SmartParcels: cross-Layer IoT planning for smart communities

Tung-Chun Chang, Georgios Bouloukakis, Chia-Ying Hsieh, Cheng-Hsin Hsu, Nalini Venkatasubramanian

SmartParcels: Cross-Layer IoT Planning for Smart Communities

Tung-Chun Chang
University of California, Irvine
tungchuc@uci.edu

Georgios Bouloukakis
Télécom SudParis, IP Paris
georgios.bouloukakis@telecom-sudparis.eu

Chia-Ying Hsieh
National Tsing Hua University
cyinghsieh@gmail.com

Cheng-Hsin Hsu
National Tsing Hua University
chsu@cs.nthu.edu.tw

Nalini Venkatasubramanian
University of California, Irvine
nalini@uci.edu

ABSTRACT

The emergence of IoT-aided smart communities has created the need for a new set of urban planning tools. The extra design process includes instrumenting infrastructures (sensing, networking, and computing devices) in smartspaces to generate information units (from data analytics) to realize a range of required services. In this paper, we propose SmartParcels, a framework that generates a comprehensive and cost-effective plan for instrumenting designated regions of smart communities (often called parcels). SmartParcels embeds an approach to solve the cross-layer IoT planning problem (shown to be NP-hard) that must consider applications, information/data, infrastructure, and geophysical layout as interdependent layers in the overall design. We develop a suite of algorithms (optimal, partial optimal, heuristic) for the problem; urban planners can compose these techniques in a plug-and-play manner to achieve performance trade-offs (optimality, timeliness). SmartParcels can be utilized for clean-slate planning (from scratch) or for retrofit of communities with existing smart infrastructure. We evaluate SmartParcels in two real-world settings: National Tsing Hua University in Taiwan and Irvine in California, USA, for clean-slate and retrofit. The evaluation results reveal that SmartParcels can enable a 2X - 7X improvement in cost/performance metrics as compared to the baseline algorithm in the clean-slate and retrofit cases.

CCS CONCEPTS

• **Software and its engineering → Layered systems**.

KEYWORDS

IoT planning, urban planning tool, placement problem

ACM Reference Format:
Tung-Chun Chang, Georgios Bouloukakis, Chia-Ying Hsieh, Cheng-Hsin Hsu, and Nalini Venkatasubramanian. 2021. SmartParcels: Cross-Layer IoT Planning for Smart Communities. In *International Conference on Internet-of-Things Design and Implementation (IoTDI '21), May 18–21, 2021, Charlottesvle, VA, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3450268.3453526

IoTDI '21, May 18–21, 2021, Charlottesvle, VA, USA

ACM ISBN 978-1-4503-8354-7/21/05.
https://doi.org/10.1145/3450268.3453526

1 INTRODUCTION

The rapid growth of the IoT ecosystem with consumer-grade devices is enabling us to embed new sensing, communication, and computation technologies pervasively in communities - this is changing how we approach everyday activities. Such smartspaces can enable new services, higher efficiency, and increased levels of community resilience. Applications include: (i) smart governance and public safety (such as smart streetlights and surveillance cameras), (ii) environmental monitoring (such as air quality and noise), (iii) smart utilities (such as energy, water, and waste management), and (iv) smart transportation (such as parking management). The worldwide smart city market is significant at $83.9 billion in 2019, and industry forecasts point to a staggering annual growth rate of 24.7% between 2020 and 2027 [19].

While various scales of smart city trials have been carried out, existing urban planning tools (such as umi [31] and CitySim [35]) are still limited in their ability to select and deploy technologies to create a comprehensive and cost-effective plan to meet the customized application needs of communities. In fact, umi and CitySim focus on environmental impact from communities, such as energy consumption; however, none of them includes the assistance of IoT. What is required is the ability to infuse and assess emerging technologies for sensing and communication that are heterogeneous (in scope, as well as connectivity and storage needs). Technology adoption comes at a cost, and city governments are often concerned about public investments when technologies may be short-lived and do not inter-operate with existing deployments. City governments and agencies, therefore, often resort to manual and ad-hoc planning for their technology deployment of IoT, computing, and network devices leading to sub-optimal planning of communities.

There are multiple trade-offs to be considered when preparing IoT deployment plans for communities. For example, wildfire-prone communities may choose to purchase robots/drones with multi-spectral capabilities (sensing ranges from 15-60 km) to monitor fire spread and provide real-time alerts [4, 7]. Another possibility is to use gas sensors [32] for air quality monitoring - such devices often have limited sensing coverage. Acoustic sensors (embedded, say in streetlights) can detect ambient noise or more complex events, e.g., gunshots, traffic accidents in urban settings, or wildlife/ecosystem monitoring in other geographies [6]. The analytics required for the above applications execute on relatively high-end computing devices. A range of communication possibilities exist today - lightweight data can be transmitted via access networks such as LoRa, Bluetooth, or Zigbee, while more voluminous multimedia data, such

as surveillance videos, should be sent via WiFi or metered cellular networks. Technology choices result in diverse trade-offs, and understanding the geospatial characteristics of a community is a critical part of IoT planning for deployment in smartspaces, which is often too complicated to be manually determined. Hence, in order to provide a comprehensive and cost-effective IoT plan with these trade-offs, the following challenges need to be addressed: (i) what/where infrastructures (sensing, networking, and computing devices) should be instrumented, (ii) how the information units (data) flow via the infrastructures, and (iii) under fixed budgets, how to reuse the infrastructures, such as reuse a sensor for various applications and install a networking device at the location with a higher communication coverage.

In this paper, we study the IoT planning problem to help urban planners make critical decisions involving IoT deployments that provide needed functionality and ensure cost-effective operations. In particular, we aim to develop algorithms to determine: (i) the types and locations of *IoT devices* to deploy for sensing physical phenomena, (ii) the locations of *computing devices* to deploy for analyzing and extracting human-readable information from sensor data, and (iii) the types and locations of *network components, e.g., switches and access points* to deploy for interconnecting the devices and collecting sensor data.

Several prior efforts in cyber-infrastructure planning have focused on techniques for sensor placement and network planning–typically to ensure sensing and communication coverage. Much of this assumes homogeneity of sensing infrastructure and data from independently deployed sensors are collated and fused in cloud platforms to generate higher-level information. A large body of research also focuses on efficient and reliable data collection to conduct such analytics. Tools for conducting what-if analyses and exploring design space options at the community scale, however, require a holistic view that works across multiple system layers.

To this end, we design, implement, and evaluate a system called **SmartParcels**. The name is derived from urban planning literature where *Parcels* refer to a piece of *designated land* slated for development with a specific use and purpose [14]–our work supports the design and instrumentation to create smart community land parcels. The following are the key contributions of this paper:

- We present a general cross-layer architecture that sets the stage for formulating IoT Planning problems–the architecture highlights the needs for distinct infrastructure and information layers for implementing diverse applications (Section 3).
- We propose SmartParcels to help urban planners deploy applications on smart communities with clean-slate (from scratch) and retrofit (with existing infrastructures) scenarios. For better scalability, our approach seeks to partition communities into service sites that have distinct requirements and geospatial characteristics (Section 3).
- We model the IoT planning problem into two distinct steps to map the requested applications to installed devices. We quantify cost-utility functions for different deployment choices, which are used to formulate an optimization problem to address the cross-layer needs for service sites within a community. The problem is shown to be NP-hard (Section 3).
- We develop a suite of algorithms for the two-step process in SmartParcels. First, an optimal technique based on enumeration and dynamic programming is developed–the complexity of such a solution prohibits its use in medium- to large-scale settings. We then develop practical planning graph generation algorithms that consider component reusability, service utility, and communication coverage. The suite of algorithms in SmartParcels offers high flexibility for urban planners to solve the IoT planning problems with different optimality, i.e., optimal, partial optimal, and pure heuristic (Section 4).
- We conduct extensive evaluations of the proposed SmartParcels framework and algorithms in two real-world community settings with associated applications–National Tsing Hua University in Taiwan and Irvine in California, USA. Our evaluation results reveal that: (i) SmartParcels outperforms the baseline algorithm by up to 2 times on overall service utility, (ii) SmartParcels improves the performance up to 7.64 times in retrofit scenarios comparing to the baseline algorithm, and (iii) successfully trade-off the optimality and efficiency (Section 5).

2 SMARTPARCELS AND CROSS-LAYER DESIGN

Related work in planning smart infrastructures has been studied at different layers of the system stack; often independently. Early work originated in the area of network planning for deploying wireless infrastructures and sensor placement in wireless sensor networks. The canonical *sensor placement problem* [36], which aims to maximize the coverage area for a wireless sensor network, has yielded multiple sensing models. Omnidirectional coverage models [13, 37, 39] assume that sensors have a 360-degree sensing angle while directional coverage models are used when sensors have limited sensing angles [8, 12, 18, 24]. The distance between the sensor and sensing target is a factor that impacts coverage; this is typically represented using models where sensing probabilities decay with distance [25]; other techniques truncate sensor values when distance thresholds are met [40, 41]. At the network layer, communication coverage problems focus on providing network connectivity to devices through gateways. Gateway placement has been studied from many perspectives - including the collection of data in neighborhood wireless networks [29], Algorithmic approaches to gateway placement problems have been designed to maximize throughput in multi-hop wireless mesh networks with multiple gateways [22] and under cost-QoS constraints [9, 16, 21]. Multinetwork solutions [20] combine expensive gateways with low-cost transmission equipment to reduce gateway deployment costs. Preliminary efforts consider sensing in a simple connectivity model [15] to address coverage concerns. Contrary to the above efforts, SmartParcels takes an integrated approach that deals with the sensing, communication, computing, and application layers using heterogeneous capabilities of devices/networks while incorporating community structure and needs.

With SmartParcels, the IoT planning decisions are made via the following steps.

(1) **Profiling the communities.** We survey the communities for *boundaries* (between two adjacent communities), *candidate locations* (for installing IoT, computing, and network devices),

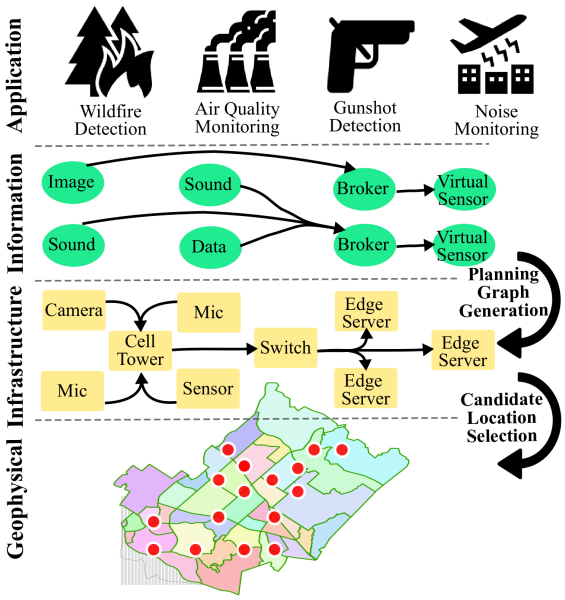


Figure 1: Overview of the IoT planning problem considered in SmartParcels.

required applications (e.g., wildfire detection, gunshot detection, noise monitoring, and air quality monitoring), deployment and operational budgets, etc.

(2) **Developing the information flows.** Each required application can be realized via different combinations of sensor data from IoT devices and analytic algorithms on computing devices (edge servers) connected by directed graphs, which are referred to as *information flows*. Different information flows of the same required application allow urban planners to trade-off the application Quality-of-Service (QoS) and (deployment and operational) cost by selecting the most suitable one to meet the community requirement.

(3) **Determining the infrastructure flows.** Each information flow can be installed on different combinations of sensors with proper sensing capabilities, edge servers with sufficient computing power, and network switches with enough bandwidth, which are referred to as *infrastructure flows*. Different infrastructure flows of multiple information flows may lead to different degrees of resource (or device) sharing, which in turn allow urban planners to exploit *reusability* for higher *efficiency*.

(4) **Running our IoT planning algorithms.** With the above inputs, urban planners can run our IoT planning algorithms for deployment plans. There are two types of such deployments: (i) *clean slate*, where no prior IoT infrastructure exists (e.g., a totally new residential community) and (ii) *retrofit*, where some IoT devices, edge servers, and network switches are already in-place (e.g., an expanding business district). Our proposed algorithms return the types and locations of new IoT devices, edge servers, and network switches to urban planners.

To rigorously solve the IoT planning problem in SmartParcels, we formulate the problem as an optimization problem to maximize the overall *service utility* of the required applications after the IoT devices, edge servers, and network switches in the computed plan are deployed. The service utility is defined as a function of: (i) *coverage*, which represents the geographical area where events can be detected and (ii) *accuracy*, which represents the probability an event can be correctly detected. The IoT plans must satisfy several constraints, including the deployment and operational budgets, sensing ranges, computing power, network bandwidth, and application QoS requirements. Solving the resulting IoT planning problem is not an easy task because of the complex interplay among the four layers: application, information, infrastructure, and geophysical as illustrated in Fig. 1. While the *mappings* across the layers offer tremendous flexibility for higher *reusability* of existing resources and higher *efficiency* of the smart city, the mappings result in an extremely large *search space* for the optimal IoT plan. Let's say a community has one hundred existing street lights as candidate locations for an application to instrument three sensors. The total possible number of mappings for the sensors is 100^3. The search space further increases drastically when the number of required applications and deploying devices increase. In fact, as we will prove later, the IoT planning problem is NP-hard and inapproximatable within a $1 - 1/e$ factor. How to develop a suite of practical IoT planning algorithms, therefore, is crucial to automate the generation of IoT plans. Our proposed IoT planning algorithms can be integrated with the urban planning tools, such as umi [31] and CitySim [35].

To reduce the complexity and avoid redundant computations, we decompose the IoT planning problem into two subproblems: (i) *geophysical mapping selection*, which chooses promising mappings between the infrastructure flows and candidate locations, and (ii) *planning graph generation*, which computes the mappings between the information flow and infrastructure flows that maximize the overall service utility. These two subproblems are solved sequentially. We propose two algorithms for geophysical mapping selection in clean-slate IoT planning problems: (i) *Enumeration* (ENUM), which exhaustively lists all mappings and (ii) *Selection* (SEL), which prunes less-promising mappings using utility and communication coverage. ENUM is needed for optimality of smaller IoT planning problems, while SEL offers urban planners a control knob to trade-off complexity (running time) and optimality (diversity for the planning graph generation subproblem) for larger problems. The two algorithms are also readily applicable to retrofit IoT planning problems after some augmentations.

We propose three algorithms for planning graph generation: (i) *Dynamic Programming* (DP), which gives the optimal IoT plan when being used with ENUM, (ii) *Maximum Reusability* (MR), which is a greedy algorithm that takes reusability of each decision in individual steps into considerations, and (iii) *Maximum Reusability Plus* (MR+), which starts building an IoT plan with DP to a certain degree and finishes up the plan with MR. Among these three algorithms, DP is only suitable for small IoT planning problems, MR is useful when urban planners do not have the luxury of optimal IoT plans, and MR+ offers urban planners a control knob to trade-off complexity (running time) and optimality (overall service utility).

3 CROSS-LAYER IOT PLANNING PROBLEM

Fig. 1 gives an overview of the considered cross-layer IoT planning problem in SmartParcels. Each *application* can be realized by multiple *information flows*, which consist of multiple software processing units. Each information flow needs to be overlaid on an *infrastructure flow*, which dictates the hardware devices to install. Last, individual hardware devices, chosen by one or more infrastructure flows, need to be placed at the selected candidate locations. The goal of our cross-layer IoT planning problem is to compute the optimal mappings across the layers, so as to maximize the service utility without incurring excessive costs.

3.1 System Models

Community profiles. The considered smart city consists of a set of communities C, where $c_i \in C$ is the i-th community. Community c_i provides a set of service sites $\mathcal{S}_i$, where $s_{i,j} \in \mathcal{S}_i$ is the j-th service site. For example, a plaza (community) has a set of service sites, including a grocery store, a restaurant, and a bank. $s_{i,j}$ demands for a set of applications $\mathcal{A}_{i,j}$, where $a_{i,j,k} \in \mathcal{A}_{i,j}$ is the k-th application. For instance, a bank demands for gunshot detection for security, and a restaurant demands for air quality for outdoor dining. Each community c_i has a tuple $(\mathcal{S}_i, \{\mathcal{A}_{i,j} | \forall s_{i,j} \in \mathcal{S}_i\})$ that indicates its service sites and corresponding applications. For simplicity, a service site is represented by the center point within its boundary. The communities jointly provide a set of candidate locations $\mathcal{L}$, like traffic poles, street lights, and even rooftops, for installing devices, including IoT devices with sensors, edge servers, and network devices. Note that different applications could have different importance levels. For each service site $s_{i,j}$ and community c_i, we assign a weight $\beta_{i,j,k}$ to each required application $a_{i,j,k}$. Without loss of generality, we assume $\sum_{\forall a_{i,j,k} \in \mathcal{A}_{i,j}} \beta_{i,j,k} = 1, \forall s_{i,j}$.

Information flows for an application. To implement $a_{i,j,k}$, a set of information flows $\mathcal{F}^{info}_{i,j,k}$ can be adopted, where $f^{info}_{i,j,k,m} \in \mathcal{F}^{info}_{i,j,k}$ is the m-th information flow. More precisely, $f^{info}_{i,j,k,m} = (V^{info}, E^{info})$ is a directed weighted graph, where $v \in V^{info}$ represents an information unit (which can be *raw data or middleware components*), and $e(u,v) \in E^{info}$ represents the data flow between two information units. Both the vertices and edges are associated with weights. The weight of a vertex $w(v)$ represents the computing resources consumed by the information unit, whereas the weight of an edge $w(e(u,v))$ is the bandwidth consumption over the edge. Furthermore, each information flow specifies the number of sensors, e.g., three microphones are needed by the gunshot detection for triangular localization.

Infrastructure flows implementing an information flow. Each information flow $f^{info}_{i,j,k,m}$ can be implemented by a set of infrastructure flows $\mathcal{F}^{ifr}_{i,j,k,m}$, where $f^{ifr}_{i,j,k,m,n} \in \mathcal{F}^{ifr}_{i,j,k,m}$ is the n-th infrastructure flow. $f^{ifr}_{i,j,k,m,n} = (V^{ifr}, E^{ifr})$ is a directed weighted graph, where $v \in V^{ifr}$ represents a device and $e(u,v) \in E^{ifr}$ represents the data flow between two devices. We consider very general devices, which could be sensors, like microphones or cameras, computing devices, like edge servers, and network switches, like LTE small cells or Ethernet switches. The weights of a vertex $w(v)$ and an edge $w(e(u,v))$ represent the computing resource and the network bandwidth offered by them, respectively. A tuple $(\mathcal{F}^{info}_{i,j,k}, \{\mathcal{F}^{ifr}_{i,j,k,m} | \forall f^{info}_{i,j,k,m} \in \mathcal{F}^{info}_{i,j,k}\})$ summarizes all information flows and the corresponding infrastructure flows for each application $a_{i,j,k}$. Given an $f^{info}_{i,j,k,m}$ and an $f^{ifr}_{i,j,k,m,n}$, each processing unit $v \in V^{info}$ is assigned to a device $v' \in V^{ifr}$ by a function $R(v) = v'$. Besides, for an edge $e(u,v) \in E^{info}$, let $\langle R(u), R(v) \rangle$ denote the shortest path on $f^{ifr}_{i,j,k,m,n}$ consisting of involved devices, i.e., the actual data flow on the infrastructure layer. Without loss of generality, we assume $\langle R(u), R(v) \rangle$ contains at least one network switch unless $R(u)$ and $R(v)$ are the same device. If u and v run on the same device, the network bandwidth between them is extremely high, thus we assume $w(e(R(u), R(v))) = \infty$.

Planning graph. We define a planning graph as a two-layer graph $G^p = (V^p, E^p)$, where the first layer $G^p_1 = (V^p_1, E^p_1)$ contains a set of information flows, and the second layer $G^p_2 = (V^p_2, E^p_2)$ consists of a set of infrastructure flows. In both layers, flows may share vertices or edges. Besides, a set of *assignment edges* E^r has each edge $e(v, R(v)) \in E^r$ indicates the assignment of $v \in V^{info} \subset V^p_1$ to $R(v) \in V^{ifr} \subset V^p_2$ for $f^{info}_{i,j,k,m}$ and $f^{ifr}_{i,j,k,m,n}$. We collectively write the planning graph as $V^p = \{V^p_1, V^p_2\}$ and $E^p = \{E^p_1, E^p_2, E^r\}$.

Geophysical mapping function of infrastructure. To select the candidate locations, we define a geophysical mapping function $f(v)$ that maps a vertex $v \in V^p_2$ of the infrastructure layer in a planning graph G^p to a candidate location $l \in \mathcal{L}$. Each $v \in V^p_2$ is captured by a tuple $t_v = (r^{tr}_v, r^{sen}_v, \tau_v)$. The device's transmission and sensing ranges are represented by r^{tr}_v and r^{sen}_v, respectively. τ_v indicates the device type, which could be *sensing*, *computing*, or *networking*. If $\tau_v \neq$ *networking*, r^{tr}_v equals the transmission range of its connected network device u, i.e., $r^{tr}_v = r^{tr}_u$, $e(v,u) \in E^p_2$. Besides, multiple devices can be mapped to the same candidate location. For the sake of presentation, we let $V^{sen}_{i,j,k,m,n}$ denote the sensors of $f^{ifr}_{i,j,k,m,n}$, i.e., $\forall v \in V^{sen}_{i,j,k,m,n}$, $\tau_v =$ *sensing*.

Service utility of an infrastructure flow. We write the Euclidean distance between two candidate locations $l_1, l_2 \in \mathcal{L}$ as $dist(l_1, l_2)$. An infrastructure flow is *connected* if all its devices are connected after mapping, i.e., $dist(f(u), f(v)) \leq \min(r^{tr}_u, r^{tr}_v)$, $\forall e = (u,v) \in f^{ifr}_{i,j,k,m,n}$; otherwise, the flow is not connected. If $f^{ifr}_{i,j,k,m,n}$ is connected, its *service utility* to a service site $s_{i,j}$ is:

$$U(f^{ifr}_{i,j,k,m,n}, s_{i,j}) = A(f^{ifr}_{i,j,k,m,n}) \times P(V^{sen}_{i,j,k,m,n}, s_{i,j}), \quad (1)$$

where $A(f^{ifr}_{i,j,k,m,n})$ and $P(V^{sen}_{i,j,k,m,n}, s_{i,j})$ are the *accuracy* and *sensing probability*. If $f^{ifr}_{i,j,k,m,n}$ is unconnected, we set $U(f^{ifr}_{i,j,k,m,n}, s_{i,j})$ to 0. Each $f^{ifr}_{i,j,k,m,n}$ implementing application $a_{i,j,k}$ comes with an accuracy model $A(f^{ifr}_{i,j,k,m,n})$ depending on the implemented method. For example, for wildfire detection, image-based detection has higher accuracy than emission-based detection.

Our sensing probability models are inspired by the truncated attenuated model [36]. First, for a sensor $v \in V^{sen}_{i,j,k,m,n}$, the probability

is attenuated (decaying) with its distance to $s_{i,j}$, $dist(f(v), s_{i,j})$, and truncated by its sensing range, r_v^{sen}. Hence, if $dist(f(v), s_{i,j}) \leq r_v^{sen}$, $\forall v \in V_{i,j,k,m,n}^{sen}$, the mean truncated attenuated sensing probability is $\bar{p} = \sum_{\forall v \in V_{i,j,k,m,n}^{sen}} e^{-\alpha_v * dist(f(v), s_{i,j})} / |V_{i,j,k,m,n}^{sen}|$, where α_v is a parameter related to v; otherwise, $\bar{p} = 0$. The sensing probability is then bounded by the sensing range of sensors as follows:

$$P(V_{i,j,k,m,n}^{sen}, s_{i,j}) = \begin{cases} \bar{p}, \ if \ dist(f(v), s_{i,j}) \leq r_v^{sen}, \\ \quad \forall v \in V_{i,j,k,m,n}^{sen}; \\ 0, \ otherwise. \end{cases} \tag{2}$$

This completes our definition of service utility. We emphasize that our proposed algorithms do not rely on the mathematical properties of the service utility. Hence, administrators have total freedom to apply different models, e.g., non-line-of-sight sensing ranges.

Costs. Each device $v \in V_2^p$ in the infrastructure layer incurs two types of cost: (i) *deployment cost* $\delta_{deploy}(v, l)$ due to deploying v at candidate location $l \in \mathcal{L}$ and (ii) *operational cost* $\delta_{op}(v)$ due to maintaining its daily operations. The deployment cost is one-time, while the operational cost is recurring. Besides, B_{dp} and B_{op} are the budgets for deploying and operating the devices.

3.2 Problem Formulation

Given the community profiles, information flows, infrastructure flows, the IoT planning problem aims to maximize the overall service quality under the cost budgets by generating the optimal planning graph G^{p^*} and selecting the optimal geophysical mapping functions $\mathcal{F}^* = \{f^*(v) | \forall v \in V_2^{p^*}\}$. More specifically, the IoT planning problem is formulated as follows.

$$\max \sum_{\forall c_i \in C} \sum_{\forall s_{i,j} \in \mathcal{S}_i} \sum_{\forall f_{i,j,k,m,n}^{ifr^*} \in G_2^{p^*}} \beta_{i,j,k} U(f_{i,j,k,m,n}^{ifr^*}, s_{i,j}), \tag{3a}$$

$$\text{subject to:} \sum_{\forall v \in V_2^{p^*}} \delta_{deploy}(v, f^*(v)) \leq B_{dp}, \tag{3b}$$

$$\sum_{\forall v \in V_2^{p^*}} \delta_{op}(v) \leq B_{op}, \tag{3c}$$

$$\sum_{\forall e(u,v) \in E^{r^*}} w(u) \leq w(v), \forall v \in V_2^{p^*}, \tag{3d}$$

$$\sum_{\forall e(u,v) \in E_2^{p^*}} w(e(u,v)) \leq \sum_{\forall e(v,u') \in E_2^{p^*}} w(e(v,u')), \ \forall v \in V_2^{p^*}, \tag{3e}$$

$$w(e(u,v)) \leq \min_{\forall e' \in \langle R(u), R(v) \rangle} w(e'), \forall e(u,v) \in E_1^{p^*}. \tag{3f}$$

The objective function in Eq. (3a) finds G^{p^*} and $\mathcal{F}^*$ to maximize the total service utility. Eqs. (3b) and (3c) are the budget constraints of deployment cost (B_{dp}) and operational cost (B_{op}). For each infrastructure $v \in V_2^{p^*}$, Eq. (3d) ensures that v has enough computing resources to process all assigned information unit u, i.e., v's weight $w(v)$ is no less than the sum of all u's weight. Besides, Eq. (3e) ensures that v's output bandwidth is no less than its input bandwidth. For each data flow $e(u,v) \in E_1^{p^*}$, Eq. (3f) ensures that the minimum bandwidth within the assigned path $\langle R(u), R(v) \rangle$ meets its bandwidth requirement.

Theorem 1. *The IoT planning problem is NP-hard and can not be approximated within* $1 - 1/e$.

Proof. We prove the NP-hardness by reducing the max K-cover problem to a special case of the IoT planning problem, which is derived by assuming unlimited resources (eliminating Eqs. (3d)–(3f)). Each application has only one *connected* infrastructure flow with the deployment and operational costs set to 1, and both budgets are K. The K-cover problem has a polynomial reduction to the special case as follows: given a set of service sites, selecting K infrastructure flows for the service sites such that the service utility is maximized. Hence, the IoT planning problem is as hard as the K-cover problem and cannot be approximated within $1 - 1/e$ [17]. □

4 SMARTPARCEL'S ALGORITHMS

We decompose the IoT planning problem of SmartParcels into two subproblems, *geophysical mapping selection* and *planning graph generation*, for better *reusability* and *efficiency*. Let $G^{p^*} = (V^{p^*}, E^{p^*})$ denote the optimal planning graph, where $G_1^{p^*} = (V_1^{p^*}, E_1^{p^*})$ contains a set of information flows, and $G_2^{p^*} = (V_2^{p^*}, E_2^{p^*})$ contains a set of infrastructure flows. The main reason we opt for decomposition is because when searching for the optimal planning graph G^{p^*}, the geophysical mappings are generated and examined repeatedly. However, the geophysical mappings are mostly static, which can and should be stored and reused. Fig. 2 shows the relation between these two subproblems, which are described as follows.

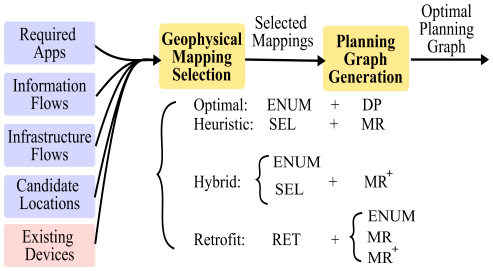


Figure 2: Problem decomposition, key inputs/outputs, and proposed algorithms.

4.1 Multi-Phase Approach

Geophysical Mapping Selection. The goal is to select the optimal geophysical mapping functions $\mathcal{F}^*$ containing a mapping $f^*(v)$ for each infrastructure $v \in V_2^{p^*}$, i.e., $\mathcal{F}^* = \{f^*(v) | \forall v \in V_2^{p^*}\}$. Geophysical mapping selection has the following inputs: (i) a tuple $(\mathcal{S}_i, \{\mathcal{A}_{i,j} | \forall s_{i,j} \in \mathcal{S}_i\})$ indicating the set of service sites and each service site's required application for each community $c_i \in C$, (ii) a tuple $(\mathcal{F}_{i,j,k}^{info}, \{\mathcal{F}_{i,j,k,m}^{ifr} | f_{i,j,k,m}^{info} \in \mathcal{F}_{i,j,k}^{info}\})$ representing the set of information flows and the infrastructure flows implementing each information flow for each application $a_{i,j,k} \in \mathcal{A}_{i,j}$, (iii) a set of candidate locations $\mathcal{L}$ for deploying the infrastructures, and (iv) a set of existing devices $\mathcal{D}$ in the community (for retrofit). Afterward, a set of possible mappings of each infrastructure flow is selected.

For each infrastructure flow $f^{ifr}_{i,j,k,m,n} = (V^{ifr}, E^{ifr}) \in \mathcal{F}^{ifr}_{i,j,k,m}$, a geophysical mapping function $f(v)$ for each infrastructure $v \in V^{ifr}$ is selected as follows: (i) for a pair of infrastructures u and v on edge $(u,v) \in E^{ifr}$, they must be within each other's transmission range r^{tr}_u and r^{tr}_v after mapping, i.e., $dist(f(u), f(v)) \leq \min(r^{tr}_u, r^{tr}_v)$, and (ii) if v is a sensor, service site $s_{i,j}$ must be within its sensing range r^{sen}_v, i.e., $dist(f(v), s_{i,j}) < r^{sen}_v$. We define an infrastructure flow $f^{ifr}_{i,j,k,m,n}$ along with the mapping $f(v)$ for each infrastructure $v \in f^{ifr}_{i,j,k,m,n}$ as a Mapped Infrastructure Flow (MIF).

Planning Graph Generation. The goal is to compute the optimal planning graph G^{p^*} based on the outputs of geophysical mapping selection, i.e., various sets of MIFs for each application $a_{i,j,k} \in \mathcal{A}_{i,j}$. The set of all generated MIF sets $\mathcal{M}^{ifr}_{i,j,k,m,n}$ for $a_{i,j,k}$ is denoted by $\hat{\mathcal{M}}_{i,j,k} = \{\mathcal{M}^{ifr}_{i,j,k,m,n} | \forall f^{ifr}_{i,j,k,m,n} \in \mathcal{F}^{ifr}_{i,j,k,m}, \forall f^{info}_{i,j,k,m} \in \mathcal{F}^{info}_{i,j,k}\}$. Besides, the set of all required applications at all service sites of all communities is $\hat{\mathcal{A}} = \{a_{i,j,k} | \forall a_{i,j,k} \in \mathcal{A}_{i,j}, \forall s_{i,j} \in \mathcal{S}_i, \forall c_i \in C\}$. In this phase, the optimal planning graph is generated by determining: (i) which application $a_{i,j,k} \in \hat{\mathcal{A}}$ to implement and (ii) which MIF $\hat{\mathcal{F}}$ (i.e., mapping) from which set $\mathcal{M}^{ifr}_{i,j,k,m,n} \in \hat{\mathcal{M}}_{i,j,k}$ (i.e., which infrastructure and information flows) to implement $a_{i,j,k}$.

In the rest of this section, we develop a suite of algorithms to solve these two subproblems.

Intermediate planning graph. During planning graph generation, an intermediate planning graph $\hat{G}(K) = (\hat{V}(K), \hat{E}(K))$ is built with K MIFs, where $\hat{G}_{1(2)}(K) = (\hat{V}_{1(2)}(K), \hat{E}_{1(2)}(K))$ is the first (second) layer. For each MIF $\hat{\mathcal{F}}$, its infrastructure flow and corresponding information flow are included in $\hat{G}_2(K)$ and $\hat{G}_1(K)$, respectively. $\langle R(u_1), R(u_2)\rangle$ represents the actual data flow at the infrastructure layer for each edge $(u_1, u_2) \in \hat{E}_1(K)$, where information units u_1 and u_2 are assigned to infrastructure $R(u_1) \in \hat{V}_2(K)$ and $R(u_2) \in \hat{V}_2(K)$, respectively. $\hat{\delta}_{dp}(K)$ and $\hat{\delta}_{op}(K)$ denote the cumulative deployment and operational costs of $\hat{G}(K)$, respectively. Besides, $\mathcal{T}_K(l)$ represents the set of infrastructures mapped to a candidate location $l \in \mathcal{L}$ for $\hat{G}(K)$.

Fusing operations. To generate the planning graph, an extra MIF $\hat{\mathcal{F}}$ is *fused* into the intermediate planning graph $\hat{G}(K)$, which is denoted as $\hat{G}(K) = \hat{G}(K-1) + \hat{\mathcal{F}}$. $\hat{\mathcal{F}} = \{f(u) | \forall u \in V^{ifr}\}$ is a mapping result of infrastructure flow $f^{ifr}_{i,j,k,m,n} = (V^{ifr}, E^{ifr}) \in \mathcal{F}^{ifr}_{i,j,k,m}$ implementing information flow $f^{info}_{i,j,k,m} = (V^{info}, E^{info}) \in \mathcal{F}^{info}_{i,j,k}$. Besides, $\hat{G}(K)$ and $\mathcal{T}_K(l)$ are set to $\hat{G}(K-1)$ and $\mathcal{T}_{K-1}(l)$ initially; similarly, $\hat{G}(K)$'s cumulative costs $\hat{\delta}_{dp}(K)$ and $\hat{\delta}_{op}(K)$ are initialized to $\hat{\delta}_{dp}(K-1)$ and $\hat{\delta}_{op}(K-1)$. The operation includes each infrastructure $v \in V^{ifr}$ and v's corresponding assigned information unit $u \in V^{info}$ into $\hat{G}(K)$ as follows.

(1) When a (previously included) infrastructure $v' \in \mathcal{T}_K(f(v))$ is identical to v in $\hat{G}(K)$, we have the following three cases.
   (a) u and v are *merged* into $\hat{G}(K)$, i.e., reuse the existing infrastructure and information unit if one of the following conditions holds: (i) v' has been assigned an information unit u', i.e., $R(u') = v'$, and u' is identical to u, (ii) u and u' have the same inward information units, i.e., $\exists e(u'_{in}, u') \in \hat{E}_1(K)$, such that u'_{in} is identical to u_{in}, $\forall e(u_{in}, u) \in E^{info}$, and (iii) the identical inward information units are mapped to the same infrastructure, $R(u'_{in}) = R(u_{in})$.
   (b) v is *merged* into $\hat{G}(K)$ while u is *branched* out of $\hat{G}(K)$, i.e., reuse the existing infrastructure while adding an extra information unit to $\hat{G}_1(K)$ and an edge to v' if: (i) all information unit assigned to v is distinct to u, i.e., u is distinct to u', $\forall u' \in \hat{V}_1(K)$ whose $R(u') = v$, and (ii) v has enough resources, i.e., Eqs. (3d)–(3f) hold. Besides, if u has an inward information unit u', i.e., $\exists(u', u) \in f^{info}_{i,j,k,m}$, an edge between u' and u is added in $\hat{G}(K)$ after u' and u are included.
   (c) u and v are not included if only condition (i) of case (b) holds. Then, $\hat{\mathcal{F}}$ is excluded from $\mathcal{M}^{ifr}_{i,j,k,m,n}$ since v has insufficient computing resources[1].

(2) When no infrastructure $v' \in \mathcal{T}_K(f(v))$ is identical to v, u and v are *branched* out of $\hat{G}(K)$, i.e., adding an extra information unit and infrastructure to $\hat{G}(K)$. Then, the connections to u and v in $f^{info}_{i,j,k,m}$ and $f^{ifr}_{i,j,k,m,n}$ are attached to the corresponding vertices in $\hat{G}(K)$. Finally, the costs of v are accumulated, i.e., $\hat{\delta}_{dp}(K) = \hat{\delta}_{dp}(K) + \delta_{deploy}(v, f(v))$ and $\hat{\delta}_{op}(K) = \hat{\delta}_{op}(K) + \delta_{op}(v)$. Then, $\mathcal{T}_K(f(v)) = \mathcal{T}_K(f(v)) \cup v$.

4.2 Optimal Solution for IoT Planning

SmartParcels derives the optimal solution by adopting a composite solution including (i) Enumeration (ENUM), which outputs every possible geophysical mapping of infrastructures for all possible infrastructure flows and (ii) Dynamic Programming (DP), which outputs the planning graph by systematically examining all possible combinations of the generated mappings.

Enumeration (ENUM). To derive the optimal solution, ENUM conducts an exhaustive search for all possible MIFs for each infrastructure flow of all required applications. Specifically, for each application $a_{i,j,k} \in \mathcal{A}_{i,j}$ required at a service site $s_{i,j} \in \mathcal{S}_i$ by a community $c_i \in C$, ENUM examines each infrastructure flow $f^{ifr}_{i,j,k,m,n} = (V^{ifr}, E^{ifr}) \in \mathcal{F}^{ifr}_{i,j,k,m}$ that implements information flow $f^{info}_{i,j,k,m} \in \mathcal{F}^{info}_{i,j,k}$ for $a_{i,j,k}$ as follows.

(1) ENUM starts from selecting all possible geophysical mapping functions $f(v)$ for each sensor $v \in V^{sen}_{i,j,k,m,n} \subset V^{ifr}$ such that $s_{i,j}$ is within v's sensing range r^{sen}_v, i.e., $dist(f(v), s_{i,j}) \leq r^{sen}_v$. That is, ENUM selects a combination of mappings for the sensors, e.g., C^{10}_4 for mapping 4 sensors to 10 candidate locations, and each possible arrangement is denoted as $\hat{\mathcal{F}} = \{f(v) | \forall v \in V^{sen}_{i,j,k,m,n}\}$.
(2) For each $\hat{\mathcal{F}}$, ENUM further selects all possible mappings $f(u)$ for each infrastructure u adjacent to the sensors in $f^{ifr}_{i,j,k,m,n}$,

[1] For simplicity, we assume a candidate location hosts one device for each device type.

i.e., $dist(f(v), f(u)) \leq \min(r_v^{tr}, r_u^{tr})$. Similarly, a combination of mappings is generated, and each possible arrangement is attached to $\hat{\mathcal{F}}$ separately as a new result, i.e., $\hat{\mathcal{F}} = \hat{\mathcal{F}} \cup \{f(u) | \forall u, \exists (v,u) \in E^{ifr}\}$.

(3) ENUM recursively performs step (2) for the infrastructures adjacent to the previously examined ones until all infrastructures are examined.

Finally, ENUM outputs a set of all possible MIFs $\mathcal{M}_{i,j,k,m,n}^{ifr}$ for each $f_{i,j,k,m,n}^{ifr}$. Each MIF is represented by $\hat{\mathcal{F}}$ containing the selected mappings for all infrastructures, i.e., $|\hat{\mathcal{F}}| = |V^{ifr}|$. That is, $\forall f(v) \in \hat{\mathcal{F}}$, $\forall \hat{\mathcal{F}} \in \mathcal{M}_{i,j,k,m,n}^{ifr}$, if $v \in V_{i,j,k,m,n}^{sen}, dist(f(v), s_{i,j}) \leq r_v^{sen}$; otherwise, if $\exists (v,u) \in E^{ifr}, dist(f(v), f(u)) \leq \min(r_v^{tr}, r_u^{tr})$.

Dynamic Programming (DP). Initially, the intermediate planning graph is empty, i.e., $\hat{G}(0)$. DP recursively builds the planning graph by fusing an extra MIF $\hat{\mathcal{F}}$ into the planning graph, i.e., $\hat{G}(K) = \hat{G}(K-1) + \hat{\mathcal{F}}$. In other words, DP recursively implements an application $a_{i,j,k} \in \hat{\mathcal{A}}$ at service site $s_{i,j} \in \mathcal{S}_i$ required by a community $c_i \in C$ by applying $\hat{\mathcal{F}}$. Particularly, DP examines each application $a_{i,j,k} \in \hat{\mathcal{A}}$ as follows.

(1) For each $a_{i,j,k}$, DP generates an intermediate planning graph $\hat{G}(1)$ for each MIF $\hat{\mathcal{F}} \in \mathcal{M}_{i,j,k,m,n}^{ifr}$, $\forall \mathcal{M}_{i,j,k,m,n}^{ifr} \in \hat{\mathcal{M}}_{i,j,k}$ by fusing $\hat{\mathcal{F}}$ into $\hat{G}(0)$, i.e., $\hat{G}(1) = \hat{G}(0) + \hat{\mathcal{F}}$.
(2) For each derived $\hat{G}(1)$, DP calculates its *service utility gain* by summing all the service utility derived by fusing $\hat{\mathcal{F}}$. That is, let $\hat{s}$ denote the set of service sites which require the same application as $a_{i,j,k}$ and within the sensing range of sensors after mapping. The gain is the sum of Eq. (1) for each $s_{i,j} \in \hat{s}$. Then, DP records the service utility of each $\hat{G}(1)$.
(3) DP: (i) excludes $a_{i,j,k}$ from $\hat{\mathcal{A}}$, (ii) excludes the application required by $s_{i,j} \in \hat{s}$ from $\hat{\mathcal{A}}$, (iii) selects the next application from $\hat{\mathcal{A}}$, and (iv) performs the same process on top of each generated $\hat{G}(1)$ and then generates a set of $\hat{G}(2)$.
(4) DP recursively repeats step (3), i.e., $\hat{G}(K+1) = \hat{G}(K) + \hat{\mathcal{F}}$, until all MIFs in $\mathcal{M}_{i,j,k,m,n}^{ifr}$, $\forall \mathcal{M}_{i,j,k,m,n}^{ifr} \in \hat{\mathcal{M}}_{i,j,k}$, $\forall a_{i,j,k} \in \hat{\mathcal{A}}$ cannot be fused, i.e., at least one of the constraints in Eqs. (3b)–(3f) is violated for all MIFs.

Last, DP outputs the planning graph with the maximum service utility. However, the execution time grows drastically when the number of the followings increase: (i) communities, (ii) service sites, (iii) required applications, (iv) implementation methods (information and infrastructure flows), and (v) candidate locations. Take a city containing 38 communities as an example. Assuming each community has only 1 service site, and each demands only 3 applications. Further, assume each application has only one implementation method, and each method has only one possible mapping. Such a simplified setting still takes DP 114! recursion procedure calls to generate the planning graph. Supposing a recursion is finished within a clock cycle, with a recently released Intel I9-10900K Processor (5.3 GHz and 20 threads), DP takes around 2.8×10^{170} days to generate the planning graph.

4.3 Clean-Slate Planning Heuristics

The optimal solution could take an extremely long time to complete. Hence, we propose a heuristic algorithm for each subproblem: (i) Selection (SEL), which selects the (more promising) mappings with higher expected service utility and then with higher communication coverage to reduce the number of mappings and (ii) Maximum Reusability (MR), which iteratively selects the infrastructure flow with the maximum reusability of the instrumented devices.

Selection (SEL). We propose novel selection policies to prune out less-promising mappings at runtime of ENUM. The policies contain the following intuitions: (i) MIFs with higher utilities should be included earlier and (ii) MIFs with higher communication coverage (how many candidate locations are covered) should be included earlier. For each $f_{i,j,k,m,n}^{ifr}$, the utility can be estimated by Eq. (1) after mapping all the sensors (assuming the graph is connected); similarly, the communication coverage of a network device can be estimated after mapping. Specifically, let M and N denote user-specified pruning criteria for top M utility and top N communication coverage. The pruning policies are applied as follows.

(1) In step (1) of ENUM, after ENUM selects all possible mapping functions for sensors, only $\hat{\mathcal{F}}$ with the top M utility are passed to step (2).
(2) In step (2) of ENUM, only the mappings with the top N communication coverage are included in $\hat{\mathcal{F}}$ if the examined infrastructure is a network device.

Hence, SEL outputs a set of mappings with a smaller size but higher service utility and communication coverage as compared to ENUM.

Maximum Reusability (MR). We propose a novel MR algorithm to generate the planning graph efficiently. Instead of examining all possible combinations of MIFs, MR iteratively: (i) selects an application $a_{i,j,k} \in \hat{\mathcal{A}}$ to implement and (ii) fuses a MIF $\hat{\mathcal{F}} \in \mathcal{M}_{i,j,k,m,n}^{ifr}$, where $\mathcal{M}_{i,j,k,m,n}^{ifr} \in \hat{\mathcal{M}}_{i,j,k}$, into the planning graph $\hat{G}(K)$ according to $\hat{\mathcal{F}}$'s *reusability*. We define a reusability index, which considers both the *investment efficiency* and *communication coverage gain* when fusing $\hat{\mathcal{F}}$ as follows.

Investment efficiency is the ratio of the *service utility gain* to the *cost gain* after fusing $\hat{\mathcal{F}}$ into $\hat{G}(K)$. That is, the more infrastructures are reused, the lower the costs while fusing $\hat{\mathcal{F}}$. Specifically, let $\Delta U(\hat{\mathcal{F}}, \hat{G}(K))$ denote the service utility gain after fusing $\hat{\mathcal{F}}$ into $\hat{G}(K)$, which can be derived by the same procedure in step (2) of DP. After fusing $\hat{\mathcal{F}}$ into $\hat{G}(K)$, the cost gain $\Delta\delta(\hat{\mathcal{F}}, \hat{G}(K)) = \hat{\delta}_{dp}(K) - \hat{\delta}_{dp}(K-1) + \hat{\delta}_{op}(K) - \hat{\delta}_{op}(K-1)$. Hence, $\mathcal{I}_{eff}(\hat{\mathcal{F}}, \hat{G}(K)) = \frac{\Delta U(\hat{\mathcal{F}}, \hat{G}(K))}{\Delta\delta(\hat{\mathcal{F}}, \hat{G}(K))}$ is the investment efficiency of fusing $\hat{\mathcal{F}}$ into $\hat{G}(K)$.

Communication coverage gain is determined by the locations of network devices. If a network device has a higher communication coverage after deploying, fewer network devices are needed. One extreme case is instrumenting all network devices at the edge of the communities, which requires the largest number of network devices. Let $\mathcal{L}_{cov}(K) \subset \mathcal{L}$ denote the candidate locations within the communication coverage of the network devices in $\hat{G}(K)$. The communication coverage gain after fusing $\hat{\mathcal{F}}$ into $\hat{G}(K)$ is $\mathcal{I}_{cov}(\hat{\mathcal{F}}, \hat{G}(K)) = \mathcal{L}_{cov}(K) - \mathcal{L}_{cov}(K-1)$.

Reusability index is defined as a weighted sum of investment efficiency and communication coverage gain when fusing a MIF $\hat{\mathcal{F}}$ into the intermediate planning graph $\hat{G}(K)$. Let α_{eff} and α_{cov} denote the weights of investment efficiency and communication coverage gain. The reusability index is written as $\mathcal{I}(\hat{\mathcal{F}}, \hat{G}(K)) = \alpha_{eff}\mathcal{I}_{eff}(\hat{\mathcal{F}}, \hat{G}(K)) + \alpha_{cov}\mathcal{I}_{cov}(\hat{\mathcal{F}}, \hat{G}(K))$. Without loss of generality, we assume $\alpha_{eff} + \alpha_{cov} = 1$.

With the above notations, MR starts with an empty planning graph $\hat{G}(0)$. MR iteratively selects an application $a_{i,j,k} \in \hat{\mathcal{A}}$ to implement by fusing a MIF $\hat{\mathcal{F}} \in \mathcal{M}^{ifr}_{i,j,k,m,n}$, where $\mathcal{M}^{ifr}_{i,j,k,m,n} \in \hat{\mathcal{M}}_{i,j,k}$, into the current intermediate planning graph $\hat{G}(K)$ as follows.

(1) For each $a_{i,j,k} \in \hat{\mathcal{A}}$, MR examines the reusability index $\mathcal{I}(\hat{\mathcal{F}}, \hat{G}(K))$ of each MIF $\hat{\mathcal{F}} \in \mathcal{M}^{ifr}_{i,j,k,m,n}, \forall \mathcal{M}^{ifr}_{i,j,k,m,n} \in \hat{\mathcal{M}}_{i,j,k}$.
(2) MR fuses the MIF $\hat{\mathcal{F}}$ with the maximum $\mathcal{I}(\hat{\mathcal{F}}, \hat{G}(K))$ into $\hat{G}(K)$ and excludes the corresponding applications from $\hat{\mathcal{A}}$ (the same application for each $\hat{\mathcal{F}}$ as step (3) in DP).
(3) MR repeats steps (1) and (2) until at least one of the constraints in Eqs. (3b)–(3f) is violated for all MIFs of the remaining applications, or all derived reusability indices are zero.

Finally, MR outputs the planning graph.

Extended MR (MR$^+$). We extend MR to MR$^+$ in order to provide a control knob for urban planners to trade-off complexity, i.e., execution time, and optimality, i.e., overall service utility. MR$^+$ exploits DP to derive an intermediate planning graph. Then, MR$^+$ executes MR and fuses additional MIFs into the generated planning graph to derive the planning graph. MR$^+$ supports user-specified termination criteria for DP, including but not limited to execution time and the number of included MIFs (i.e., implemented applications). MR$^+$ works with ENUM and SEL to generate the final planning graph, as shown in Fig. 2.

4.4 Planning for Retrofit

Communities may have been equipped with some devices that urban planners can exploit. By reusing these devices, the urban planners can derive a higher service utility. Hence, we extend SmartParcels for the retrofit scenario, which includes existing devices. We propose a RETrofit (RET) algorithm that takes a set of existing devices $\mathcal{D}$ into consideration while selecting the mappings for each infrastructure flow. Let l_d denote the location of each existing device $d \in \mathcal{D}$. We modify ENUM into RET as follows.

(1) In step (1) of ENUM, if (i) $d \in \mathcal{D}$ is identical to sensor v and (ii) service site $s_{i,j}$ is within d's sensing range r^{sen}_d, i.e., $dist(l_d, s_{i,j}) < r^{sen}_d$, where $r^{sen}_d = r^{sen}_v$, an extra mapping result, $f(v) = l_d$ is included in $\hat{\mathcal{F}}$.
(2) If (i) $d \in \mathcal{D}$ is identical to infrastructure u (step (2) of ENUM) and (ii) d and sensor v or previously examined adjacent infrastructure u' are within each other's transmission range (step (3) of ENUM), i.e., $dist(l_d, f(v)) \leq \min(r^{tr}_d, r^{tr}_v)$ or $dist(l_d, f(u')) \leq \min(r^{tr}_d, r^{tr}_{u'})$, where $r^{tr}_d = r^{tr}_u$, an extra mapping result, $f(u) = l_d$ is included in $\hat{\mathcal{F}}$.

After the modifications, RET generates a subset of all possible MIFs for each infrastructure flow. We notice that the pruning techniques of SEL can be applied to RET as well. Like ENUM and SEL, RET works with DP and MR, as shown in Fig. 2.

4.5 Generalizability and Limitations

We propose a general framework, SmartParcels, for the cross-layer IoT planning problem with relatively static demands. The proposed modeling method of information and infrastructure flows supports a range of applications, such as (i) simple flows from data/sensor to the analytic model/processing unit and (ii) complex flows containing a database, sensor fusion, and multiple various data sources. Moreover, by decoupling the applications and communities, urban planners can easily apply the modeled flows to other communities requiring the same applications. However, SmartParcels does not support IoT planning in dynamic scenarios, where information and infrastructure flows change over time, such as mobile IoT devices and in wireless sensor networks with cluster head changes over time. We leave the dynamic scenario to our future work.

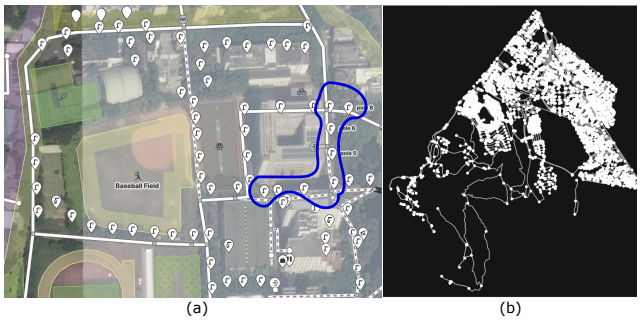
5 PERFORMANCE EVALUATIONS

5.1 Setup

Settings. We conduct the experiments in two real-world settings: (i) a *smart campus setting* consisting of a faculty housing community and the EECS college of National Tsing Hua University in Taiwan and (ii) a *smart city setting* consisting of four diverse communities of Irvine in California, USA. Fig. 3(a) shows the smart campus testbed with eight smart streetlights (mirrored L-shape region on the right) and 63 regular streetlights. The following IoT, computing, and network devices are installed on the smart streetlights: (i) 5 cameras, (ii) 1 motion sensor, (iii) 4 WiFi routers, and (iv) 2 edge servers. The smart streetlights are connected through a gateway to the EECS building via a Gigabit Ethernet cable. The university deploys an auto-dimming application for the streetlights to save energy. There are two ways to achieve auto-dimming: (i) image-based analysis and (ii) motion sensor detection. The sets of service sites and candidate locations are identical, including the smart and regular streetlights. The details of streetlight locations and installed devices are added to the publicly available OpenStreetMap (OSM) database [26], which is queried in our evaluations.

In the smart city, we consider the following communities: (i) A, Irvine Spectrum, which is an outdoor shopping center, (ii) B, Quail Hill, which is a residential area close to a highway, (iii) C, Shady Canyon Open Space Preserve, which is a wildland, and (iv) D, Shady Canyon, which is a residential area located next to a wildland. C and D are identified as *very high fire hazard severity zones* by the Orange County fire authority [5]. The applications required by each community are as follows: (i) A demanding for gunshot detection, (ii) B demanding for air quality and noise monitoring, (iii) C demanding for wildfire detection, and (iv) D demanding for wildfire detection and air quality monitoring. Moreover, we derive service sites and candidate locations using OSM through the OSMnx [11] library as follows. First, we retrieve the boundary of each community and draw the smallest square that can completely surround the community. Then, we split the square into 16 equal-size cells and use the center of each cell to represent a service site of the corresponding community. Second, we consider the public facilities (found in OSM) as candidate locations since they

are managed by the authority with easier access. We also consider road intersections as candidate locations because they are often close to city infrastructures, such as electricity and the Internet. We query the OSM database to get all road segments for identifying intersections, which are shown in Fig. 3(b). Table 1 lists the number of public facilities and intersections in each community. Last, we uniformly sample the candidate location from these locations across the communities so that communities have similar numbers of candidate locations.



(a) (b)

Figure 3: Two real-world settings: (a) streetlights on a smart campus and (b) road segments in a smart city.

Table 1: Numbers of the Public Facilities and Intersections

Community	A	B	C	D	Community	A	B	C	D
Traffic signal	185	0	0	13	Power pole	44	0	52	0
Train station	3	0	0	0	Fire station	1	0	0	0
Hospital	1	0	0	0	School	1	0	0	2
Retail	34	0	0	6	Intersection	9189	1721	734	2051

Information and infrastructure flows. Fig. 4 shows three sample flows used in our evaluations. We use a colon to indicate the corresponding infrastructure/information unit for a flow, e.g., *{sensing data:sound}*. Moreover, we use parentheses to indicate multi-network choices, e.g., *{network device:(wifi,lora)}*. In addition, we use: (i) *sd* for sensing data, (ii) *a* for analysis model, (iii) *s* for sensors, and (iv) *n* for network devices. With the notations, we can write the information/infrastructure flows in a concise manner, such as the three sample flows in Fig. 4. In addition, we also consider the following information/infrastructure flows in our evaluations.

(1) **Object detection:** *{sd:image,a:object detector via image, s:camera,n:wifi}* and *{sd:motion,a:object detector via motion, s:motion sensor,n:wifi}*.
(2) **Noise monitoring:** *{sd:sound,a:noise monitor,s:mic.,n:wifi}*.
(3) **Air quality monitoring:** *{sd:emission,a:air quality monitor,s:gas sensor,n:(wifi,lora)}*.

Parameters of information units and infrastructures. We derive the parameters via specifications, empirical experiments, or reasonable assumptions to the best of our knowledge, which are summarized in Table 2. For the costs, we refer to online retails, specifications, minimum wages, cost of power, or reasonable assumptions. Table 3 lists the deployment (one time) and operational (for 24 hours) costs of each infrastructure.

Table 2: Parameters Used in the Experiments

Parameters		Values
Bandwidth consumption	Image	10 Mbps
	Motion sample	1.92 Kbps
	Emission reading	0.64 Kbps
	Sound	128 Kbps
Computing resource requirement	Image	80 Mbps
	Motion sample	3.84 Kbps
	Emission reading	1.28 Kbps
	Sound	384 Kbps
Computing resource	Edge server	6.8 GHz (4 × 1.7 GHz)
Transmission range & bandwidth	WiFi AP	50 m/100 Mbps
	Lora gateway	1 km/50 Kbps
Sensing range	Camera	15 m (campus) 400 m (suburban)
	Motion sensor	10 m
	Gas sensor	600 m
	Microphone	300 m
Sensing parameter α_v in Eq. (2)	All sensors	Reciprocal of sensing range

Table 3: Deployment and Operational Costs in USD

Infrastructure	Camera	Motion sensor	Gas sensor	Mic.	WiFi AP	Lora Gateway	Edge Server
Deployment cost (campus)	1399	360	-	-	236	-	376
Deployment cost (city)	1863	-	735	686	700	632	840
Operational cost (both)	12.02	8.38	5.51	15.75	8.02	20.04	67.03

5.2 Measurement Studies for Accuracy Models

Accuracies achieved for applications depend on the implemented models. We describe our approach to deriving the accuracies.

Smart campus. According to the literature, the auto-dimming application leveraging image and motion samples to achieve different accuracy levels: 92.7% (images) [27] and 80% (motions) [38]. To validate the numbers, we conduct experiments in our smart streetlight testbed. First, we take a 10-minute surveillance video on a weekday. We then randomly select 5 frames per minute and manually label the ground truth of whether there exists any object next to a streetlight. Finally, we use YOLOv3 [30] to detect the objects, which leads to an accuracy level of 96%. In addition, we get the readings of the motion sensor during the same time and adopt a threshold-based method to detect objects. This results in an accuracy level of 86%. We notice that our experiments give similar accuracy levels to those in the literature. Hence, we apply the accuracy levels in the prior work [27, 38] in our evaluations.

Smart city. Because the monitoring applications merely compute the aggregated sensor readings without complicated analysis, we set their accuracy levels to 100%. For the gunshot detection, prior reports indicate an 89.6% accuracy [23]. For wildfire detection, we use image- and emission-based detection. We gather data from two real datasets: HPWREN [3], containing surveillance videos of several wildland fires, and use World Air Quality Index (WAQI) [1], for air quality. Specifically, we select six wildfire events that occurred in California, USA [2]. Then, we retrieve 5538 images capturing the wildfire events and 5844 images without a wildfire from HPWREN. We uniformly divide the dataset into 80% training and 20% testing sets. We train a CNN model, which achieves an accuracy

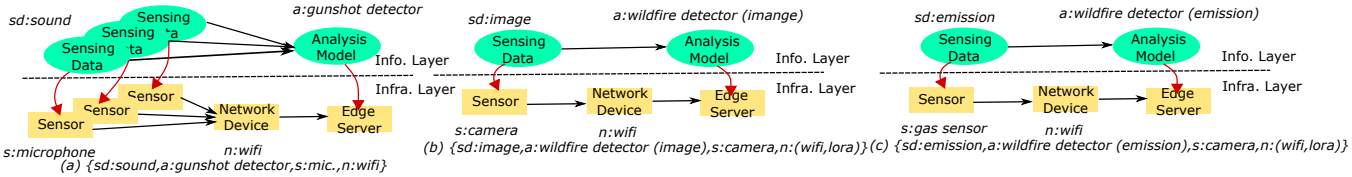


Figure 4: Sample information/infrastructure flows used in our evaluations: (a) gunshot detection and (b), (c) wildfire detection.

level of 94.41%. Besides, we collect the emission data (pm 2.5, O3) for one day from the stations closest to the wildfire events in WAQI database. We label the data during wildfire events as *true* using the open data from the government [2] and the daily emission data on other days as *false*. We get 576 days with wildfire and another 576 days without wildfire and divide them into 80% training and 20% testing sets. We use SVM (Support Vector Machine) to predict wildfire events, which achieves an accuracy level of 62.62%.

Last, we note that the accuracy levels of different applications carry diverse meanings. Therefore, using absolute accuracy levels leads to biased service utility and IoT plans. For instance, the monitoring applications always give perfect accuracy levels and will be favored by IoT planning algorithms. To mitigate this issue, we normalize the accuracy levels to the maximum achievable accuracy of each application and use the normalized accuracy in the following evaluations. Moreover, the weight $\beta_{i,j,k}$ in Eq. (3a) of both wildfire detection and gunshot detection is 0.35, and each monitoring application has 0.15 weight.

5.3 Clean-Slate Algorithms

To the best of our knowledge, the cross-layer IoT planning problem has not been solved in the literature. Therefore, we develop a baseline algorithm called Maximum Utility (MU), which mimics the behaviors of the urban planners. Specifically, MU generates a planning graph by iteratively selecting an application's MIF with the maximum utility value until running out of resources. We compare SmartParcels with MU in both smart campus and smart city settings. We consider the following metrics: (i) the overall service utility, (ii) the cost performance index, which is the ratio of the overall service utility to the overall cost, (iii) the number of MIFs included in the planning graph, (iv) the services rate, which is the fraction of services provided by the planning graph, and (v) the execution time. We vary the following parameters: (i) the number of candidate locations, (ii) the deployment and operational budgets, (iii) the termination criteria of MR$^+$, (iv) the pruning conditions of SEL, and (v) the weights of MR to study their implications on various performance metrics. We sample random candidate locations and run different IoT planning algorithms. For statistically meaningful results, we repeat the experiment of the same parameters ten times and report the average results with one standard deviation interval if applicable.

Results for clean-slate planning (smart campus). We compare MR and MU generating planning graphs from ENUM's results for clean-slate IoT plans in the smart campus. The deployment and operational budgets are 25000 and 300 USD. Besides, we evaluate MR with various weights: (i) MR (eff), where $\alpha_{eff} = 1$, $\alpha_{cov} = 0$, (ii) MR (cov), where $\alpha_{eff} = 0$, $\alpha_{cov} = 1$, (iii) MR (e25c75), where $\alpha_{eff} = 0.25$, $\alpha_{cov} = 0.75$, (iv) MR (e50c50), where $\alpha_{eff} = 0.5$, $\alpha_{cov} = 0.5$, and (v) MR (e75c25), where $\alpha_{eff} = 0.75$, $\alpha_{cov} = 0.25$. Fig. 5 shows that MR (e75c25) achieves the best overall performance. In Fig. 5(a), MR outperforms MU by up to 2 times on the overall service utility. That is, MR instruments infrastructure at proper locations to *double* the overall service utility. Even though maximizing the communication coverage, i.e., MR (cov), seems to be futile, the importance of communication coverage manifests when the number of candidates grows. In fact, MR (e75c25) derives higher overall service utility than MR (eff), which only considers efficiency and ends up with deploying more network devices. Besides, MR (e75c25) is a steady approach for urban planners because it has a low performance variation. Fig. 5(b) shows that MR (e75c25) has the highest number of infrastructure flows in the planning graph, which explains its high service utility. Fig. 5(c) shows that MR (e75c25) supports the most services in the smart campus: around 60% of the demands. Last, MR (e75c25) invests carefully and derives a higher service utility per unit cost, as shown in Fig. 5(d). In summary, we empirically compare MR's performance under different weights and find the best weights of $\alpha_{eff} = 0.75$ and $\alpha_{cov} = 0.25$, which will be applied in the following experiments if not otherwise specified.

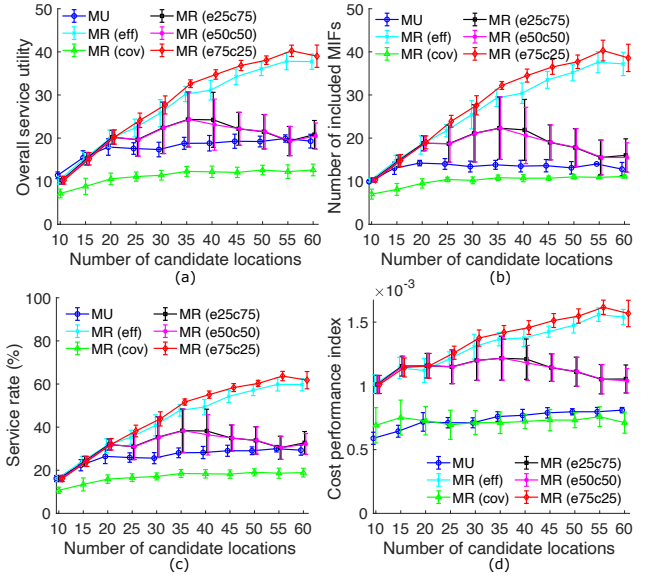


Figure 5: Experiment results of clean-slate plan in the smart campus: (a) overall service utility, (b) number of included MIFs, (c) service rate, and (d) cost performance index.

Implication of diverse budgets. Intuitively, higher budgets lead to higher overall service utility. We study the impact of different budgets on the same smart campus setting with 35 candidate locations. Figs. 6(a)–(b) show that MR achieves much better utilization of the budgets as compared to MU. In fact, MR obtains the maximum overall service utility that can be provided by the candidate locations under lower budgets. However, MR's overall service utility saturates even when the budgets exceed 30K/350. This is because the quantity (number) and quality (distance to service sites) of candidate locations restrict the performance of MR.

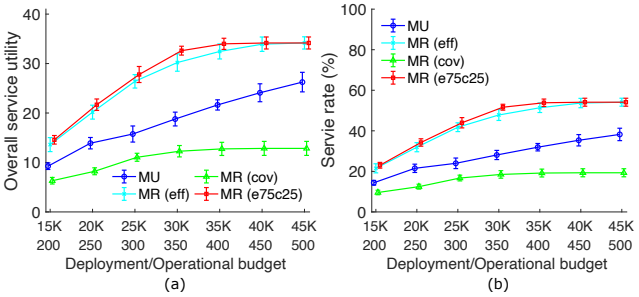


Figure 6: Experiment results under different budgets: (a) overall service utility and (b) service rate.

Results for clean-slate planning (smart city). We then evaluate our algorithms in a city-level environment. The deployment and operational budgets are 65000 and 450 USD, respectively. We compare ENUM with SEL to demonstrate the effect after pruning when MR and MU are exploited to generate the planning graph. First, Fig. 7(a) reports the number of generated mappings by SEL with different pruning criteria compared to ENUM. The number of all possible mappings (ENUM) increases exponentially when the number of candidate locations grows. Though the number of generated mappings from SEL also grows exponentially, SEL prunes up to 90.57% of them as shown in this figure. Fig. 7(b) shows that MR has a tolerable loss in performance; however, MU's performance deteriorates when the number of options decreases. Interestingly, as shown in Fig. 7(c), MR identifies the most reusable MIFs and derives a high service utility. This means that ENUM-MR includes fewer MIFs, yet achieves the highest overall service utility. Last, Fig. 7(d) shows that SEL imposes minimal influence on MR in terms of services rate, whereas MU is incapable of pinpointing the MIFs for higher reusability and higher overall service utility. In summary, we empirically confirm that SEL effectively prunes less-promising mappings, and MR carefully identifies MIFs with high reusability to maximize the overall service utility in larger (city-level) problems.

Comparison to the optimal solutions. We compare the performance of MR$^+$ against that of DP, where ENUM is adopted for geophysical mapping selection. In such setup, DP gives Optimal (OPT) planning graphs, while MR$^+$ gives sub-optimal ones. MR$^+$, however, offers a controllable *termination criterion* K of its initial DP phase for trading optimality off efficiency. In particular, the initial DP phase terminates after adding K MIFs and outputs the intermediate planning graph. Afterward, MR finalizes the generation process with the weight of reusability indices, $\alpha_{eff} = 0.75$ and

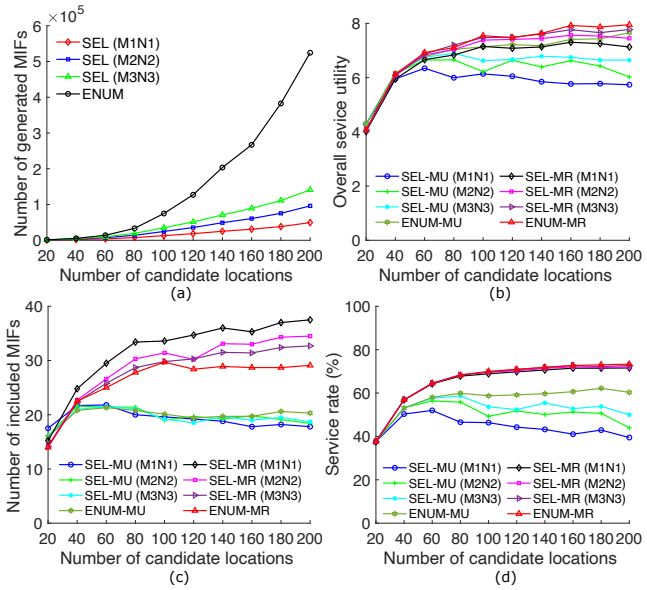


Figure 7: Experiment results in the smart city: (a) amount of generated mappings, (b) overall service utility, (c) number of included MIFs, and (d) service rate.

$\alpha_{cov} = 0.25$ in the MR phase. Figs. 8(a)–(b) show that DP produces OPT solutions but with a long execution time, especially when the number of candidate locations increases. In contrast, Figs. 8(a)–(b) manifest MR$^+$'s trade-off between optimality and efficiency. When K increases, the performance of MR$^+$ increases; however, in the meantime, the execution time increases drastically.

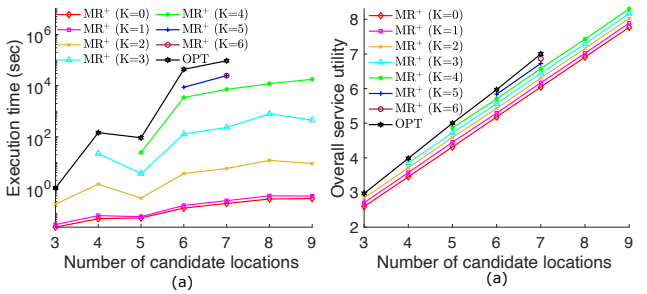


Figure 8: Compared to OPT, MR$^+$ successfully trades-off: (a) execution time and (b) overall service utility.

5.4 Results for Retrofit

We evaluate SmartParcels by solving the retrofit problems in the smart campus with the existing smart streetlight testbed. We compare the performance of our proposed algorithms in retrofit and clean-slate problems. For a fair comparison, we reuse the settings of the clean-slate problems in the smart campus, except adding the existing devices on the smart streetlights to the retrofit problems. The resulting IoT plans in the retrofit problems are, therefore, quite similar to those in the corresponding clean-slate problems. Fig. 9(a)

shows the overall service utility derived by MR and MU, while Fig. 9(b) reveals the retrofit's gain of service utility, i.e., the difference of service utility between clean-slate and retrofit. The figures depict that MR exploits the existing devices for a significant performance improvement, which is as high as 7.64 times compared to MU.

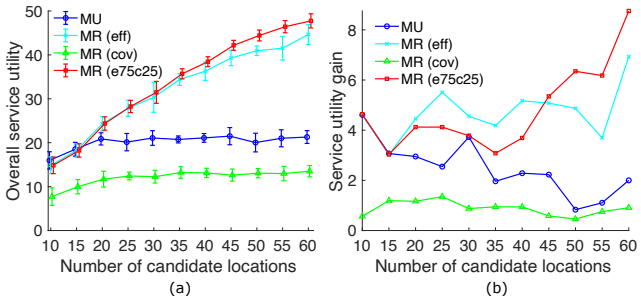


Figure 9: Experiment results from retrofit-smart campus: (a) overall service utility and (b) service utility gain.

5.5 Scalability Studies

We study the scalability of SmartParcels since the previous simulations are conducted by assuming that an event occurs only at the service site. We implement a simulator in python that: (i) generates events uniformly within the boundary of the testbeds, (ii) determines whether an event is captured by a planning graph, and (iii) calculates the service utility for the captured events using Eq. (1). We first generate 10000 events for each application and compare the planning graph generated by ENUM+MR and ENUM+MU in both testbeds: the smart campus and smart city. Fig. 10(a) shows sample clean-slate results in the smart campus. We observe that MU derives higher overall service utility with fewer candidate locations, which can be attributed to the more than sufficient budgets (since the campus is relatively small). However, when the number of candidate locations increases, MR performs better since MR carefully instruments the devices with higher reusability. Fig. 10(b) reports sample clean-slate results from the smart city, which is larger than the smart campus. This figure reveals that MR captures the events and derives higher overall service utility, which can be explained by the relatively insufficient budgets. In summary, the results confirm that our proposed algorithms outperform the baseline algorithm under system dynamics, especially when the budgets are scarce.

6 FUTURE RESEARCH DIRECTIONS

We propose SmartParcels, a framework for IoT planning in smart cities that SmartParcels provides solutions to clean-slate (from scratch) and retrofit (with existing devices) scenarios for urban planners. This platform supports a design-space exploration of possibilities to trade off efficiency (execution time) and optimality (overall service utility) for customized applications that are relevant to the community at hand. While SmartParcels can help urban planners find a valuable plan, the nature of events and data needs in communities changes dynamically, and analysis tools that allow us to explore the unpredictability and its impact on performance in an interactive manner is useful. In ongoing work, we are exploring

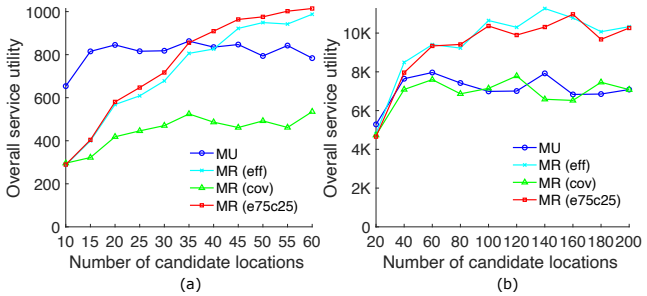


Figure 10: Overall service utility from our event-driven simulator, sample results from the clean-slate problems in: (a) smart campus and (b) smart city.

the design of a model-driven lightweight analytics tool where variability in a planning graph can be studied. In particular, we adopt a queueing network for our lightweight SmartParcels platform in two parts: (i) modeling the infrastructures (transmission and processing capability) and (ii) modeling variability (network and data burstiness). We exploit methods for rapid analysis of well-known queueing models to impose end-to-end analytical models that provide approximate performance metrics, such as latency, throughput, and utilization. Our lightweight SmartParcels provides prompt analysis on variability in a generated plan, and urban planners can react accordingly. Two theoretical extensions further explore: (i) the complexity analysis in Theorem 1 to determine approximation ratios of SmartParcels for performance guarantees and (ii) providing formal guarantees in placement [33]. Finally, we are exploring the dynamicity of the system in the following topics: (i) how Software Defined Network (SDN) can help for the effective management in the multinetwork data flows [28] and (ii) the ability to dynamically choose edge analytics [10, 34].

ACKNOWLEDGEMENT

We acknowledges the funding support from the University of California Laboratory Fees Research Program funded by the UC Office of the President (UCOP), grant ID LFR-20-653572 and NSF IIS Award #40284388.

REFERENCES

[1] Air Quality Open Data Platform . https://aqicn.org/data-platform/register/.
[2] CAL Fire . https://www.fire.ca.gov/.
[3] HPWREN . https://hpwren.ucsd.edu/.
[4] IQ FireWatch . https://www.iq-firewatch.com/technology.
[5] Orange County Fire Authority . https://https://www.ocfa.org/.
[6] Securaxis Acoustic Monitoring . https://securaxis.com/sounds-analytics.
[7] Wildfire Detection System InsightFD + Insight Globe. https://www.insightrobotics.com/en/services/wildfire-detection-system/.
[8] Ai, J., et al. Coverage by directional sensors in randomly deployed wireless sensor networks. *Journal of Combinatorial Optimization 11*, 1 (2006), 21–41.
[9] Aoun, B., Boutaba, R., Iraqi, Y., and Kenward, G. Gateway placement optimization in wireless mesh networks with QoS constraints. *IEEE Journal on Selected Areas in Communications 24*, 11 (2006), 2127–2136.
[10] Benson, K., Fracchia, C., Wang, G., Zhu, Q., Almomen, S., Cohn, J., D'arcy, L., Hoffman, D., Makai, M., Stamatakis, J., et al. Scale: Safe community awareness and alerting leveraging the internet of things. *IEEE Communications Magazine 53*, 12 (2015), 27–34.
[11] Boeing, G. Osmnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks. *Computers, Environment and Urban Systems 65* (Sep 2017), 126–139.

[12] Cai, Y., et al. Target-oriented scheduling in directional sensor networks. In *IEEE INFOCOM* (2007), pp. 1550–1558.

[13] Cardei, M., et al. Maximum network lifetime in wireless sensor networks with adjustable sensing ranges. In *IEEE WiMob* (2005), vol. 3, pp. 438–445.

[14] Davidson, M., and Dolnick, F. *A planners dictionary.* American Planning Association, Planning Advisory Service, 2004.

[15] Dou, R., and Nan, G. Optimizing sensor network coverage and regional connectivity in industrial iot systems. *IEEE Systems Journal 11*, 3 (2015), 1351–1360.

[16] Drabu, Y., et al. Gateway placement with qos constraints in wireless mesh networks. In *International Conference on Networking* (2008), IEEE, pp. 46–51.

[17] Feige, U. A Threshold of ln n for Approximating Set Cover. *Journal of the ACM (JACM) 45*, 4 (1998), 634–652.

[18] Fusco, G., and Gupta, H. Selection and orientation of directional sensors for coverage maximization. In *6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks* (2009), IEEE, pp. 1–9.

[19] Grand View Research. Smart cities market size & analysis, industry report, 2020-2027, 2020. https://www.grandviewresearch.com/industry-analysis/smart-cities-market.

[20] Gravalos, I., Makris, P., Christodoulopoulos, K., and Varvarigos, E. A. Efficient network planning for internet of things with qos constraints. *IEEE Internet of Things Journal 5*, 5 (2018), 3823–3836.

[21] He, B., et al. Optimizing the internet gateway deployment in a wireless mesh network. In *IEEE International Conference on Mobile Adhoc and Sensor Systems* (2007), pp. 1–9.

[22] Li, F., et al. Gateway placement for throughput optimization in wireless mesh networks. *Mobile Networks and Applications 13*, 1-2 (2008), 198–211.

[23] Lim, H., Park, J., and Han, Y. Rare sound event detection using 1d convolutional recurrent neural networks. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events Workshop* (2017), pp. 80–84.

[24] Liu, L., et al. On directional k-coverage analysis of randomly deployed camera sensor networks. In *IEEE ICC* (2008), pp. 2707–2711.

[25] Megerian, S., et al. Exposure in wireless sensor networks: Theory and practical solutions. *Wireless Networks 8*, 5 (2002), 443–454.

[26] OpenStreetMap contributors. Planet dump retrieved from https://planet.osm.org . https://www.openstreetmap.org, 2017.

[27] Peng, Q., et al. Pedestrian detection for transformer substation based on gaussian mixture model and yolo. In *8th international conference on intelligent human-machine systems and cybernetics (IHMSC)* (2016), vol. 2, IEEE, pp. 562–565.

[28] Qin, Z., Denker, G., Giannelli, C., Bellavista, P., and Venkatasubramanian, N. A software defined networking architecture for the internet-of-things. In *IEEE network operations and management symposium (NOMS)* (2014), pp. 1–9.

[29] Qiu, L., et al. Optimizing the placement of integration points in multi-hop wireless networks. In *Proceedings of ICNP* (2004), vol. 4.

[30] Redmon, J., and Farhadi, A. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767* (2018).

[31] Reinhart, C., et al. Umi-an urban simulation environment for building energy use, daylighting and walkability. In *13th Conference of International Building Performance Simulation Association, Chambery, France* (2013), vol. 1.

[32] Solórzano, A., et al. Fire detection using a gas sensor array with sensor fusion algorithms. In *2017 ISOCS/IEEE International Symposium on Olfaction and Electronic Nose (ISOEN)* (2017), IEEE, pp. 1–3.

[33] Venkatasubramanian, N., Talcott, C., and Agha, G. A. A formal model for reasoning about adaptive qos-enabled middleware. *ACM Transactions on Software Engineering and Methodology (TOSEM) 13*, 1 (2004), 86–147.

[34] Venkateswaran, P., Hsu, C.-H., Mehrotra, S., and Venkatasubramanian, N. Ream: Resource efficient adaptive monitoring of community spaces at the edge using reinforcement learning. In *IEEE International Conference on Smart Computing (SMARTCOMP)* (2020), pp. 17–24.

[35] Walter, E., et al. A verification of citysim results using the bestest and monitored consumption values. In *Proceedings of the 2nd Building Simulation Applications conference* (2015), Bozen-Bolzano University Press.

[36] Wang, B. Coverage Problems in Sensor Networks: A Survey. *ACM Computing Surveys (CSUR) 43*, 4 (2011), 1–53.

[37] Wang, J., and Medidi, S. Energy efficient coverage with variable sensing radii in wireless sensor networks. In *Third IEEE international conference on wireless and mobile computing, networking and communications* (2007), IEEE, pp. 61–61.

[38] Wu, L., Wang, Y., and Liu, H. Occupancy detection and localization by monitoring nonlinear energy flow of a shuttered passive infrared sensor. *IEEE Sensors Journal 18*, 21 (2018), 8656–8666.

[39] Zhou, Z., Das, S. R., and Gupta, H. Variable radii connected sensor cover in sensor networks. *ACM Transactions on Sensor Networks (TOSN) 5*, 1 (2009), 1–36.

[40] Zou, Y., and Chakrabarty, K. A distributed coverage-and connectivity-centric technique for selecting active nodes in wireless sensor networks. *IEEE Transactions on Computers 54*, 8 (2005), 978–991.

[41] Zou, Y., et al. Sensor deployment and target localization in distributed sensor networks. *ACM Transactions on Embedded Computing Systems 3*, 1 (2004), 61–91.