Matching supply and demand in online parking reservation platforms

Merkouris Karaliopoulos, *Member, IEEE,* Orestis Mastakas and Wei Koong Chai, *Senior Member, IEEE*

***Abstract*—Our work concerns online parking reservation platforms proposed in the last decade to cope with the parking challenge in cities worldwide. Enlisting parking resources from commercial operators (*e.g.*, lots) and individuals (*e.g.*, doorways) and letting drivers make online reservations through mobile apps, the goal of those platforms is to ease transactions between the two sides and best match parking supply with parking demand. This way they maximize their value for drivers and parking space providers but also their revenue out of charged commissions.**

We distinguish between two types of commissions these platforms typically charge, fixed per transaction and proportional to its value, and formulate the respective optimization problems for the platform revenue maximization. We show that the two problems are NP-hard and design a novel algorithm that can treat both, combining greedy and dynamic programming principles. We study its optimality properties both analytically and experimentally, showing that the algorithm closely tracks optimal solutions for small and moderate problem sizes at run times that are several orders of size smaller than off-the-shelf ILP solvers. We then analyze real parking data we collected for the period 2018-2020 from the Bournemouth city in UK to realistically model the rich spatiotemporal dynamics of parking demand such as the location, start times and duration of parking events. These data drive the experimental evaluation of the proposed algorithm, which reports gains of up to 35% compared to the *de facto* reservation policy in use in such platforms. Notably, these gains are higher when the platform operates under constrained supply conditions.

***Index Terms*—Parking reservation, resource sharing, optimization, sharing economy**

I. Introduction

Despite the huge progress made in transportation and information/communication technologies over the last decades, parking still marks a societal challenge that we have not managed to convincingly cope with. In 2017 alone, in the pro-Covid era, British drivers spent, on average, 44 hours and an equivalent £733 per year in terms of wasted time, fuel and emissions [1]. In Covid-19 times, as people avoid public transportation and return to the use of private cars to protect against Covid [2], the situation tends to get worse.

At the heart of the problem lies the general *imbalance* between parking demand and parking supply across time and space. This imbalance gets particularly problematic in areas of high parking demand (*e.g.*, city hotspot areas), where parking space is not only harder to find but also more expensive [3]. The deployment of new infrastructures that could curb parking demand (public transportation infrastructure) or increase parking supply is costly, not least because they need to occupy urban space that is typically very scarce in those areas.

The obvious alternative to deploying new infrastructures is maximizing the efficiency in the management of *existing* parking resources. This is the goal of online parking reservation platforms that have emerged during the last decade, such as Justpark[1], Parkopedia[2], YourParkingSpace[3] to mention but a few. These platforms attempt to ease *transactions* between parking resource providers and drivers seeking for parking space in the same way Airbnb does for lodging: they register parking spots from parking lot operators with dedicated parking infrastructure but also individuals or businesses willing to share parking spots in their backyard, doorway or private property, respectively. Hence, those platforms attack the parking problem in two ways: first, they augment the parking supply by recruiting and enlisting more parking resources that are not directly "visible"; second, they let drivers reserve parking space, circumventing the parking cruising problem [4].

In this paper, we study these online parking reservation platforms focusing on their algorithmic challenges. We model parking demand as a set of requests with specific spatial and timing context and an upper bound on the price they are willing to pay for being served. Parking supply, on the other hand, is abstracted as a set of parking spots, each associated with a distinct location, specific hours within the day that it is available for use and the parking fee it charges as a function of time. The platform then is responsible for matching the submitted parking requests with proper parking spots and directing the drivers to them. Each such *transaction* bears a commission for the platform, whose goal is to maximize its revenue while keeping both drivers and spot owners happy to secure its long-term sustainability.

Proposed solutions in literature have followed different directions. Auction-based solutions (*e.g.*, [5][6]) let high flexibility in the matching process but their complexity has prohibited their adoption in parking reservation platforms, in line with a broader decline in their use in online markets [7]. On the other hand, under simpler pricing mechanisms, the matching process is almost always (*e.g.*, [8][9][10]) reduced to Mixed-Integer Linear Program (MILP) instances. To the best of our knowledge, the research community has overlooked the need for smart algorithmic solutions to these problems routinely pointing to off-the-shelf solvers that cannot scale to practical scenarios.

M. Karaliopoulos and O. Mastakas are with the Athens University of Economics and Business, Dept. Informatics, Athens, Greece

W. K. Chai is with the Bournemouth University, Dept. Computing & Informatics, Bournemouth, UK

[1]https://www.justpark.com

[2]https://www.parkopedia.com

[3]https://www.yourparkingspace.co.uk

978-1-7281-7374-0/20/$31.00 © 2020 IEEE

Main contributions and paper organization: We contribute to this research thread in three ways. *First*, we formulate optimization problems for the two ways online parking platforms may charge commissions for their services, as a fixed amount or a fixed percentage of the price paid by the driver for parking. We characterize the complexity of the problems showing that they are NP-hard. We lay out our modeling assumptions for the parking demand and supply in section II and continue with the problem formulation and its complexity characterization in section III. *Secondly,* in section IV, we design a lightweight algorithm that treats both, combining greedy and dynamic programming principles. We study its optimality analytically for one special case of the problem and experimentally for small and moderate problem sizes. In the latter case, the algorithm closely tracks the optimal solutions at run times that are several orders of size smaller than those off-the-shelf ILP solvers achieve. *Finally*, we provide a thorough evaluation of our proposed algorithm that draws on real data about parking supply and demand. These data are collected from various parking facilities run by the municipality of Bournemouth, in UK, a touristic location recording over 17 million visitors each year that is often plagued with traffic and parking issues especially during the peak holiday season[4]. We analyze these data to parameterize our models of parking demand and supply and capture their strong spatiotemporal variations in section V. Then in section VI we compare our solution with the de-facto policy for handling parking requests and we obtain platform revenue gains of up to 35%. At the same time, we evaluate the benefits of the algorithm for the other two parties involved in the platform, *i.e.*, the parking spot owners and the drivers.

We position our work with respect to related work in this area in section VII. Finally, we conclude in section VIII with a summary of our findings.

II. System model

We consider a parking reservation system, which is run by an online platform and roughly works as follows:

- It maintains a list of available parking resources, which may include spots in parking lots, on street parking spots as well as privately owned parking spots (*e.g.*, in a private, garage, backyard or doorway). For each parking resource, the platform logs its location, the hours it is available and the parking fee it charges for different stay durations.
- Users registered with the platform issue requests for parking, specifying the time interval of interest, destination area and what they are they willing to pay.
- The platform periodically (could be once per day or more frequently) considers the availability of parking resources and the demand for parking space and makes allocations/reservations. Drivers who have issued parking requests and the owners of parking spots are notified about the reservations in due time.

The platform essentially gives rise to a two-sided market where owners of parking resources ("sellers") can serve the demand raised by drivers ("buyers"). It matches drivers with the parking resources taking into account the preferences of the drivers and the constraints of the resources.

Formally, let $\mathcal{R}$ be the set of parking requests submitted to the platform, with $|\mathcal{R}| = n$. Each request $r \in \mathcal{R}$ is a tuple $r = ([s_r, e_r], d_r, p_r)$, where $[s_r, e_r]$ is the time interval of the requested parking event (s_r: start time, e_r: end time), d_r is the driver's travel destination (possibly obfuscated) and p_r is the maximum amount she is willing to pay for a parking spot.

Let also $\mathcal{S}$ denote the set of parking resources that listed in the platform and $|\mathcal{S}| = m$. Each spot $s \in \mathcal{S}$ is a tuple $s = ([a_s, b_s], l_s, c_s)$ described by its window of availability $[a_s, b_s]$ (a_s/b_s : start/end of availability time), its location l_s and $c_s(t)$ is the *tariff function* of parking spot s yielding the amount charged by the spot for parking stay of time t. The minimum assumption we make about these functions is that they are weakly increasing functions of parking stay duration: for all $s \in \mathcal{S}$, $t_1 \leq t_2$ implies that $c_s(t_1) \leq c_s(t_2)$. In practice, these functions typically exhibit decreasing returns to scale, *i.e.*, the marginal cost of parking decreases as the parking stay duration decreases.

The platform seeks to match the parking requests with the available spots under constraints related to:

- the physical distance between spots and users' travel destinations: the requirement is that

$$dist(d_r, l_s) \leq \epsilon_d \tag{1}$$

  namely, their distance should not exceed a threshold ϵ_d so that spot s be relevant for the parking request r.
- the time context of the parking request r: the time interval of the parking event should lie within the availability of the spot s, *i.e.*,

$$a_s \leq s_r \quad \text{and} \quad b_s \geq e_r \tag{2}$$

- financial (mis)matches: a match between request r and spot s is feasible only if the fees charged by s is less than what the user issuing the request is willing to pay,

$$c_s(e_r - s_r) \leq p_r \tag{3}$$

Hence, for each request r, we can define the set C_r of candidate spots as

$$C_r = \{s : (1) \cap (2) \cap (3)\} \tag{4}$$

and, likewise, for each parking spot s, we can define the set A_s of admissible parking requests as

$$A_s = \{r : (1) \cap (2) \cap (3)\} \tag{5}$$

III. The parking reservation problem

The platform makes its revenue by charging commissions for each transaction (a parking request served by a parking resource) carried out through it. We distinguish between two types of commission: (a) fixed commission, h_f, per transaction; and (b) commission as a fixed percentage, h_r, of the transaction value. The way the platform charges its commissions directly affects its overall goal when matching parking requests to resources. When the platform earns a

[4]https://www.bournemouthecho.co.uk/news/19369584.parking-chaos-bournemouth-thousands-flock-beach/

This article has been accepted for publication in IEEE Transactions on Intelligent Transportation Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TITS.2022.3230087

fixed amount out of each transaction, its goal is to maximize the number of distinct transactions it facilitates. When its commission is proportional to the revenue produced out of a transaction, it favors matchings that generate higher revenue.

Formally, let us introduce binary control variables

$$x_{rs} = \begin{cases} 1 & \text{if request } r \text{ is matched with spot } s \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

and define the RxS matrix $\mathbf{X} = [x_{rs}]$. When the platform charges a fixed commission *per transaction*, we can formulate the optimization problem (OPT_1) it faces as:

$$\max_{\mathbf{X}} \quad \sum_{s\in\mathcal{S}} \sum_{r\in A_s} x_{rs} \qquad (OPT_1)$$

$$s.t. \quad \sum_{s\in C_r} x_{rs} \leq 1 \quad \forall r \in \mathcal{R} \tag{7}$$

$$x_{rs} \in \{0,1\} \quad r \in \mathcal{R}, s \in \mathcal{S} \tag{8}$$

where constraint (7) reflects that each request can only be served once or not at all.

On the other hand, when its commission is a fixed portion of the transaction value, the platform seeks to

$$\max_{\mathbf{X}} \quad \sum_{s\in\mathcal{S}} \sum_{r\in A_s} x_{rs} \cdot c_s(e_r - s_r) \qquad (OPT_2)$$

$$s.t. \quad (7), (8) \tag{9}$$

In the special case that all parking resources charge the same constant rate c per hour, *i.e.*, $c_s(t) = c{\cdot}t$ in OPT_2, maximizing the aggregate revenue out of the parking resources coincides with maximizing their utilization.

We can show that:

Theorem 1. *Problems OPT_1 and OPT_2 are NP-hard.*

Proof. It suffices to show that the corresponding decision problems are NP-complete. The proof proceeds in three steps.

First, we formulate the decision problems that correspond to these optimization problems, say $OPT_{1,d}$ and $OPT_{2,d}$: *"Given a set $\mathcal{R}$ of requests and a mapping between $\mathcal{R}$ and the set of parking spots $\mathcal{S}$, as defined by sets $\mathcal{C}_r, r \in \mathcal{R}$, can we satisfy at least k requests (for $OPT_{1,d}$) or can we achieve a revenue of at least K (for $OPT_{2,d}$)?"*.

Then, we construct the problem that can emerge as a special case of the more general $OPT_{2,d}$ problem when all parking spots have identical availabilities and charge the same constant fee for every parking request they serve, irrespective of its duration. In this case, the two problems coincide, whether we consider their maximization or decision variant. Hence, problem $OPT_{2,d}$ "degenerates" to $OPT_{1,d}$.

In the third step, we prove that:

Lemma 1. *Problem $OPT_{1,d}$ is NP-complete.*

Proof. To prove NP-completeness for $OPT_{1,d}$, we need to show that (a) $OPT_{1,d}$ is in NP; (b) there is an NP-complete problem Y that reduces to $OPT_{1,d}$.

The reduction part is trivial. If parking requests are mapped to jobs with fixed start and end times and parking resources correspond to non-identical machines that are only appropriate for subsets of the full job set, we can identify the $OPT_{1,d}$ problem with the job scheduling problem that is introduced and shown to be NP-complete in [11].

Regarding the first requirement, we need to establish that for any input of the decision problem it is possible to verify in polynomial time whether it responds positively or negatively to the decision problem. In ([11], Thm. 3), the authors explain how a weighted Directed Acyclic Graph (DAG) can be constructed in polynomial time for every instance of the $OPT_{1,d}$ problem. We can then find the value V of the longest path in this DAG. If $V \geq k$, then the decision problem is answered in the affirmative; otherwise, the reply is negative. □

□

IV. An approximate algorithm for $OPT_{1,2}$

A. Description of the algorithm

In Algorithm 1, we present an algorithm that can be used for both problems OPT_1 and OPT_2. The algorithm parses sequentially the parking resources and seeks to assign to each of them the optimal set of parking requests. To this end, for each parking resource (spot) s, Algorithm 1 calls the dynamic programming (DP) algorithm in Algorithm 2.

Input to the DP algorithm are the availability start and end times of s, the set $\mathcal{A}_s$ of admissible parking requests and a switch variable (mod) that specifies whether Algorithm 2 is used for problem OPT_1, when $mod = 1$, or problem OPT_2, when $mod = 2$. All times passed to the DP algorithm are discretized by dividing the actual continuous values with the minimum booking duration parameter T (typical values for T are 30min or 1hr). We use accents to denote the discrete values, *i.e.*, $b_s^{'} = \lceil \frac{b_s}{T} \rceil$.

The DP algorithm first ranks the admissible requests in order of increasing start times. It then parses them one by one to row-wise fill the $(|\mathcal{A}_s| + 1) \times (b_s^{'} - a_s^{'})$ matrix u, after initializing its first row to zero. The entry $u(k,t)$ holds the maximum utility that can be accumulated by (discretized) time t after considering the first k parking requests: for OPT_1, this is the number of served requests by the spot, whereas for OPT_2 the total generated revenue out of them. If the parking request r is k^{th} in order of parsing, with value $v_r = 1$ for OPT_1 and $v_r = c_s\big((e_r - s_r)^{'} \cdot T\big)$ for OPT_2, then the entries of the k^{th} row of matrix u are given by

$$u(k,t) = \begin{cases} u(k-1,t) & t \in [a_s^{'}, s_r^{'}) \\ \max\big(u(k-1,t), u(k-1,s_r^{'}) + v_r\big) & t \in [s_r^{'}, e_r^{'}] \\ \max\big(u(k-1,t), u(k,e_r^{'})\big) & t \in (e_r^{'}, b_s^{'}] \end{cases}$$

The maximum achievable utility after processing all $|\mathcal{A}_s|$ parking requests is stored at the matrix element $u(|\mathcal{A}_s|+1, b_s^{'})$. Algorithm 2 is essentially an adaptation of the DP for the 0-1 Knapsack Problem (*e.g.*, see [12], 6.4) to the constraints induced by the fixed start and end times of parking requests. Moreover, it logs the booked requests q_s as it runs rather than in a separate computational step after the matrix u is filled in. After each run of Algorithm 2, those requests are removed from further consideration and the admissible request sets of the remaining resources are updated (Algorithm 1, line 5).

This article has been accepted for publication in IEEE Transactions on Intelligent Transportation Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TITS.2022.3230087

The order in which the different parking resources are parsed is a degree of freedom of the algorithm. One possibility is to run Algorithm 1 many times, each parsing the parking resources in a different random order, and keep the best of the resulting solutions. This is a technique that is practised broadly in randomized algorithms. An alternative is to use heuristic rules for ranking the parking resources. For instance, when commissions are charged per transaction OPT_1, we can rank the spots in order of increasing availability windows $(b_s - a_s), s \in \mathcal{S}$, thus prioritizing resources that are less flexible in serving parking requests. We call this the *Least Flexible First (LFF)* rule. Likewise, when commissions are charged as a fixed part of the transaction value OPT_2, we can rank spots in order of decreasing (average) fees they charge. We call this the *Most Profitable First (MPF)* rule.

Algorithm 1 Algorithm for solving problems OPT_1 and OPT_2

Input: Set of parking resources $\mathcal{S}$; set of admissible requests, $\mathcal{A}_s, s \in \mathcal{S}$; time discretisation step T; switch variable mod
Output: Booked requests $b_s, s \in \mathcal{S}$; platform revenue Rev

$Rev = 0$
for $s \in \mathcal{S}$ **do**
  (*Time discretisation:*) $a_s = \lfloor a_s/T \rfloor$, $b_s = \lfloor b_s/T \rfloor$ and $s_r = \lfloor s_r/T \rfloor$, $e_r = \lfloor e_r/T \rfloor$, $r \in \mathcal{A}_s$
  $[q_s, U_s] =$ Algorithm2 $(s, \mathcal{A}_s, mod)$
  for $s' \in \mathcal{S} \setminus s$ **do**
    $\mathcal{A}_{s'} = \mathcal{A}_{s'} \setminus q_s$
  end for
  if $mod == OPT_1$ **then**
    $Rev = Rev + h_f \cdot U_s$
  else
    $Rev = Rev + h_r \cdot U_s$
  end if
end for

B. Properties of the algorithm

1) Accuracy: We first derive a bound for the worst-case performance of the Algorithm 1 in the special case that the tariff function is proportional to the parking time and identical across all parking spots, *i.e.*, $c_s(t) = c \cdot t \ \forall s \in \mathcal{S}$. This could be the case if the platform determined itself the parking fees for the parking spots, much as ride-sharing platforms do [13].

Approximation ratio: We can prove the following result.

Proposition 1. *Under identical parking charging rates, Algorithm 1 yields an approximation ratio equal to 1/2.*

Proof. Let $q = \bigcup q_s, s \in \mathcal{S}$ be the requests booked under Algorithm 1 and $q* = \bigcup q_s^*$ the respective ones under the optimal algorithm. Then the overall revenue achieved by Algorithm 1 is

$$R(q) = \sum_{s \in \mathcal{S}} R(q_s) \tag{10}$$

whereas it is

$$R(q*) = \sum_{s \in \mathcal{S}} R(q_s^*). \tag{11}$$

Algorithm 2 Finding the optimal set of requests that can be served by a single parking resource s

Input: Discretized availability start and end times (a_s, b_s) and tariff function $c_s(t)$ for slot s; discretized start/end times $(s_r, e_r), r \in \mathcal{A}_s$; switch variable mod
Output: Set of booked requests q_s; achieved spot utility U_s

Sort the set $\mathcal{A}_s$ in order of increasing start times
Initialize: $u(0,t) = 0$, $a_s \le t \le b_s$, $k = 0$
for every $r \in \mathcal{A}_s$ **do**
  $k = k + 1$
  if $mod == OPT_1$ **then**
    $v_r = 1$
  else
    $v_r = c_s(e_r - s_r)$
  end if
  $u(k,t) = u(k-1,t),\ 1 \le t \le e_r - 1$
  if $u_r + u(k, s_r) > u(k-1, e_r)$ **then**
    $u(k, e_r) = v_r + u(k, s_r)$
    $q(e_r) = q(s_r) \cup r$
  else
    $u(k, e_r) = u(k-1, e_r)$
  end if
  for $t = e_r + 1 : b_s$ **do**
    if $u(k-1, t) \ge u(k, e_r)$ **then**
      $u(k,t) = u(k-1,t)$
    else
      $u(k,t) = u(k, e_r)$
      $q(t) = q(e_r)$
    end if
  end for
end for
return $U_s = u(|\mathcal{A}_s| + 1, b_s)$, $q_s = q(b_s)$

We need to show that it always holds

$$R(q) \ge R(q^*)/2. \tag{12}$$

Let us split the requests booked under the optimal algorithm into two parts $q^* = q_1^* + q_2^*$, with

$$R(q^*) = R(q_1^*) + R(q_2^*) \tag{13}$$

such that $q_1^* \subseteq q$ are the requests that are also booked by Algorithm 1, not necessarily on the same parking resources, and $q_2^* \not\subseteq q$, are those which are not. Then, it apparently holds

$$R(q_1^*) \le R(q). \tag{14}$$

Assume that the remaining requests q_2^* are booked under the optimal algorithm at resources $s_1, s_2, ...s_m$ as $q_2^* = q_{2,s_1}^* + q_{2,s_2}^* + ... + q_{2,s_m}^*$. Since these requests are not chosen by Algorithm 1 at the respective resources, for sure it holds:

$$R(q_{2,s_1}^*) \le R(q_{s_1}), R(q_{2,s_2}^*) \le R(q_{s_2}), ..., R(q_{2,s_m}^*) \le R(q_{s_m})$$

and summing on both sides of these inequalities, we get

$$R(q_2^*) \le R(q). \tag{15}$$

Then (12) follows directly from (13)-(15). □

This article has been accepted for publication in IEEE Transactions on Intelligent Transportation Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TITS.2022.3230087

Table I: Approximation error of Algorithm 1 for OPT_1 under three ways to order parking spots: 10^4 problem instances.

Spots' parsing order	Approximation error, η, statistics median	95th prctile	worst	num instances with $\eta = 0$
random once	0.107	0.142	0.213	734
LFF rule	0.088	0.125	0.195	1619
multiple random keep the best	0.022	0.073	0.1714	3008

Table II: Approximation error of Algorithm 1 for OPT_2 under three ways to order parking spots: 10^4 problem instances.

Spots' parsing order	Approximation error, η, statistics median	95th prctile	worst	num instances with $\eta = 0$
random once	0.123	0.28	0.76	86
MPF rule	0.081	0.18	0.727	259
multiple random keep the best	0.069	0.16	0.727	488

Extending this bound to more general settings, where the revenue each served parking request generates depends on the assigned parking spot, is not trivial. Hence, to get insights to the accuracy of the algorithm, we experimentally assess how closely it matches the optimal solution.

Experimental comparison with the optimal solution: We compare the solutions of Algorithm 1 with the optimal ones, obtained for "small" problem instances of a few tens of parking spots and requests with an off-the-shelf ILP solver, the MATLAB R2018a *intlinprog* solver. We run two sets of simulations, one to assess the accuracy of Algorithm 1 for problem OPT_1 and one to do the same for OPT_2. In each set, the number m of parking spots uniformly samples the interval $\{20, 40\}$ and for each value of m, we let the number n of parking requests vary in $[1.5m, 4m]$. We set $T = 1hr$ and match requests with spots over intervals of 24hrs. The start and end times of the spot availability windows, the endpoints of the parking request intervals and the spot fees are generated randomly from uniform distributions and determine the candidate spot set of each request[5]. We generate 10000 distinct problem instances.

Tables I and II report the comparison outcomes for the two simulation experiments. We run three variants of Algorithm 1 corresponding to three different ways to parse parking spots: once randomly (or), multiple times randomly keeping the best solution (mrb), and according to the LFF/MPF rule (lff/mpf), depending on whether we solve (OPT_1) or (OPT_2), respectively. If opt is the optimal solution and $approx_j, j \in \{or, mrb, lff/mpf\}$ the solution of Algorithm 1 when parking spots are parsed in line with variant j, the two tables report statistics of the *approximation error*, defined as

$$\eta_j = \frac{opt - approx_j}{opt}. \tag{16}$$

Overall, Algorithm 1 performs much better on average than the worst-case bound obtained for the special case in Proposition 1 or the worst-case encountered over these experiments, in line with what we know about greedy algorithms (*e.g.*, [14]).

[5]In these experiments, the emphasis is on assessing the accuracy of the algorithm. We take into account the physical layout of spots and user travel destinations in section V.

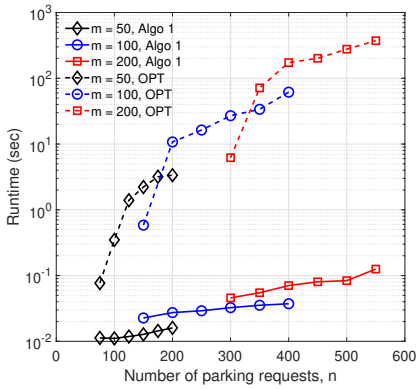


Figure 1: Run times of Algorithm 1 for (OPT_1): each point is the average over 150 problem instances.

Moreover, in line with intuition, the LFF and MPF heuristics clearly improve over a naive random choice of parsing order, while trying many different random orders yields the best scores. The approximation error statistics of Algorithm 1 for (OPT_1) are consistently better than for (OPT_2). Its solutions are optimal for roughly an order of magnitude more (OPT_1) problem instances than (OPT_2) ones.

2) Complexity: The algorithm first indexes requests in order of increasing start times in time $O(nlogn)$. Then, for each spot s, (a) it determines its set of admissible requests A_s in time $O(n)$; it runs an instance of the DP algorithm in Algorithm 2 in time $O(nW_s)$, where $W_s = b_s - a_s + 1$; and reduces the set of requests by the set that was assigned to s in time $O(n)$. Overall, it takes time $O(nlogn) + mO(nW) = O(n(logn + mW))$, where $W = \max_{s \in \mathcal{S}} W_s$. Fig. 1 reports how the runtimes of Algorithm 1 scale from small to moderate problem instances with a couple of hundred spots and requests. Runtimes are measured on a desktop PC with an Intel Core i7-7700 CPU at 3.60GHz and 16GB installed RAM and each plot point is the average over 150 problem instances. Even for the bigger problem size, $(n, m) = (200, 550)$, the runtime of Algorithm 1 remained a fraction of a second, when we measured worst-case runtimes of ∼30mins with the MATLAB ILP solver.

V. Data-driven evaluation methodology

In this section, we describe our overall methodology for evaluating the proposed algorithm, insisting on our models for parking supply and demand and how we parameterize them after real-world parking data.

A. Parking demand

To characterize parking demand, we obtained parking data spanning the period 2018-2020 from the Bournemouth city council. The data concerns various council-managed parking facilities including multi-storey lots and on-street parking space. The data for gated parking facilities, in particular, include fine details such as individual vehicle arrival/departure times, that let capture the spatial and temporal dynamics of the parking demand across the city area.

Figure 2 illustrates sample distributions of vehicle *arrival times* at two parking facilities in the Bournemouth city center,

This article has been accepted for publication in IEEE Transactions on Intelligent Transportation Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TITS.2022.3230087

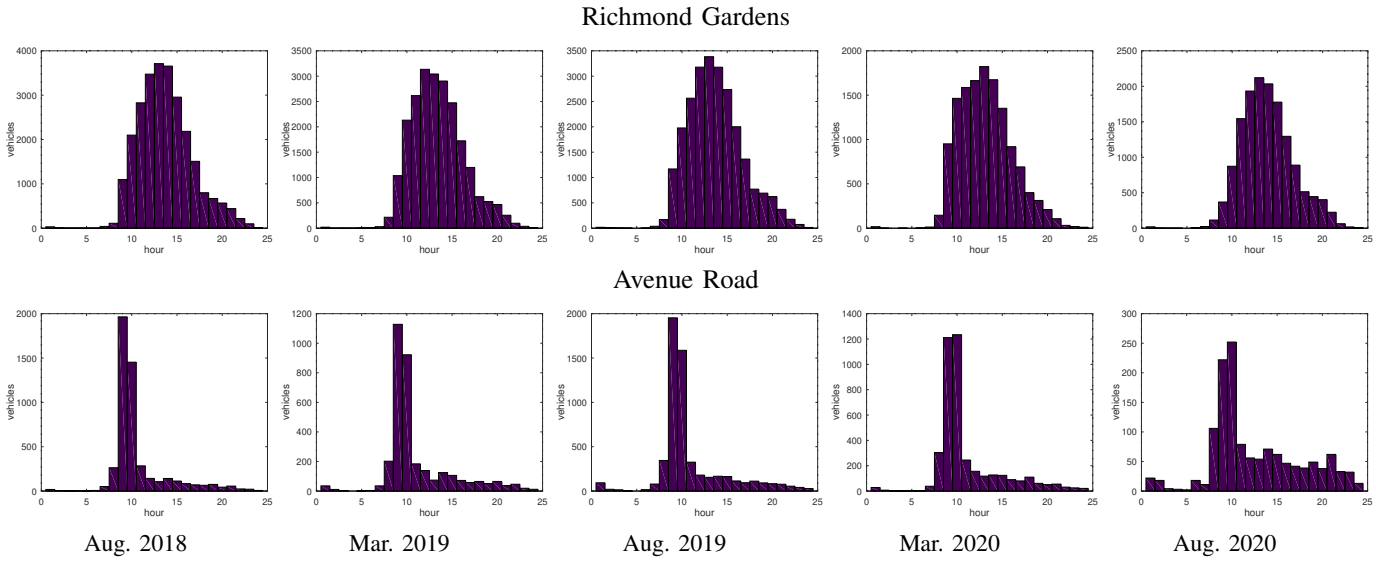


Figure 2: Distribution of vehicle arrival times over time of the day for two multi-storey parking facilities.

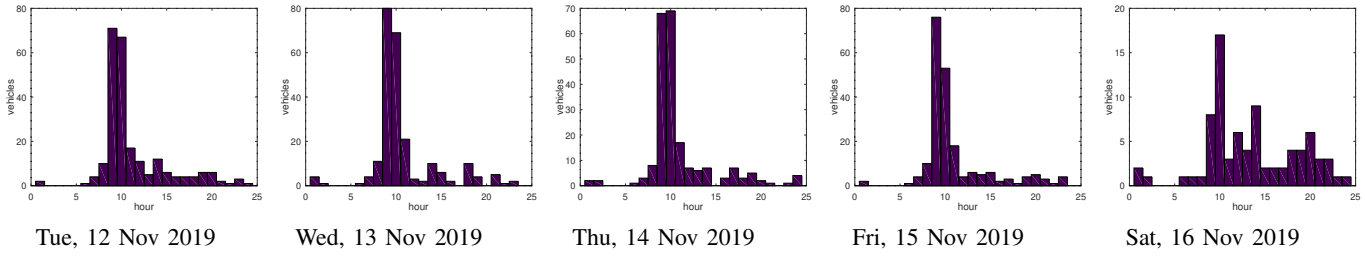


Figure 3: Distribution of vehicle arrival times over time of the day between 12-16 Nov. 2019 for Avenue Road parking facility.

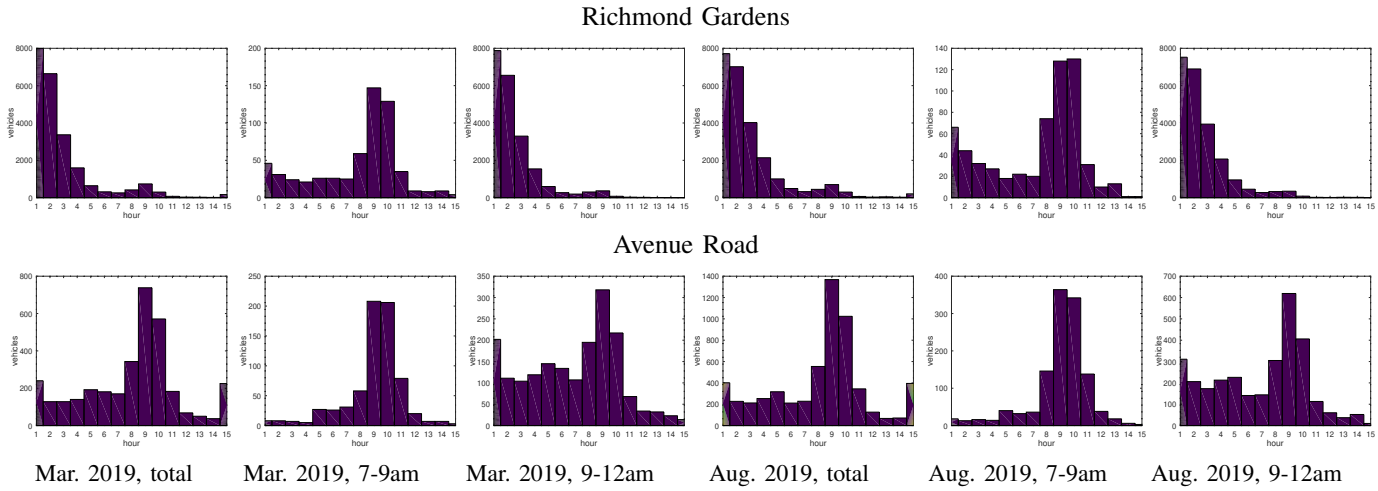


Figure 4: Distribution of parking duration in March and August 2019 for (i) all parking events (columns 1 and 4), (ii) parking events started between 7-9am (columns 2 and 5) and (iii) parking events started between 9-12am (columns 3 and 6).

namely Richmond Gardens and Avenue Road, in different months during the 2-year long period. Both facilities are multi-storey, allow long stay parking and share the same parking tariffs. Moreover, there is no on-street parking, at least managed, available nearby[6]. We can, first, see that these distributions vary considerably between the two locations even though they are not far from each other. The arrival times at Avenue Road exhibit a sharp peak between 9-11am while at Richmond Gardens, they are more uniformly spread over 8am-6pm with a peak around 12-1pm. Second, the distributions are strongly consistent across different months and years. Despite the reduction in the *volume* of arrivals, the *shapes* of those distributions remain intact even amid the Covid-19 pandemic. Notably, these shapes persist over different timescales, *i.e.*, monthly and daily (allowing for the weekday *vs.* weekend effect), as shown in Fig. 3.

[6]https://www.bournemouth.gov.uk/Parking/Parking.aspx

This article has been accepted for publication in IEEE Transactions on Intelligent Transportation Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TITS.2022.3230087

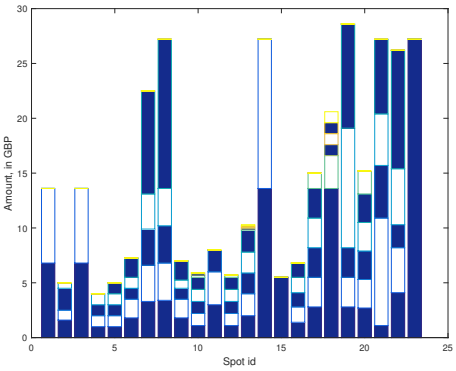


Figure 5: Tariff functions of individual private spots in the Bournemouth city area, as manually retrieved from the JustPark and Parkopedia platforms. Each bar corresponds to one spot and the height of the bottom k stacked pieces in each bar denotes its charge for k hours of parking. If there are n pieces in the bar, the marginal charge after n hours is zero.

Figure 4 shows the distributions of *parking duration* for both Richmond Gardens and Avenue Road parking facilities. We found correlation between vehicle arrival time and parking duration. Vehicles arriving in the early morning hours (between 7-9am) tend to stay for 8-10hrs, pointing to people who routinely go to work. Vehicles arriving after 9am tend to stay for shorter times. For Richmond Gardens, the arrivals after 9am are far more than the early morning ones and give rise to a geometrical distribution for the overall distribution of parking duration, both within a day and cumulatively across a month. For Avenue Road, stays tends to be longer for vehicles arriving after 9am so that the overall distribution of parking duration throughout the day is more spread out in time.

B. Parking supply

To obtain information about availability windows, locations and corresponding parking charges of individual private parking spots such as driveways and backyards, we have collected the data on parking spots registered with platforms such as JustPark and Parkopedia within the Bournemouth area.

Figure 5 presents the tariff functions of 23 spots located in Bournemouth. These are non-decreasing functions of parking duration and almost all of them charge nothing extra if the parking duration exceeds m hours, $m \in [2..10]$. Interestingly, for many of the spots, the per hour parking fee increases when parking duration exceeds some threshold, to probably discourage long continuous use of the spot by a single vehicle. We sample these data to generate tariff functions, $\{c_s(t)\}_{s\in\mathcal{S}}$, for our simulations (ref. section V-C).

C. Simulation scenarios and parking request generation

To best account for the spatiotemporal dynamics of parking demand in our evaluation, we divide the Bournemouth city area into a grid of 24×10 cells, as shown in Fig. 6. Each cell roughly corresponds to a $300m \times 300m$ square area. The red shaded areas, accounting for $K_h = 57$ cells,

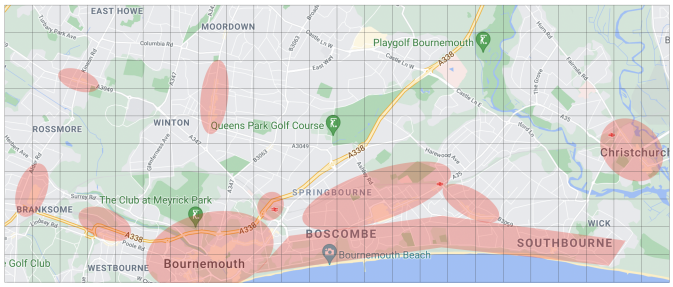


Figure 6: Grid layout over a large part of the Bournemouth town. Parking demand hotspot areas are marked red.

have been identified as *hotspot* areas considering whether they are residential or business districts and their vicinity to urban attractors (*e.g.*, shopping malls, train stations and recreation places). We then define our simulation scenarios as (n, m, r_h, s_h) quadruples, where r_h denotes the portion of parking requests with destinations in hotspot cells and s_h denotes the portion of spots that are located in hotspot cells, with $0 \leq r_h, s_h \leq 1$.

For given (n, m, r_h, s_h) scenario, we take the following steps to simulate the parking allocation process over a reservation period (typically one day):

Step 1: $n \cdot r_h$ parking requests are uniformly directed towards one of the K_h hotspot cells and $n(1 - r_h)$ requests towards one of the non-hotspot cells. Likewise, $m \cdot s_h$ parking spots are uniformly assigned to one of the K_h hotspot cells, whereas $m(1 - s_h)$ are placed in one of the non-hotspot cells.

Step 2: Depending on the travel destination of each parking request, its start time is sampled from one of the two distributions of vehicle arrival times in Fig. 2 for the Avenue Road and Richmond Gardens facilities. To reduce the number of distributions we need to store and sample, we group the cells into two clusters according to the similarity of the vehicle arrival distribution in each cell with the respective distributions for Avenue Road and Richmonds Gardens. Namely, these two distributions serve as the respective cluster centroids.

Step 3: The request start time from step 2 falls in one of the two time zones (7-9am or 9-12pm) in Fig. 4. The combination of cluster ("Avenue Road" or "Richmond Gardens") and request start time (*i.e.*, time zone) determine which parking duration distribution will be sampled for determining the request end time. For instance, if the request start time is 8am for a destination in a cell assigned to the "Avenue Road" cluster, the parking duration is sampled from a distribution like the one in the bottom of the second column of Fig. 4.

Then, the sets $\{C_r\}_{r\in\mathcal{R}}$ and $\{A_s\}_{s\in\mathcal{S}}$ are derived for the parking requests and spots at hand (see section II) and fed to Algorithm 1. We do not further thin out these sets via pricing considerations, namely how much drivers are willing to pay for parking *vs.* how much spot owners charge for given parking duration. The complexity of realistically modeling the fine contextual/behavioral details of the *drivers' willingness to pay* is incommensurate with its added value for the algorithm's evaluation. Hence, whereas the *absolute* platform revenue values in section VI should be viewed as optimistic estimates of the actual ones, the focus of our subsequent evaluation is

on the *relative* performance gains of Algorithm 1.

VI. Evaluation results

We compare our algorithm against the First Come First Served (FCFS) scheme that processes parking requests sequentially, in the order they are issued to the platform. For each arriving request r, the scheme ranks all candidate spots in set C_r (see section II) according to their proximity and assigns the request to the closest available spot at that time. FCFS is the *de facto* scheme to serve parking requests *online*, as they arrive to the platform. Any performance gains of our scheme also set a reference for what is achievable with parking reservation platforms that work offline, *i.e.*, they collect parking requests up to some deadline and then serve them in batches.

Besides the platform revenue achieved by the two schemes, we report the portion of spots that are not serving any request over the reservation period (*idle spots*) and the portion of requests that the platform fails to serve (*blocked requests*). Idle spots hint at spot owners who make no profit out of the platform and are closely linked to the sustainability of the platform supply. Blocked requests imply drivers who did not benefit from the platform and impact the longer-term sustainability of the platform demand. Each point in the plots that follow is the average of 1500 runs; the related confidence intervals are negligible. We have used the Matlab R2018a student version for our numerical simulations.

A. Parking spots without time availability constraints

We first assume that parking spots are fully available within the day. This represents maximum flexibility on the supply side and avoids the blocking of parking requests due to timing incompatibilities (see Eq. 2). Fully available are typically spots in dedicated parking infrastructures such as parking lots rather than parking space at private doorways or backyards.

1) Fixed commission per parking transaction: Figure 7 compares the number of parking requests that can be satisfied by Algorithm 1 and the FCFS scheme under different demand-supply ratios, n/m; the platform revenue is a fixed multiple of this number. Figure 8 then plots the number of idle spots under the two algorithms together with the number of *a priori* idle spots, *i.e.*, spots with empty sets of admissible requests, as discussed in section II. We can make the following remarks.

First, for each plot (*i.e.*, n/m and s_h values) the satisfied parking requests are maximized when the demand distribution across hotspot and non-hotspot areas best matches the supply counterpart; not in terms of parking request/spot *numbers* but rather total hours requested *vs.* available and their precise timing. When this happens, the number of spots attracting drivers also grows and the interests of all three parties, *i.e.*, drivers, spot owners and the platform, are aligned. On the contrary, both quantities (served requests and utilized spots) deteriorate when there is mismatch between supply and demand. The spatial distribution of blocked requests and idle spots varies significantly depending on how this mismatch emerges. For $s_h = 0.4$, for example, most parking spots lie away from hotspot areas. As r_h varies from 0.4 towards 1 and the demand shifts towards hotspot areas, several spots in the periphery fail to attract drivers, also reflected in the high number of *a priori* idle spots, and many parking requests for hotspot destinations cannot get satisfied. For $s_h = 1$, it is the other way round. As the demand shifts towards hotspot areas, more requests can be served and fewer spots remain idle. The idle spots at low r_h values are located in hotspot areas and the blocked requests are for destinations away from hotspot cells. For $s_h = 0.7$, the r_h^* value that simultaneously maximizes satisfied requests and utilized spots lies within the interval [0.4, 1]. Idle spots and blocked requests emerge in both hotspot and non-hotspot areas. For $r_h < r_h^*$, blocked requests are mainly for destinations in the non-hotspot areas and most idle spots are located in hotspot areas. The opposite holds for $r_h > r_h^*$.

Second, the extent to which the two platform parties, drivers and spot owners, are kept happy is highly determined by the demand-supply ratio. Hence, at smaller n/m values, the platform can satisfy higher portions of drivers. The competition is on the supply side and several spots remain idle without attracting parking events. As n/m grows, the situation is reversed: more spots can be matched with at least one parking event and the competition is fiercer on the demand side so that significant part of the requests cannot be served.

Third, the platform revenue generated by Algorithm 1 is always at least as high as what the FCFS scheme generates and almost always higher than that. The gain across all (N, M, r_h, s_h) value combinations is up to 35% (see Fig. 7d, $s_h = 0.4$) depending on parking supply and demand volumes (n, m) as well as their distribution (r_h, s_h). Notably, it increases with the demand-supply ratio n/m, *i.e.*, Algorithm 1 turns out to be more resilient to supply-constrained scenarios.

2) Commission in proportion to the parking charges: Figure 9 compares the platform revenue achieved by Algorithm 1 and the FCFS scheme when the platform charges commissions as a fixed percentage of the overall parking fee for $n/m = 2.5$ and $n/m = 4$. This is the overall revenue the platform generates over a day for the spot owners and itself; its own revenue is a fixed percentage of this amount.

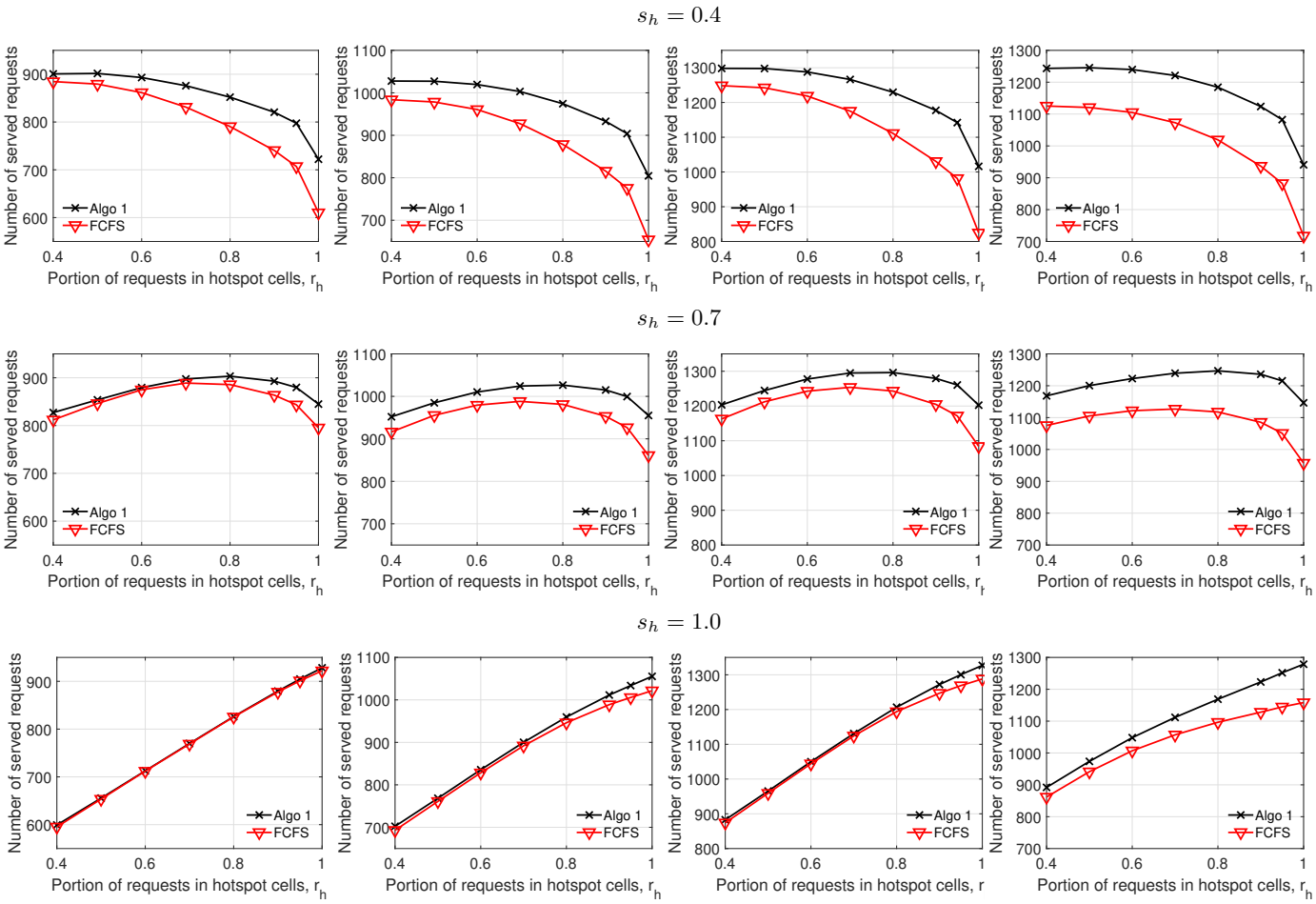
Algorithm 1 again outperforms FCFS, revenue gains ranging from 6% to 35% (for $s_h = 0.4$ and $s_h = 0.7$ when $(n, m) = (1600, 400)$) and increasing with higher demand-supply ratios. As seen in Fig. 10, Algorithm 1 satisfies more parking requests but also extracts more revenue out of them.

As with fixed commissions, the spatiotemporal dynamics of parking supply and demand already set hard lower bounds on the numbers of idle spots and blocked requests. These are shown as *a priori* idle spots and blocked requests in Fig. 10, *i.e.*, parking requests with empty sets of candidate spots (see section II). The portions of drivers and spot owners that eventually benefit from the platform then exhibits similar dependence on the demand-supply ratio and the (r_h, s_h) parameters as with fixed platform commissions.

B. Parking spots with time availability constraints

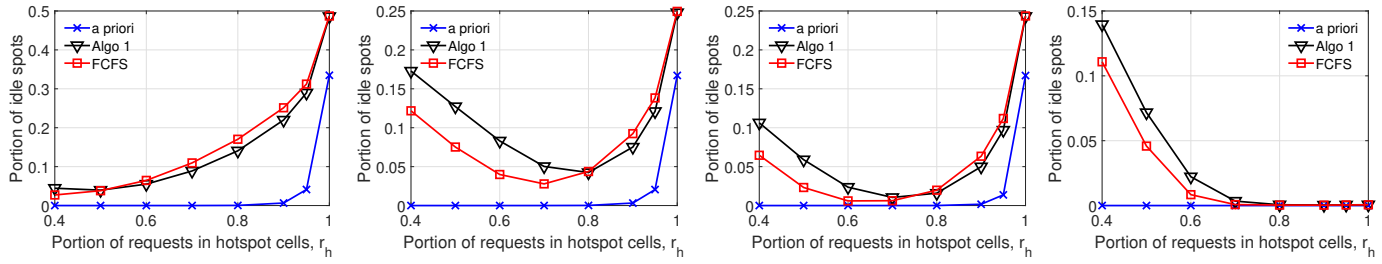
We synthetically generate constrained availability windows for the parking spots with the help of the beta distribution of the first kind, Beta(α, β). The distribution is highly flexible in that with proper manipulation of the parameters α and

This article has been accepted for publication in IEEE Transactions on Intelligent Transportation Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TITS.2022.3230087



(a) $n = 1000, m = 400$ (b) $n = 1200, m = 400$ (c) $n = 1500, m = 500$ (d) $n = 1600, m = 400$

Figure 7: Satisfied parking requests under the two algorithms: fixed commission per transaction, full spot time availability.



$(n, m, s_h) = (1000, 400, 0.4)$ $(n, m, s_h) = (1000, 400, 0.7)$ $(n, m, s_h) = (1200, 400, 0.7)$ $(n, m, s_h) = (1600, 400, 1)$

Figure 8: Idle spots under the two algorithms: fixed commission per transaction, full spot time availability.

β, it yields a broad range of distributions ranging from the standard uniform to both right- and left-tailed ones across the [0,1] interval. Hence, to model spot availability start times distributed in $[a_1, a_2]$, we draw a sample u from the Beta$(1, \beta)$ distribution and set the start time to $a_s = a_1 + (a_2 - a_1) \cdot u$. As β grows larger, the distribution tends to concentrate its mass towards a_1. Likewise, to model the finish times of spot availability, we draw a sample v from the Beta$(\alpha, 1)$ distribution and set $b_s = b_1 + (b_2 - b_1) \cdot v$. Note that as $\alpha \to \infty$ and $\beta \to \infty$, $a_s \to a_1$, $b_s \to b_2$ and with proper choices of $(a_1 = 0, b_2 = 24)$ we fall back to scenarios without spot availability constraints (ref. section VI-A).

In Fig. 11, we show representative plots of achievable platform revenue under the two scenarios for platform commissions, when $(a_1, a_2, b_1, b_2) = (5, 16, 17, 24)$ (columns a, c, d) and when $(a_1, a_2, b_1, b_2) = (5, 13, 15, 24)$ (column b). The following are worth noting. First, Algorithm 1 continues outperforming the FCFS scheme as we introduce constraints in the availability of the parking supply. In all plots of Fig. 11, the top-right points with $\alpha(\beta) = 20$ and $\beta(\alpha) = 5$, respectively, approximate the performance of the two alternatives under full time availability of the parking resources (compare with respective points in Figs. 7 and 9). Departing from those points either left- or bottom-wards the performance of both schemes

This article has been accepted for publication in IEEE Transactions on Intelligent Transportation Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TITS.2022.3230087

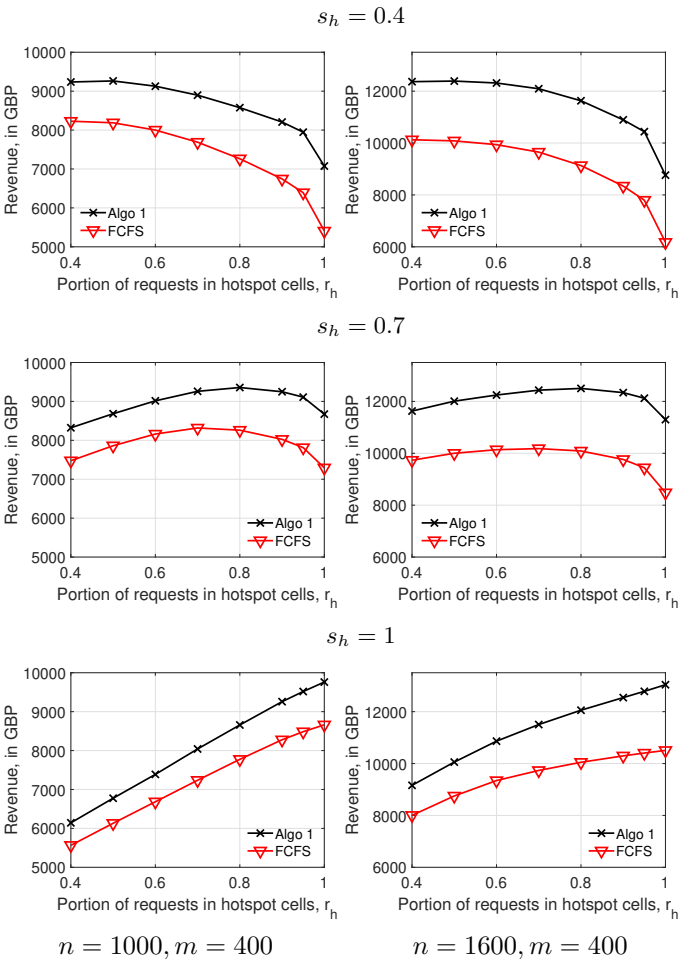
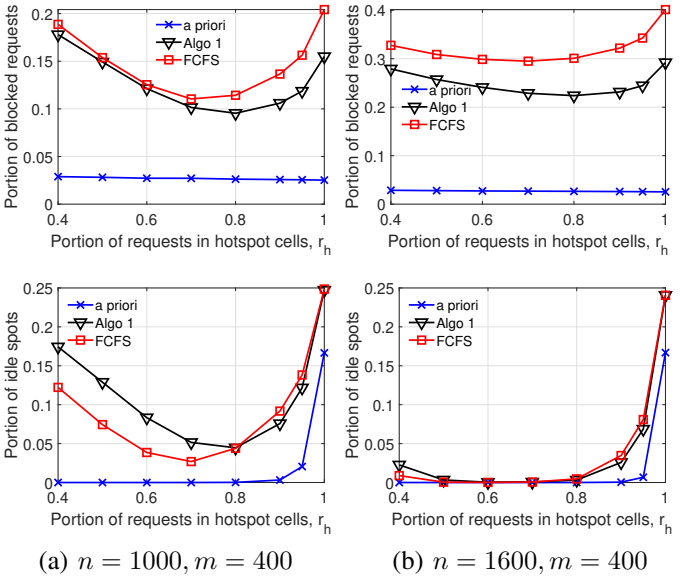


Figure 9: Platform revenue under our algorithm and the FCFS scheme: commission proportional to parking charge, full spot time availability.



(a) $n = 1000, m = 400$ (b) $n = 1600, m = 400$

Figure 10: Blocked requests/idle spots under our algorithm and the FCFS scheme: commission proportional to parking charge, full spot time availability, $s_h = 0.7$.

degrades but Algorithm 1 always fares better.

Secondly, and more interestingly, the two schemes exhibit different sensitivity to the way the time availability constraints are introduced. Our algorithm is more resilient than FCFS to the randomization of the start times (smaller α values). As more clearly seen in Fig. 11a and c, the collected revenue decreases more sharply for FCFS, broadening the performance gap between the two alternatives. On the other hand, FCFS turns out to be less sensitive to the randomization of the spot availability end times (smaller β values). Or, rephrasing this, our algorithm can leverage far better any additional flexibility in terms of end times and increase the revenue gains over the FCFS scheme for the platform.

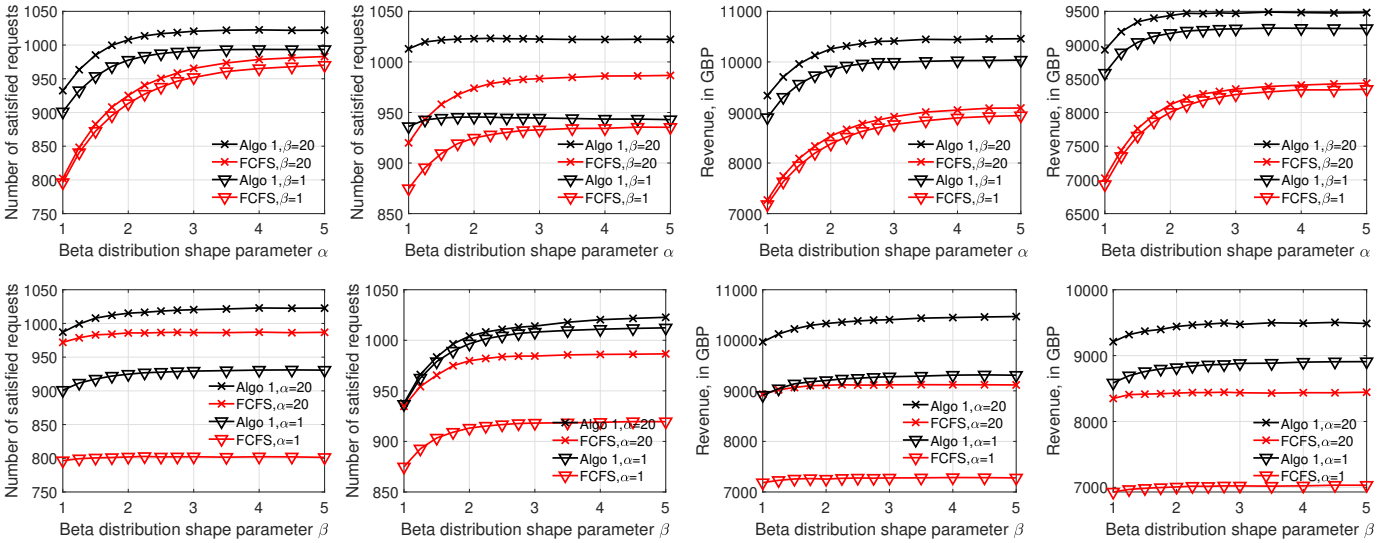
VII. Related work

Aspects of the parking problem have been attracting research interest for many decades now addressing, among others, how drivers search (or *should* search) for parking resources, either on their own (*e.g.*, [15][16]) or leveraging information from parking assistance systems (*e.g.*, [17]); where should parking infrastructure be located (*e.g.*, [18] [19]) and how should it be dimensioned (*e.g.*, [20]); and how could parking resources be efficiently priced (*e.g.*, [18][21]). The common assumption in almost all those studies is that parking search is a drivers' task. On the contrary, with online parking platforms and related apps, the search task is offloaded to them. More recent research work on those platforms mainly addresses their two main operational tasks, *i.e.*, the matching of parking demand and supply and the pricing of parking resources [5][6][8][9][10][22][23].

The studies lying modeling-wise closer to our work are [8][9] and [10]. In all three studies, parking resources are available over distinct time windows, parking requests have concrete timing and spatial context and drivers are willing to pay fees that may vary over time and space. In [9], spot assignments are made offline once, whereas in [8] and [10], these assignments may be updated online as drivers move towards the originally assigned spots, if better alternatives are available (*i.e.*, spots closer to their destinations). In all three studies, the authors formulate Integer (Linear) Programs and solve them with generic off-the-shelf solvers. To cope with the computational issues, they fragment the operations area into smaller subareas and solve multiple smaller problem instances.

Different yet interesting approaches to the private parking slot sharing are found in [22] and [23]. In [22], private parking slots are shared through money-compatible extensions of the top-trading-cycles (TTC) mechanism, the most popular mechanism without money (transfer). Drivers that fail to exchange their own parking spot or do not have one can lease it to the platform or rent one from it, respectively, and the platform should set the prices to ensure budget balance. Unless the parking demands of individual drivers mix really well in time and space, the solution degenerates to a conventional two-sided market. On the other hand, the work in [23] accounts for uncertainty in parking demand. The authors use historical data to predict the arrival of parking requests of different duration and partition the parking resources accordingly. Their method

This article has been accepted for publication in IEEE Transactions on Intelligent Transportation Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TITS.2022.3230087



(a) $(1200, 400, 0.7, 0.7, 1)$ (b) $(1200, 400, 0.7, 0.7, 1)$ (c) $(1200, 400, 0.7, 0.7, 2)$ (d) $(1200, 400, 0.5, 0.8, 2)$

Figure 11: Platform revenue under Algorithm 1 and the FCFS scheme with interval parking spot availability and different $(n, m, r_h, s_h, mode)$ tuples. For columns a,c,d $(a_1, a_2, b_1, b_2) = (5, 16, 17, 24)$; for column b $(a_1, a_2, b_1, b_2) = (5, 13, 15, 24)$.

finds it hard to compete with the simpler FCFS policy as supply and demand scale up to values of practical interest.

Pricing the use of parking spots is the main theme in [5] and [6]. The first two studies work with auctions. In [6], users have unique valuation functions over different parking spots, which are separated into reservable and non-reservable ones. Auctions are used to manage the former, whereas the platform regulates the price for the latter. In [5], the platform puts truthful double auctions mechanisms in place to manage parking demand and supply. Drivers can bid for more than one parking slot and aim to maximize the difference between the value they receive from renting slots minus the monetary cost for doing so. Spot owners seek to maximize the difference between their monetary gains minus the utility they lose by temporarily renting their slots. The platform's goal is then to set the price so as to match demand and supply and maximize its net payoff, *i.e.*, the difference between the total payments from drivers and the rewards paid to the spot owners.

Our work is focused on platforms that aim to realize a sharing economy of parking resources. As in [8][9][10], we explore the way the spatiotemporal dynamics of parking demand and supply can be best captured in the demand-supply matching process. Similar to [5][6][9][22], our reservations are offline, *i.e.*, they are determined in batches over some finite reservation periods. Contrary to [6][8]-[10][22][23], the platform does not interfere with pricing: the spot owners set the prices for their own spots, possibly with the help of recommendations from the platform. This is in line with current practices in parking sharing platforms (*e.g.*, JustPark or YourParkingSpace) and in sharp contrast with practises in ride-sharing platforms such as Lyft or Uber [13]. On the other hand, auctions' complexity is not, at least yet, acceptable for such kind of platforms/"markets" [7]. We have instead taken a more pragmatic approach and (a) proposed and analyzed an algorithmic solution for matching parking demand with supply that yields significant gains for the platforms and its two parties, spot owners and drivers; (b) invested serious effort to assess the benefits of this solution under realistic scenarios drawing on real datasets. Table III positions our work in literature regarding its modeling and algorithmic approach.

VIII. Conclusions

Parking problems remain largely unsolved in many cities around the world. At least in the foreseeable future, the answer to these problems will not come from a single disruptive solution but rather through taking small coordinated progress steps on several fronts (technology, urban planning, policy-making). Our work has focused on one important ingredient of the overall solution: parking reservation platforms that (a) augment the effective parking space leveraging sharing economy practices; and (b) more efficiently manage this augmented space. We have looked into the algorithmic optimization of those platforms, proposing an algorithm that matches parking demand with supply more efficiently than the commonly practised method, at run times that scale dramatically better than generic off-the-shelf ILP solvers. In our experiments, we measured platform revenue gains up to 35%, depending on how the platform charges commissions for its services. Notably, the gains increase with higher demand-supply ratios implying that the algorithm becomes even more attractive in supply-constrained settings. Finally, these platform gains come hand in hand with benefits for both parties contributing to the platform, *i.e.*, the drivers (more satisfied parking requests) and the spot owners (more profit from serving parking requests). This is a necessary condition for the sustainable involvement of the two parties in the platform.

In this paper, the proposed algorithm has been described in the context of a platform that schedules parking requests over finite reservation periods. One straightforward extension is to integrate it into hybrid platforms that combine reservations with real-time parking spot allocation, see [8][10]. It would also be tempting to compare its performance against intelligent

Table III: Related work on parking reservation platforms.

Studies \ Model	Spot time availability		Information about requests		Request-spot preferences		Who determines pricing of parking requests			Reservation frequency		Algorithmic approach
	full	interval	perfect	imperf.	no	yes	platform	owner	auction	∼1hr	∼hrs	
Xiao *et al.* [5]*	✓			✓	✓				✓		✓	Linear relaxation
Wang *et al.* [6]	✓		✓		✓		✓		✓		✓	combination***
Geng *et al.* [8]		✓	✓		✓		✓	✓			✓	OTS** ILP solver
Shao *et al.* [9]	✓		✓		✓		✓			✓		OTS MILP solver
Kotb *et al.* [10]	✓		✓		✓		✓			✓		OTS MILP solver
Xu *et al.* [22]	✓		✓			✓	✓				✓	Top Trade Cycles
Bai *et al.* [23]	✓			✓	✓		✓			✓		OTS MINLP solver
Our work		✓	✓			✓		✓			✓	Dynamic Programming

* non-trivial modeling assumptions, *e.g.*, parking requests may be served at other times than requested by multiple resources with car relocation)
**OTS : off-the-shelf ***stochastic control for non-reservable resources, iterations with linear relaxation for auction-regulated resources

optimization algorithms rooted in the artificial intelligence field such as swarm intelligence algorithms [24][25].

ACKNOWLEDGMENT

The research work of M. Karaliopoulos received funding from the Hellenic Foundation for Research and Innovation (HFRI) and the Hellenic General Secretariat for Research and Innovation (GSRI), under grant agreement No 892

REFERENCES

[1] "Searching for parking costs the UK £23.3 billion a year." [Online]. Available: https://inrix.com/press-releases/parking-pain-uk/
[2] "Cars make a Covid comeback, and that means burning More Oil." [Online]. Available: https://www.bloomberg.com/news/articles/2021-04-28/covid-boost-for-global-car-sales-to-spark-a-surge-in-oil-demand
[3] "Global parking index 2019." [Online]. Available: https://business.parkopedia.com/2019-global-parking-index?hsLang=en
[4] D. C. Shoup, "Cruising for parking," *Transport Policy*, vol. 13, pp. 479–486, 2006.
[5] H. Xiao, M. Xu, and Z. Gao, "Shared parking problem: A novel truthful double auction mechanism approach," *Transport. Res. B-Meth.*, vol. 109, pp. 40–69, 2018.
[6] P. Wang, H. Guan, and P. Liu, "Modeling and solving the optimal allocation-pricing of public parking resources problem in urban-scale network," *Transport. Res. B-Meth.*, vol. 137, pp. 74–98, 2020.
[7] L. Einav, C. Farronato, and J. Levin, "Peer-to-peer markets," National Bureau of Economic Research, Inc, NBER Working Papers 21496, 2015.
[8] Y. Geng and C. G. Cassandras, "New "smart parking" system based on resource allocation and reservations," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1129 – 1139, 2013.
[9] C. Shao, H. Yang, Y. Zhang, and J. Ke, "A simple reservation and allocation model of shared parking lots," *Transp. Res. Part C Emerg. Technol.*, vol. 71, pp. 303–312, 2016.
[10] A. O. Kotb, Y.-C. Shen, X. Zhu, and Y. Huang, "iparker—a new smart car-parking system based on dynamic resource allocation and pricing," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 7, pp. 2637 – 2647, 2016.
[11] E. M. Arkin and E. B. Silverberg, "Scheduling jobs with fixed start and end times," *Discrete Applied Mathematics*, vol. 18, no. 1, pp. 1–8, 1987.
[12] S. Dasgupta, C. H. Papadimitriou, and U. Vazirani, *Algorithms*, 1st ed. USA: McGraw-Hill, Inc., 2006.
[13] "How surge pricing works," https://www.uber.com/us/en/drive/driver-app/how-surge-works/, last accessed: 2021-12-10.
[14] M. Dyer, A. Frieze, and B. Pittel, "The average performance of the greedy matching algorithm," *The Annals of Applied Probability*, vol. 3, no. 2, pp. 526–552, 1993.
[15] J. MacQueen and R. G. Miller, "Optimal persistence policies," *Operations Research*, vol. 8, no. 3, pp. 362–380, 1960.
[16] T. Lambe, "Driver choice of parking in the city," *Socio-Economic Planning Sciences*, vol. 30, no. 3, pp. 207–219, 1996.
[17] E. Kokolaki, M. Karaliopoulos, and I. Stavrakakis, "Leveraging information in parking assistance systems," *IEEE Trans. on Veh. Tech.*, vol. 62, no. 9, pp. 4309–4317, 2013.
[18] J. Y. Wang, H. Yang, and R. Lindsey, "Locating and pricing park-and-ride facilities in a linear monocentric city with deterministic mode choice," *Transport. Res. B-Meth.*, vol. 38, no. 8, pp. 709–731, 2004.
[19] P. Wu, F. Chu, N. Saidani, H. Chen, and W. Zhou, "IoT-based location and quality decision-making in emerging shared parking facilities with competition," *Decision Support Systems*, vol. 134, p. 113301, 2020.
[20] S. F. Franco, "Downtown parking supply, work-trip mode choice and urban spatial structure," *Transport. Res. B-Meth.*, vol. 101, pp. 107–122, 2017.
[21] F. He, Y. Yin, Z. Chen, and J. Zhou, "Pricing of parking games with atomic players," *Transport. Res. B-Meth.*, vol. 73, pp. 1–12, 2015.
[22] S. X. Xu *et al.*, "Private parking slot sharing," *Transport. Res. B-Meth.*, vol. 93, pp. 596–617, 2016.
[23] M. Bai, S. Zhong, P. Yan, Z. Chen, and Z. Zhang, "A data-driven near-optimization approach for smart parking management platforms," in *IEEE Int'l Conf. on Netw., Sens. and Ctrl.*, vol. 1, 2021, pp. 1–6.
[24] G. Tian, Y. Ren, and M. Zhou, "Dual-objective scheduling of rescue vehicles to distinguish forest fires via differential evolution and particle swarm optimization combined algorithm," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 11, pp. 3009–3021, 2016.
[25] J. Tang, G. Liu, and Q. Pan, "A review on representative swarm intelligence algorithms for solving optimization problems: Applications and trends," *IEEE/CAA Journal of Automatica Sinica*, vol. 8, no. 10, pp. 1627–1643, 2021.

Merkouris Karaliopoulos received his Diploma in Electrical and Computer Engineering from the Aristotle University of Thessaloniki, Greece, and a Ph.D. in Electronic Engineering from the University of Surrey, UK. Since 2016 he has been a Senior Researcher at the Athens University of Economics and Business, Greece. Prior to that, he held research appointments at the University of Athens (Greece), ETH Zurich (Switzerland), and the University of North Carolina at Chapel Hill (United States). His research interests lie in performance analysis and resource allocation problems emerging in wireless networks and online/mobile platforms.

Orestis Mastakas is a Management PhD student at the University of St. Gallen and Research Associate in the Athens University of Economics and Business. He completed his Bachelor and Master studies at the Electrical and Computer Engineering department of the National and Technical University of Athens in 2018. His research interests include algorithms and optimization.

Wei Koong Chai is a Principal Academic in Bournemouth University (BU), UK. He heads the Future & Complex Networks Research Group (FlexNet) in the Dept. Computing & Informatics in BU. He was also a visiting academic in the Dept. Electronic & Electrical Engineering, University College London (UCL), UK. Dr. Chai received the B.Eng. (Hons) degree in electrical engineering from the Universiti Teknologi Malaysia, Malaysia, in 2000, and the M.Sc. (Distinction) and Ph.D. degrees from the University of Surrey, U.K., in 2002 and 2008, respectively. He has research interest in networks in general, both applied (*e.g.*, information-centric networking, content caching, infrastructure networks such as transportation, smart grid) and theoretical (*e.g.*, network science, epidemic theory).