location set is obtained, the candidate location among all agents with the maximum coverage is selected. If a location is covered by more than one candidate location set, the agent with a smaller energy cost is assigned. This process is repeated until all agents are assigned.



| Steps | $T_{cur}$ | $T_{cur+1}$ | $T_{cur+2}$ | ⋯ | $T_n$ | $T_{n+1}$ |
|---|---|---|---|---|---|---|
| $a_1$ | [(2,6),12] | [(2,2),8] | [(2,2),8] | ⋯ | [(2,2),8] | [(4,4),4] |
| $a_2$ | [(6,6),7] | [(5,5),5] | [(5,5),5] | ⋯ | [(5,5),5] | [(4,4),3] |

Fig. 8. Policy Initialization Example.

**Example.** For example, in Figure 8, we have 2 agents and a charge station. Suppose $r = 1$, and the energy cost between two locations is Manhattan distance. At time step $T_{cur}$, the two agents are at $(2, 6)$ and $(6, 6)$ with remaining energy 12 and 7 respectively. The future averaged crowd flow is shown in Figure 8(a), in which the candidate locations are marked. In the first iteration, the location with the maximal coverage value $(5, 5)$ is selected. Although both agents can reach this location, we assign it to the closest one, i.e., $a_2$. Then we assign the location with the secondary large coverage value $(2, 2)$ to $a_1$. Finally, we obtain a feasible policy as Figure 8(b).

**Step 2. Greedy Adjustment.** To get more coverage, a greedy adjustment is performed by adjusting locations for agents in future time steps, until there is no coverage gain, as shown in Algorithm 2 line 2-4. The coverage gain is defined as the difference of crowd coverage by replacing an original location with a new location while the other locations selected remain unchanged. In each iteration, a location $l$ with the maximal coverage gain for agent $a_k$ in a future time step $T_t$ is selected, and we update the assignment for agent $a_k$. This process continues until there is no further coverage improvement within agents' energy limitations.

The function *SelectBest* is described in Algorithm 3, which iterates over all agents and future time steps, to select an adjustment with the maximal coverage gain. In line 7, a candidate location set for adjustment is generated for each agent. Line 8 selects a location in the candidate set with the maximal coverage gain. Finally, the best adjustment with overall maximal coverage gain is returned.

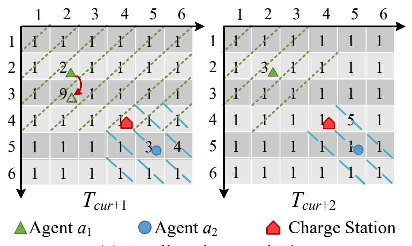
**Algorithm 3** SelectBest.

```
    Input: The current status of agents $S^{cur}$; the current policy $\{p_k^{cur+1,n}\}_1^K$;the predicted crowd flows $\hat{\Lambda}^{cur+1,n}$;
    Output: Adjusted agent $a^*$; adjusted time $T^*$; adjusted location $g^*$; adjusted coverage gain $gain^*$
1:  $a^*, T^*, g^* \leftarrow null$;
2:  $gain^* \leftarrow 0$;
3:  for $a_k \in \mathcal{A}$ do
4:      $r \leftarrow remain(s_k^{cur}, p_k^{cur+1,t_n}, g_\omega)$;          ▷ the remaining energy of the current schedule
5:      for $T_t \in \mathcal{T}$ do
6:          $avail \leftarrow r + cost(p_k^{t-1}, p_k^t) + cost(p_k^t, p_k^{t+1})$;   ▷ the total available energy by unfixing location assignment at $T_t$
7:          $C \leftarrow GetCandiLocations(p_k^{t-1}, p_k^{t+1}, avail)$;   ▷ obtain feasible candidate location within the energy limitation
8:          $g, gain \leftarrow SelectMax(C, \hat{\Lambda}^t, \{p_k^t\}_1^K, a_k, r)$;   ▷ select a location with maximum coverage gain among candidates
9:          if $gain > gain^*$ then
10:             $a^*, T^*, g^*, gain^* \leftarrow a_k, T_t, g, gain$;
11: return $a^*, T^*, g^*, gain^*$
```

The idea of generating candidate locations is similar to the approach in *policy initialization*, in which we can derive a candidate location upper bound using the current location and the remaining energy of an agent. When adjusting the location for agent $k$ in time step $T_t$, we keep the assignments in other time steps unchanged. To this end, we set the locations of the agent at $T_{t-1}$ and $T_{t+1}$ as the center/foci, and original remaining energy before adjustment plus the consumption between $[T_{t-1}, T_{t+1}]$ as the available energy for this adjustment, as shown in Line 6.

**Example.** Following the example in Figure 8, to get the candidate adjustment locations of $a_1$ at $t_{cur+1}$, we first calculate the available energy by recycling the energy cost from $T_{cur}$ to $T_{cur+2}$ to the final remaining, i.e., $4 + 4 + 0 = 8$. The bound of candidate adjustment locations is a ellipse with $(2, 6)$ and $(2, 2)$ as its foci, and 8 as the length of major axis. After refinement, the candidate adjustment locations of both agents at $T_{cur+1}$ and $T_{cur+2}$ are marked in Figure 9(a). We then derive the locations with max coverage gain in all time steps and agents, as shown in Figure 9(b). In this example, if we move $a_1$ from $(2, 2)$ to $(3, 2)$, the max coverage gain, $9 - 2 = 7$, is achieved. Finally, we update the current policy, as shown in Figure 10.

**Analysis.** The hill-climbing based heuristic algorithm is not bound to find the optimal solution, and there exist many meta-heuristic algorithms like tabu search [10], simulated annealing [41] to avoid getting stuck in local



(a) Predicted Crowd Flow.

| Time | Loc | Gain |
|---|---|---|
| $T_{cur+1}$ | (3,2) | 7 |
| $T_{cur+2}$ | (2,2) | 0 |
| ··· | ··· | ··· |

$a_1$

| Time | Loc | Gain |
|---|---|---|
| $T_{cur+1}$ | (5,6) | 1 |
| $T_{cur+2}$ | (4,5) | 4 |
| ··· | ··· | ··· |

$a_2$

Max Gain

| Agent | Time | Loc | Gain |
|---|---|---|---|
| $a_1$ | $T_{cur+1}$ | (3,2) | 7 |

(b) Select Max Coverage Gain Location.

Fig. 9. Greedy Adjustment Example.

| Steps | $T_{cur}$ | $T_{cur+1}$ | $T_{cur+2}$ | ··· | $T_n$ | $T_{n+1}$ |
|---|---|---|---|---|---|---|
| $a_1$ | [(2,6),12] | [(3,2),7] | [(2,2),6] | ··· | [(2,2),6] | [(4,4),2] |

Fig. 10. Update Allocation Policy for $a_1$.

optima. However those optimization techniques are not used for following three reasons: 1) the initial policy already reaches a good crowd coverage due to the spatial hot spots of the crowd flows explained in Figure 6; 2) those algorithms use random "noise" to escape the local optima, which introduce higher computational costs to find a better solution and are not efficient in real-time scheduling; 3) the predictions of the crowd flows are not totally perfect, even if we spend huge efforts to obtain a higher crowd coverage based on the predictions, the improvement towards the true crowd flows is also limited. Therefore, those methods are not discussed in the paper, but we do compare the performance of EADS with the optimal solution in Section 5.3.

## 5 EXPERIMENTS

In this section, we conduct extensive experiments to evaluate the effectiveness and efficiency of our system. We first describe the real dataset used and the experimental setup in the paper. Then the comparisons of different deploy strategies and algorithms under different parameter settings are evaluated. Finally, a case study in Happy Valley demonstrates the effectiveness of our system.

### 5.1 Experimental Settings

**Datasets.** The following 4 datasets are used in the experiments.

- *Crowd Flows.* We scraped the crowd flow data in Beijing Happy Valley from 1/1/2018 - 10/31/2018 from Tencent crowd flow heat map [3]. Each location (a $10m \times 10m$ grid) has a crowd flow observation each hour. The area of interest is about $5.5 \times 10^5 m^2$, containing 51×108=5,508 uniform grids. The average hourly crowd flows are 6,470, and the average daily crowd flows are 77,650, which are the accumulation of hourly crowd flows. In order to train a more accurate prediction model, we also obtain some external features from public websites, including the weather [4], which contains outlook, temperature, and wind speed in every hour, and the promotional information (i.e., time-variant ticket prices) from the account of Happy Valley in WeChat.
- *Trash Bins.* We collect the real locations of 95 trash bins in Happy Valley by an on-field investigation. The spatial distribution of trash bins is visualized in Figure 1.

[3]https://heat.qq.com/
[4]http://www.weather.com.cn/weather/101010300.shtml

**Baselines.** We compare our proposed algorithm *EADS* against the following two dynamic scheduling baselines. The first one the optimal solution, and the second one is an aggressive scheduling strategy.

- *B&B.* We employ the commonly used branch and bound [18] algorithm to obtain the optimal solution of the ILP problem through a modern ILP solver PuLP[5].
- *MYOPIC*, which is an aggressive allocation strategy. It schedules agents only based on the predicted crowd flows in the next time step. The maximal coverage location within the energy limitation is selected and assigned for each agent in the remaining energy increasing order.

**Variants.** We also compare *EADS* with two different types of variants.

- *EADS-Inf.* It is the *EADS* scheduling strategy with adequate energy case, which means only the multi-level max cover scheduling will be executed at each decision time.
- *EADS-nMPC.* Instead of adjusting the scheduling decision at each decision time following the MPC principle, this method plans the schedules only *once* at the beginning of the service time.

For dynamic policies, the crowd flow prediction model is used by default. But in order to evaluate their theoretical performances if the prediction model is perfect, the scheduling results based on future ground-truth crowd flows are also shown with methods in *(Theo.)* suffix.

**Implementations.** All the above algorithms are implemented in Python. The multi-step crowd flow prediction model is implemented by MXNet and trained with one NVIDIA 1080Ti GPU. Besides, experiments are conducted on a workstation with an Intel(R) Xeon(R) CPU E3-1225 @ 3.3GHz, 16GB memory, and Windows 10 OS.

**Evaluation Methods.** The allocation performance is evaluated during 10/1/2018-10/31/2018. According to the opening hours of Happy Valley, the first service time interval is 10:00-11:00, while the last service time interval is 21:00-22:00. Besides, the decision interval is an hour, the charge station is located in the center of the study area, and the energy consumption calculation is based on Euclidean distance due to no extra geographical data. We conduct our dynamic allocation on each day, and the averaged daily crowd coverage and the averaged running time per decision over evaluation days are used as the evaluation metrics. The averaged daily crowd coverage (ADCC) is defined as:

$$ADCC = \frac{\sum_{d \in \mathcal{D}} \sum_{T_t \in \mathcal{T}} cov^d_{T_t}}{|\mathcal{D}|} \tag{3}$$

where $\mathcal{D}$ is the evaluation day set, $\mathcal{T}$ is service time intervals of each day, and $cov^d_{T_t}$ is the crowd coverage during time interval $T_t$ at day $d$.

**Tranining Details & Hyperparameters.** To obtain the multi-step crowd flow prediction model, we use the mean squared error as the loss function and leverage Adam [17] to perform network training. The initial learning rate is set as 0.01, and we apply the learning rate decay every 10 epochs with a ratio of 0.1. We use a ratio of 9:1 to split the dataset in temporal range 1/1/2018-9/30/2018 for training and validation. For hyperparameters, we use records of previous 7 hours ($l_{hist} = 7$) to predict crowd flows of future 12 hours ($l_{pred} = 12$) because we need to look 12 steps ahead till the end of the service time at the first decision time of each day. We use 1 MFDense layer, and the region embedding size is set to 4. Those are two internal hyperparameters of MF-STResNet [33].

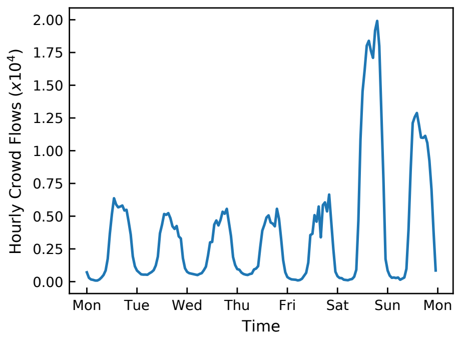
[5]https://pythonhosted.org/PuLP/

## 5.2 Dataset Descriptions

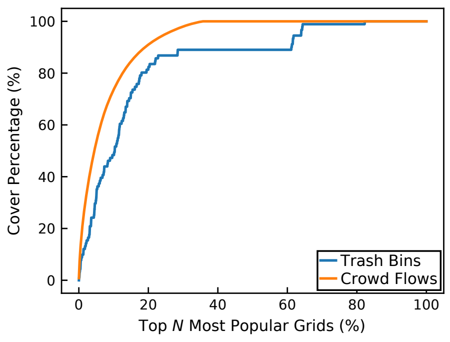
In this subsection, we provide some statistics of the crowd flows and the collected trash bins.

**Crowd Flow Distributions.** The temporal distribution of hourly crowd flows in the area of interest during one week (10/08/2018-10/14/2018) is depicted in Figure 11(a). It is obvious that the crowd flows in the weekend is about 2 ∼ 3 times more than workdays because the Happy Valley is a place for entertainment. In addition to that, though the changes of crowd flows in weekdays (or on weekends) are similar, they are still slightly different from day to day, which indicates the necessity of using the prediction model. We also characterize the spatial distribution of crowd flows in Figure 11(b). We rank the spatial grids according to historical crowd flows in descending order, and calculate the crowd cover percentage in the whole area with the increasing of the taking up of top grids. The result shows that 80% crowds are distributed in the top 12.6% grids, which indicates the spatial distribution of crowd flows is highly skewed.

**Correlation between Crowd Flows and Trash Bins.** As we want to evaluate the effectiveness of public resource reduction using the trash bin as an evaluation example, we further demonstrate the correlation between the spatial distribution of crowd flows and trash bins. For comparison convenience, we also plot the trash bin cover percentage in the whole area with the increasing of the taking up of top popular grids in Figure 11(b). We found that 80% trash bins are distributed in the top 18.0% grids, which is similar to the result of crowd flows. It is also interesting to see that about 16% trash bins are distributed in grids with rare crowd flows. We guess it is due to the trash bins are deployed in the early time of the park. With the rapid development and facility adjustment of the park, those trash bins fail to serve the crowds nowadays.



(a) Temporal distribution in one week.



(b) Spatial distributions.

Fig. 11. Dataset Descriptions.

## 5.3 Effectiveness Evaluation

In this subsection, we demonstrate the necessity of dynamic resource allocation, and the effectiveness as well as the efficiency of the proposed *EADS* compared with the optimal solution.

**Different Deploy Strategies.** We compare different deploy strategies in the average daily crowd coverage (ADCC) when service radius $r = 1$ (about 10m) and energy limitation $E = 50$ (about 500m) in Figure 12. The *Current* strategy is the current deployment of 95 trash bins in the theme park. And the *Static* strategy selects maximal coverage locations based on the historical average crowd flows, and fixes agents on those locations.

With the increase in resources, all methods obtain higher daily crowd coverage. Results indicate that all dynamic methods are significantly superior to the *Static* deployment approach. Comparing *EADS* and *EADS-nMPC*, the effectiveness of MPC principle is demonstrated, which mitigates the prediction and schedule error using the re-planning strategy. To meet the same coverage value as the current trash bin deployment, 44 resources are required for *Static*, 33 resources are required for *EADS-nMPC*, 19, 13, 7 resources are needed for *EADS*, *EADS-Inf*, and *EADS-Inf (Theo.)*, respectively. The gap between *EADS-Inf (Theo.)* and *EADS-Inf* is due to the prediction error, which could be reduced if more training data is given. However, even based on the current prediction model and the energy limitation, *EADS* achieves 80.0% and 56.8% resource reduction with respect to *Current* and *Static* deployment strategies. The comparison demonstrates the effectiveness and necessity of the proposed dynamic scheduling.

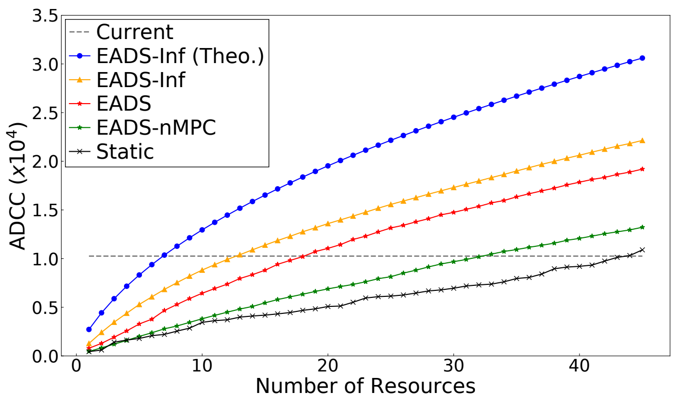


Fig. 12. Deploy strategy comparison.

**Comparison with the Optimal Solution.** Since *EADS* is a heuristic-based solution, we are interested to know its effectiveness and efficiency compared with the optimal algorithm. Nevertheless, with the large action space, the optimal algorithm is not practical in real-world scenarios. Therefore, we aggregate the study area into 5x10=50 grids, and vary the number of agents from 1 to 3, while the other parameters remain unchanged. Note that when we increase the number of agents to 4, no result returned from the *B&B* algorithm after an hour waiting, which is impractical in real scenarios, so we didn't add it into comparison. The results of *EADS* and *B&B*, as well as their theoretical versions based on the ground-truth crowd, flows with respect to the different number of resources are reported in Figure 13. The heuristic baseline *MYOPIC* is also added for comparison. From the perspective of the ADCC, the proposed *EADS* has very similar scheduling quality compared with the optimal solution. If the prediction model is perfect, *B&B (Theo.)* is only 1.5% better than *EADS (Theo.)* when the number of resources is 3. Interestingly, when the prediction model is used, *EADS* is even a little bit better. This is because the optimal schedule "overfits" the prediction results. Therefore, we believe that in the in the real-world scenarios, *EADS* is effective enough comparing with the optimal solution. *MYOPIC* is better when there is only one resource but shows much worse performance when there are multiple resources. But its theoretical version is definitely worse than others. We think the reason might be when the resource is very few, the error caused by prediction dominates the inappropriate scheduling issue. As for the average decision time, the running time of *B&B* grows exponentially due to larger action space, while the decision time for *EADS* doesn't exceed 12ms in the same settings. *MYOPIC* is the fastest, because only the crowd flows of the next time step are considered when it makes decisions, while others are based on long-term crowd flow changes.

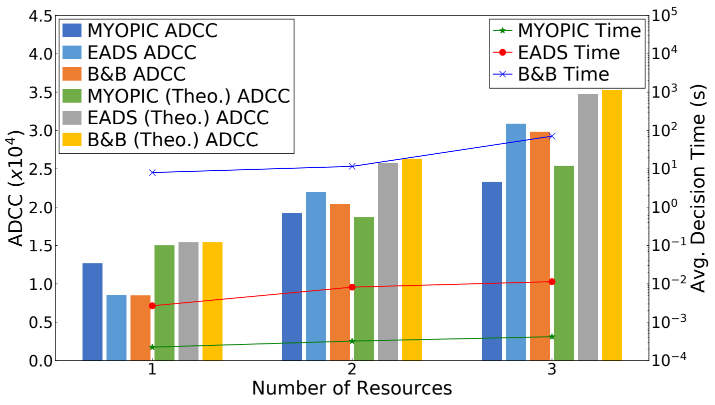


Fig. 13. Comparison with the optimal.

## 5.4 Effect of Different Parameters

In this subsection, we further evaluate the effectiveness and efficiency of *EADS* under different parameter settings. *MYOPIC*, which is also a dynamic policy, is used as the competitor.

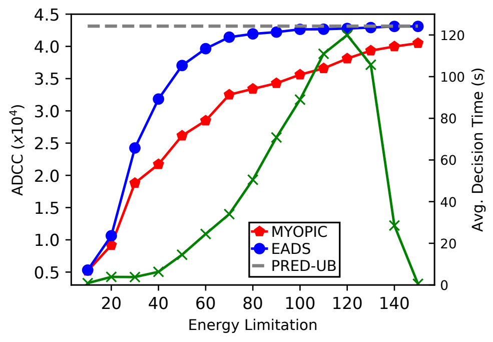
**Parameters.** Table 1 shows all parameters in the following experiments, where the default settings are highlighted.

Table 1. Parameter Settings.

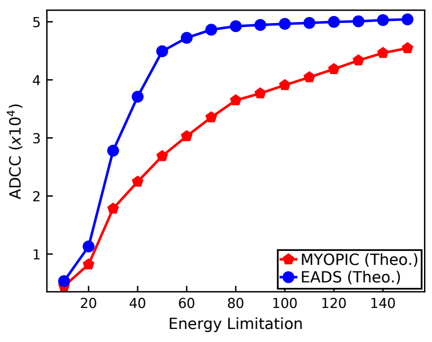
| **Parameters** | **Settings** |
|---|---|
| Energy Limitation | 10,20,...,**50**,...,150 |
| Number of Resources | 10,**20**,30 |
| Service Radius | 1,**2**,3 |

**Different Energy Limitation.** Figure 14(a) gives the average daily crowd coverage with respect to different energy limitations of each agent, and the coverage value upper bound without energy limitation is given in the grey dash line. With larger energy limitations, both methods lead to higher crowd coverage, but *EADS* is faster to reach the coverage upper bound with smaller energy limitations. The average decision efficiency is given in the same figure as the green line. With the increase of the energy limitation, the decision time first increases, because there are more locations feasible for an agent to go. However, when the energy continues getting larger, the running time decreases. This is because, with larger energy, the multi-level max cover scheduling plan becomes feasible, which requires much less planning time. In general, the maximal decision time doesn't exceed 2min, which shows the energy adaptive property of our proposed algorithm. We also evaluate the performance of *EADS* and *MYOPIC* based on the future real crowd flows in Figure 14(b). Under the perfect prediction of crowd flows, the advantage of *EADS* is more obvious when the energy is limited. The reason is that when *EADS* makes allocation decisions, the changes of crowd flows in the future are also taken into consideration, and the energy is used wisely. But *MYOPIC* makes the decision blindly, which only takes the crowd flows at the next time step into consideration.

**Different Number of Resources.** Figure 15(a) gives the average daily crowd coverage with respect to different numbers of resources. As can be seen, *EADS* outperforms *MYOPIC* in different resource settings. Besides, more crowd coverage can be obtained with more agents, because more locations can be served at the same time. The



(a) Different Energy Limitation.

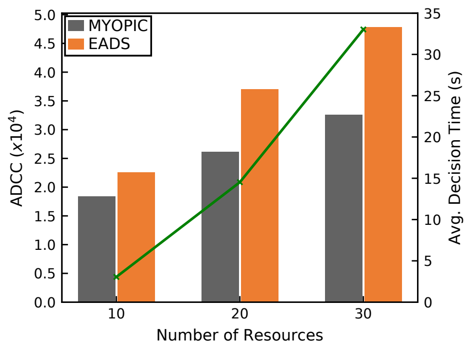


(b) Under Perfect Prediction.

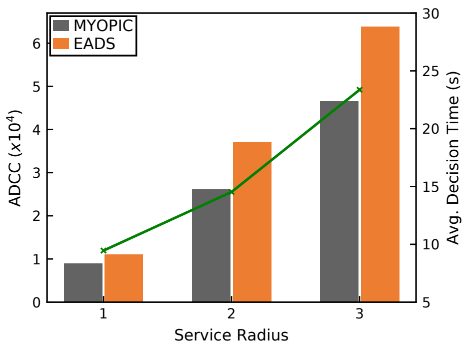
Fig. 14. Different energy limitation.

average decision time of *EADS* is also shown in the same figure as the green line. With the increasing of agents, the running time increases due to more decision choices to be made.

**Different Service Radius.** Figure 15(b) gives the average daily crowd coverage with respect to different service radius of each agent. *EADS* outperforms *MYOPIC* in different radius settings. Furthermore, with a larger service radius, more crowd coverage is obtained. The service radius is an important factor that influences the coverage value with the same number of resources since a larger service radius means much fewer resources are needed to deploy. We also report the efficiency results with the green line. With the increase of the service radius, the running time grows, since each agent has more location choices in order to serve a certain location.



(a) Different Number of Resources.



(b) Different Service Radius.

Fig. 15. Different number of agents and service radius.

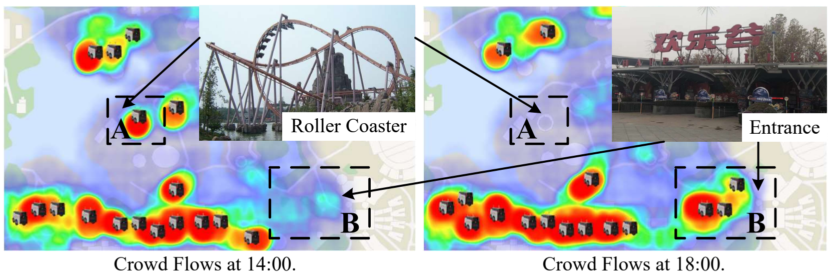
## 5.5 Case Study

In order to evaluate the effectiveness of our dynamic resource allocation system, we conduct the dynamic allocation on a Saturday, 10/13/2018 in simulation.

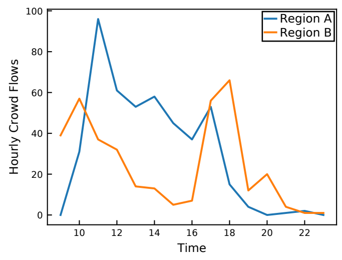
The spatial and temporal distribution of crowd flows is given in Figure 16. It can be found in Figure 16(a) that region A is popular at 14:00, but the flows decline and vanish at 18:00. In contrast, the crowd flows in the region B are not dense at 14:00, but the region becomes a hot spot at 18:00. The temporal distributions of hourly crowd flows in two different functional grids over the day are also given in Figure 16(b). It shows the temporal distributions of crowd flows are significantly different in those regions during the day.

After a thorough investigation in the Happy Valley, we believe that high crowd flows are related to the operation time of entertainment facilities. In region A, a suspended flying roller coaster is closed at night for security issues, and it explains why the crowd flow drops at 18:00. In region B, the crowd flows correlates with the price of daytime and nighttime ticket. Since the price of the nighttime ticket is much cheaper than the price of the daytime ticket, there are fewer people entering the park at 14:00. And at 18:00, some people leave the park after playing in the daytime, and some people enter the park with night tickets, which makes the region popular again.

As shown in the figure, our allocation system can successfully predict such variant of crowd flows and allocate the agents to the target locations at different time intervals. On that day, using *EADS*, the daily crowd coverage is increased by 87.25% compared with the static maximal coverage deployment strategy based on historical crowd flows under the same number of resources setting.



(a) Visualization.



(b) Temporal distributions.

Fig. 16. Case study.

## 6 RELATED WORKS

**Optimal Control of Dynamic Systems.** Reinforcement learning (RL) and model predictive control (MPC) are the major methods for optimal control of dynamic systems [7, 12]. Most RL algorithms are model-free, and directly learn the policy from the interaction with the environment. In our problem, the action space of agents is prohibitively large, even if defined in a multi-agent way [24, 25], which leads to huge computational cost. Besides, the constraint handling of RL is immature [12, 34].

On the contrary, MPC is model-based, and the feasibility is naturally guaranteed. MPC is particularly helpful for a stochastic and time-varying model. In some literature [15, 29], model predictive control is also regarded as a model-based reinforcement learning approach. An MPC-based approach is proposed in [13] to compute the optimal re-balancing strategy of autonomous using short-term demand forecasts. The problem is formulated as a mixed integer linear programming problem, and the optimal solution is found using MILP solver. However, the number of locations in their settings is much smaller than ours, which makes the optimal solution solvable in real time.

**Orienteering Problem.** The main objective of orienteering problem (OP) [11, 40] selects a subset of nodes and define the sequence of selected nodes so that the score is maximized without exceeding the maximum total travel time (or distance). One of its variants is team orienteering problem with time window (TOPTW) [28, 42], when the number of route considered is more than one, and each node has a time window constraint. All of these solutions focus on optimizing robust solutions in a deterministic environment. [14] solves an OP which takes the profit with normal distribution into consideration, which focuses on finding a robust *preplanned* solution. To the our best knowledge, [26] is the only work to find a policy for TOPTW that reacts to real-time stochastic visiting duration time. However, our problem is very different, as in our settings, the score is stochastic. Besides, all of orienteering problems assume that the score of each node is entirely addictive [43], which is not true in our problem, especially when $r > 1$.

**Public Resource Allocation.** The allocation of public resources can be divided into two parts, i.e., static allocation and dynamic allocation. The static public resource allocation, is similar to location selection with fixed number of resources, which has been extensively studied [3, 5, 23]. The objective varies according to different requirements, e.g., minimize the average (or max) cost to reach all clients [3, 5], maximize the number of covered trajectories [23]. There are few works focusing on the dynamic public resource allocation. Existing works of dynamic resource allocation is mainly discussed in the cloud environment [37, 44] due to its applications in data center. [35] dynamically allocates the computing resources in different base station for mobile edge computing scenario. In this paper, we focus on dynamically allocating the energy constrained mobile agents to maximize the crowds that can be served.

## 7 CONCLUSION

In this paper, we provide the first attempt to study the dynamic public resource allocation problem and present a solution framework incorporating long-term crowd flow prediction and multi-agent scheduling. We define the scheduling task as a multi-agent long-term maximal coverage scheduling (MALMCS) problem, which is formulated into an integer linear programming (ILP) problem. MALMCS takes the crowd coverage and energy limitation during the whole day into consideration, and we prove its NP-hardness. Due to the long-term prediction is not accurate, we employ the principle of MPC, which makes the schedule for a long-term time horizon, but only executes actions in the first step. To overcome the challenges of maximizing the crowd coverage with limited resources, energy limitations, and large action space, a two-stage heuristic algorithm, i.e., energy adaptive scheduling (EADS) is proposed. In our system, we first build a multi-step crowd flow prediction model, which incorporates external factors. Then, EADS is employed to address MALMCS in real-time scenarios. Extensive experiments based on real crowd flow data in Beijing Happy Valley are performed, which demonstrates the effectiveness and efficiency of our proposed dynamic public resource allocation system. Our system achieves 80.0% and 56.8% resource reduction with respect to the current and the static maximal coverage deployment strategies, respectively, and EADS significantly outperforms the aggressive allocation strategy. At the same time, EADS has competitive performance against the optimal solution, however, the planning time is orders of magnitude faster. In this paper, the traveling time cost and the capacity (if applicable) of the agents are not considered. In the future, with fine-grained crowd flow data and real-time demands, those constraints can also be taken into consideration.

## ACKNOWLEDGMENTS

This work was supported by the National Key R&D Program of China (2019YFB2101805), the NSFC Grant (61672399, 61761136020), and NSFC-Zhejiang Joint Fund for the Integration of Industrialization and Informatization (U1609217). Yanhua Li was supported in part by NSF grants CNS-1657350 and CMMI-1831140, and a research grant from DiDi Chuxing Inc. Yingcai Wu was also supported by Zhejiang Provincial Natural Science Foundation (LR18F020001).

## REFERENCES

[1] Rainer Burkard, Mauro Dell'Amico, and Silvano Martello. 2012. *Assignment problems, revised reprint.* Vol. 106. Siam.

[2] P Carraresi and Giorgio Gallo. 1984. A multi-level bottleneck assignment approach to the bus drivers' rostering problem. *EJOR* (1984).

[3] Zitong Chen, Yubao Liu, Raymond Chi-Wing Wong, Jiamin Xiong, Ganglin Mai, and Cheng Long. 2014. Efficient algorithms for optimal location queries in road networks. In *SIGMOD.* ACM, 123–134.

[4] Cedric De Cauwer, Joeri Van Mierlo, and Thierry Coosemans. 2015. Energy consumption prediction for electric vehicles based on real-world data. *Energies* 8, 8 (2015), 8573–8593.

[5] Ke Deng, Shazia Sadiq, Xiaofang Zhou, Hu Xu, Gabriel Pui Cheong Fung, and Yansheng Lu. 2012. On group nearest group query processing. *TKDE* 24, 2 (2012).

[6] Trivikram Dokka, Anastasia Kouvela, and Frits CR Spieksma. 2012. Approximating the multi-level bottleneck assignment problem. In *International Workshop on Algorithms and Computation.* Springer, 64–75.

[7] Damien Ernst, Mevludin Glavic, Florin Capitanescu, and Louis Wehenkel. 2009. Reinforcement learning versus model predictive control: a comparison on a power system problem. *Cybernetics* 39, 2 (2009), 517–529.

[8] Carlos E Garcia, David M Prett, and Manfred Morari. 1989. Model predictive control: theory and practice a survey. *Automatica* 25, 3 (1989), 335–348.

[9] Xu Geng, Yaguang Li, Leye Wang, Lingyu Zhang, Qiang Yang, Jieping Ye, and Yan Liu. 2019. Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting. In *2019 AAAI Conference on Artificial Intelligence (AAAI'19).*

[10] Fred Glover. 1989. Tabu search—part I. *ORSA Journal on computing* 1, 3 (1989).

[11] Bruce L Golden, Larry Levy, and Rakesh Vohra. 1987. The orienteering problem. *Naval Research Logistics (NRL)* 34, 3 (1987), 307–318.

[12] Daniel Görges. 2017. Relations between Model Predictive Control and Reinforcement Learning. *IFAC-PapersOnLine* 50, 1 (2017), 4920–4928.

[13] Ramon Iglesias, Federico Rossi, Kevin Wang, David Hallac, Jure Leskovec, and Marco Pavone. 2018. Data-driven model predictive control of autonomous mobility-on-demand systems. In *ICRA.* IEEE, 1–7.

[14] Taylan Ilhan, Seyed MR Iravani, and Mark S Daskin. 2008. The orienteering problem with stochastic profits. *Iie Transactions* 40, 4 (2008), 406–421.

[15] Sanket Kamthe and Marc Peter Deisenroth. 2018. Data-efficient reinforcement learning with probabilistic model predictive control. *AISTATS.*

[16] Samir Khuller, Anna Moss, and Joseph Seffi Naor. 1999. The budgeted maximum coverage problem. *Information processing letters* 70, 1 (1999), 39–45.

[17] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[18] Eugene L Lawler and David E Wood. 1966. Branch-and-bound methods: A survey. *Operations research* 14, 4 (1966), 699–719.

[19] Seungseob Lee, SuKyoung Lee, Kyungsoo Kim, and Yoon Hyuk Kim. 2015. Base station placement algorithm for large-scale LTE heterogeneous networks. *PloS one* 10, 10 (2015).

[20] Ruiyuan Li, Huajun He, Rubin Wang, Yuchuan Huang, Junwen Liu, Sijie Ruan, Tianfu He, Jie Bao, and Yu Zheng. 2020. JUST: JD Urban Spatio-Temporal Data Engine. IEEE.

[21] Ruiyuan Li, Huajun He, Rubin Wang, Sijie Ruan, Yuan Sui, Jie Bao, and Yu Zheng. 2020. TrajMesa: A Distributed NoSQL Storage Engine for Big Trajectory Data. In *ICDE.* IEEE.

[22] Yuhong Li, Jie Bao, Yanhua Li, Yingcai Wu, Zhiguo Gong, and Yu Zheng. 2016. Mining the most influential k-location set from massive trajectories. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems.* ACM, 51.

[23] Yuhong Li, Jie Bao, Yanhua Li, Yingcai Wu, Zhiguo Gong, and Yu Zheng. 2018. Mining the Most Influential $k$-Location Set from Massive Trajectories. *TBD* (2018).

[24] Yexin Li, Yu Zheng, and Qiang Yang. 2018. Dynamic Bike Reposition: A Spatio-Temporal Reinforcement Learning Approach *(KDD).*

[25] Kaixiang Lin, Renyu Zhao, Zhe Xu, and Jiayu Zhou. 2018. Efficient Large-Scale Fleet Management via Multi-Agent Deep Reinforcement Learning *(KDD '18).*

[26] Yi Mei and Mengjie Zhang. 2018. Genetic Programming Hyper-Heuristic for Stochastic Team Orienteering Problem with Time Windows. In *CEC.* IEEE, 1–8.

[27] Wil Michiels, Emile Aarts, and Jan Korst. 2007. *Theoretical aspects of local search.* Springer Science & Business Media.

[28] Roberto Montemanni and Luca Maria Gambardella. 2009. An ant colony system for team orienteering problems with time windows. *Foundation Of Computing And Decision Sciences* 34, 4 (2009), 287.

[29] Anusha Nagabandi, Gregory Kahn, Ronald S Fearing, and Sergey Levine. 2018. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *ICRA.* IEEE, 7559–7566.

[30] Myriam Neaimeh, Graeme A Hill, Yvonne Hübner, and Phil T Blythe. 2013. Routing systems to extend the driving range of electric vehicles. *IET Intelligent Transport Systems* 7, 3 (2013), 327–336.

[31] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. 1978. An analysis of approximations for maximizing submodular set functions-I. *Mathematical programming* 14, 1 (1978), 265–294.

[32] Zheyi Pan, Yuxuan Liang, Weifeng Wang, Yong Yu, Yu Zheng, and Junbo Zhang. 2019. Urban Traffic Prediction from Spatio-Temporal Data Using Deep Meta Learning. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19)*. ACM, New York, NY, USA, 1720–1730.

[33] Zheyi Pan, Zhaoyuan Wang, Weifeng Wang, Yong Yu, Junbo Zhang, and Yu Zheng. 2019. Matrix Factorization for Spatio-Temporal Neural Networks with Applications to Urban Flow Prediction. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. ACM, 2683–2691.

[34] Tu-Hoa Pham, Giovanni De Magistris, and Ryuki Tachibana. 2018. OptLayer-practical constrained optimization for deep reinforcement learning in the real world. In *CRA*. IEEE, 6236–6243.

[35] Jan Plachy, Zdenek Becvar, and Emilio Calvanese Strinati. 2016. Dynamic resource allocation exploiting mobility prediction in mobile edge computing. In *PIMRC*. IEEE, 1–6.

[36] Sijie Ruan, Cheng Long, Jie Bao, Chunyang Li, Zisheng Yu, Ruiyuan Li, Yuxuan Liang, Tianfu He, and Yu Zheng. 2020. Learning to generate maps from trajectories. AAAI.

[37] AT Saraswathi, Y RAb Kalaashri, and S Padmavathi. 2015. Dynamic resource allocation scheme in cloud computing. *Procedia Computer Science* 47 (2015), 30–36.

[38] Salvatore Scellato, Anastasios Noulas, Renaud Lambiotte, and Cecilia Mascolo. 2011. Socio-spatial properties of online location-based social networks. In *Fifth international AAAI conference on weblogs and social media*.

[39] Ravi Shankar and James Marco. 2013. Method for estimating the energy consumption of electric vehicles and plug-in hybrid electric vehicles under real-world driving conditions. *IET intelligent transport systems* 7, 1 (2013), 138–150.

[40] Theodore Tsiligirides. 1984. Heuristic methods applied to orienteering. *Journal of the Operational Research Society* 35, 9 (1984), 797–809.

[41] Peter JM Van Laarhoven and Emile HL Aarts. 1987. Simulated annealing. In *Simulated annealing: Theory and applications*. Springer, 7–15.

[42] Pieter Vansteenwegen, Wouter Souffriau, Greet Vanden Berghe, and Dirk Van Oudheusden. 2009. Iterated local search for the team orienteering problem with time windows. *Computers & Operations Research* 36, 12 (2009), 3281–3290.

[43] Pieter Vansteenwegen, Wouter Souffriau, and Dirk Van Oudheusden. 2011. The orienteering problem: A survey. *EJOR* (2011).

[44] Andreas Wolke, Martin Bichler, and Thomas Setzer. 2016. Planning vs. dynamic control: Resource allocation in corporate clouds. *IEEE Transactions on Cloud Computing* 1 (2016), 1–1.

[45] SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. 2015. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*. 802–810.

[46] Huaxiu Yao, Xianfeng Tang, Hua Wei, Guanjie Zheng, and Zhenhui Li. 2019. Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction. In *AAAI Conference on Artificial Intelligence*.

[47] Jiani Zhang, Xingjian Shi, Junyuan Xie, Hao Ma, Irwin King, and Dit-Yan Yeung. 2018. Gaan: Gated attention networks for learning on large and spatiotemporal graphs. *arXiv preprint arXiv:1803.07294* (2018).

[48] Junbo Zhang, Yu Zheng, and Dekang Qi. 2017. Deep Spatio-Temporal Residual Networks for Citywide Crowd Flows Prediction.. In *AAAI*. 1655–1661.

[49] Ping Zhang, Zhifeng Bao, Yuchen Li, Guoliang Li, Yipeng Zhang, and Zhiyong Peng. 2018. Trajectory-driven influential billboard placement. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2748–2757.