Travelling Officer Problem: Managing Car Parking Violations Efficiently Using Sensor Data

Wei Shao, Flora D. Salim, Tao Gu, Ngoc-Thanh Dinh and Jeffrey Chan

***Abstract*—The on-street parking system is an indispensable part of civic projects, as it provides travellers and shoppers with parking spaces. With the recent in-ground sensors deployed throughout the Melbourne CBD, there is a significant problem on how to use the sensor data to manage parking violations and issue infringement notices efficiently in a short time-window. In this paper, we use a large real-world dataset with on-street parking sensor data from the local city council, and establish a formulation of the Travelling Officer Problem with a general probability-based model. We propose two solutions using a spatio-temporal probability model for parking officers to maximize the number of infringing cars caught with limited time cost. Using real-world parking sensor data and Google Maps road network information, the experimental results show that our proposed algorithms outperform the existing patrolling routes.**

***Index Terms*—Parking Sensor Data, Parking Violation Management, Smart Cities, Intelligent Transportation Systems (ITSs)**

I. Introduction

WITH cities growing rapidly in population and traffic volume, traditional parking management techniques face many challenges such as inefficient resource management, high human capital needs, and data noise. The Internet of Things (IoT) provides the capacity to deal with such challenges, as it can be designed to capture sensor data for monitoring areas of interest in smart cities. Recently, researchers have explored the potential use of the IoT in public transportation services and urban computing [1]. An increasing number of cities try to use an IoT-based framework in local transportation system management. Most existing works focus on finding car parking spaces for drivers; several models have been proposed to provide drivers with real-time information about available car parking bays nearby [2]. Only a few studies have been done to help authorities manage on-street parking more effectively and efficiently. Over the last few years, the Melbourne City Council has installed thousands of in-ground sensors in car parking bays located in the Melbourne central business district (CBD) [3]. These sensors can detect car parking events by recording the arrival time and departure time of a car. The parking system can check whether a car has overstayed the maximum permitted period within parking rules. These sensors will send a signal to the central station within five minutes of a car being in violation. Parking officers will be dispatched to specific locations to issue parking infringement notices on cars in violation.

Due to a lack of parking officers, drivers are often able to escape infringement notices. Therefore, catching violators in time is a critical issue for Melbourne City Council, not only for the potential financial benefits, but also for ensuring public compliance with local laws, and reducing repeat offences.

The current patrolling methodology works on a 'first-come first-served' (FCFS) basis. The parking officers will go to the next parking space with the earliest violation time. The existing system works well in a small area with fewer violations, though it is inefficient in busy areas. This is because the existing system does not consider the temporal and spatial information of violations, or the probability of a car leaving after a violation, and before officers have arrived.

In this paper, we formulate car parking fine collection as a Travelling Officer Problem (TOP), which aims to find an optimized path to maximize the probability of catching cars in violation, with limited time cost. We take into account the walking speed and behaviour of an officer, and also use spatio-temporal and historical information (probability of violation period). Based on this model, we propose two solutions using observed spatio-temporal information. The first solution is a greedy algorithm that employs probability estimation, and the second solution is a path-finding algorithm based on Ant Colony Optimization (ACO) [4]. The results show that the ACO-based algorithm performs more stably than both the existing approach, and the greedy algorithm. Both algorithms utilize Google Maps road network information, and an observed distribution of parking violations. We also build a system to simulate the real case, and conduct extensive experiments using real-world parking data provided by the Melbourne City Council [5]. The results show that both algorithms perform much better than the current approach.

Figure 1 illustrates a simple example of our problem and results from different algorithms. There are four car parking slots (P1 to P4) in violation around the CBD. The patrolling officer needs to go through these four parking slots to give infringement notices to each car in violation. The information centre sends the events sequentially as 'P1, P2, P3, P4. The first diagram of Figure 1 shows the officer's path under the current methodology, denoted by the blue line. The length of the complete trajectory is 1km, as measured by Google Maps. Using the greedy algorithm with a car-leaving probability estimation, the patrolling officer tends to look for car parking plots with the highest probability of the car not leaving. The green line (around 928 m) denotes the walking path

Wei Shao, Tao Gu, Jeffrey Chan and Flora.D.Salim are with the School of Science (Computer Science), RMIT University, Melbourne, Victoria, Australia. e-mail: (wei.shao@rmit.edu.au). Ngoc-Thanh Dinh is from Soongsil university, Seoul, Korea

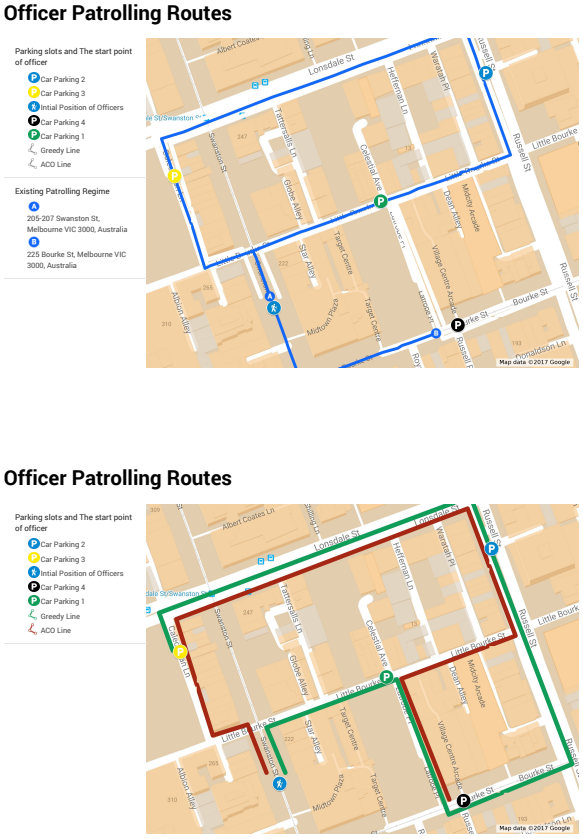


Fig. 1. Officer patrolling routes through three different algorithms. The officer patrols from the initial position to each parking slot by three lines. Blue line: existing patrolling regime; green line: greedy algorithm; red line: ACO algorithm.

determined by the greedy algorithm, shown in the second diagram of Figure 1. The ACO-based algorithm (red line) leads to the shortest path, with a length of 750 m. It takes advantage of leaving probability estimation and integrates with the concept of pheromone in the ACO algorithm. The example shows that the proposed methods perform better than the existing regime.

In summary, this paper makes the following contributions:

- We define a new problem called the Travelling Officer Problem: a touring problem faced by officers who need to monitor parking spaces within a given time constraint.
- We propose two algorithms by taking advantage of spatio-temporal information and probability estimation, to issue infringement notices more efficiently by maximizing the number of violators caught within a limited travelling cost.
- We build a system to evaluate our model and both algorithms. We conduct extensive experiments using a large public dataset that has been published online by Melbourne City Council, which has been published online. The results show both algorithms outperform the baseline.

The paper is organized as follows: Section II discusses related work; Section III presents an overview of on-street car parking in the city; Section IV formally defines the travelling officer problem; Section V gives the details of the two algorithms we propose; Section VI presents the experiments and comparison studies; Section VII assesses limitations and identifies future work; Section VIII concludes the paper.

II. Related Work

Orienteering problems such as the travelling salesman problem (TSP) [6], [7] and its variants [8] are popular in the optimization area. The travelling thief problem (TTP) is an optimization problem which combines two classic problems: one is the TSP, the other is the multiple knapsack problem [10]. It gives more constraints to TSP, which makes it more applicable in real world. TOP is a problem are based on both TSP and Travelling thief problem(TTP) [9]. This general model is also useful for introducing different types of interdependencies in a more strategic way rather than simply putting different problems together to generate new benchmarks.

Spatio-temporal based problems have also become popular recently. Using temporal features as the constraint of an optimization problem is a new trend in the transportation area [11], [12]. Spatio-temporal data is one type of data with spatial and temporal features which usually generated from sensors and ubiquitous devices. It is different from tradition methodology which is used to solve spatial information based orienteering problems. It can be more complex when the time constraint is dynamic. Dynamic time features need more elaborate methods [13].

III. Overview of On-street Parking in City

A. Background and Motivation

The Melbourne Transportation Council set up in-ground sensor systems around CBD areas. For each car parking area, the sensor can detect parking space availability, and check its violation state with parking rules. The sensors report parking events to information centre periodically, and the system would send the message to the patrolling parking officer who supervises this area when the nearby parking cars are in violation state. Then the patrolling parking officer will go to check the cars in violation, and dispatched to issue an infringement notice.

B. Parking Sensor Data

The parking events data recently has been published online, which attracts researchers to study and analyse [5]. The parking events data were recorded from October 1, 2011, to September 30, 2012 (12 months). A total number of 12,208,178 records were logged. Each record comprises the information of a parking event including area name, street name, street segment and some other parking information. It also provides the spatial information such as the latitude and longitude of the parking spaces. The entire CBD is divided into 23 areas by the city council, and each area is monitored by one officer.

Figure 2 reveals the distribution of parking violations within one month. More violations happened in the darker colour

Fig. 2. The monthly parking violation map

areas for that month. The sensors on those streets recorded over 500 parking violations within a month while other street segments (e.g. the street segment on Flinders Street from Williams Street to King Street) recorded much less parking violations. Figure 3 shows the distribution of cars in violation is unbalanced. Most violation events are located in some specific positions.

The local transportation council arranges parking officers to take responsibility for each region separately. As shown in Figure 3. We denote parking spaces in each region with different colours.

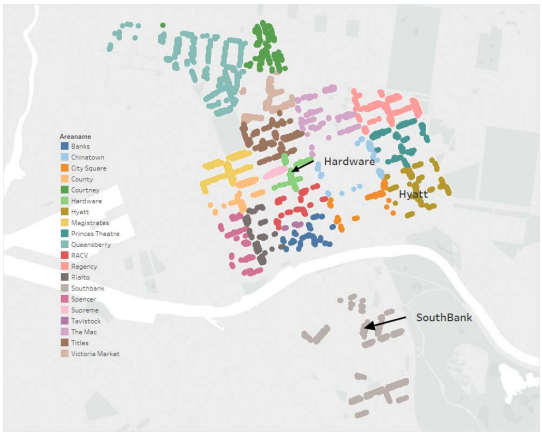


Fig. 3. The parking regions in the CBD area

In real-life scenarios, officers can only move along the roads and cannot cross blocks; therefore, road network information must be taken into account when modelling officer movements.

C. Distribution of Violation Period

Figure 4 shows the distribution of violations with the length of time in violation. The horizontal axis denotes the length of time from the beginning of the violation to the car leaving, and the vertical axis shows the total violation numbers within a month. Figure 4 shows that there are only a small part of cars in violation beyond 100 minutes, and most violations between five and 60 minutes. This is the key observation that we take into account in our spatio-temporal-based optimization approach. Using such observations we build a probabilistic model that can estimate the likelihood of a car in violation leaving which allows us to optimise the strategy that officers use to catch cars in violation.

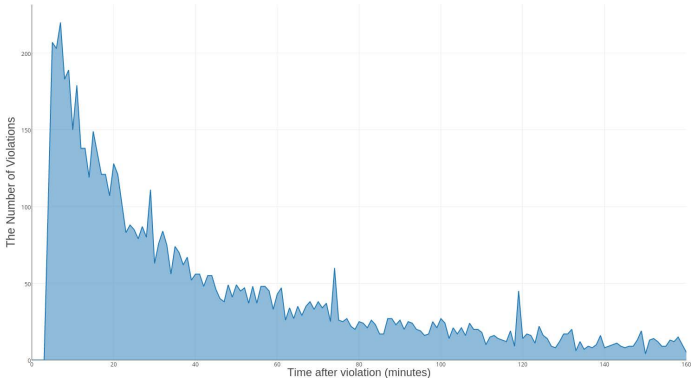


Fig. 4. The numbers of violation with total violation time

IV. Travelling Officer Problem

We describe the Travelling Officer Problem as follows. We have a set of parking spaces in a region, and there are M parking officers. We present each parking space as a node. Parking nodes with cars can be divided into two classes: in violation or in the legal state. Each parking officer takes responsibility for one area. Each officer starts at a random fixed point in the area and collects fines from parking nodes in a violation state. In this paper, we assume they collect fines with a reasonable walking speed during working hours. They can also take bicycles or other vehicles. The violation state of cars in the graph can be changed with the time. Our objective is to find a path that maximizes infringing cars caught on this route with limited travelling cost. The cost metric is defined in terms of working hours or distance.

A. Prime Model Formulation

We define the map of car parking bays as a weighted completed directed graph $G = (V, E)$, where V is the set of n nodes (vertices) and E is the set of edges. Let $C = (c_{ij})$ be the cost matrix associated with E. The cost in the travelling problem can be the travelling time. Another matrix associated with V is $B(v_i) = B(f_{v_i}(t_i))$, which defines the number of cars in violation collected from node i at time t.

The state of car park bays can be defined as a binary variable in Eq. 1:

$$v_i = \begin{cases} 1 \text{ the node i is in violation state} \\ 0 \text{ otherwise} \end{cases} \tag{1}$$

The state of v_i varies with time, which is defined as $f_{v_i}(t)$. The solution of the problem is defined as a set of edges $S = \{x_{ij}\}$, where x_{ij} is a binary variable as follows.

$$x_{ij} = \begin{cases} 1 \text{ the path goes from node i to node j} \\ 0 \text{ otherwise} \end{cases} \tag{2}$$

There also exists a subset of V, denoted as V^s. V^s is a set of nodes located on path S. We assume there are m nodes in V^s.

We formulate our model using an assignment based linear programming model. We aim to seek the best solution S with constraints. The general model is given as follows:

$$Max \quad \sum_{i=1}^{m} B(v_i) \quad = B(f_{v_i}(t_i)) \tag{3}$$

$$s.t. \quad \sum_{x_{ij} \in S} C(x_{ij}) \leq T \tag{4}$$

$$x_{ij} \in S \tag{5}$$

$$t_i = \sum_{j=1}^{i+1} C(e_{j,j+1}) \tag{6}$$

where T is the maximum travel cost for the solution. In this case, it is the maximum working hours of parking officers in a day, t_i represents the time that an officer arrives at node i, and $B(v_i)$ represents how many infringement notices officers issued on node i.

B. Dynamic Temporal Probability Model

As mentioned in Section III, there is a distribution of violation period associated with each area. We denote it as $P_{area}(t)$. Here, we have

$$t = t_{dep} - t_{vio} \tag{7}$$

where t_{dep} denotes the time when a car leaves the parking bay, and t_{vio} denotes the beginning time of the cars in violation.

The car leaving event is a spatio-temporal based event, and it is dynamic and uncertain. Therefore, the model should be a time-based probabilistic model. We generalize the Eq. 7 as follows:

$$\begin{aligned} P_{area}(t) &= P(t_{dep} - t_{vio}) \\ &\approx P(F(t_{vio}, \lambda)) \end{aligned} \tag{8}$$

where $(t_{dep} - t_{vio}) \sim F(t_{vio}, \lambda)$. The $F(t_{vio}, \lambda)$ is an exponential distribution as we observed in Figure 4. In Eq. 3, $f_{v_i}(t_i)$ varies with the $P_{area}(t)$. When $t = t_i$, P_{area} is the probability that a car in violation is still in the parking slot at time t_i.

The probability of violation period in each area is essential to our model. We can estimate the violation period of each parking bay with such model. For example, once a signal sent from a parking bay, our model estimates the probability of car leaving before the officer gets there. If the probability is high (i.e., close to 100%), the officer may choose to go other nodes marked in violation because the car may have left when the officer arrives.

The dynamic temporal probability model is likely to be applied to other cities because the probability density distribution model matches our assumption: the probability that drivers in violation leave a parking slot is decreased exponentially with time. Most drivers know they are likely to be captured if they stay in violation for a longer time: therefore, they would prefer to leave within a short time, and only a minority of drivers will stay in violation for a long time.

C. Challenges

1) Model of the Computational Complexity: Given a graph G, an upper bound m, and a possible solution in the form of a cascade path, it is possible to verify or reject that solution with n additions and a single numerical comparison. This can be accomplished in polynomial time. Because a potential solution can be verified or rejected in polynomial time, the Travelling Officer Problem is an NP-problem [14].

As we have proved that the travelling officer problem is an NP-problem, it is extremely difficult to use a polynomial computational complexity algorithm to solve it. Therefore, intelligent search strategies are needed to solve this problem [15].

2) Unexpected States: The state of each node varies with time. In the real world, these changes may be unknown to officers, and difficult to predict. Additionally, the patten is slightly different for each area.

V. Path Finding Algorithm

In this section, we present the existing patrolling regime which is currently in operation by officers. We propose two solutions: a greedy-based algorithm with dynamic temporal probability model, and an ant colony optimization framework.

A. Existing Patrolling Regime

Our work is supported by Melbourne City Council and we have had discussions on the existing patrolling approach. The existing approach employs a sequential notification of violations, as described by the domain expert; therefore, we use it as the baseline method. The existing patrolling regime is simple and straightforward. When a parking violation is detected, sensor signals are sent to the central system, and an in-charge officer is dispatched [16].

Algorithm 1 captures the regime of existing patrolling method. It works as follows: once a violation occurs, the system will push the event to the in-charge officer's mission queue. The officer always processes the tail event in the queue. Once an event has been processed, it will be removed from the queue. The officer can have a break or rest when the queue is empty.

For the existing approach, we have the following definitions. Q is a queue to store the violation events by time.

Q has some functions associated with it:

- $updateQ(v_i)$: Insert or remove one or more v_i at the tail of the queue.
- $dequeue(v_i)$: Remove the node from the head of the queue and return the node.
- $QisEmpty()$: Check whether Q is empty or not.

This framework is formally described in Algorithm 1.

To explain further, adding a NULL to S means that if there are no cars in violation for time t, the officers will wait until the next event.

B. Greedy Algorithm with Probability Estimation

The existing patrolling regime is inefficient. In this section, we apply a greedy based algorithm with a proposed probability

Algorithm 1 The Existing Patrolling Regime

Input: a given graph $G = (V, E)$
a solution $S \leftarrow$ NULL
a queue Q
a set Ω of constraints among the variables
Output: a solution S
while $Cost(S) <$ T **do**
  if Q is Empty **then**
    add a NULL to S
  else
    add dequeue(v_i) to S
  end if
  updateQ()
end while

model. The greedy algorithm follows the heuristic of making the locally optimal choice at each stage [17], with the hope that it will end up with a globally optimal answer. The greedy algorithm is one of the best solutions to solving NP-hard problems [18]. The general idea is simple. The patrolling officer chooses the parking space with the highest probability of collecting fine as their next destination.

To make the question more precise, we extract some operations below.

calProbability (V) Calculate the probability for each potential node $v_i \in V$. The probability of V_i guides parking officers to the next node because it suggests whether the car in violation can be caught or not. Here we use the Eq. 8 to estimate the probability.

updateV$(C(S))$ Update the potential nodes which connects to the current one. V consists of potential points. S is the existing solution that needs to be updated. For each iteration, the state of each node in the graph should be updated with the latest information from the control centre. All information is collected by in-ground sensors.

The complete algorithm is described in Algorithm 2.

Algorithm 2 The Greedy Algorithm with Probability Estimation

Input: a given graph $G = (V, E)$
a solution $S = \emptyset$
a set of potential next nodes V
a set Ω of constraints among the variables
Output: a solution S
while $Cost(S) <$ T **do**
  if V is Empty **then**
    add a NULL to S
  else
    calProbability (V)
    add v_i with highest probability to S
  end if
  updateV()
end while

C. Ant Colony Optimization with Probability Estimation

1) Background and Motivation: Ant colony optimization (ACO) is a highly compatible, probabilistic swarm intelligence method which is usually used to address meta-heuristic optimizations [19], [20]. ACO can employ heuristic knowledge to find an optimal solution in the search space.

We chose ACO to solve the travelling officer problem for the following reasons:

- ACO is a swarm intelligence searching algorithm which is mainly used to address NP-hard graph optimization problems such as TSP [21].
- The probability model of the problem matches the meta-heuristic in the ACO framework.
- ACO performs well in the global optimization which is suitable for our purpose—to collect as many fines as possible.
- ACO is a widely accepted optimization algorithm to solve the TSP. TOP is a variant of TSP. [22]

2) ACO-based Algorithm: There are two key factors of an ACO algorithm: pheromone and heuristic knowledge, and we define them as τ and η, respectively.

The complete algorithm works as follows. At each iteration, n_a ants construct a solution in the current search space based on previous knowledge (probability model) and a given pheromone model. Then, before the next iteration starts, the pheromone is updated. Finally, we find the best solution and give the next node on this path. Once the search space is changed (i.e. a new violation occurs or is eliminated, we restart the algorithm to find the new path. The algorithm is explained in more details below.

IniPheromoneModel() At the beginning of each step, the pheromone values are all initialized to a constant value $c > 0$.

MoveToNextNode() The ant walks to the next node depending on heuristic knowledge and pheromone distribution. The probability for the choice should be proportional to $[\tau(x_{ij})]^\alpha \bullet [\eta(x_{ij})]^\beta$, where η is a probability that a car leaves its parking bay. The values of parameters α and β determine the relative importance of pheromone, and the car leaving probability model. Therefore, in this case, the probabilities for choosing the next node (i.e., transition probabilities) [21] are defined as follows:

$$\begin{aligned} P(x_{ij}|s^p) &= \frac{[\tau(x_{ij})]^\alpha \cdot [\eta(x_{ij})^\beta]}{\sum_{x_{kl}\in E}[\tau(x_{kl})]^\alpha \cdot [\eta(x_{kl})^\beta]} \\ &= \frac{[\tau(x_{ij})]^\alpha \cdot [P(v_j)^\beta]}{\sum_{x_{kl}\in E, v_p \in V}[\tau(x_{kl})]^\alpha \cdot [P(v_p)^\beta]} \\ &= \frac{[\tau(x_{ij})]^\alpha \cdot [P_{v_j}(t_{dep} - t_{vio})^\beta]}{\sum_{x_{kl}\in E, v_p \in V}[\tau(x_{kl})]^\alpha \cdot [P_{v_p}(t_{dep} - t_{vio})^\beta]} \end{aligned} \tag{9}$$

where $P_{v_p}(t_{dep} - t_{vio})$ is the probability model mentioned above and s^p is the constructed path in the map.

PheromoneUpdate() The aim of the pheromone value updating rule is to increase the pheromone values on solution components that have been found in high quality solutions [23]. In this case, we define it as follows.

$$\tau(x_{ij}) = (1-\rho)\cdot\tau(x_{ij}) + \frac{\sum_{v_i \in S_p} B(v_i)}{\sum_{v_i \in V} B(v_i)} \quad (10)$$

where $\rho \in (0, 1]$ is called evaporation rate [23]. It has the function of uniformly decreasing all the pheromone values. From a practical point of view, pheromone evaporation is needed to avoid a rapid convergence of the algorithm towards a local optimized region.

The completed algorithm is given in Algorithm 3

Algorithm 3 The ACO based Fine Collection Algorithm with Probability Estimation

Input: a given graph $G = (V, E)$
a best solution $S_{bs} = \emptyset$
IniPheromoneModel()
Probability model η
Output: The best solution so far S_{bs}

```
while Cost(S) < T do
  while Iteration < nIteration do
    if V is Empty then
      add a NULL to S
    else
      for i = 1; i < n; i++ do
        calProbability (V_i)
        Constrctred Solution S^p
        if B(S^p) < B(S_bs) then
          S_bs ← S^p
        end if
      end for
    end if
    PheromoneUpdate()
  end while
  MoveToNextNode()
end while
```

VI. EVALUATION

We evaluate the greedy and ACO-based approaches in this section. Using the parking sensor dataset provided by Melbourne City Council [5]. At the beginning, we present some rules and assumptions made in association with this dataset, which are summarized as follows:

- The violation sensor data is only sent to one officer. Other parking officers are not able to receive them.
- The officer does not know whether a car has left until they arrive at the spot.
- All officers work from 7am to 7pm in a day.
- We assume a normal walking speed of 70 meters per minute (m/min).
- The system updates the parking space states in real-time: however, the algorithm calculates the solution only when the officer is available due to the limited computational resources.
- We select the nearest public transport stops or stations near to each area as the starting point of officers every day.

TABLE I
ATTRIBUTE LIST FOR PARKING EVENTS

Attribute Name	Description
Street Marker	The signs placed on the side of parking bays
Area Name	City area, - used for administrative purposes
Arrive Time	Time that the sensor detected a vehicle over it
Departure Time	Time that the sensor detected a vehicle is leaving
In Violation	Indicates that the parking event exceeded the legally permissible time
Sign	Parking sign in effect at the time of the parking event.

TABLE II
OTHER ATTRIBUTES USED IN EXPERIMENTS

Attribute Name	Description
Location	The longitude and latitude of the parking slot
Violation Time	The violation time based on signs
Violation Period	The length of time that car overstays

We conduct an experiment involving car parking violations in 23 areas for a complete week in the first week of September 2011. Each parking officer takes responsibility for issuing parking tickets to cars in violation within his zone. We use the default setting of the ACO, where $\alpha = 1.0$, $\beta = 2.0$ and $\rho = 0.3$. Although the performance can be boosted with parameters tuning, we did not conduct many experiments to test our model because we aim to propose a general solution which can be applied to most cities.

A. System Implementation

We implemented and tested the system (including the TOP model) in C++. The route information was acquired through web service calls to Google Maps APIs. The code ran on a Quad-core laptop running the Windows operating system. We did not use any external code or libraries.

B. Dataset

1) Parking Dataset from the CBD: As mentioned in Section III, the dataset consists of all parking events in the CBD's on-street car parking bays over a year.

To apply our model to the dataset, we used six attributes extracted from the dataset (as shown in Table I), and a list of attributes we defined from that dataset (Table II).

2) Google Maps: Distance is the most important attribute that we use to calculate the cost and find the solution in our model. Since we conducted experiments in real scenarios, it was important to measure the distance between two points on the map. Therefore, we extracted records that reflected accurate positions and driving distance (the distance calculated on the street path using Google Maps) between car parking bays.

C. Evaluation Methodology

We propose two criteria to measure the solutions we propose. One is the fines that can be collected, and the other is the length of time that officers can have a rest between events. The main purpose of our solutions is to collect as many fines as possible. Hence, the benefits index is the most important criteria, and it is defined in Eq.11.

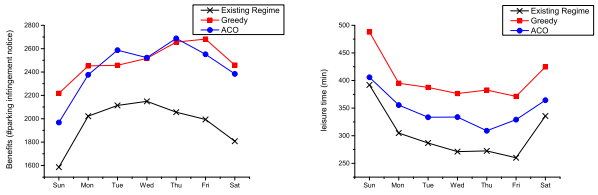


Fig. 5. Left figure is the weekly fine collections in the CBD through three algorithms. Right figure shows the weekly break/rest time in the CBD through three algorithms

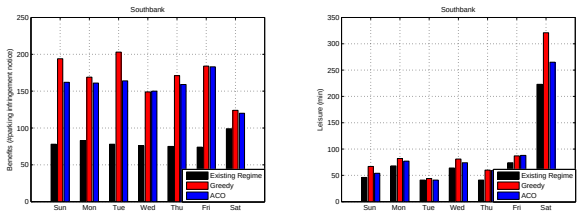


Fig. 6. The weekly benefits and break time in Southbank

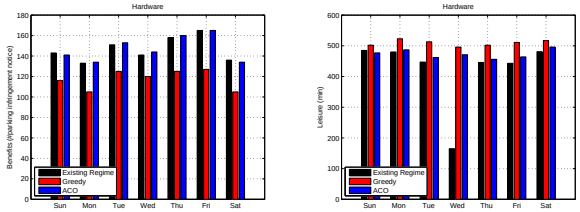


Fig. 7. The weekly benefits and break time in Hardware

$$Benefit = \sum_{t=T_s}^{T_e} B(S) \tag{11}$$

where T_s is the start time of work in the day, and T_e is the end of working hours.

The other important criterion is the rest or break time, which is defined in Eq.12.

$$Rest = \sum_{t=T_s}^{T_e} t(S = \emptyset) \tag{12}$$

$S = \emptyset$ means that there is no violation during such time period. If an area has low violation density on that day, the higher performance algorithm is able to help officers to spend less time on patrolling but have more break or rest time. This is because algorithms with higher efficiency take less time on unproductive path, and find the shortest path to reach car parking bays in violation.

D. Performance Results

We conducted two main experiments to evaluate our models and algorithms. In the first experiment, we evaluated the performance of weekly fine collection. The other experiment explores the relationship between the walking speed of parking officers and the performance of the algorithm. The algorithm performed differently in different areas due to large difference in the distribution of car parking bays and violations. Therefore, we also conducted experiments in each area, and show results for some typical areas.

1) Comparison Studies: In this section, we compare three solutions with two proposed criteria for one week.

The left diagram of Figure 5 shows the overall weekly fine collection results. Compared to the benefits from fine collection using the existing patrolling regime, both ACO and greedy with probability estimation significantly improved the gains in benefit. The greedy algorithm and ACO performed similarly in general. Compared with the greedy algorithm, ACO performed more stably in gaining benefits on the weekdays. The ACO-based algorithm performed better one weekends. We analysed the difference between the parking data during the weekdays and weekends, and found that the distribution of violations on weekends was significantly different from the distribution on weekdays. The violations on weekdays usually occur at specific times, such as after lunch or after work. We plan to explore more temporal information in the future.

The right diagram of Figure 5 shows the average weekly break time for all areas. Both ACO and greedy outperformed the existing patrolling regime; however, the greedy algorithm performed best in terms of break or rest time. This is because the greedy algorithm focuses on local optimization, choosing the closer parking nodes, which leads to less time consumption on the way. For the ACO algorithm, the path computation takes account of both the probability of a car leaving the bay, and maximizing the fines that can be collected. ACO covered more car parking bays than the greedy algorithm; however, it also spent more time on travelling.

2) Weekly Fine Collection in Main Areas: There are many areas in the CBD. We applied three algorithms to all areas, and computed the gains in benefits and break time. We observed that these three algorithms performed differently in different areas. We classified areas into three categories: greedy algorithm domain, ACO-domain, and areas where they have similar performance (refereed to as balanced area). In each category, we present some representative areas to show the results of benefit and break time by applying three different algorithms. **Greedy domain areas** Figure 6 illustrates the weekly benefits and break time in the region Southbank, which has a small number of violations. The graph shows that the greedy algorithm with probability estimation performed better than both ACO and the existing regime. The performance of ACO is slightly lower than the greedy algorithm, and much better than the existing regime. Most of the greedy domain areas show a similar trend. In term of break time, the greedy algorithm looks more promising. It has an enormous advantage in spending less time on navigating to car parking bays in violation. Greedy areas occupy around 30% of total areas.

ACO domain areas Figure 7 illustrates the weekly benefits and break time in the Hardware region. The graph shows that the ACO algorithm outperformed others in these areas. ACO domain areas occupy around 20% of total areas.

Balanced areas Figure 8 illustrates the weekly benefits and break time in the Hyatt region. ACO and the greedy algorithm had similar performance in more than half of the areas. In

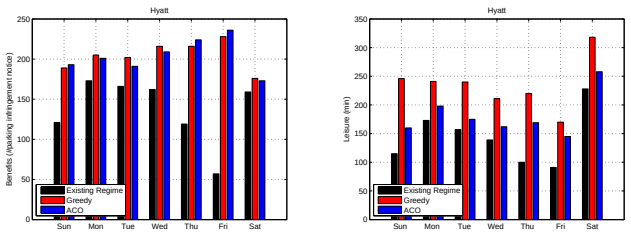


Fig. 8. The weekly benefits and break time in Hyatt area

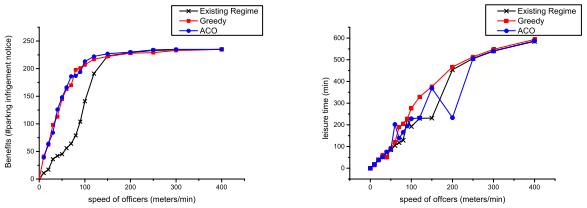


Fig. 9. Left figure shows the benefits in relation to the speed of parking officers, Right figure shows the relationship between break time and speed of parking officers

terms of achieving longer break time, the greedy algorithm performed much better than the other two algorithms.

3) Comparison of Different Walking Speed: The walking speed of parking officers also influences the benefit and break time of our fine collection model. With an increasing speed of parking officers, they are able to cover more nodes during a shorter period. Lower speed suggests that more tasks are being undertaken, and raises more challenges to the algorithm because officers need to spend more time on the travelling route.

The left diagram of Figure 9 shows the result of benefits gained with an increasing walking speed. Both ACO and the greedy algorithm performed much better than the existing regime at any speed. Within 100 m/min, ACO performs better than the greedy algorithm.

The right diagram of Figure 9 shows the result of longer break time with an increasing walking speed. The rest or break time is likely to be longer if parking officers move faster because officers spend less time on the route. The greedy based algorithm performed the best in this experiment. Both ACO and the existing algorithm showed a similar trend with increasing walking speed. Interestingly, the break time of ACO and the existing regime decreased between 150 m/min and 250 m/min. This will be explored in future work.

In summary, both the greedy algorithm and ACO performed much better than the existing regime in increasing benefits and reducing time wasted navigating to destinations. They both performed better than the original algorithm, which suggests that our model and proposed solutions work well for the TOP.

E. Discussion

According to the experimental results, we draw two conclusions. Firstly, both the ACO-based algorithm and greedy-based algorithm can be used in real application for improving fines collection. The ACO-based algorithm can be applied on weekends and Greedy-based algorithm can be used in the weekdays. Secondly, if we take break time into account, the greedy-based algorithm performs better; that is, it provides parking officers with lower workloads.

VII. Limitations and Future Work

In this section, we discuss some limitations of the current model and algorithm, along with our ongoing work and potential future directions.

Firstly, the existing parking management system deployed by Melbourne City Council still depends on parking officers physically issuing infringement notices x though thousands of in-ground sensor have been installed. It is recommended that to an automatic system be designed to issue electronic infringements using these sensor data. Also, the algorithm design in this paper is based on pre-defined areas that are set by Melbourne City Council. We believe that there is a better way to segment the CBD into new areas for more efficient parking management. We plan to apply clustering methods based on violations, and geographic areas. We also plan to design more efficient algorithms to solve the Travelling Officer Problem.

Secondly, the current design is based on the assumption that only one officer knows that a violation occurs nearby. This assumption is a result of one of the limitations of the existing parking system deployed by Melbourne City Council, which is a lack of communication between parking officers. In this case, although the proposed algorithms work best for one officer, it may not achieve the best overall result. In the future, we plan to study multiple officers collaborating to solve this problem, aiming to hire fewer officers for saving tax payers money.

Finally, people may be concerned about the fairness issue in the parking management system (i.e. whether drivers with a parking violation for a longer period should receive a greater penalty). The current design of our algorithms does not distinguish between drivers with longer-period violation or shorter-period violation. Instead, the system offers drivers five minutes grace period (i.e., if they leave within five minutes, the system does not consider they are in violation). In the future, we may look into this fairness issue.

VIII. Conclusion

In this paper, we propose an accurate and efficient model for the Travelling Officer Problem (TOP) and two innovative solutions, a greedy algorithm and an ant colony optimization algorithm, to allocate resources for managing parking areas and collecting fines. They both take advantage of heuristic knowledge to improve the efficiency of parking officers in performing patrolling tasks. To verify our model and solutions, we build a system that implements our model and solutions using a real-world parking sensor dataset from on-street parking bays provided by Melbourne City Council. We also propose two meaningful criteria to measure and evaluate the performance of the proposed TOP solutions. In the future, we plan to apply clustering methods based on violations and geographic areas, and design more efficient algorithms to solve the Multiple Travelling Officer Problem. Spatio-temporal clustering methods based on the knowledge can solve the regional division problem.

REFERENCES

[1] M. Handte, S. Foell, S. Wagner, G. Kortuem, and P. J. Marrón, "An internet-of-things enabled connected navigation system for urban bus riders," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 735–744, 2016.

[2] I. Samaras, N. Evangeliou, A. Arvanitopoulos, J. Gialelis, S. Koubias, and A. Tzes, "Kathodigos-a novel smart parking system based on wireless sensor network," in *Intelligent Transportation Systems*, vol. 1, 2013, pp. 140–145.

[3] D. Tacconi, D. Miorandi, I. Carreras, F. Chiti, and R. Fantacci, "Using wireless sensor networks to support intelligent transportation systems," *Ad Hoc Networks*, vol. 8, no. 5, pp. 462–473, 2010.

[4] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," *Computational Intelligence Magazine, IEEE*, vol. 1, no. 4, pp. 28–39, Nov 2006.

[5] "Parking bay arrivals and departures 2011," https://data.melbourne.vic.gov.au/Transport-Movement/Parking-bay-arrivals-and-departures-2011/8nfq-mtcn, accessed: 2017-07-21.

[6] G. Babin, S. Deneault, and G. Laporte, "Improvements to the or-opt heuristic for the symmetric travelling salesman problem," *Journal of the Operational Research Society*, vol. 58, no. 3, pp. 402–407, 2007.

[7] L. M. Adleman, "Molecular computation of solutions to combinatorial problems," *Nature*, vol. 369, p. 40, 1994.

[8] D. S. Johnson and L. A. McGeoch, "The traveling salesman problem: A case study in local optimization," *Local search in combinatorial optimization*, vol. 1, pp. 215–310, 1997.

[9] M. R. Bonyadi, Z. Michalewicz, and L. Barone, "The travelling thief problem: the first step in the transition from theoretical problems to realistic problems," in *Evolutionary Computation (CEC), 2013 IEEE Congress on*. IEEE, 2013, Conference Proceedings, pp. 1037–1044.

[10] C. Chekuri and S. Khanna, "A polynomial time approximation scheme for the multiple knapsack problem," *SIAM Journal on Computing*, vol. 35, no. 3, pp. 713–728, 2005.

[11] H. Hu, Z. Ma, and Y. Wang, "Optimization of spatial, temporal and amplitude resolution for rate-constrained video coding and scalable video adaptation," in *Image Processing (ICIP), 2012 19th IEEE International Conference on*. IEEE, 2012, pp. 717–720.

[12] W. Shao, F. D. Salim, A. Song, and A. Bouguettaya, "Clustering big spatiotemporal-interval data," *IEEE Transactions on Big Data*, vol. 2, no. 3, pp. 190–203, 2016.

[13] F. C. Anderson and M. G. Pandy, "Dynamic optimization of human walking," *Journal of biomechanical engineering*, vol. 123, no. 5, pp. 381–390, 2001.

[14] J. K. Lenstra and A. Kan, "Complexity of vehicle routing and scheduling problems," *Networks*, vol. 11, no. 2, pp. 221–227, 1981.

[15] B. Selman, H. J. Levesque, and D. G. Mitchell, "A new method for solving hard satisfiability problems," in *AAAI*, vol. 92, 1992, Conference Proceedings, pp. 440–446.

[16] "On-street parking in the city," 2014. [Online]. Available: http://www.melbourne.vic.gov.au/parking-and-transport/parking/Pages/parking.aspx

[17] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. MIT press Cambridge, 2001, vol. 2.

[18] A. Vince, "A framework for the greedy algorithm," *Discrete Applied Mathematics*, vol. 121, pp. 247–260, 2002.

[19] J. Kennedy, "Particle swarm optimization," in *Encyclopedia of Machine Learning*. Springer, 2010, pp. 760–766.

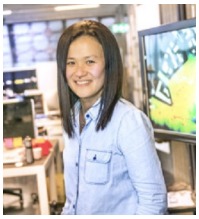
[20] S.-I. Amari, "Natural gradient works efficiently in learning," *Neural computation*, vol. 10, no. 2, pp. 251–276, 1998.

[21] M. Dorigo and L. M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," *Evolutionary Computation, IEEE Transactions on*, vol. 1, no. 1, pp. 53–66, 1997.

[22] H. Neyoy, O. Castillo, and J. Soria, "Dynamic fuzzy logic parameter tuning for aco and its application in tsp problems," in *Recent Advances on Hybrid Intelligent Systems*. Springer, 2013, pp. 259–271.

[23] M. Dorigo and C. Blum, "Ant colony optimization theory: A survey," *Theoretical computer science*, vol. 344, no. 2, pp. 243–278, 2005.

Wei Shao Wei Shao is current a PhD student in the RMIT, Australia. His interest research area focused on data mining, spatio-temporal data analysis and device-free activity recognition. He received the Master of Science in the University of Hong Kong. Previously, he received a BEng in Software Engineering from Xidian University, China.



Flora Salim Dr. Flora D. Salim is a Senior Lecturer at the Computer Science and IT department, School of Science, RMIT University. Previously, she was a Researcher Fellow at RMIT Spatial Information Architecture Laboratory and an Honour Research Fellow and Associate Lecturer at Faculty of Information Technology, Monash University. She obtained her PhD in Computer Science from Monash University in 2009. Her research interests area mobile data mining, context-aware computing, activity and behaviour recognition, and context and semantic learning.

Tao Gu Dr. Gu currently an Associate Professor in the School of Computer Science and Information Technology at RMIT University. Prior to joining RMIT, He is an Assistant Professor in the Department of Mathematics and Computer Science at the University of Southern Denmark. He obtains his Ph.D. degree in Computer Science from National University of Singapore in 2005, M.S. degree in Electrical and Electronic Engineering from Nanyang Technological University, Singapore in 2001, and B.Eng. degree in Automatic Control from Huazhong University of Science and Technology in 1990.

Ngoc-Thanh Dinh Thanh Dinh received his Master degree from Soongsil university, Korea. He has been pursuing a PhD degree at the Graduate School of Electronic and Telecommunication in Soongsil university, Seoul, Korea. He was a PhD candidate of research in School of Computer Science and IT, Royal Melbourne Institute of Technology University. His current researching interests include Internet of Things and cloud, data analytics, machine learning, mobility and next generation networks (5G, ICN, DTN).

Jeffrey Chan Jeffrey is a Lecturer in the School of Science (Computer Science) at RMIT University, Australia. His research interests include machine learning, graph and social network analysis, social computing and novel applications of optimisation and machine learning. He holds a PhD in computer science from the University of Melbourne, Australia.