Incentivizing User-centric Resource Allocation in Wireless Networks in Realtime

Submitted in partial fulfillment for the requirements for
the degreee of
Doctor of Philosophy
in
Electrical & Computer Engineering

Madhumitha Harishankar
B.S., Electrical & Computer Engineering, Rutgers University

Carnegie Mellon University
Pittsburgh, PA

December 2020

I dedicate this thesis to my late grandfather, Keemu.
Your love and blessings mean the world to me.

Acknowledgments

I express my gratitude, first and foremost, to my advisors Prof. Carlee Joe-Wong and Prof. Patrick Tague for their support, encouragement and advise throughout my doctoral study. I am deeply thankful for the long rope they gave me to explore my research interests as well as for the timely direction and insights they provided that kept my pace steady. They have entertained innumerable requests for meetings and discussions at a ten minute notice, and have been extremely patient as I struggled through learning curves and sometimes wavered on the problem statement to pursue. A single question posed by them would often expose critical gaps in my carefully constructed argument, and get me rethinking my approach. Carlee and Patrick, I am a better thinker because of you. During the five years of my doctoral study, numerous life-changing events unfolded in my personal life. Words cannot express the gratitude I feel for the genuine compassion and understanding that my advisors showed during these times and for giving me the room to process these changes and evolve as a person. Being advised by Prof. Patrick Tague and Prof. Carlee Joe-Wong has taught me that the most effective professional counsel from a mentor comes from a place of truly caring for the protege. Patrick and Carlee, I will always be deeply grateful to have had you as my advisors and strive to be a mentor like you.

I would also like to take this opportunity to thank my thesis committee members, Prof. Aron Laszka and Dr. Anand Raman. I am very grateful for their invaluable feedback, time, and help.

This research was supported in part by the National Science Foundation under grant CNS-1645759 and in part by Carnegie Mellon University's Electrical and Computer Engineering Department. The views and conclusions contained here are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either express or implied, of CMU, NSF, or the U.S. Government or any of its agencies.

I am thankful to my wonderful colleagues and collaborators, including

past and current members of the LIONS and MEWS research groups. I am especially grateful for the multiple collaborations with Jun Han in the initial days of my doctoral study, which gave me invaluable insights into how to frame a problem statement and execute a research project. I also thank Dimitrios-Georgios Akestoridis of the MEWS group and Jinhang Zuo of the LIONS group for their valuable contributions to my research projects. I would also like to thank the M.S. students who collaborated with me on various projects - Nagarjun Srinivasan, Sireesha Pilaka, Pragya Sharma, Faisal Ali Alqarni, Huijing Zhou, Qingyu Chen, Roberto Del Valle Rodriguez and Aarushi Wadhwa. I am also grateful to Prof. Anupam Datta, Prof. Bob Iannucci, Prof. Osman Yagan and Prof. Pei Zhang - they spent valuable time sharing their insights on my research, giving me feedback on various talks and applications, and helping recommend me for numerous opportunities to further my doctoral study.

The friendships that I made during this time helped make my PhD journey less lonely. I want to thank my friends - Akshay Chandrasekaran, Abhinav Jauhri, Aniruddha Basak, Aishwarya Prem Renu, Manika Murali, Swati Goswami, Sarita Ghosh, Ravali Yannamaddi, Manisha Mukherjee, Tyler Nuanes, Dolly Duan and Vinodh Pramesh. I especially thank Akshay, Aniruddha and Abhinav for the timely help they always rendered, whether over whiteboarding sessions, sharing their GPUs or over coffee and dinner. I thank my friends Manika and Aishwarya for the wonderful times together and for dutifully cheering me up after each rejection.

My husband Sriram V. Iyer has been my steadfast companion and unyielding supporter every step of the way during my graduate studies. He inspired me and gave me confidence, often burnt the midnight oil to review my applications and papers when I was stressed, brainstormed with me when I was stuck, critically validated (and sometimes invalidated) my ideas when I was uncertain of them, taught himself new topics to be able to follow along my research and contribute, helped with my experiments when I had too much on my plate, bought me Starbucks and food when I did not have time to step out of the lab, and made sure food was delivered at my door minutes after my paper deadlines even when he was in a different country, knowing full well I would not have eaten all day. He lent not only

his towering intellect and precious time to my graduate studies, but also his unconditional love. Lots of love to you too, Sriram!

I am always indebted to my beloved parents and sister, Saraswathi Lakshmanan, Harishankar Krishnamurthy, and Niveda Harishankar, for their unconditional love, prayers, and encouragement. I could not have made the leap to quit a promising career in industry and start on a long-winding path towards an unclear research goal without their complete support. Neither could I have withstood the downs of this journey without them. My mom's conviction in the value of a sound and meaningful education, and my dad's fearless attitude in the face of adversities became my mental crutches to lean on during my doctoral studies. In my sister, I have always had a shoulder to lean on without fear of judgement. She has been my companion in right and wrong, in ups and downs, in joy and sadness throughout my life and so too during my doctoral study. I love you fam! And thanks Nivi for making illustrations for my papers and working up your photoshop magic on a moment's notice! And for the PhD-care-basket filled with pens, sharpies, notebooks, postits, paper clips and more!

I also thank the kind and loving folks of my extended family, including grandparents, aunts, uncles, nieces, nephews, in-laws and cousins - a call or visit to any of them put a smile on my face even during the most stressful of times. Special thanks to my aunt Meenakshi Venkateswaran for the numerous discussions about market theory in economics and for always being just a phone call away to address any doubts I had on the subject. My special thanks also to Babu for his timely tutorial on elliptic curve cryptography that helped push my project along.

I would also like to thank Shree Lal (Didi) and members of the Sai family for their constant word of encouragements during my studies. They helped me pick myself back up after the toughest paper rejections and I'm indebted to them for the stream of affection, support and prayers that they have sent my way during these five years.

Finally, I am deeply grateful to have encountered Swami Paramarthananda's lectures which, filled with patience, clarity, and humility, introduced me to the philosophy of Vedanta and allayed the tempests of my mind in the

most difficult of times.

"Shraddha aur Saburi"(Conviction and Patience) - Shirdi Sai Baba

Abstract

In this thesis, I propose mechanisms for user-centric resource allocation in wireless networks. I consider a series of practical motivating contexts that progressively require lesser trust and reliance on the network provider and allow for more flexible connectivity schemes benefitting end-devices, especially for emerging connectivity use-cases like the IoT. The granularity of typical month-long mobile data plans is such that users must forecast their network usage over a month and assign a single monetary value to its utility. Finer-grained real-time information about user needs does not play a role in resource allocation, though users determine their needs and launch mobile applications only in realtime. This results in unrealized value for both the end-user and the network operator and further restricts the user to availing resources that belong only to their subscribed network(s).

Inspired by Verizon's recent PopData offering, I first consider supplementing typical monthly subscription plans with ad-hoc discount offers, wherein users may consume unlimited data for the offered hour for a small fixed fee. This allows users to realize any additional resource needs for their sessions in realtime by utilizing these simple offers without the risk of incurring a data overage, while also affording the network a predictable contract revenue. Second, I consider a user-driven approach to acquiring network resources by proposing a model wherein a slice of resources is dynamically created and assigned to a device based on the session needs it specifies. Devices can then reliably estimate their session performance at the onset. I explore how these models can be made incentive-compatible for the network and the user, show that they can be executed in realtime albeit at a steep cost to users since they are unable to plan spending optimally in realtime, and that this suboptimality can be alleviated with reinforcement learning techniques. Finally, I remove the inherent device-network trust relationship that exists in these models by allowing devices to seamlessly authenticate with any access point (without subscriptions) and make real-time payments for consumed data, using public and permissionless blockchains, in a scalable and secure manner.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

End-devices and last-mile wireless networks that provide internet connectivity to them treat each other as untrusted black-boxes and hence require a-priori identity, trust and payment setup in the form of long-term subscriptions to interact. This model does not allow finer-grained real-time information about a user's network needs to play a role the resource allocation process, though users often determine their network needs sand launch mobile applications in realtime. Network resources available to users are inherently limited relative to their needs, atleast during certain congested times-of-day and locations, and hence users inability to influence the resource allocation process with their realtime valuations results in a poor network experience. This model further restricts users to primarily connecting only to networks with which they have subscriptions established. The lack of any real-time and incentive-compatible information exchange between devices and the network introduce significant suboptimalities in user utility from data consumption. It occludes the evolution of a more dynamic, decentralized and scalable network that is suitable for the explosion of data and devices that is expected with the Internet of Things (IoT). This thesis identifies the core research challenges in solving these issues, and introduces and validates techniques to address them.

1.1 Issues with Subscriptions for Internet Access

As new internet paradigms like the IoT and, more broadly, Cyber-Physical Systems (CPS) emerge, modern wireless networks are faced with an increasing heterogeneity of resource demands and performance requirements. Machine to machine scenarios

(e.g., tactile internet [74], telepresence [101]) that require low latency, high bandwidth and high reliability simultaneously are integral CPS use-cases, as are smart-city scenarios [191] that require periodic transmission of IoT data to the cloud and low-latency computing resources at the edge for making real-time actuation decisions. Simultaneously, the end users' diversity in network requirements and volume of data use also grows. Multimedia applications with bandwidth and/or latency sensitive traffic make up the vast majority of mobile traffic today. In fact, as per Cisco's forecast [50], video will constitute 78% of all mobile data traffic by 2021. Supporting these highly diversified network needs of existing and emerging applications continues to pose significant challenges to wireless networks and forms the basis of 5G's vision for wireless networks [28].

Even as these applications proliferate, our daily experiences with them continue to be filled with *uncertainty*. For example, a smartphone user making a video call while on the bus is entirely uncertain about the call quality. In-fact, the call may drop altogether due to poor signal strength, noise or congestion. The user's resulting experience with the application is indeterminate. This inadequacy results directly from the black-box interaction paradigm between the network and users today, wherein applications launched by users have no means of gauging the network's resource availability, and the network conversely cannot ascertain the user's app-usage intentions. This inadequate interaction model is enabled by long-term subscription contracts wherein no significant real-time communication of network requirements or associated payments is required between devices and the network. The network typically relies on a few passively inferred indicators about network flows to ascertain their Quality of Service (QoS) needs and attempts to schedule resources to fulfil them, subject to preemption factors like congestion. Monthly subscription contracts establish the necessary identity and payment relationships between these otherwise untrusting entites and allow them to interact (albeit suboptimally) in realtime.

Specifically, we identify two critical shortcomings of today's networking model.

First, there exists **no mechanism for real-time communication of network needs and corresponding valuations** by devices and **for the guaranteed procurement of corresponding resources** from the Internet Service Provider (ISP). With typical 4G/LTE/5G connections, the network flows of a device are passively analyzed and classified by the cellular network into a QoS class (e.g. QCI in the LTE) which is intended to satisfy a specific SLA (in terms of bitrate, packet drop,

jitter etc). However, flows complete for available network resources at any QCI level, and can hence be starved or preempted at any time (e.g. during congestion) since network resources are (periodically) allocated only for the span of a few milliseconds (e.g. TTI in LTE and NAV in IEEE 802.11). On the other hand, end-user engagement with applications, especially multimedia apps, occurs over the span of a few minutes[1][2] [42, 159]. Since devices are unable to relay any real-time session-oriented resource needs or corresponding payments to the network and instead only make a monthly payment for an approximate estimation of their total data utilization (as with typical monthly subscriptions), they are unable to influence the actual resource allocation process in real-time in accordance with their session-oriented needs. This leads to *indeterminate* quality of experience for devices connecting to wireless networks. If, instead, user-driven session-aware resource procurement in realtime were possible, we would see the emergence of application protocols that are *proactively network-aware*, rather than *reactively* network-aware like DASH. In fact, Zou et al. [200] show that existing adaptive bit-rate algorithms achieve only $69-86\%$ of the optimal possible QoE from their realized bandwidth allocations simply due to not knowing the bandwidth availability information beforehand.

Network Slicing, one of 5G's key architectural innovations to handle these diverse and potentially stringent resource needs of internet applications, involves allocations of sufficient physical resources to a QoS level (aka slice) to satisfy the traffic demand [79]. These virtual slices potentially span resources all the way from the edge to the core of the network and are dedicated to satisfying demands of a specific service level end-to-end. However, this model continues to result in the same indeterminate network experiences that today's networks do since a central content provider or network operator owns slices and controls the admission of end-devices into slices *based on subscription contracts instead of real-time device valuations*. In other words, the slice owner does not know how to *prioritize among its users*. For instance, a content provider like Skype contracts a slice from the network operator that delivers low latency and high bandwidth. This enables high-quality Skype calls for users admitted to the slice but physical resources within the slice continue to be limited. Presumably, when multiple users make Skype calls at congested times, they compete

[1] https://vertoanalytics.com/chart-week-winning-mobile-video-app-war/

[2] https://www.statista.com/statistics/579411/top-us-social-networking-apps-ranked-by-session-length/

for admission into the slice and have no way to influence the outcome, just as is the case today. For example, a user cannot demonstrate to the slice allocation algorithm their higher call *utility value* for a job interview over a recreational call from another user. **Hence, while admission into a slice largely guarantees slice-specified SLA, admittance itself is entirely controlled by the centralized operator or the content provider in any case.**

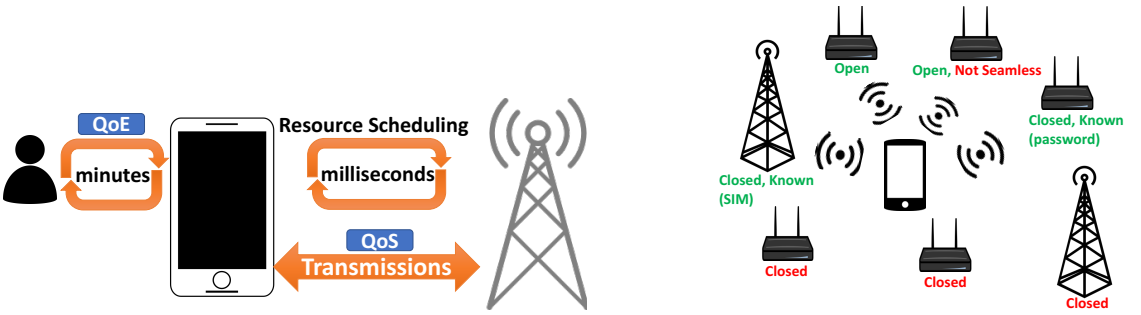
Further envisioned Internet of Things (IoT) devices have considerable variations in their data needs, which also makes different usage-based contracts appropriate for different devices [54]. For example, a camera continuously sending a video stream, a temperature sensor sending a single measurement every two hours, and radar sensors on an autonomous vehicle requiring millisecond latencies all upload varying quantities of data at varying frequencies and qualities-of-service (QoS). Further, a device's resource needs may not be fixed, for e.g., if it uploads data in response to some environment triggers. The current model of estimating utilization (simply in terms of net download/upload bytes) at a month-level granularity and signing up for a corresponding QoS-agnostic payment tier fails to meet the diverse requirements of these devices. Providing for this will become even more challenging as IoT deployments grow: the number of smart cities worldwide is expected to grow at a rate of 26% through year 2022 [94], making up a 34 billion USD market in 2019 [19]. As discussed earlier, while network slicing promises to allow devices to subscribe to specific network-supported QoS levels of service in their subscription contracts, proposed models continue to occlude end-devices from influencing the allocation process in realtime, thereby subjecting them to indeterminate network experience.

The second significant shortcoming of today's subscription-based interaction model between devices and networks is the resulting **inability of devices to exploit available network resources belonging to unsubscribed networks**. Subscription contracts establish a-priori identity relationships (e.g. through SIM) between the network and the device and serve to introduce trust in the interaction. The device trusts the usage accounting done by the network and the network trusts that the device will pay for its usage at the end of the billing cycle. Such a model strictly restricts end-devices to connecting only to known networks with which identity associations and usage/payment agreements have been setup a-priori (and out-of-band), or to open networks. Devices hence have a severely constrained view of network resources available for consumption at their location at any time. In-fact, our analysis of the

LifeMap mobility dataset [48] that contains fine-grained mobility information collected over the course of a few months for a set of students in South Korea shows that at any given location, participants are in range of $2-17$ closed WiFi hotspots and almost no open ones. They are, however, presumably restricted to accessing at-most one of these (e.g. if they own the hotspot and thereby have a subscription with the ISP). Even taking into account cellular networks, users are likely able to access only one network that they already have a SIM-based subscription with. Such a severely restrictive connectivity model scales even more poorly to the IoT. The rapid and dense deployment of IoT devices (e.g. in smart cities) based on the current subscription-based connectivity model will require device owners to manage increasingly complex usage contracts with operators' LoRaWAN (Low-power long-range wide area network) or NB-IoT (Narrowband - Internet of Things) networks, posing a prohibitively unscalable and expensive bottleneck for large-scale IoT deployments. The overhead of provisioning dedicated contracts for each device may accelerate as 5G networks are more widely installed. Indeed, such networks are expected to include multiple access points of different radio access technologies, potentially with different operators, making it even more difficult to pre-specify contracts for individual IoT/user devices with each operator. In-fact, a significant portion of IoT devices in smart-city deployments are expected to be WiFi-equipped and could potentially realize their variant data needs by accessing readily available (and potentially closed) WiFi hotspots nearby. Indeed, a study from 2016 conducted in the city of Turin found that approximately 50% of streets around a block were within transmission range of in-home [175] WiFi networks. Exploiting these private hotpots, however, is impossible with the prevalent model wherein only open or known hotspots (for which the owner has entered into a subscription with an ISP) are seamlessly accessible.

1.2 Identifying Core Research Challenges

Incentive compatibility issues underpin these identified shortcomings with using subscriptions for accessing wireless networks. Moving away from the long-term subscription model to a dynamic negotiation of network requirements, resource allocation, costs and payment structure introduces several theoretical as well as practical research challenges in aligning the network's incentive to maximize revenue with the user's incentive to maximize their cost-sensitive utilities from internet usage. These challenges span



(a) QoS vs QoE timescale

(b) Custodial Auth and Payment per Network

Figure 1.1: (a) It is unclear how to make session-level performance guarantees in a wireless medium when resource allocation happens in timescales that is orders of magnitude lower. (b) Devices are restricted to connecting only to subscribed or open networks, using network-specific authentication and payment mechanisms.

various stages of the device-network connectivity process, as illustrated in Figure 1.2.

First, long-term data contracts afford predictable revenues to the ISPs. A pricing strategy based purely on real-time network usage introduces, atleast initially, a substantially high degree of uncertainty in the ISP revenue. This makes it crucial to ensure that users' truthful valuations and requirements are extracted, ensuring that the real-time pricing of network's resources are competitive and adequately reflect competition. Second, such truthful elicitation of resource requirements and valuations is non-trivial to achieve in real-time, i.e. in the timescales where users launch applications and spontaneously expect to be allocated sufficient network resources for their session. Several strongly truthful auction allocation strategies, for instance, require solving NP-hard problems [108] that would require several minutes or hours before users can be informed of their allocation outcome. These techniques are therefore a poor fit for real-time determination of allocation outcome for spontaneously launched user sessions. Further, these assume that users are perfectly rational, which may well not be true when they are budget constrained and make decisions in realtime that are influenced by perceived risks of over-spending for data consumption over time. Third, users may, in-fact, end up making more suboptimal choices under the constraints of repeated real-time decision making for their data usage as opposed to a simple monthly usage and utility forecast. Dynamic estimation of future requirements, prices, and budget constraints is highly challenging and, in-fact, the network may design its pricing strategy to exploit the suboptimality in user decisions that results

from inherent lack of knowledge about future prices the network plans to set. Fourth, it is unclear whether predictable QoS can be ensured in the timescales typical of session durations in a wireless medium. As Figure 1.1(a) depicts, resources in a cellular network are periodically scheduled every few milliseconds, which is hence the timescale for QoS guarantees, whereas user-facing Quality of Experience (QoE) with the session is experienced in the timescale of minutes [42, 159]. Indeed, available spectrum resources must be quantified and accurately reconciled with session requirements presumably expressed in terms of bitrate, latency and duration. Even then, it is unclear whether performance guarantees of reasonable accuracy are possible despite uncontrollable wireless influences like fast fading.

Fifth, for devices to be able to connect seamlessly to non-subscribed networks and make reliable payments, non-custodial identity and payment management is necessary. Today's networks rely on network-specific setup to authenticate a device into the network and process its payments. For instance, as Figure 1.1(b) depicts, an end-device can connect either through SIM-based auth to its subscribed network with which a payment structure has already been setup off-band, or to a closed WiFi network with which credentials and payment structure have been pre-established or to open hotspots which may prompt the user to setup identity and payments through a captive portal. To truly enable seamless connectivity to any network in realtime, we need a non-custodial identity and payment management framework that is not network-specific and does not involve setup overhead. Sixth, since such an envisioned non-custodial system will be widely used by end-devices and networks ubiquitously for authentication and payments, it must be easy to scale. Finally, it is unclear how to enforce payments as per the dynamically agreed usage terms in this setup when the device and network do not have prior trust relationships. Without proprietary hardware that guarantees honest usage monitoring, neither the end-device nor the network's report of measured usage can be trivially used as ground-truth to then enforce corresponding payments as both may have incentives to misreport this information.

Several of these challenges involve dynamic pricing of network resources to reflect real-time congestion, which has been well-studied in literature [157]. However, proposed techniques have historically been challenging to realize in practice [128] due to the difficulty users incur in making real-time spending decisions with incomplete information about future prices and consumption. Indeed, recent studies have

shown [87] that dynamic pricing is challenging for end-users who are budget constrained; imperfect planning can lead to substially higher data costs at the end of the month than what users may otherwise incur with a fixed monthly subscription. In identifying the core challenges here and addressing them, we specifically focus on *practical solutions* by identifying imminent scenarios where users may stand to benefit substantially from dynamic negotiation of utilization and costs despite the overhead of real-time planning, such as when using real-time multimedia applications (which cannot be bufferred) and for IoT use-cases (where the subscription model scales poorly). In the techniques we develop, we emphasize session-oriented resource pricing and procurement for meaningful QoS at user-perceived timescales, the mitigation of suboptimality incurred in making real-time spending decisions, and seamlessness when connecting to non-subscribed network and paying for these dynamically negotiated contracts.

1.3 Towards Incentivizing User-centric Resource Allocation in Realtime

We now codify these research challenges based on the various stages of a device's interaction with a wireless network and subsequently propose a methodology to tackle them.

1.3.1 Framework

Figure 1.2 depicts the framework that we arrive at. The first stage of the device-network connectivity process is **Authentication**, where the device typically uses network-specific credentials like SIM or PSK to authenticate with the network. This lets the network ascertain the corresponding user's identity and map the initiated session to an existing user account/contract. Here, our goal is to instead enable a non-custodial identity management solution, wherein a universal set of credentials can be used to authenticate against various networks. This overcomes any network-specific out-of-band setup activity like SIM or key establishment as typically done via subscriptions. Such an authentication scheme should utilize well-established networking standards, generalize across radio access network types, and work seamlessly.

Following Authentication, the network executes an Authorization check where it

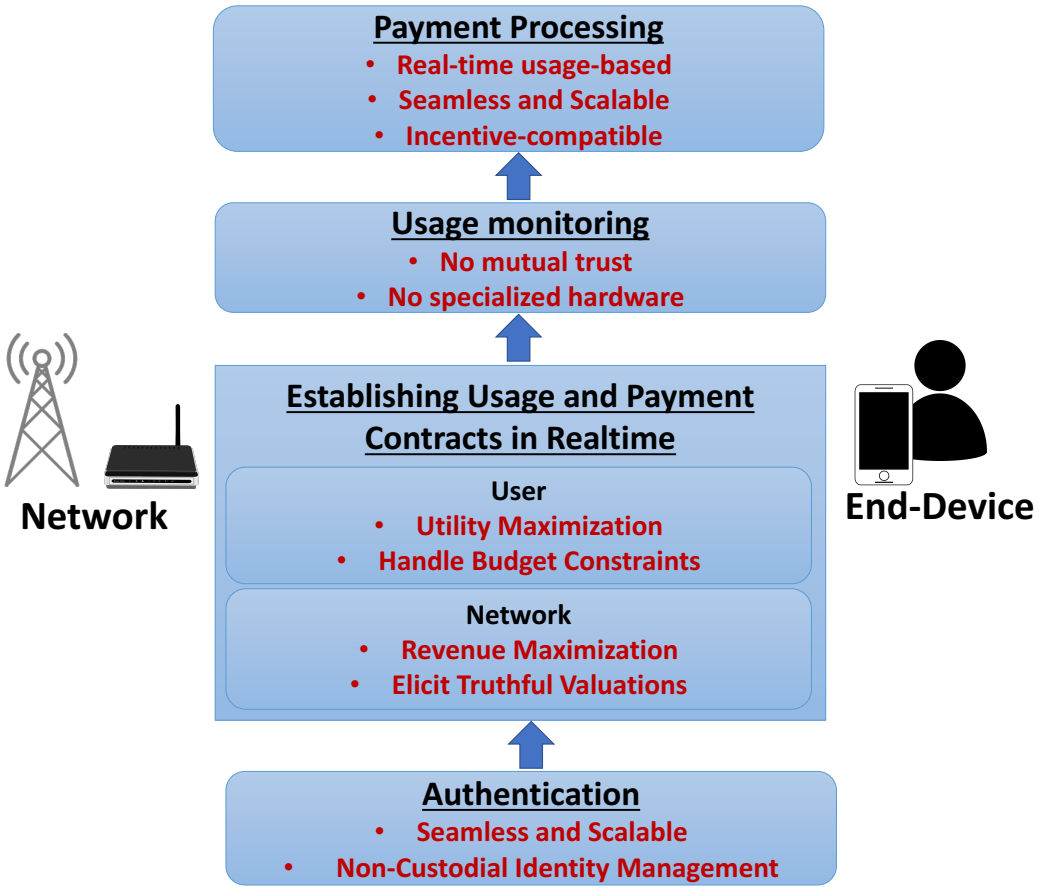


Figure 1.2: Starting from Authentication to Payment Processing, the research challenges in enabling user-centric and incentive-compatible resource allocation affects multiple stages of the device-wireless network connectivity process.

checks the status of the ascertained user's contract (e.g. whether the data quota is reached already or the last bill was paid) and authorizes the establishment of the session if the checks pass. With typical subscription models, such a contract is established out-of-band over a long-term basis and does not capture real-time needs. Here, instead, we aim to facilitate the **establishment of usage and payment contracts in realtime**. We wish to enable dynamic negotiation of resource allocation and costs between the user and the network that reflects the end-user's session needs, utilities, valuations, and buget constraints. The realtime aspect here specifically introduces research challenges both for the network and the user in maximizing their revenues and utilities respectively. If the network moves entirely to a realtime negotiation model, it may well lose any predictable notion of monthly revenue based on the dynamic pricing model used. The user, on the other hand, is forced to routinely forecast future

network pricing and resource availability to make optimal decisions in realtime, at any time. Indeed, the user also loses the predictability of fixed monthly costs for their data usage.

Once a real-time usage contract has been established (correspondingly, the Authorization done), the user's data consumption begins and leads to the question of **usage monitoring**. Subscription contracts also serve to bootstrap trust between the device and the network; the device trusts the network to perform accurate Accounting of the data session, based on which the network levies agreed-upon usage-based costs on the user at the end of the billing cycle. Without subscriptions and the a-priori trust they bootstrap, it is unclear how any usage-based payment structure agreed in the contract just established can be enforced. Hence, our goal here is to enable a data utilization monitoring mechanism that can be reconciled with both the device and the network, without relying on any proprietary purpose-specific trusted hardware. Finally, we aim to facilitate **payment processing** based on utilization measured in realtime, as per the terms of the established contract. Though the network and the device may not have associated previously and have performed no setup ceremony establishing Billing details or authorization, we nonetheless wish to enable seamless and scalable payments from the device to the network that ensures that the data services provided by the network is incentive-compatible.

1.3.2 Desired Functionality and Corresponding Scenarios

Together, these four modules enable new functionalities in wireless networking. Note these these modules simply correspond to the ubiquitous *Authentication, Authorization, Accounting and Billing* framework in telecommunications. In the current legacy implementation based on static subscription models, these functions are simply performed using network-specific identity management (e.g. SIM) for Authentication, assessing the subscription status for Authorization, trusted network hardware for Accounting and out-of-band payment setup process via the susbcription for Billing. As Figure 1.2 illustrates, we instead aim to enable these core functionalities dynamically without relying on pre-established long-term subscriptions. We further categorize these new features into three distinct motivating network scenarios, shown in Figure 1.3. These three network contexts form the basis of our concrete problem statements and further study:

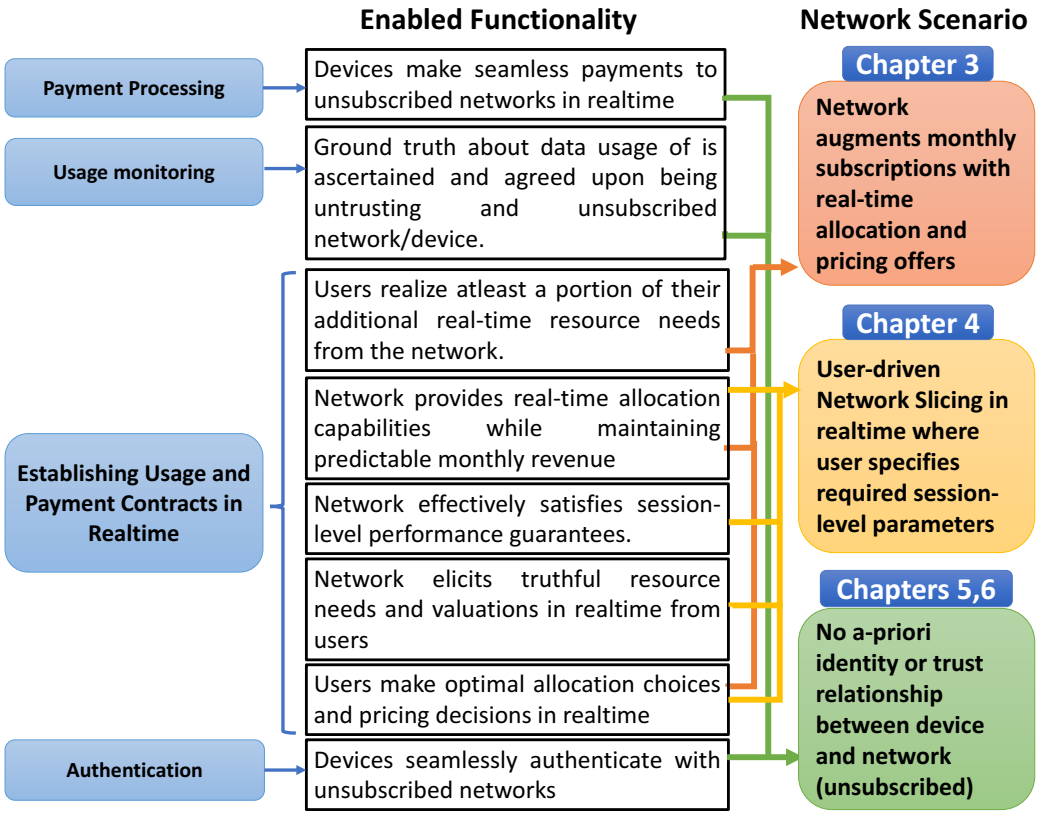


Figure 1.3: We identify three distinct connectivity models that enable an increasingly user-centric resource allocation process and involve decreasing levels of network centralization. Together, these three contexts enable the study of our core modules of interest, namely, non-custodial authentication, establishment of dynamic usage/price contracts, truthworthy usage monitoring of an untrusted device-network session, and in-band payment processing based on real-time usage.

- **Supplementing Subscriptions with Realtime Offers** We first consider a model that captures the notion of influencing the network's resource allocation process with user-preferences in realtime. The ISP introduces a real-time pricing/allocation strategy *on top of* a subscription-based model in the form of *data discount offers*. This allows us to retain the benefits of subscriptions, namely predictable monthly network revenue and predictable monthly data costs for the user, while incrementally introducing real-time session-oriented pricing dynamics. These offers, when made by the ISP, provide a low flat-rate cost for the next hour's data usage and thereby allow users to acquire resources for

their session (typically lasting minutes[3][4]) at this cheaper rate as opposed to the risk of incurring expensive overages. This context captures the challenges in real-time determination of offer decisions for ISPs that seek to maximize their revenue as well as the challenges in real-time decision-making for cost-sensitive users that are unable to accurately predict future offers that the network may run. We show that, with perfect information, these discount offers can be quite beneficial for both ISPs and users; however, the *ad-hoc* nature of these offers results in information asymmetry between ISPs and users which the former can exploit to increase its revenue without proportional increase in data benefit to users. Finally, we illustrate the effectiveness of certain reinforcement learning techniques that users can employ to predict the ISP's offer schedule and mitigate their spending suboptimality.

- **User-driven Network Slicing for Real-time Session Guarantees** We next consider a more user-driven model where, instead of waiting for the ISP to make real-time data discounts in addition to a static monthly data plan, users proactively request the network for the exact network resources they need so as to secure their *session's performance*. We retain the assumption of trusted usage monitoring by the ISP and off-band payments from the subscription model without its pre-agreed utilization and pricing terms. At any time, the network allocates a *slice* [77] of resources to a device that corresponds to its session-level needs; hence, this model requires considering practical challenges of whether session-level performance guarantees can be successfully made in a wireless medium where flows have externalities on each other and the medium itself is subject to noise and fading effects. Further, this model also captures challenges in incentivizing truthful user reports in real-time; without the stability of monthly subscription revenue, we now contend with the necessity to ensure that users report their resource needs and valuations truthfully so as to ensure that such a fully user-driven allocation model is incentive-compatible for the network to offer in terms of revenue. Further, since users must repeatedly decide their valuation for each session's resources in realtime, ensuring that their overall spending during the course of the billing cycle stays within a prespecified budget

[3] https://vertoanalytics.com/chart-week-winning-mobile-video-app-war/

[4] https://www.statista.com/statistics/579411/top-us-social-networking-apps-ranked-by-session-length/

becomes challenging but crucial. We use our learnings from the earlier model to mitigate this significant source of suboptimality stemming from real-time decision making with incomplete future information (regarding future resource availability and prices).

- **Seamlessly Connecting to Unsubscribed Networks and Making Payments** Finally, we remove the assumption of trusted usage monitoring by the ISP and off-band billing systems with monthly payments that we had retained previously. Indeed, practically facilitating session-based incentive-compatible association between a device and a generic network requires also circumventing contractual trust models between devices and access points (cellular or otherwise). The research challenges here span the Authentication, Accounting and Billing functions. A scalable non-custodial identity management system becomes necessary for an end-device to identify and authenticate itself with a generic network; SIM-based access methods are specific to the cellular network in question while PSK can only be established with known hotspots and differ across networks, hence rendering both schemes unsuitable. Further, when the device has no prior trust on the connected network, the device vs network's measurements of data usage becomes challenging to reconcile. While the previous model provides an incentive-compatible mechanism for the network to allocate session-oriented resources to a device in realtime, it is unclear how such a dynamically established contract can be monitored to ascertain whether the terms of the contract were successfully met and the corresponding payment enforced, without a mutually trusted usage monitoring method. With this model, we hence seek to facilitate trustworthy usage measurements of device data usage that both parties agree on and that the device can may payments for in realtime without pre-established off-band payment setup.

1.4 Contributions of This Thesis

The overarching thesis statement spans both theoretical and systems research questions and is framed as follows:

Thesis statement. *Enable end-devices like smartphones, laptops and IoT devices to connect to wireless data networks without a-priori identity or trust relationships, establish a dynamic session-oriented contract specifying usage and pricing terms, and*

provide a mechanism for its enforcement. Ensure that the established contract terms are incentive-compatible for both the network (heeding revenue maximization) and the user (heeding utility maximization and budget constraints), and any suboptimalities for the user stemming from incomplete information during real-time decision-making is mitigated.

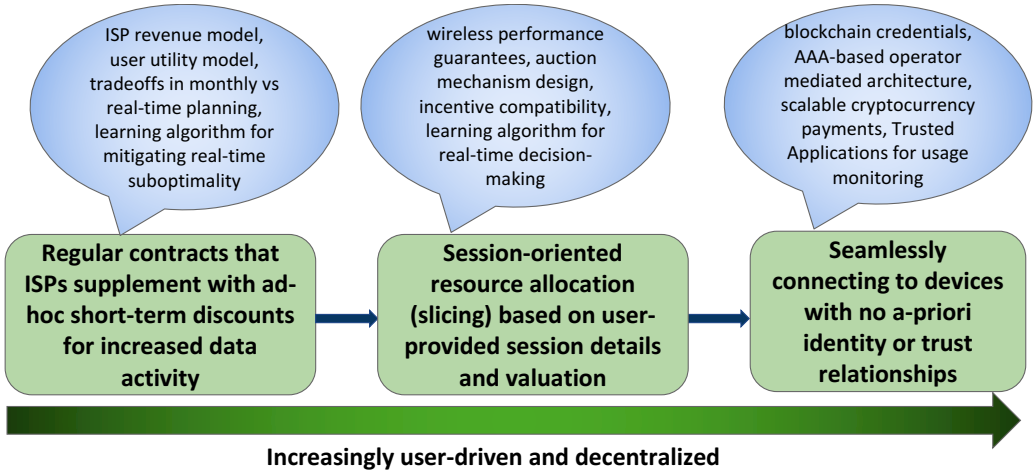


Figure 1.4: I propose to explore the research questions of interest through a variety of networking models that differ in the degree of user-control and network centralization.

Figure 1.4 illustrates the roadmap of this thesis. The three scenarios of study proposed involve networking models that *progressively allow for more user control over the network's resource allocation process and progressively diminish the centralized and trusted role of ISPs today.* The research work in this thesis is presented in this order, elaborating on each scenario's formal definition, research questions, methodology and findings. I now summarize the research contributions made in each of these three studies:

Supplementing Subscriptions with Realtime Offers – As demand for Internet usage increases, Internet service providers (ISPs) have begun to explore pricing-based solutions to dampen data demand. However few explicitly consider the dual problem of monetizing idle network capacity at uncongested times. PopData [178] is a recent initiative from Verizon that does so by offering supplemental discount offers (SDOs) at these times, in which users can pay a fixed fee in exchange for unlimited data in the next hour. This work is the first of its kind to assess the benefits and viability of SDOs by modeling user and ISP decisions as a game, considering both overall monthly decisions and hour-to-hour decisions throughout the month. We first use our monthly

model to show that users are generally willing to accept some SDO offers, allowing the ISP to increase its revenue. We then show that users face a complex hourly decision problem as to which SDOs they should accept over their billing cycles, since they are unaware of their exact future needs or when future SDOs will be made. They must plan their decisions over the billing cycle, despite not knowing their future usage needs or when future SDOs will be made. The ISP faces a similarly challenging problem in deciding when to offer SDOs so as to maximize its revenue, subject to users' decisions. We develop optimal decision criteria for users and ISPs to decide whether to make or accept SDO offers. Our analysis shows that both users and ISPs can benefit from these offers, which we verify through numerical experiments on a one-week trace of 20 cellular data users. We find that ISPs can exploit user uncertainty in when future SDOs will be made to optimize its revenue, but show that reinforcement learning techniques can be employed by users to mitigate this and optimize their real-time spending decisions.

User-driven Network Slicing for Real-time Session Guarantees – Real-time multimedia applications such as interactive gaming, live video streaming, and augmented reality have strict latency and bitrate requirements. However, unpredictable network conditions like congestion and link quality can severely degrade the Quality of Experience (QoE). While buffer-based mitigations cannot be applied to real-time applications due to their immediate resource needs, recent innovations in network slicing have demonstrated the feasibility of dedicating specified amounts of network resources to individual sessions in the radio access network. Encouraged by this, we propose to reserve network resources for multimedia sessions *in real time* according to their declared needs, thereby providing *ad hoc session-level performance guarantees*. Through WiFi experiments and trace-driven LTE simulations, we show that such session-level resource provisioning is robust to real-time channel fluctuations and congestion externalities over the lifetime of a session. This approach, however, raises challenges: how can the network ensure that users are honest about their resource needs and optimally allocate its limited resources to users, *under uncertainty in future sessions' resource needs*? We derive a novel Multi-Unit Combinatorial Auction (MUCA) model with a unique structure that can be exploited for fast winner determination and yet incentivize truthful bidding, properties not simultaneously achieved in a generic MUCA but essential to making *real-time* session guarantees. Further, since dynamic bidding in real time is challenging for end-users who are budget-constrained,

we develop a Reinforcement Learning based utility-maximizing strategy to distribute their budget across sessions, and show that it yields high user utility.

Seamlessly Connecting to Unsubscribed Networks and Making Payments – Relying on dedicated contracts with specific network operators for Internet access significantly limits connectivity options for devices. As new usecases for internet access emerge, e.g., with the Internet of Things in smart-cities, managing such individual contracts for each deployed device with varying data needs is prohibitively cumbersome and highly expensive. In this work, we enable contract-less connectivity between end-devices and access points that have no a-priori trust relationship (without a trusted intermediary). Our core insight is that exchange of services and payments can be *trustlessly* enforced by distributed ledger technologies; the credentials that blockchains use for account management can also be used for TLS-based authentication in networks. However, this raises several challenges.

First, making cryptocurrency payments in such envisioned bandwidth-sharing marketplaces in non-trivial. Decentralized marketplace applications demand fast, cheap and easy-to-use cryptocurrency payment mechanisms to facilitate high transaction volumes. The standard solution for off-chain payments, state channels, are optimized for frequent transactions between two entities and impose prohibitive liquidity and capital requirements on payment senders for marketplace transactions. We propose PayPlace, a scalable off-chain protocol for payments between consumers and sellers. Using PayPlace, consumers establish a virtual unidirectional payment channel with an intermediary operator to pay for their transactions. Unlike state channels, however, the PayPlace operator can reference the custodial funds accrued off-chain in these channels to in-turn make tamper-proof off-chain payments to merchants, *without locking up corresponding capital in channels with merchants.* Our design ensures that new payments made to merchants are guaranteed to be safe once notarized and provably mitigates well-known drawbacks in previous constructions like the data availability attack and ensures that neither consumers nor merchants need to be *online* to ensure continued safety of their notarized funds. We show that the on-chain monetary and computational costs for PayPlace is $\boldsymbol{O(1)}$ in the number of payment transactions processed, and is near-constant in other parameters in most scenarios. PayPlace can hence scale the payment throughput for large-scale marketplaces at *no marginal cost* and is orders of magnitude cheaper than the state-of-art solution for non-pairwise off-chain payments, Zero Knowledge Rollups.

Even with the PayPlace mechanism in place to facilitate fast cryptocurrency payments between devices and routers in real-time, it is unclear how the data session can be metered without using special-purpose trusted hardware at the access point whose measurements a digital ledger or an intermediary can use as ground truth. However, specialized hardware or even software modifications at the access point to integrate with the blockchain significantly hinders solution adoption. Further, the blockchain's ability to enforce transaction rules is limited by the extent to which the underlying exchange of services is digitally trackable, which is susceptible to manipulation in this case, and blockchains suffer from stringent throughput and latency limitations and may presumably scale poorly as adoption of this system increases. Using a AAA-based remote authentication architecture that also takes advantage of PayPlace's operator-mediated payment structure and trusted execution environments that devices today come equipped with, we address these challenges to design and build DataNet, a system providing seamless and incentivized connectivity between untrusting end-devices and APs, without significant computation or network overhead.

1.4.1 Thesis Outline

The remainder of this thesis is organized as follows. We present related work with respect to the four core modules of interest (Authentication, Authorization, Accounting and Billing) in Chapter 2. Subsequently, we present our study of the three identified network contexts from most to least degree of network centralization in Chapters 3, 4, 5 and 6. Chapters 5 and 6 both address the third network scenario, with the former dealing with Authentication and Accounting (usage monitoring) challenges and the latter with Billing (payment) challenges. In Chapter 7, we summarize my contributions, and present future directions of this research.

Chapter 2

Related Work

Our goal is to enable end-devices and wireless data networks to connect trustlessly (without any prior contracts), establish session-oriented resource consumption contracts, and facilitate real-time payments in accordance with this. This end-to-end flow has four core modules involved that present research challenges of interest as shown in Figure 1.2, namely, Authentication, Authorization, Accounting and Billing. We present an overview of state-of-the-art techniques proposed in prior work and compare them with the contributions of this thesis, with reference to this framework. This is illustrated in Figure 2.1. We also discuss related work in detail in individual chapters that each deal with one of the three network scenarios identified in Chapter 1.

We recall that the first and second network models (see Figure 1.3) retain the assumption of trusted device-network interaction, thereby retaining the authentication, usage monitoring and billing systems that come with subscription models, while tackling questions of incentive-compatibility and real-time decision making for dynamic usage contract establishment. While authorization in legacy networks involves verifying the device's contract status based on its identity and payment status, authorization here is instead substituted by the prices and resource allocation that the network and device negotiate in realtime. We now briefly review state-of-the-art techniques in this space of dynamic pricing and allocation.

To limit usage during congested times, some industry [63, 68] and academic [99, 192, 195] research has advocated for time-dependent pricing (TDP) for mobile data. Under TDP, users are charged higher rates when the network is congested and lower rates during times of low network utilization. These previous studies assessed the benefits of TDP compared to static pricing [192, 195], e.g., with game-theoretic

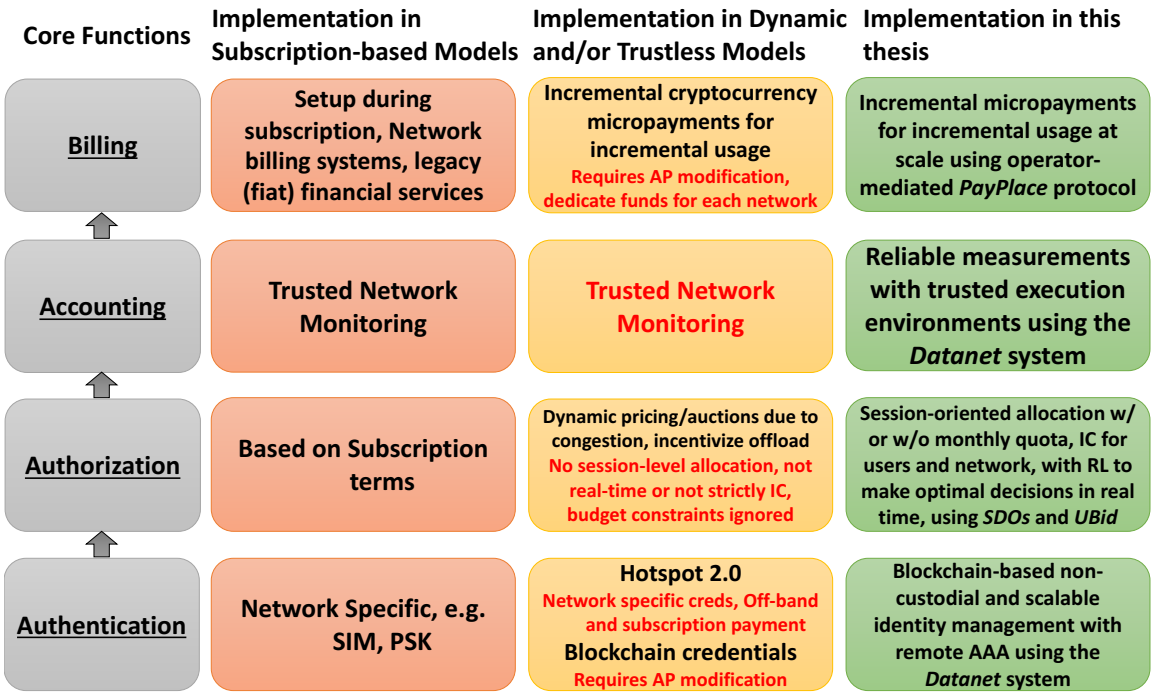


Figure 2.1: With respect to the four core stages of the device-network connection process that we are concerned with, we compare existing techniques based on the ubiquitous subscription model against the state-of-the-art proposals in dynamic resource allocation and trustless networking. Finally, we compare these against our contributions.

models [99]. Complementary work has focused on offloading users' data traffic from cellular to WiFi [134], e.g., creating auctions for ISPs to dynamically purchase WiFi capacity at times of cellular network congestion [62]. None of these works focus on **session-level allocation schemes** and guaranteening session-performance through incentive-compatible resource reservation. Auctions have also been employed for QoS-aware real-time channel allocation to primary users in mobile networks, but here as well, such approaches [69, 187] focus on sub-carrier allocation with millisecond granularity and interference mitigation while a session's timescales is on the order of minutes[1][2] [42, 159]. Indeed, resource reservations at session timescales introduce combinatorial characteristics in the allocation and pricing problem that are often intractable and cannot be solved in realtime. Approximation algorithms such as the one in [177] to solve these quickly typically involve tradeoffs in important incentive-

[1] https://vertoanalytics.com/chart-week-winning-mobile-video-app-war/

[2] https://www.statista.com/statistics/579411/top-us-social-networking-apps-ranked-by-session-length/

compatiblilty properties which are required to ensure that the network's limited resources are well-allocated and that rational users are incentivized to participate in the system. Finally, few works have heeded end-users' monetary constraints when considering dynamic spending decisions [86, 97], and none have considered the use of learning techniques in the context of spending decisions for network resource reservations. Our work on *supplemental discount offers (SDOs)* and *user-driven real-time network slicing* (that we call *UBid*) addresses these questions.

Finally, we address the Authentication, Accounting and Biling modules (Figure 2.1) by facilitating seamless connection of devices and networks with no a-priori identity or trust relationships (i.e. without subscriptions). We consider the state-of-the-art techniques in facilitating such trustless networking for each of these functions. Non-custodial identity management in wireless network authentication has received little attention. The Hotspot 2.0[3][4] specification provides for seamless long-tail WiFi discovery and aims to enable cellular-style roaming for WiFi networks; it is based on the IEEE 802.11u standard. However, the end-device can seamlessly connect to the hotspot only if has valid credentials specific to a domain/vendor who has partnered with the hotspot. Otherwise, the device is taken to an Online Sign-Up process for the user to register with a supported provider and agree to a utilization contract wherein the device trusts the usage measurements made by the hotspot. We note that 5G introduces support for PKI-based EAP-TLS authentication but requires the end-device to present SIM-based credentials (the Subscription Permanent Identifier) and a certificate signed by the network, presumably installed by the network as a result of the subscription process [194].

While these authentication techniques hence do not enable non-custodial identity management and integrally assume that the usage measurements made by the network are trustworthy, other recent works [89, 152, 171] have relied on public and permisionless blockchain systems to circumvent these issues. A user's blockchain keys can be used to identify the user's escrow account on the blockchain to the network and subsequently use this escrow account to process cryptocurrency *micropayments* [45, 144] made by the user. While the use of these credentials for authentication indeed makes the process network-agnostic, proposed techniques involve modifying

[3] https://www.cwnp.com/hotspot-2-0-and-the-next-generation-hotspot

[4] https://gigaom.com/2014/03/07/hotspot-2-0-inches-its-way-into-public-wi-fi-networks/

the access point to integrate with the blockchain to perform the necessary verification. Further, this payment process based on micropayments overcomes the challenge of getting trustworthy utilization readings by simply requiring that the device make frequent and incremental micropayments to the network for incremental network resources consumed. This circumvents the need for trusted utilization measurement, by instead simply capping the network's loss to atmost one incremental unit of service if the device fails to provide the next micropayment (in which case the network may disconnect the device) and the device's loss to atmost one incremental unit of payment if the network fails to provide the corresponding service (if the device is to make the payment before consumption). There are two significant challenges that this model poses when implemented in practice for a large number of access points. First, it requires access points to be modified to process these frequent micropayments from connected devices, thereby posing deployment challenges. It also requires end-devices to frequently send cryptographically signed messages to the network they are connected to; the battery impact of this, especially on lightweight devices, has not yet been studied. Second, proposed techniques require the establishment of pairwise payment channels with sufficient deposits between end-devices and each access point (AP) that they connect to. The *PayPlace* cryptocurrency payment protocol and the *Datanet* system that utilizes this protocol address these challenges.

Chapter 3

Supplemental Discount Offers

Inspired by Verizon's recent PopData offering, we consider supplementing monthly subscription plans with ad-hoc discount offers, wherein users may consume unlimited data in the next hour for a small fixed fee. This allows users to realize any additional real-time resource needs for their sessions that is not accounted for in the monthly forecast by utilizing these simple offers without the risk of incurring a data overage. This also affords the network a predictable contract revenue. We study the ISP and user's revenue and utility maximization problems respectively, illustrate the suboptimalities in user choices introduced by frequent realtime decision making and propose techniques to alleviate it.

3.1 Problem Definition

As mobile data usage continues to grow, with a 66% increase in 2016 [51] alone, Internet service providers (ISPs), mobile service providers in particular, are exploring ways to handle this rising demand. In the U.S., many ISPs have advocated changes to pricing plans; even "unlimited" data plans force users to submit to lower throughputs upon exceeding specified monthly data quotas [33, 164]. Internationally, most ISPs still offer quota-based plans with additional fees for exceeding the quota, e.g., Orange's EE in the U.K. [131]. Such pricing plans incentivize users to limit their overall mobile data demands so that they stay within ISPs' available capacity. However, they do not address the fact that congestion on ISP networks is concentrated at specific times of the day [95]. By reducing overall usage, they can thus have the unintended effect of increasing the amount of idle capacity, and its associated unrealized ISP revenue, at

uncongested times.

Much recent research has proposed ways to reduce usage at congested times, e.g., by charging users more at these times [157] or incentivizing them to use WiFi instead [134]. However, few of these explicitly consider the dual problem of monetizing idle capacity [88], and many of them have proven complex for users to understand [128, 156]. *Supplemental discount offers* (SDOs) offer a solution to both problems. SDOs have recently been deployed by Verizon as PopData, a supplement to Verizon users' primary data plans [178]. Under PopData, a user pays an additional fee for unlimited data usage for a limited period of time, e.g., $3 for one hour of unlimited usage. Over the month, the ISP occasionally makes these SDOs to subscribed users; by making offers in less congested times, it can offer predictable service quality/QoS to users and since an SDO spans a duration of thirty minutes to an hour, it allows a user to realize an entire session's resource needs with it. These SDOs may be particularly attractive for users who prefer to use the cellular network instead of public WiFi due to security concerns. Users can easily understand and react to such SDOs; they simply decide whether to accept offers when they are made. In-fact, future variants of time-dependent pricing (TDP) schemes are likely to follow a similar format to address concerns that TDP is too complex for users to understand [156].

Further inspection reveals, however, that fully understanding or even *optimizing* a user's acceptance of SDOs is quite complex. Such optimization requires a user to *plan their acceptance decisions over the month.* For instance, if a user knows she will not reach her data plan quota, it is better to ignore SDOs. In practice, however, users would not know their exact usage needs for the rest of the month, nor would they know when SDOs will be offered in the future. They thus need to optimize over both sources of uncertainty.

The uncertainty in user decision making leads to an equally challenging decision problem for the ISP. Namely, the ISP wants to offer SDOs at times and prices that maximize revenue, subject to network availability and the fact that user SDO acceptance is based on uncertain future data needs and future SDOs. Yet it is unclear what this optimal schedule would be. For instance, offering SDOs late in the billing cycle may or may not maximize ISP revenue: at that time, only users who know they will exceed their data plan quotas would accept the SDO to avoid overage fees. On the other hand, these users could be more likely to accept SDOs at the end of the month, when they know they will otherwise incur overage fees, than at the beginning

of the month, which may increase ISP revenue.

In this work, we model user and ISP actions in accepting and making SDOs as a game in the presence of uncertainty, allowing us to assess SDO benefits for users and ISPs. By handling the uncertainty challenges discussed above, we address five fundamental questions:

- Which types of users would be most affected by SDOs?
- How should the ISP price its SDOs?
- When should ISPs offer and users accept SDOs?
- Are SDOs viable in practice?
- Can users improve their net data consumption utility with ad-hoc SDOs?

To address the question of **which types of users would be most affected by SDOs**, we first consider a model that abstracts away the hour-to-hour SDOs by considering user utility and ISP revenue on a monthly basis. Under this model, we derive closed-form expressions for users' optimal decisions. In this study, we reach the two important conclusions that (1) *subscribers always accept a nonzero number of SDOs* and (2) *users who consume more data per accepted SDO also use more of their data plan*, so heavier users are more affected.

To address the question of **how the ISP should price SDOs**, we extend our model to include the ISP's ability to optimize the SDO price at the beginning of the month. We find that *when all users have limited data demands, the ISP should charge a high price.* However, in a more diverse mix of users, ISPs may reduce fees to incentivize users to accept offers.

To understand **when ISPs should offer SDOs and when users should accept them**, we model user and ISP hourly decisions with an iterative Stackelberg game. We then *derive conditions under which users would accept SDOs.* The ISP's decision problem in this model is NP-hard, so we *provide a near-optimal heuristic based on dynamic programming.* These user and ISP decision algorithms employ online learning to optimize over uncertainty in users' future data needs.

To assess **SDOs' practical viability**, we conduct extensive *trace-driven simulations* with real usage data to measure the effectiveness of our decision algorithms. We find that *ISPs can exploit user uncertainty in future SDO offers*, and can compute an optimal SDO schedule such that users, in their limited ability to be optimal without knowing the schedule, spend higher with SDOs to realize the same data needs than

without SDOs using only overages.

Finally, we show that users can **mitigate this suboptimality** that arises from their lack of knowledge of future SDO offers by using reinforcement learning techniques. Indeed, using Double Deep Q Networks and Deep N-Step Advantage Actor Critic Models, **users render ineffective the ISP's gamed SDO schedule** that is designed to exploit their overage averseness and had caused them to spend more with SDOs for the same amount of data consumption. Users learn to selectively accept those SDOs over the billing cycle that results in a net higher utility for no or marginal increase in spending.

3.2 Related Work

To limit usage during congested times, some industry [63, 68] and academic [99, 192, 195] research has advocated for time-dependent pricing (TDP) for mobile data. Under TDP, users are charged higher rates when the network is congested and lower rates during times of low network utilization. These previous studies assessed the benefits of TDP compared to static pricing [192, 195], e.g., with game-theoretic models [99]. TDP has been shown to be effective in user trials for cellular networks [88, 156] and smart grids [133]. Users under TDP not only reduced their usage at congested, high-price times, but also increased their usage at uncongested, low-price times. We focus on this latter effect in our work. Variations on TDP include incorporating location into pricing models and using lotteries to offer time-dependent rewards for reducing usage at congested times [113]. Many works show that ISPs can reduce congestion and increase revenue by offering different prices at different times of the day, but the apparent complexity for users has so far prevented deployment.

Complementary work has focused on offloading users' data traffic from cellular to WiFi [134], e.g., creating auctions for ISPs to dynamically purchase WiFi capacity at times of cellular network congestion [62]. Yet while these measures can decrease congestion for ISPs, they may also decrease ISP profits, not only due to the cost of purchasing WiFi capacity, but also due to the reduction in usage on cellular networks. To model this loss in revenue in our discount offers scenario, we include the presence of WiFi in users' hourly decisions in Section 3.4.1. Other work has used large-scale usage datasets to model how users consume their data quotas over a month [29]. We leverage similar frameworks in developing user and ISP decision algorithms in Section 3.4.

Table 3.1: We summarize the notation used in the paper.

Symbol	Definition
(η, d, p)	ISP Data Plan
η	Fixed monthly charge
d	Data limit
p	Overage charge per GB beyond data limit
ρ	SDO price
n	Number of times ISP offers SDO over a month
β	Fraction of SDOs accepted
x	Monthly data usage by user
x_{max}	Maximum data consumed by a user during an SDO period
α	User price sensitivity
γ	Desired monthly maximum data consumption
$x_c(t)$	User's accrued consumption under their data plan until time t.

3.3 Monthly SDO Decision Model

To assess the benefits of SDOs, we model the ISP and users respectively as the leader and followers in a game. The ISP offers and prices SDOs, and users decide whether to accept them. We assume a monopolistic ISP that offers a quota-based data plan to users, imposing a usage-based overage fee p per unit of data used over the monthly data quota d, with flat fee η. In addition, the ISP periodically makes SDOs; a user who accepts an SDO pays a fixed price ρ for unlimited data usage in the next time slot (e.g., one hour). Although a user's data use during this time slot is contractually unbounded, usage is still subject to network constraints and would in practice be finite. We assume that a user consumes a maximum of x_{max} data during an SDO session. We further assume there are N users in the system. Table 3.1 summarizes our notation.

In this section, we derive a monthly model of user and ISP behavior using a Stackelberg game. While this model is an approximation that abstracts away hourly dynamics, it provides qualitative insights into user benefits and SDO pricing. Under this model, the ISP sets the number of SDOs n offered during the month and chooses the optimal SDO price ρ in anticipation of user decisions. In the model developed in Section-3.4, the ISP implicitly chooses n, or how many SDOs to offer over the billing

cycle, by making hourly decisions on whether to offer SDOs. Given n and ρ at the start of the month, each user further makes two decisions: the fraction β of accepted SDOs and their monthly data plan usage x.

3.3.1 Modeling User Utility

We model users' utilities as having two components: utility from data plan usage and utility from SDOs. We use the standard α-fair models for user utility from monthly data usage [100, 197] to obtain the utility function

$$u(x, \beta) = C_1 \frac{x^{1-\alpha}}{1-\alpha} + \beta n \left[C_2 \frac{x_{max}^{1-\alpha}}{1-\alpha} - \rho \right] - \eta - p(x-d)^+, \tag{3.1}$$

where C_1 and C_2 are scaling factors capturing relative utility between data plan and SDO usage, and $\alpha \in [0, 1)$ indicates the user's price sensitivity. The first terms in this utility function represents the overall utility from a user's regular monthly data plan and from SDOs, respectively. Since β represents the fraction of SDOs that the user accepts and n the number of offers that are made, we can interpret the utility from SDOs as the user receiving a utility of $C_2 x_{max}^{1-\alpha}/(1-\alpha) - \rho$ each time an offer is accepted. C_2 scales the utility from x_{max} usage depending on the received quality of service (QoS). If the average QoS during SDO periods is high, then the user will receive a higher utility from consuming data at that time. Similarly, C_1 can be scaled to represent the average QoS at non-discount times. By using different scaling factors for SDO and non-SDO times, we can model ISPs' choice of making SDOs only at uncongested hours of the day. The last term in (3.1) represents the cost of data plan usage with $(x-d)^+ \equiv \max\{x-d, 0\}$ denoting the amount of users' overage data.

We further suppose that the user's overall data usage is constrained by a monthly maximum γ, imposing the constraint

$$x + \beta n x_{max} \leq \gamma. \tag{3.2}$$

In abstracting away from hourly dynamics, we assume that users know some monthly statistics about their usage (*e.g.,* x_{max} and γ). For instance, we could take $\gamma = x_{max} T$, where T is the total number of time periods in a month. This maximum usage indicates the inherent limit on the amount of data that a user would consume even if not charged for this data usage. Since users in reality would limit their data consumption so as to

avoid paying more for data, we assume that $\gamma \geq \max\left\{d, (C_1/p)^{1/\alpha}\right\}$, i.e., maximum usage γ without data costs is no less than the user's optimal data plan usage.

3.3.2 Optimizing User Utility

In maximizing the utility (3.1) subject to the constraint (3.2), the user jointly optimizes the data x consumed under the regular data plan and the fraction β of accepted SDOs for the month.

Optimizing Monthly Data Usage x. We initially consider β as given and identify the optimal values of x under different conditions, yielding the following.

Lemma 1. *The user's optimal data plan usage x^* is given by*

$$x^* = \begin{cases} d, & \text{if } (\frac{C_1}{p})^{1/\alpha} \leq d \\ (\frac{C_1}{p})^{1/\alpha}, & \text{if } d \leq (\frac{C_1}{p})^{1/\alpha} \leq \gamma - \beta n x_{max} \\ \gamma - \beta n x_{max}, & \text{if } (\frac{C_1}{p})^{1/\alpha} \geq \gamma - \beta n x_{max}. \end{cases}$$

Thus, if no SDOs are made ($n = 0$), the user would consume $x^* = \max\left\{d, (C_1/p)^{1/\alpha}\right\}$ amount of data.

Proof. In the event that $x \leq d$, the utility $u(x, \beta)$ can be written as

$$u(x, \beta) = C_1 \frac{x^{1-\alpha}}{1-\alpha} + \beta n \left[C_2 \frac{x_{max}^{1-\alpha}}{1-\alpha} - \rho \right] - \eta, \tag{3.3}$$

for $x \leq d$, where u is strictly increasing in x. Hence, the user utility is maximized at $x^* = d$, at which point the user always consumes the data quota, as it is already paid for. Considering the case where $x \geq d$, the user utility expression is

$$u(x, \beta) = C_1 \frac{x^{1-\alpha}}{1-\alpha} + \beta n \left[C_2 \frac{x_{max}^{1-\alpha}}{1-\alpha} - \rho \right] - \eta - p(x - d), \tag{3.4}$$

for $x \geq d$. In this region, $u(x, \beta)$ is convex and hence a maxima exists. However, this maxima is the optimal x for (3.4) only if it lies beyond d. Else, the function is strictly decreasing is this region and the optimal x is simply d. Assuming the maxima

is beyond d, we find the utility-maximizing x by equating the derivative of the utility function to 0, yielding

$$\begin{aligned} \frac{\partial u}{\partial x} &= 0 \\ C_1 x^{-\alpha} - p &= 0 \\ x^* &= \left(\frac{C_1}{p}\right)^{\frac{1}{\alpha}} \end{aligned}$$

The optimal value obtained, $x^* = (C_1/p)^{1/\alpha}$, is subject to two constraints: it is lower bounded by d and upper bounded by $\gamma - \beta n x_{max} \geq d$ due to (3.2). By considering these bounds, we obtain the desired result. □

From Lemma 1, we observe that if $(C_1/p)^{1/\alpha} \leq \gamma - \beta n x_{max}$, then the user's data plan usage would not change with SDOs. Thus, *heavy users' data plan consumption is most affected by SDOs*; light users would not change their usage behavior. These "light" users would have lower C_1 values, indicating that their marginal value from data consumption is low compared to the cost of their data plan.

Optimizing the Discount Acceptance Rate β. The above insight into lighter and heavier users is also reflected in the fraction β of accepted SDOs, as follows.

Proposition 1. *Table 3.2 gives the optimal (x^*, β^*) that maximize the utility (3.1) subject to the usage constraint (3.2).*

Proof. We separately consider two cases. If $(C_1/p)^{1/\alpha} \leq d$, then $x^* = d$, and the user utility as a function of β is:

$$u(\beta) = C_1 \frac{d^{1-\alpha}}{1-\alpha} + \beta n \left[C_2 \frac{x_{max}^{1-\alpha}}{1-\alpha} - \rho \right] - \eta, \tag{3.5}$$

for $d \geq (C_1/p)^{1/\alpha}$.

From (3.5), we see that the utility function is linear in β. If $C_2 \frac{x_{max}^{1-\alpha}}{1-\alpha} - \rho$ is negative, (3.5) decreases with β indicating that the satisfaction obtained from using PopData is less than the cost of PopData, and hence $\beta^* = 0$. A positive co-efficient for β, however, implies that the user utility increases linearly in β, and hence the difference between

γ and x (which is, by definition, d in this region) in this region is accommodated by PopData. The optimal β in this region is hence:

$$\beta^* = \frac{\gamma - d}{nx_{max}} H\left[C_2 \frac{x_{max}^{1-\alpha}}{1-\alpha} - \rho\right], \tag{3.6}$$

where H denotes the *unit step function* that equals one if the argument is greater than 0, and 0 otherwise.

We now consider the second case in which $(C_1/p)^{1/\alpha} > d$, for which $u(\beta)$ is

$$\begin{aligned} u(\beta) =& C_1 \frac{((\frac{C_1}{p})^{\frac{1}{\alpha}})^{1-\alpha}}{1-\alpha} + \beta n \left[C_2 \frac{x_{max}^{1-\alpha}}{1-\alpha} - \rho\right] \\ & - \eta - p((\frac{C_1}{p})^{\frac{1}{\alpha}} - d), \end{aligned} \tag{3.7}$$

for $d \leq (C_1/p)^{1/\alpha} \leq \gamma - \beta n x_{max}$. As in the first case, we see that the optimal β is either 0 or the upper-bound from the constraint in (3.2),

$$\beta^* = \frac{\gamma - (\frac{C_1}{p})^{\frac{1}{\alpha}}}{nx_{max}} H[C_2 \frac{x_{max}^{1-\alpha}}{1-\alpha} - \rho] \tag{3.8}$$

Finally, we jointly optimize β and x over the remaining region. If $(C_1/p)^{1/\alpha} \geq \gamma - \beta n x_{max}$, $u(\beta)$ is given by

$$\begin{aligned} u(\beta) =& C_1 \frac{(\gamma - \beta n x_{max})^{1-\alpha}}{1-\alpha} + \beta n \left[C_2 \frac{x_{max}^{1-\alpha}}{1-\alpha} - \rho\right] \\ & - \eta - p(\gamma - \beta n x_{max} - d), \end{aligned} \tag{3.9}$$

for $d < \gamma - \beta n x_{max} \leq (C_1/p)^{1/\alpha}$.

Upon equating the derivative of (3.9) to 0, we have:

$$\begin{aligned}
\frac{\partial u}{\partial \beta} &= 0 \\
\frac{C_1 n x_{max}}{(\gamma - \beta n x_{max})^{\alpha}} &= n\left[C_2 \frac{x_{max}^{1-\alpha}}{1-\alpha} - \rho\right] + p n x_{max} \\
\gamma - \beta n x_{max} &= \left(\frac{C_1 x_{max}}{\left[C_2 \frac{x_{max}^{1-\alpha}}{1-\alpha} - \rho\right] + p x_{max}}\right)^{1/\alpha} \\
\beta^* &= \frac{\gamma - \left(\frac{C_1 x_{max}}{\left[C_2 \frac{x_{max}^{1-\alpha}}{1-\alpha} - \rho\right] + p x_{max}}\right)^{1/\alpha}}{n x_m a x} \qquad (3.10) \\
x^* &= \left(\frac{C_1 x_{max}}{\left[C_2 \frac{x_{max}^{1-\alpha}}{1-\alpha} - \rho\right] + p x_{max}}\right)^{1/\alpha} \qquad (3.11)
\end{aligned}$$

We now note that Region 2 is a special case of Region 3 when SDO has negative utility. To see this, substitute $[C_2 x_{max}^{1-\alpha}/(1-\alpha) - \rho] = 0$ in (3.10) and (3.11), thus setting utility from SDO to 0. Then, x^* and β^* take the values of x^* and β^* for Region 2. However, if the utility of SDO is 0, H in (3.8) would put β^* as 0, which is not the case as seen. Thus Region 2, in fact, does not apply when the Utility from SDO is non-negative, in which case, Region 3 accounts for the values of x^* and β^*. However, when the utility from SDO is negative, i.e, $[C_2 x_{max}^{1-\alpha}/(1-\alpha) - \rho] < 0$, then (3.8) correctly results in zero PopData usage and optimal x^* of $(C_1/p)^{1/\alpha}$.

We note as well that, by definition of $(C_1/p)^{1/\alpha}$ in Region 3, utility from SDO cannot be negative in Region 3. That is, if $[C_2 x_{max}^{1-\alpha}/(1-\alpha) - \rho] < 0$ in Region 3, then the optimal x^* given by (3.11) exceeds $(C_1/p)^{1/\alpha}$, in which case that x^* is infeasible as it violates usage constraint (3.2). This means that if utility from SDO is negative and $(C_1/p)^{1/\alpha} > d$ (i.e, we are not in Region 1), then the user must necessarily be in Region 2. On the other hand, if the utility from SDO is greater than 0 and $(C_1/p)^{1/\alpha} > d$ (i.e, we are not in Region 1), then the user must necessarily be in Region 3. These conditions yield the final result given in Table 3.2. □

This table defines the different boundary conditions under which distinct utility-maximizing solutions emerge. We see that users with $(C_1/p)^{1/\alpha} \leq d$ would not change their data plan usage based on SDOs, rather supplementing their data plan with SDOs

Table 3.2: Optimal x^* and β^* that maximize user utility (3.1) under different conditions on d (columns) and ρ (rows).

Conditions	$d \geq (C_1/p)^{1/\alpha}$	$d < (C_1/p)^{1/\alpha}$
$\rho \geq \frac{C_2 x_{max}^{1-\alpha}}{1-\alpha}$	$x^* = d$ $\beta^* = 0$	$x^* = (C_1/p)^{1/\alpha}$ $\beta^* = 0$
$\rho < \frac{C_2 x_{max}^{1-\alpha}}{1-\alpha}$	$x^* = d$ $\beta^* = \min\left\{\frac{\gamma - d}{n x_{max}}, 1\right\}$	$x^* = d'$ $\beta^* = \min\left\{\frac{\gamma - d'}{n x_{max}}, 1\right\}$
In the above, $d' = \left(\frac{C_1 x_{max}}{C_2 x_{max}^{1-\alpha}/(1-\alpha) - \rho + p x_{max}}\right)^{1/\alpha}$.		

as needed. However, heavier users, as identified in Lemma 1, with $(C_1/p)^{1/\alpha} > d$, would change their data plan usage. Without SDOs, these users would consume $x^* = (C_1/p)^{1/\alpha}$ including overage usage. By inspection of Table 3.2, we conclude that they always consume less than that when SDOs are made.

Corollary 1. *If $C_2 x_{max}^{1-\alpha}/(1-\alpha) > \rho$, i.e., the user gains positive utility from SDOs, then $\beta^* > 0$ and the user accepts at least some SDOs. However, data plan usage reduces with SDOs, as $x^* < \max\left\{d, (C_1/p)^{1/\alpha}\right\}$.*

We observe from this corollary that if users would have consumed overage data without SDOs, then *no matter how small their utility from the SDOs, they would replace some of their overage data consumption with SDO usage.* However, light users would still consume their data quota d (cf. Lemma 1), though they might accept SDOs on top of this usage. We next focus on how heavy users' data plan consumption with SDOs depends on their individual characteristics. In particular, we find that users' data plan usage x^* can increase with x_{max}.

Corollary 2. *If $(C_1/p)^{1/\alpha} > d$ and $\alpha < \rho(1-\alpha)/(C_2 x_{max}^{1-\alpha}) < 1$, usage x^* is minimized when $x_{max} = \rho(1/\alpha - 1)^{1/(1-\alpha)}$. When $x_{max} \geq \rho(1/\alpha - 1)^{1/(1-\alpha)}$, x^* increases with x_{max}.*

Proof. Under the stated conditions, users' data plan usage is given by

$$x^* = \left(\frac{C_1 x_{max}}{C_2 \frac{x_{max}^{1-\alpha}}{1-\alpha} - \rho + p x_{max}} \right)^{\frac{1}{\alpha}}.$$

Thus, it suffices to show that $x^{*\alpha}$ reaches its minimum value at $x_{max} = (\rho(1 - 1/\alpha)^{1/(1-\alpha)}$. We do so by taking the first derivative and setting it equal to zero, which is equivalent to

$$C_1 \left(C_2 \frac{x_{max}^{1-\alpha}}{1-\alpha} - \rho + p x_{max} \right) = C_1 x_{max} \left(p + C_2 x_{max}^{-\alpha} \right)$$
$$\frac{C_2 \alpha}{1-\alpha} x_{max}^{1-\alpha} = \rho,$$

from which the result follows directly. □

This result is somewhat surprising; we would expect larger x_{max} to lead to higher β, with less data plan usage. However, the opposite effect occurs when x_{max} is large. We can partially explain this latter result by noting that as x_{max} increases, users would approach their monthly data quota γ faster with each SDO. Thus, they would prefer to accept fewer offers, spreading their data more evenly throughout the month by consuming more of their data plan. This is particularly true for less price-sensitive users (with higher α), whose utility from an SDO session would increase slowly as x_{max} increases. They could then realize larger marginal utilities from usage on their data plans, compared to SDO usage.

We next examine the effect of the maximum usage γ in more detail. In particular, we observe that γ may be larger for users with a larger x_{max}, since both represent bounds on the user's desired data consumption.

Proposition 2. *If $\gamma = c x_{max}$ for a fixed $c > 0$ and a user has positive utility from SDOs, then both x^* and β^* increase as x_{max} increases, when $x_{max} \geq \rho(1/\alpha - 1)^{1/(1-\alpha)}$.*

Proof. Corollary 2 shows that x^* increases as x_{max} increases, for x_{max} above the given threshold, regardless of the value of γ. To show that β^* increases with γ, we consider

two cases. First, if $(C_1/p)^{1/\alpha} \leq d$, then

$$\beta^* = \frac{\gamma}{n} - \frac{d}{nx_{max}},$$

which is increasing in x_{max} by inspection. Second, if $(C_1/p)^{1/\alpha} > d$, then we find that

$$\beta^* = \frac{\gamma}{n} - \frac{1}{n}\left(\frac{C_1 x_{max}^{1-\alpha}}{C_2 \frac{x_{max}^{1-\alpha}}{1-\alpha} - \rho + p x_{max}}\right)^{\frac{1}{\alpha}}$$

thus, it suffices to show that

$$\frac{d}{dx_{max}}\left(\frac{C_1 x_{max}^{1-\alpha}}{C_2 \frac{x_{max}^{1-\alpha}}{1-\alpha} - \rho + p x_{max}}\right) < 0.$$

Taking this derivative, we find that it is proportional to

$$\begin{aligned}
&\left(C_1 \frac{x_{max}^{1-\alpha}}{1-\alpha} - \rho + p x_{max}\right)(1-\alpha) C_1 x_{max}^{-\alpha} \\
&- C_1 x_{max}^{1-\alpha}\left(C_1 x_{max}^{1-\alpha} + p\right) \\
&= -p C_1 x_{max}^{-\alpha} - \alpha p C_1 x_{max}^{1-\alpha}
\end{aligned}$$

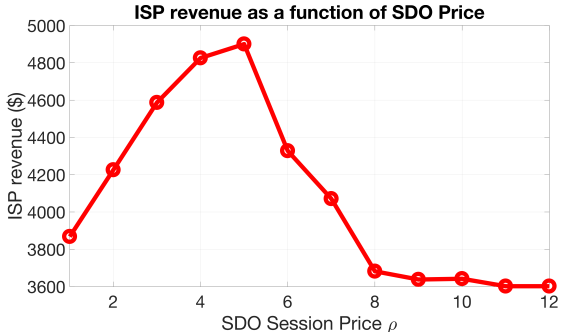
which is negative by inspection. □

In this scenario, *a larger x_{max} would lead to a larger maximum usage γ, allowing users to both accept more SDOs and consume more of their data plan.* Thus, even though users would consume more data per SDO as x_{max} increases, they would still increase both types of usage. However, users' data plan usage is still bounded by their usage without SDOs (Corollary 1); even as $x_{max} \to \infty$, $x^* \to \max\left\{d, (C_1/p)^{1/\alpha}\right\}$.

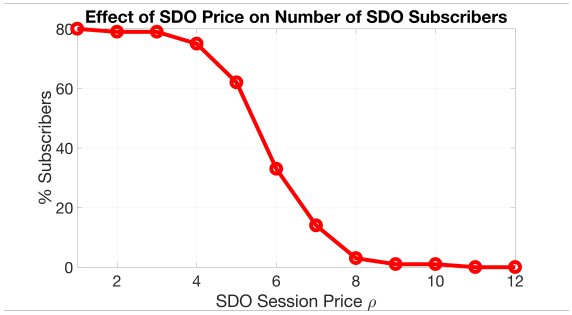
3.3.3 Maximizing ISP Revenue

Given the optimal user decisions in Proposition 1, we next find the optimum SDO price ρ to maximize ISP revenue. Since the ISP would set ρ at the beginning of the month, the monthly model guides this choice for a given number of SDO offers n. The ISP's choice of n is further considered in Section 3.4.2.

The ISP's revenue function is the sum of the revenue obtained from each user over



(a) ISP Revenue as a function of SDO Price



(b) Effect of SDO Price on Number of SDO Subscribers

Figure 3.1: ISP revenue (a) fluctuates as the SDO price ρ increases, since (b) fewer users accept SDO offers for large ρ. Users are distributed with mean $\alpha = 0.5$ and $x_{max} = 0.5$GB.

the billing cycle, so the objective is to choose ρ to maximize this revenue, formulated as

$$\begin{aligned} &\max_{\rho} \sum_{i \in U} \left(\eta + p(x_i^*(\rho) - d)^+ + \beta_i^*(\rho) n \rho\right) \\ &\text{s.t. } \rho \geq 0, \end{aligned} \tag{3.12}$$

where the subscript i is added to indicate user-specific values. We thus see that (3.12) is a complex optimization problem; the set of users whose x^* and β^* expressions fall into the different categories in Table 3.2 depend on ρ. We do not derive an analytical solution, since a line search will suffice to find the optimal ρ^*. We can, however, observe that when all users are light users, the ISP would charge them as much as possible.

Proposition 3. *When all users are homogeneous light users who do not consume overage data (i.e., $(C_1/p)^{1/\alpha} \leq d$, where C_1, C_2, α, and d are the same for all users), the optimal price ρ^* in (3.12) is $\rho = C_2 x_{max}^{1-\alpha}/(1-\alpha)$.*

From Table 3.2, we see that as long as these users have positive utility from SDOs, they would accept as many offers as necessary to realize their maximum usage γ. Thus, the ISP would have an incentive to charge as much as possible for these accepted offers. However, when there is a more diverse mix of users, the largest ρ may not be optimal. Figure 3.1(a) shows the ISP revenue as a function of ρ for a distribution

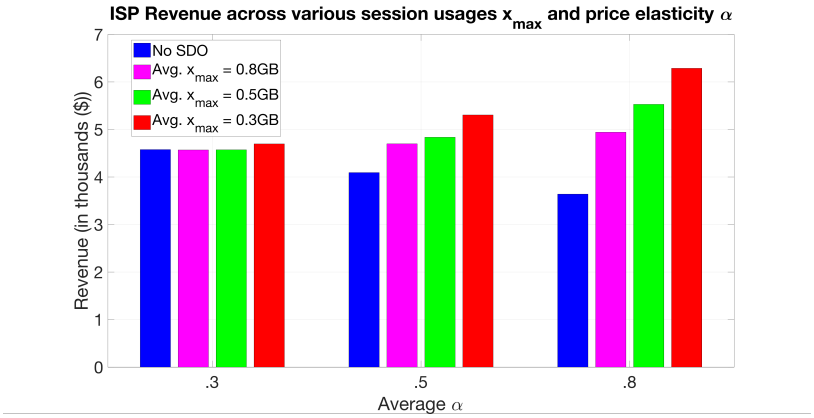


Figure 3.2: The optimal ISP revenue always increases compared to revenue without SDOs, and is higher for less price-sensitive (higher α) users who consume more data with SDOs (lower x_{max}).

of 100 light and heavy users. The optimal $\rho^* = \$5$ is lower than if all users were "light" users (Proposition 3), since the ISP can decrease ρ to encourage heavy users to accept more SDOs. Figure 3.1(b) also shows the decrease in the percentage of SDO subscribers (i.e., users who derive positive utility from an SDO) with SDO session price ρ. There is a steep drop-off in the subscription rate around $\rho = \$6$, indicating that many users no longer derive positive utility from SDOs ($C_2 x_{max}^{1-\alpha}/(1-\alpha) < \rho$).

As in our analysis of SDOs' benefits in Section 3.3.2, Figure 3.2 shows the optimal ISP revenue for user populations with different α and x_{max} values, compared to a scenario without SDOs. ISPs always increase their revenue by offering SDOs, especially for users with a higher price sensitivity; these users will accept more SDOs to avoid overage charges. As x_{max} increases, ISPs also earn more revenue, as indicated by Corollary 2, leading to more SDO revenue.

3.4 Hourly Stackelberg Game

Building on the high-level insights provided by the monthly model, we develop a game between users and ISPs to model hour-by-hour SDO decisions. In what follows, we derive a decision criterion for users to accept SDOs and propose an algorithm to optimize ISP SDO schedules.

We break the monthly billing cycle into T time steps, e.g., $T = 720$ hours in a 30-day month. At the start of each time step t, the ISP notifies users if an SDO

is offered ($y_t = 1$) or not ($y_t = 0$). An ISP's *SDO schedule* is the resulting set of decisions $\{y_t, t = 1, \ldots, T\}$. If an SDO is offered at time t, users respond by accepting or declining the SDO at the fixed price ρ. We model overage as an addition of d_O to the user's data quota at a cost p, as offered by most ISPs [179][1].

Suppose that at time t, a user has previously consumed $x_c(t)$ of their data plan quota and the total data quota currently sits at D_t, including any previously incurred overages. During time slot t, the user intends to consume x_t additional data at a desired QoS level $\phi_t \in [0, 1]$. For instance, a videoconference session may warrant a high ϕ_t, while accessing email may tolerate a low ϕ_t. x_t is the hourly counterpart of the monthly desired data usage γ from Section-3.3, and is independent of y_t. Since the ISP reveals only the current time-slot's SDO decision to the user instead of the SDO schedule for the rest of the month, the user makes a *reactive* decision at the beginning of each t rather than optimally planning their acceptance decisions and usage for the month as in Section-3.3. In other words, at the start of t, the user desires exactly x_t consumption during t, and must choose between cellular data plan, public WiFi if available, or SDO if offered ($y_t = 1$).We account for congestion and price sensitivity effects as follows. We define the congestion level θ_t of the cellular network, which is known to both the user and ISP, as well as the typical congestion θ_W for public WiFi networks. We also define $\delta_W \in [0, 1]$ as a user-specific parameter that captures the user's public WiFi preference, ranging from complete aversion ($\delta_W = 0$) to no aversion ($\delta_W = 1$), e.g., due to WiFi's greater security vulnerabilities. As ϕ_t increases and θ_t decreases, users experience more utility from their usage. A detailed list of this notation is presented in Table 3.3.

3.4.1 User Decision Criteria

At each time t, given an inherent x_t, users must decide how to realize this consumption without the ability to optimally plan their choices for the entire billing cycle (since they do not know when SDOs will be offered, this would be prohibitively difficult). They hence make decisions based on perceived utility at the current time, with awareness of the risk of incurring future overages.

The user's utility at time t from SDOs, her data plan, and public WiFi are

[1]Note that our monthly model in Section 3.3.1 uses continuous overage costs, but at the finer hourly timescale, our overage amounts are discrete.

Table 3.3: We provide a list of additional symbols and definitions for the dynamic interaction model.

Symbol	Definition
t	indexes the time intervals that the billing cycle has been divided into
a_t	Binary variable indicating a user's SDO decision for t^{th} period
y_t	Binary variable indicating the ISP's SDO decision in the t-th period
x_t	User intended data consumption in t^{th} period
ϕ_t	QoS needs of a user's $x_t \in [0, 1]$
θ_t	Cellular network congestion measure for the t-th period $\in [0, 1]$
θ_W	Typical Public Wifi congestion measure in the region of interest $\in [0, 1]$
δ_W	User-specific Public WiFi preference metric $\in [0, 1]$

respectively given by

$$u_P(t) = (1 - \theta_t\phi_t)x_t - \rho(1 - \alpha), \tag{3.13}$$

$$u_D(t) = (1 - \theta_t\phi_t)x_t - R_t p(1 - \alpha) + N_t u_O(t), \tag{3.14}$$

$$u_W(t) = (1 - \theta_W\phi_t)x_t\delta_W, \tag{3.15}$$

respectively, with corresponding costs of access scaled by the user's price sensitivity[2]. R_t represents the risk of incurring a new overage in the remainder of the billing cycle (i.e., at time $\tau \geq t$), which depends on cumulative data plan usage up to time t, as well as x_t. Thus, usage decisions at time t affect the future risk of overage R_τ for $\tau \geq t$, as this risk evolves over the billing cycle. We account for the user's utility from the extra data quota earned when incurring another overage charge by defining an overage utility $u_O(t)$. $N_t = 1$ indicates that the user incurs a new overage at time t (and 0 otherwise), so $u_O(t)$ is only realized if $N_t = 1$. We next discuss how a user would estimate the overage factors R_t and $u_O(t)$.

Modeling Risk R_t of New Overage. We define R_t as the probability that the user will incur a new overage charge in the remainder of the billing cycle. Computing this probability, however, is difficult, as the user would not know exactly how much data they would consume in the rest of the month. We thus propose to estimate this future usage by leveraging the user's historical usage patterns. We suppose the user has a typical pattern of data usage during the billing cycle, e.g., consistent usage

[2]At a finer time scale, the concavity of a user's monthly utility as in Section 3.3.1 does not appear; thus, we assume a utility linear in x_t.

throughout the cycle or gradually ramping up usage toward the end [29]. We model these consumption trends over the billing cycle as a random process $X(t) \sim \mathcal{F}_{\sigma(t)}(at^b)$ representing the user's cumulative (non-SDO) data consumption until time t. $\mathcal{F}_{\sigma(t)}$ represents a distribution around the mean cumulative usage at^b, parameterized by σ_t, e.g., a normal distribution with variance σ_t. We can learn the parameters a, b, and σ_t for each user from previous usage patterns[3].

The probability R_t of incurring a new overage in the current cycle can then be written as

$$R_t = \mathbb{P}\left(X(T) > D_t | X(T) \geq x_c(t) + x_t\right), \tag{3.16}$$

where $X(T)$ is the total usage in the billing cycle.

Modeling Utility $u_O(t)$ from Overage. We define $u_O(t)$ as analogous to users' data plan utility in (3.14):

$$\begin{aligned} u_O(t) = H \min(&D_t + d_O, \\ &\mathbb{E}[X(T) | X(T) \geq x_c(t) + x_t]), \end{aligned} \tag{3.17}$$

where $\mathbb{E}[\cdot]$ denotes expectation. The argument of the min function in (3.17) represents users' expected utility from the d_O data added to their quotas with an overage subject to their typical monthly consumption. The factor $H \in [0, 1]$ qualitatively captures any decrease in the actual utility realized in the future from the leftover data, e.g., due to future values of $(1 - \phi_t\theta_t)$. Predicting these exact values is likely impossible, as the user does not know their future data needs, but including H abstracts from the exact details.

Optimizing User Utility. At the start of time step t, the user chooses to consume data on an SDO, data plan, or WiFi to maximize utility. Given the utilities from each choice (3.13), (3.14), and (3.15), we derive the user's optimal decision criterion.

Proposition 4. *The user's optimal choice c^* of data access during t when overage is*

[3]The distribution $\mathcal{F}_{\sigma(t)}$ can be induced by an underlying random process on the parameters of users' utility functions, which will drive their demands x_t. However, these utility parameters are not directly observable by the user or ISP, so we model the directly observable usage itself as a random variable.

not incurred ($N_t = 0$) is given by

$$c^* = \begin{cases} SDO, & \text{if } \rho < R_t p \text{ and } v > \rho\alpha' \\ Public\ WiFi, & \text{if } v < \rho\alpha' \text{ and } v < R_t p\alpha' \\ Data\ plan, & \text{otherwise,} \end{cases}$$

while the optimal choice c^* during t when overage is incurred ($N_t = 1$) is given by

$$c^* = \begin{cases} SDO, & \text{if } u_O(t) < (p-\rho)\alpha' \text{ and } v > \rho\alpha' \\ Public\ WiFi, & \text{if } v < \rho\alpha' \text{ and } v < p\alpha' - u_O(t) \\ Data\ plan, & \text{otherwise.} \end{cases}$$

where $v = x_t(1 - \phi_t(\theta_t - \theta_W\delta_W) - \delta_W)$ and $\alpha' = 1 - \alpha$.

From Proposition 4, we see that when the user is not expected to go into overage at time t ($N_t = 0$), SDO is the dominant choice over data plan if it costs less than the expected overage price $R_t p$. Between WiFi and SDO, we see that SDO is the dominant choice only when the congestion in the cellular network is lower than WiFi's, subject to how important QoS is to the user (ϕ_t) and the user's affinity (or lack thereof) for WiFi δ_W. The overage case in Proposition 4 results in $R_t = 1$, and SDO is better than the data plan only if the estimated future utility from overage $u_O(t)$ is less than additional cost incurred by an overage over SDO, subject to the user's price sensitivity.

3.4.2 ISP Revenue Formulation

We next consider the ISP's decision of when to offer SDOs, given that users will respond according to Proposition 4. The ISP's revenue $r_{i,t}$ from user i in time period t is given by

$$r_{i,t} = a_{i,t} y_t \rho + (1 - a_{i,t} y_t)\omega_{i,t} N_{i,t} p \tag{3.18}$$

where $a_{i,t}$ is the user's binary decision to accept an SDO, depending on whether an SDO is offered at time t, and $\omega_{i,t} = 1$ if the user does not offload to WiFi. These can be found from each user's decision c^* in Proposition 4. Hence if $(1 - a_{i,t}y_t)\omega_{i,t} = 1$, the user does not accept an SDO but continues to use her data plan. If a new overage is incurred by i at t, then $N_{i,t} = 1$, and the ISP earns the overage price p.

While choosing the optimal y_t for (3.18) would maximize the ISP's revenue in time slot t, this could be sub-optimal in regard to the monthly billing cycle. The ISP must then account for the fact that its decision to offer an SDO at time t will affect users' risk of incurring an overage and hence the future acceptance of SDOs and future revenue. Hence, even though the ISP does not reveal the future SDO schedule to users, the current SDO decision is a function of the optimal schedule over the entire cycle. Therefore, this must be calculated at $t = 0$ for maximizing revenue over the entire billing cycle. The ISP thus aims to maximize the total revenue by optimizing the SDO schedule $\mathbf{y} = \{y_1, \ldots, y_T\}$ as

$$\mathbf{y}^* = \underset{y_1,\ldots,y_T}{\operatorname{argmax}} \mathbb{E}\left(\sum_{i\in U}\sum_{t=1}^{T} r_{i,t}\right) \tag{3.19}$$

In the revenue optimization in (3.19), note that the revenue terms $r_{i,t}$ are necessarily dependent on each other over time, seen by the inclusion of overage and conditional decision terms in (3.18). Most importantly, the expectation appears in (3.19) to capture the effects of the uncertainty in user decisions. In practice, the ISP could execute y_t^* at each time t and then re-compute its optimal schedule for the rest of a billing cycle given updated estimates of user parameters.

3.4.3 Optimizing ISP Revenue

To solve (3.19), the ISP must compute the distributions of $N_{i,t}$ and $a_{i,t}$ for each user so as to derive the expectation of $r_{i,t}$ in (3.18), noting that both depend on previous values of y_t. To do so, the ISP must estimate the parameters θ_t, α_i, and $\phi_{i,t}$ that influence users' SDO acceptance decisions in Proposition 4. While the ISP would know the cellular and WiFi congestion levels θ_t, it would need to use historical data from the user to estimate the user-specific $\phi_{i,t}$ and α_i parameters. The ISP must then estimate the distribution of users' future usage $x_{i,t}$. We suppose that it does so using the same method as the user in Section 3.4.1. Given this knowledge of user behavior, we can then recast (3.19) as a dynamic program and derive a heuristic algorithm to compute an approximate solution.

Dynamic Programming Formulation. The solution to (3.19) can be found by formulating the following Bellman equation for computing the optimal revenue V_t^* at t. It can be expressed as a function of the current time step decision y_t that

maximizes the current time-step revenue r_t as well as the revenue from the next time-step $V_t^*(\mathbf{D_{t+1}}, t+1)$, which in turn is a function of y_{t+1} and so on. That is,

$$V_t^*(\mathbf{D_t}, t) = \max_{y_t} \left(r_t + V_t^*(\mathbf{D_{t+1}}, t+1) \right), \tag{3.20}$$

where the boldface $\mathbf{D_t}$ is a vector of all users' data quotas and all of the terms depend on current and past values of y_t. The corresponding y_t value becomes the t^{th} entry in $\mathbf{y}^*$. User quotas $\mathbf{D_{t+1}}$ at time $t+1$ are a function of y_t given by:

$$D_{i,t+1} = D_{i,t} + (1 - a_{i,t} y_t) \omega_{i,t} N_{i,t} d_O \tag{3.21}$$

wherein the quota only increases if the user consumes under the regular data plan and further incurs an overage due to this. Thus, this data quota state update mechanism at every time step captures the tradeoff between overage and SDO revenue, dependent on both y_t and $\mathbf{D_t}$. The results of solving (3.20) are presented in Section 3.5.

Fast Pruning Algorithm. Finding an optimal dynamic programming solution is known to be difficult. Since the ISP's decision variables y_t are binary, our problem is NP-hard. We thus develop an approximation algorithm for (3.20) that efficiently prunes the search space of possible SDO schedules. Our near-optimal numerical results are given in Section 3.5.

Algorithm 1 presents the details of the algorithm. To facilitate our discussion, we define an *outcome state* $O_{t,\vec{y}}$ at time t as the vector of estimated accrued consumption for each user and accrued revenue for the ISP, given the y_τ decisions chosen at previous times $\tau \leq t$. At each time t, we consider both possible ISP decision choices: $y_t = 0$ (do not offer SDO) and $y_t = 1$ (offer SDO). For each choice, we prune among the possible SDO schedules by retaining only one outcome state of the option under consideration.

At $t = 1$, we start with one initial state of no usage or revenue. We then consider decisions $y_1 = 1$ and $y_1 = 0$ with resulting outcome states $O_{1,1}$ and $O_{1,0}$. At the next time step, $t = 2$, we again consider $y_2 \in \{0, 1\}$ and end up with two outcome states for each. For example, we could move to $y_2 = 1$ from either $O_{1,1}$ (ending up in $O_{1,(1,1)}$) or from $O_{1,0}$ (ending up in $O_{1,(0,1)}$). For each choice of y_2, we pick the outcome state that has higher aggregate revenue (hence implicitly choosing the associated parent state from $t = 1$). We continue until time T, when we chose the final outcome state $O_{1,(y_1,y_2\ldots,1)}$ or $O_{1,(y_1,y_2\ldots,0)}$ with higher accrued revenue. By *not* pruning between the two y_t options in each time step, but instead pruning between their possible outcome

Algorithm 1: ComputeSDOSchedule: Fast Pruning Algorithm for SDO Schedules.

Input : $\mathbf{a}, \mathbf{b}, \sigma$
Output : T-sized array of binary values

▷ Columns of all matrices are 0-indexed
$usageState[0, :] \leftarrow [0]$
$revenue[0, :] \leftarrow 0$
for $t \leftarrow 1, \ldots, T$ **do**
  for $y \leftarrow 0, 1$ **do**
    for $prevY \leftarrow 0, 1$ **do**
      $currUsage \leftarrow usageState[t-1, prevY]$
      $currRev \leftarrow revenue[t-1, prevY]$
      $incRev \leftarrow 0$
      for each $u \in \{Users\}$ **do**
        $[newUserUsage, newRev] \leftarrow$
         $estIncUsageThisHour(currUsage[u], y, a[u], b[u], \sigma[u, t])$
        $incRev \leftarrow incRev + newRev$
        $newUsage[u] \leftarrow newUserUsage$
      $revVsYDecision[prevY] \leftarrow incRev + currRev$
      $usageVsYDecisions[prevY] \leftarrow newUsage$
    if $revVsYDecision[0] > revVsYDecision[1]$ **then**
      $ySchedule[t, y] \leftarrow 0$
      $usageState[t, y] \leftarrow usageVsYDecisions[0]$
      $currRev[y] \leftarrow revVsYDecision[0]$
    else
      $ySchedule[t, y] \leftarrow 1$
      $usageState[t, y] \leftarrow usageVsYDecisions[1]$
      $currRev[y] \leftarrow revVsYDecision[1]$
if $currRev[T, 0] > currRev[T, 1]$ **then**
  return $ySchedule[0]$
else
  return $ySchedule[1]$

states, we account for the effect of accruing outcomes between the decision branches for $y_t = 1$ and $y_t = 0$.

While the ISP calculates optimal SDO schedule at $t = 0$, it strategically does not reveal this to the users, hence gaining the advantage (amongst others detailed in Section 3.5) to observe users' accrued consumption in the current billing cycle and measure any significant deviations from the learnt a, b and σ_T^2. This deviation from typical historic trends could be especially considerable when the ISP first introduces SDOs, as offloading to SDOs impacts the usage trend under the regular data plan. To accommodate such externalities, the ISP might use the following *online learning* procedure to recompute user characteristics and subsequently the SDO schedule for leftover timesteps.

Update Criteria. The ISP can periodically calculate the likelihood of the observed x_cs over the duration of the billing cycle and determine whether the user's consumption trend in the current month is in keeping with the learnt model. Given a vector of observed $\vec{\mathbf{x_c}}$ and corresponding time-intervals $\vec{\mathbf{t}}$, the update criteria is defined as:

$$p(\mathbf{x_c}(\vec{\mathbf{t}})|\mu(\vec{\mathbf{t}}), \mathbf{\Sigma}(\vec{\mathbf{t}})) = \prod_{i=1}^{t} \frac{1}{\sqrt{(2\pi)^t|\Sigma|}} \tag{3.22}$$

$$exp\frac{-(x_c(i) - \mu(\vec{\mathbf{t}}))'\mathbf{\Sigma^{-1}}(x_c(i) - \mu(\vec{\mathbf{t}}))}{2} \tag{3.23}$$

$$u(\mathbf{x_c}(\vec{\mathbf{t}})) = \begin{cases} 1, & \text{if } p(\mathbf{x_c}(\vec{\mathbf{t}})|a(t)^b, \sigma_t^2) \geq T_U \\ 0, & \text{otherwise} \end{cases} \tag{3.24}$$

where u is the update decision, and T_U is a pre-defined empirical threshold for the likelihood of observations $\mathbf{x_c}(\vec{\mathbf{t}})$, below which the user is determined to be significantly deviant from their expected trend.

Update Algorithm. If the update decision u is affirmative, the ISP can use *weighted Maximum Likelihood* estimation to recalculate the learned parameters, where the observations of the current cycle $\vec{\mathbf{x_c}}$ are assigned a weight inversely proportional to the likelihood of the observations, and the rest of the historic observations are weighed equally. *i.e.,*

$$W_i(S) = \begin{cases} \frac{(1-p(\mathbf{x_c}(\vec{\mathbf{t}})|a(t)^b,\sigma_t^2))}{|S|}, & \text{if } S = \mathbf{x_c}(\vec{\mathbf{t}}) \\ \frac{p(\mathbf{x_c}(\vec{\mathbf{t}})|a(t)^b,\sigma_t^2)}{|S|}, & \text{otherwise} \end{cases} \tag{3.25}$$

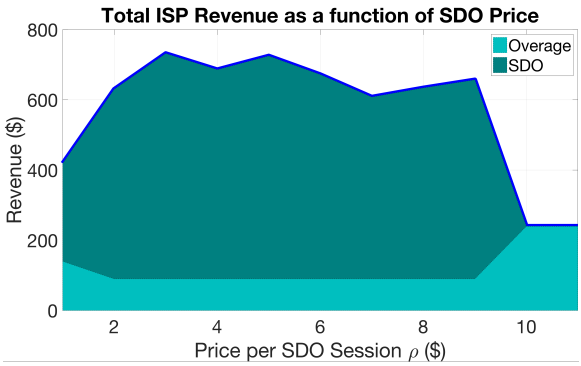


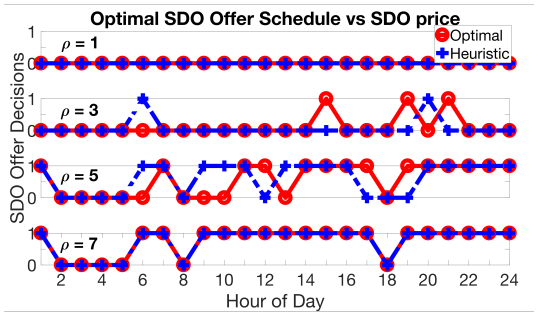
Figure 3.3: The revenue from SDOs far exceeds that from overage when the ISP plans its SDO schedule optimally.

3.5 Trace-Driven Evaluation of the Hourly Model

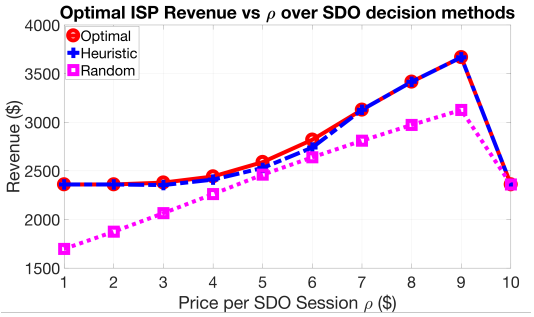
In this section, we illustrate user and ISP decisions in our hourly model. We use a cellular usage trace from 20 users to show that ISPs gain revenue from making SDOs and that the SDO schedule computed by our pruning heuristic (Algorithm 1) is close to the optimal. We then examine the effect of the SDO price ρ. We show that *ISPs can exploit user uncertainty to earn more overage revenue as* ρ *increases* and that *ISPs experience a tradeoff between maximizing their revenue and their network utilization* in making SDO offers. We also draw comparisons between our findings and Verizon's existing PopData deployment.

Simulation Setup. For illustration, we reduce the duration of the billing cycle to 24 hours, with an associated data overage threshold of 50MB (equivalent to a 1.5GB monthly quota). Our user-specific consumption patterns are taken from a one-week cellular usage trace of 20 users. The availability of public WiFi hotspots to users is drawn from a Rayleigh distribution with parameter 0.25 (where an availability below 0.5 is considered unavailable), as are users' price sensitivities α. We set $\theta_w = 0.5$ and draw θ_t and δ_W from a uniform distribution between 0 and 1. The ISP never offers SDO during hours 2-5 as typical network use is very low during these hours of the night; it also does not offer SDOs at 8AM and 6PM due to already high network congestion as done by Verizon with PopData [163]. These configurations apply to the following results unless noted otherwise.

Optimal ISP Revenue We first analyze the revenue calculated by the ISP as a function of the optimal SDO schedule for each $\rho \in [1, 9]$; for larger ρ, users do



(a) Optimal SDO Schedule vs. SDO Price



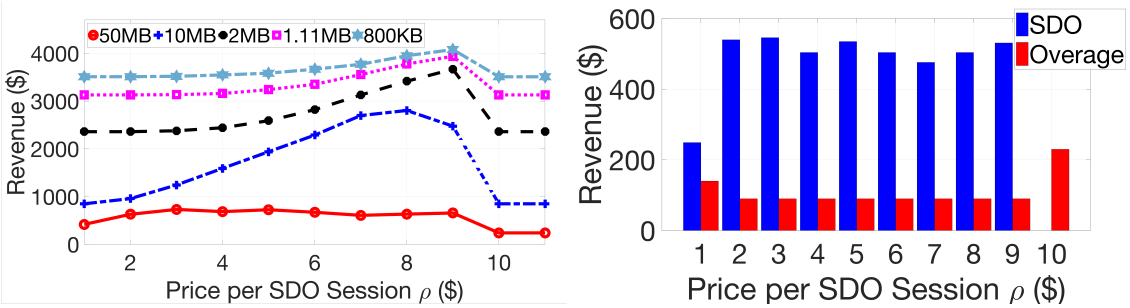
(b) ISP Revenue vs. ρ over SDO decision methods

Figure 3.4: We illustrate the dependence of ISP revenue on the SDO price ρ. Our results indicate that (a) our heuristic SDO schedule closely matches the optimal one, with an exact match for very low or high fees ρ, and (b) our heuristic yields nearly the same revenue as the optimal SDO schedule, with significant improvement over a random schedule.

not accept SDOs as they would prefer to incur overage fees. Figure 3.3 shows that the optimal revenue is non-convex in ρ, with the maximum revenue at $\rho^* = 3$. The ISP earns substantial revenue from making SDOs compared to not making them (represented by the revenue at $\rho > 9$, where users would not accept SDOs), with this additional SDO revenue exceeding lost revenue from overage. Since users do not *increase* their data usage when they avail SDOs (as opposed to consuming under their regular data plan or under WiFi), this substantial increase in revenue that the ISP experiences at certain values of ρ with optimal planning is largely attributed to their successful exploitation of user's suboptimal and myopic risk assessment (cf. Propositon 4) stemming from their overage-averseness.

SDO Schedules. Figure 3.4(a) compares the optimal ISP schedule for each value of ρ to the schedule generated by the fast pruning algorithm (Algorithm 1). Our pruning algorithm yields the optimal schedule when ρ is very low or high, and it closely trails the optimal schedule in other cases. When $\rho = 1$, the ISP does not offer any SDOs. Even though this SDO price is low enough to attract many users, the resulting SDO revenue does not compensate for the ISP's loss in overage revenue. As ρ increases, the ISP selectively makes SDOs in more hours. When ρ is sufficiently high, at \$7, the ISP makes an SDO in all hours, as its revenue from users' acceptance of an SDO exceeds any resulting loss in overage fees.

We next examine the revenues achieved by our pruning algorithm in Figure 3.4(b), with a low data overage threshold of 2MB. Our algorithm nearly achieves the revenue



(a) Effect of Overage Threshold on Optimal ISP Revenue

(b) Users' Overage vs SDO spending over SDO price

Figure 3.5: As users' data quota decreases, (a) ISP revenue is maximized at higher SDO fees ρ. As ρ increases, (b) ISP's continue to make steady income from SDOs as in Figure 3.4(a). For each ρ, the ISP exploits user uncertainty in when SDOs will be offered, choosing its SDO schedule so as to induce users to myopically accept SDOs, even though the SDO fees incurred exceed users' future overage charges.

with the optimal schedule at all prices ρ. Both significantly improve the ISP revenue compared to a random schedule, with a 20% increase at the optimal $\rho^* = 9$, emphasizing the ISP's benefit from optimizing its SDO schedule. We next examine ISP benefits in more detail by comparing their overage and SDO revenues and considering the effect of SDOs on network utilization. These results use the optimal SDO schedule.

Overage vs. SDO revenue. We first examine the effect of users' overage thresholds on ISP revenue. Figure 3.5(a) shows that users incur more overage charges, increasing ISP revenue, as the overage threshold decreases from 50MB to 800KB. Moreover, the optimal SDO price ρ^* also increases as the ISP would discourage them from accepting SDOs and lowering its overage revenue. Hence, only higher values of ρ incentivize the ISP to offer SDOs as more users go into overage. In Verizon's PopData deployment, each PopData session costs \$2, indicating that few users would incur overage charges.

To confirm this intuition, we visualize user spending on overage and SDO fees in Figure 3.5(b) for an overage threshold of 1.5MB. Surprisingly, users spend more money overall under most regions of ρ with SDO than without. Without SDOs (at $\rho = \$10$ when no users accept SDOs), users spend approximately \$200 total on overage fees. At 20 users and \$10 for an overage, this implies 20 overages overall in the billing cycle. For the same data needs, users spend significantly more when offered SDOs. We show below that this substantial increase in revenue is not due to any significant shift from

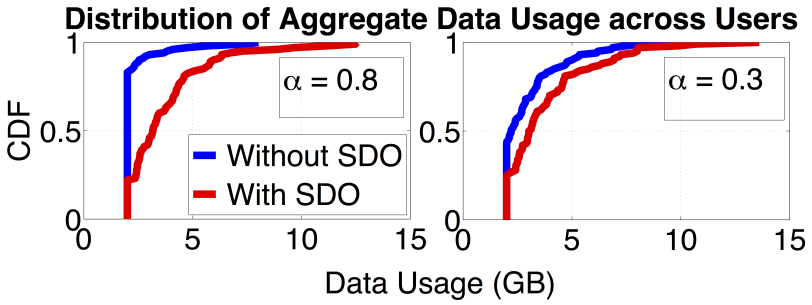


Figure 3.6: Distribution of aggregate user usage across the population. The Distributions are representatives of two populations. One, with price sensitivity 0.8 and another with price sensitivity 0.3

WiFi to SDOs. Instead, it is a direct consequence of users' inability to predict when future SDOs will be offered.

As users approach a new overage, i.e., R_t from (3.14) increases, they are more likely to accept SDOs. They do not, however, anticipate this increase in R_t in advance. As shown by users' myopic hourly utilities in (3.13–3.15), lack of information about future SDOs forces users to make bounded-rationality choices. Aversion to future overages then biases users towards accepting the SDO, allowing the ISP to plan its SDO schedule such that users' myopic decisions yield much higher revenue than the ISP could otherwise gain. While some users may avoid these charges, Figure 3.5(b) shows that most spend more under SDOs. If users, as in the monthly model, could plan their optimal usage up-front knowing the future SDO schedule, they could avoid these charges. If users, as in the monthly model, could plan their optimal usage up-front with the knowledge of the future SDO schedule, then they would properly balance SDO spending. In Figures 3.6 and 3.7, we show that in our monthly model, users consume more data with SDOs compared to without. Despite this increase in usage, however, they spend only slightly less with SDOs than without, indicating that they better balance their SDO spending with overage charges.

Network utilization vs. revenue. We finally examine the effect of WiFi availability on ISPs' revenue and network utilization. Though SDOs could incentivize users to consume cellular instead of WiFi data, thus allowing ISPs to monetize this otherwise "lost" usage, we find that there is a tradeoff between maximizing ISP revenue and the network utilization.

Figure 3.8(a) depicts the amount of data traffic onboarded onto the ISP's network from WiFi, as a function of ρ as well as the distribution of users' WiFi preference

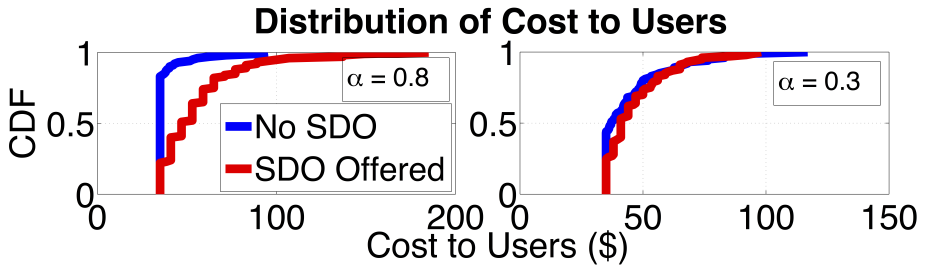
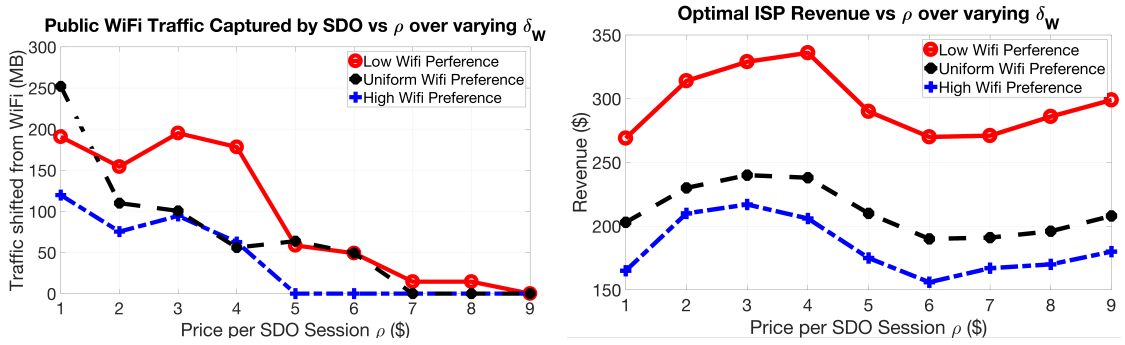


Figure 3.7: Distribution of total user cost across the population. The Distributions are representatives of two populations. One, with price sensitivity 0.8 and another with price sensitivity 0.3



(a) (b)

Figure 3.8: (a) The amount of public WiFi data captured by the ISP's network due to SDOs is non-monotonic in ρ, reflecting the ISP's strategic choices in computing the optimal SDO schedule. (b) The ISP can make more revenue from users with lower WiFi preferences, since these users would be more likely to accept SDOs. Comparing the revenue with the network utilization in (a), the revenue maximizing ρ does not maximize network utilization.

factor δ_W. Though the overall network utilization decreases as ρ increases, which we would expect since a higher SDO price ρ would lead to fewer users accepting SDOs instead of using WiFi, this decrease is non-monotonic. This is a direct effect of the ISP jointly optimizing ρ and the SDO schedule such that the optimal SDO offerings at each price are made strategically in hours that balance the ISP's predicted revenue from cellular onboarding and overage fees. Moreover, comparing Figures 3.8(a) and 3.8(b) shows that while network utilization is maximized at $\rho = 1$, ISP revenue is maximized at higher prices.

Our result realizes a key consequence of the dynamics of hourly SDO games. The ISP is able to learn user intentions from historical data and strategically choose the SDO schedule and price to maximize its revenue. Users are then at a disadvantage; even though they may increase their utility by switching from WiFi to SDOs, the ISP's offered SDO schedule and price does not maximize this utility increase. Thus, the ISP is able to control the information revealed about SDOs to profit from users' consequential myopic actions.

3.6 Mitigating User Suboptimality in the Hourly Model

As the empirical study above shows, the ISP is able to exploit users' averseness to expensive overages (along with the ISP's knowledge of their historical data consumption trends) to increase its revenue without corresponding increases in user data consumption utility. Indeed, with the optimal ISP schedule and price, the ISP times SDOs such that users that rely on the overage-averse framework in Proposition 4 to decide their SDO acceptances end up spending significantly more for equivalent data consumption as earlier (i.e. without SDOs). However, our analysis of the monthly abstraction in Section 3.3 shows that users can substantially benefit from SDOs in practice with complete up-ahead information that allows for optimal planning. This indicates that simply being overage-averse by accounting for typical monthly utilization when making real-time decisions about ad-hoc SDOs is insufficient to compensate for the lack of up-ahead knowledge about future SDOs. Users' inability to forecast future SDOs and plan their data consumption optimally effectively eliminates any advantages they may otherwise gain from SDOs and, in-fact, actively hurts them.

We hence propose to mitigate this suboptimality in users' real-time SDO decisions by using reinforcment learning techniques to *learn the ISP offer schedule* and make optimal decisions in realtime by using the learnt information. Since individual users have no prior knowledge either of the ISP offer decision model or of other users' parameters (e.g α, σ) that the ISP's offer model depends on, we utilitize *model-free* reinforcement learning techniques that directly learn the optimal choices to make by interacting with the ISP rather than by explicitly attempting to construct a model of the ISP decision process and then perform optimal planning with it. Since each billing cycle spent exploring results in monetary costs to the user, we devise techniques to hasten convergence of the learning algorithm by exploiting partial model knowledge of how state transitions occur in this environment.

3.6.1 Defining the MDP

We first define the user's decision process. Each episode spans one billing cycle; let $t = 1, 2, \ldots, 24$ index the hours within the current billing cycle. A user's state $\mathcal{S}_t$ at the beginning of any hour t consists of the following information $\mathcal{S}_t = (x_t, \phi_t, \psi_t, y_t, D_t - x_c(t), t)$, where ψ_t is a binary variable indicating whether public WiFi is available during t or not. The action $\mathcal{A}_t$ that the user takes at t in state $\mathcal{S}_t$ is indicated by the tuple (a_t, w_t) where w_t is a binary variable indicating whether the user chooses Public WiFi; note that $a_t = 1$ implies $w_t = 0$ and $w_t = 1$ implies $a_t = 0$. $a_t = 0$ and $w_t = 0$ implies that the user chooses to consume data under their regular Data Plan for that hour. Note that these state transitions are Markovian; x_t, ϕ_t and ψ_t are stochastic variables parametrized by t, the remaining data quota $D_t - x_c(t)$ can be derived from previous state's value and action taken, while the ISP's offer decision y_t is independent of previous states given the last state $\mathcal{S}_{t-1}$ of all users (see Algorithm 1). In-fact, we further assume that the ISP's SDO schedule is computed using historical user data before the billing cycle begins. Hence different users' actions are independent since the ISP's SDO schedule is not affected by any individual user's choice of actions during the billing cycle/RL game; that is, the environment is stationary. We drop the user-specific subscript i from $a_{i,t}$ and correspondingly $w_{i,t}$ since we implicitly consider a generic user i here.

The reinforcement learning (RL) agent (i.e. the user) receives a reward $\mathcal{R}_t$ after executing action $\mathcal{A}_t$ from state $\mathcal{S}_t$ that reflects the data consumption benefit from the

action less the costs incurred. We now define this reward function as

$$\mathcal{R}_t = \begin{cases} (1-\theta_t\phi_t)x_t - \mathcal{C}\rho(1-\alpha), & \text{if } y_t a_t = 1 \\ (1-\theta_W\phi_t)x_t\delta_W, & \text{if } \psi_t w_t = 1 \\ (1-\theta_t\phi_t)x_t - \mathcal{C}N_t p(1-\alpha), & \text{if } a_t + w_t = 0 \\ -10000 & \text{otherwise} \end{cases} \tag{3.26}$$

where $\mathcal{C}$ is a empirically determined cost scaling factor that serves to make the data benefit term on the LHS (denominated in Bytes) comparable with the data cost term on the RHS (denominated in \$). Note that we abuse the notation of N_t here and use it to refer to the exact *number of overages* incurred at t from executing $\mathcal{A}_t$ in $\mathcal{S}_t$ rather than simply indicating whether an overage occurred or not. Finally, we set a large negative reward (-10000) when the RL agent attempts to make invalid state transitions such as choosing SDO when there is no SDO offer ($a_t = 1, y_t = 0$) or choosing WiFi when there is no WiFi available ($\psi_t = 0, w_t = 1$).

3.6.2 24-hour Billing Cycle

We first consider the exact scenario used in Section 3.5 where we perform trace-driven evaluation with 20 users with a reduced billing-cycle length of 24 hours. This allows us to compare the effectiveness of the reinforcement learning technique in making optimal SDO choices with the overage-averse user-decision model (see Proposition 4) whose results have been analyzed in Section 3.5. Subsequently, we consider a realistic 720-hour billing cycle length; the larger state space that this results in demands more sophisticated learning techniques that we then study.

Learning Framework. Since the user's MDP is finite (each episode terminates after 24 hours of the billing cycle), stationary, and all actions can be repeatedly sampled in all states, Q-learning[167] is guaranteed to find an optimal action-selection policy given sufficient exploration time. With this technique, the agent learns the *maximum expected value of the total reward possible* for the rest of the episode from being in a state $\mathcal{S}_t$ and taking an action $\mathcal{A}_t$ from the set of available actions $\mathbb{A}_t$ at t. Based on the Bellman equation of the Q value [167], this Q value, i.e. $Q(\mathcal{S}_t, \mathcal{A}_t)$, is updated after every state transition. However, the presence of real-valued variables like x_t and ϕ_t make our state space continuous and hence impossible to

exhaustively traverse and explore in finite time. We hence utilize a deep neural network for Q function approximation, which aids in generalizing past experiences to yet unexplored states. We retain an experience replay buffer of fixed size in memory [122, 123] where traversed episodes are stored, and randomly sample state transitions from the replay buffer during every training epoch. Such Deep Q Networks (DQN) though have been shown [176] to be subject to significant overestimation bias and consequently converge on poor action-value policies or not converge at all. We therefore utilize Hasselt et al.'s Double DQN technique [176], where we have a running neural network model Q and a target neural network model Q'. When a state transition $(\mathcal{S}_t, \mathcal{A}_t) \rightarrow (\mathcal{S}_{t+1}, \mathcal{A}_{t+1})$ is sampled from the replay buffer for training, the target model Q' is used for selecting the optimal action from transitioned-to state $\mathcal{S}_{t+1}$ and the running model Q is used for evaluating this action. That is, the updated Q value $Q^*(\mathcal{S}_t, \mathcal{A}_t)$ is given by $\mathcal{R}_t + \mathcal{D}Q(\mathcal{S}_{t+1}, \text{argmax}_{\mathcal{A}'} Q'(\mathcal{S}_{t+1}, \mathcal{A}'))$, where $\mathcal{D}$ represents the discount factor. We minimize the mean squared error between Q^* and Q and, at lower frequency, periodically copy the parameters of the running model Q to the target model Q'.

Increasing Training Samples Per State Transition. Every state transition that the agent makes where it explores the environment is a source of potential monetary loss to the user; on the other hand, prematurely minimizing exploration may result in the agent getting trapped in a local optimum and making consistently poor SDO acceptance choices going forward, which also results in significant monetary loss to the user. It is hence highly desirable to extract as much learn-able information as possible from each state transition to hasten the agent's convergence to the optimal solution. Here, we achieve this by *generating three training samples instead of just one from each state transition by using our partial a-priori knowledge of how state transitions occur.*

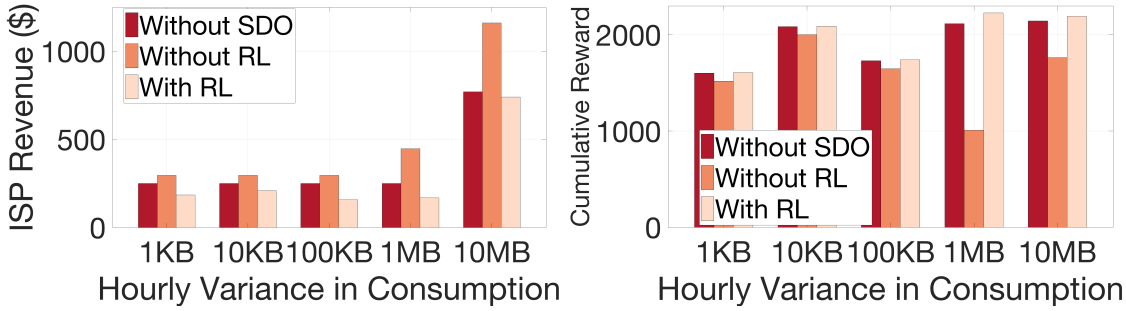
Note that at any time t, given $\mathcal{S}_t$ and the reward function as defined in Eq 3.26, the agent can fully calculate the reward it would receive from *taking any of the available actions* (i.e. choosing SDO, Public WiFi or Data). However, the agent cannot compute the subsequent state $\mathcal{S}_{t+1}$ at t since x_{t+1}, ϕ_{t+1}, ψ_{t+1} and y_{t+1} are not known until the beginning of the next hour $t+1$; note that we consider the challenging case of stochastic x_{t+1} that makes the user's hourly consumption variable. The cellular and WiFi network's QoS parameters θ_t and θ_W required to compute Eq 3.26 are known up-ahead through historical observation (as assumed for the evaluation of

the overage-averse hourly model in Section 3.5 as well) and $D_{t+1} - x_c(t+1)$ can be deterministically computed given $\mathcal{S}_t$ and $\mathcal{A}_t$.

Given a complete state transition $(\mathcal{S}_t, \mathcal{A}_t) \rightarrow \mathcal{S}_{t+1}$ however, the agent now also has knowledge of the previously unknown variables x_{t+1}, ϕ_{t+1}, ψ_{t+1} and y_{t+1}. Crucially, note that these components of the state *do not* depend on the action executed in the previous state. At $\mathcal{S}_{t+1}$ then, the agent can retrospectively calculate the hypothetical reward $\mathcal{R}_t^{hyp}$ and hypothetical state transition $\mathcal{S}_{t+1}^{hyp}$ that would have been realized for each available action $\mathcal{A}_t^{hyp} \in \mathbb{A}_t$ where $\mathcal{A}_t^{hyp} = \mathcal{A}_t$ has already been executed and realized as current state $\mathcal{S}_{t+1}$.

Exploiting this feature of our environment, we are able to extract three well-formed state transitions for each actual transition that the agent makes. The actual transition is realized by picking one of the three available actions, i.e. SDO, Public WiFi or Data Plan. The hypothetical transitions for the other two actions not executed are than computed ex-post. Three training samples are extracted per interaction that the agent has with the environment and all three are added to the relay experience buffer. A total of 72 state transitions are acquired from one 24-hour billing cycle that the agent steps through.

Evaluation. For our DQN (running as well as target models), we implement a 3-layer neural network. The input consists of the six values in a state tuple; the first two layers are 24-unit fully connected dense layers and use ReLu activations. The final layer uses a linear activation and outputs a 3-sized vector where each value correponds to the predicted Q value of that action. All layers use the Xavier normal initializer. We minimize the mean squared error and use the Adam optimizer, with a learning rate of .0001. We set the discount factor $\mathcal{D}$ to .95. We use a replay experience buffer that has a maximum size of 10000 state transitions and initiatize it with 2000 transitions made only using valid actions (for e.g. we do not allow the agent to pick WiFi when WiFi is not available). During training time, we follow an ϵ-greedy technique for exploration; we set $\epsilon = .5$ initially and decrease it by $4.5e-4$ every episode. We continue to restrict the agent only to valid actions in each state, whether exploring or exploiting, but note that the unexplored actions (whether valid or invalid) are accounted for in the additional 2-state extraction process described above. Hence the agent also learns which actions are invalid in specific states. During training time, we continue adding the agent's state transitions to the experience replay buffer; once every 8 hours (or state transitions) in the billing cycle, we pick $128 * 3$ samples from the buffer and



(a) ISP Revenue (aggregated across users) in One Billing Cycle after Training completes

(b) Aggregate Reward across agents in One Billing Cycle after Training completes

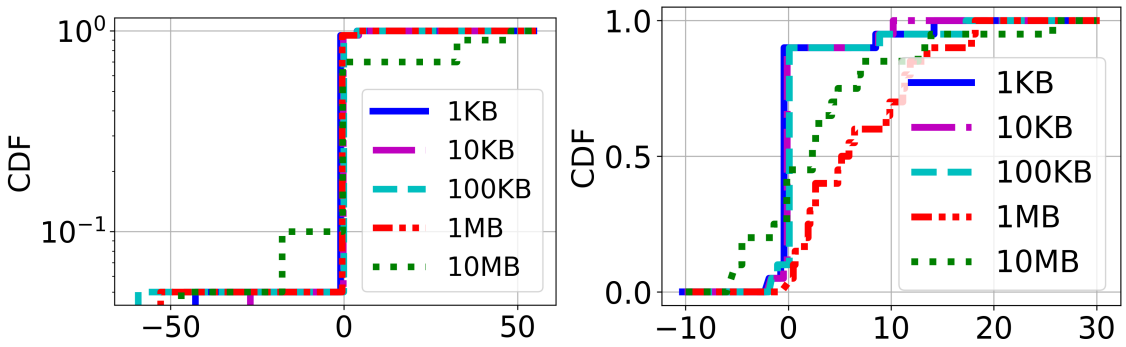
Figure 3.9: The policy learnt by the RL agents results in (a) net aggregate lower spending (or ISP revenue) and (b) equivalent or higher cumulative rewards, in comparison with the regime without SDOs as well as the overage-averse decision strategy to accept SDOs.

train the running model over 12 epochs (with shuffling). At the end of every episode, we update the target model with the weights of the running model. Since the reward function values (i.e. Q values) fall in a wide range and are unbounded, we stabilize the loss function by reducing the reward associated with each state transition with a multiplicative reward scaling factor. We set this reward scaling factor to 10^{-5} and decrease it upto 10^{-7} for some users whose rewards evaluate to orders of magnitude larger. We set the cost scaling factor $\mathcal{C}$ mentioned in Eq 3.26 to 35; note that we used the same cost scaling factor in Section 3.5 for the experiments on the overage-averse hourly user decision model. Finally, we normalize the unbounded parameters of the state tuple, namely x_t and $D_t - x_c(t)$, before training the DQN by dividing these by an estimate of the maximum values they can take in the experiment. We normalize the sixth state parameter t by dividing it by 24.

We compare users' performance in terms of their spending and reward with the RL technique against performance without RL (i.e with the hourly decision model being used to decide SDO acceptance) and performance without SDOs (i.e. with the hourly decision model being used to decide between consuming under WiFi and Data Plan). In the latter two cases, though there is no reinforcement learning being done by users, reward for user actions are still calculated using the same reward function as the one in (3.26); these rewards do not inform user actions but serve as a meaningful metric of how the user's realized utility changes over the month when compared to the utility resulting from the RL-derived policy. Figure 3.9 depicts the total user spending (i.e. ISP revenue) and rewards in one billing cycle; we compare

performance without SDOs and with SDOs but without RL (i.e. when the hourly decision model is used) to performance with the RL agent when its training is finished (i.e. the agent always exploit in each state of the billing cycle). First, we observe that the RL technique results in the lowest aggregate user spending and highest total agent reward even as the variance in users' hourly desired consumption increases (x_t). Second, we note that performance degrades with the overage-average hourly decision model compared to the scenario where SDOs are not offered, reaffirming the observations in Section 3.5 that the myopic decision-making strategy is exploited by ISPs to have users spending more for the same or marginally lower data benefit (resulting in a net lower reward). In-fact, the overage-averse model performs worse as users' variance in hourly consumption increases, rendering the strategy unsuitable for practical scenarios. The trained RL agent, on the other hand, is effective in maximizing users' aggregate utilities compared to the overage-averse one. It even results in slightly higher aggregate reward compared to the regime without SDOs (Figure 3.9(b)) by lowering users' spending for equivalent data benefit (Figure 3.9(a)). Indeed, this the best possible outcome; since we do not modify users' desired data consumption for an hour in response to that hour's SDO offer decision, the amount of consumed data per user is same under all regimes. However, with optimal actions learnt using RL, some users start to reduce overages they originally incurred (in the regime without SDOs) and instead strategically accept SDOs in certain hours that result in a net lower cost for them. Note that this substantiates findings from the analysis of the monthly model in Section 3.3 where, with full information from the ISP about SDO offers and optimal planning, users could potentially benefit from SDOs by reducing spending.

We further delve into this by studying reward and spending changes for individual users in the simulation when they employ RL for making SDO choices compared to the regime without SDOs. Figure 3.10 depicts the cumulative distribution function of percent change in user spending and reward with RL compared to without SDOs, for different values of variance in hourly user consumption. We observe that the RL agent sometimes results in increased net spending by accepting SDO offers while also resulting in decreased net spending for some users (by decreasing overages and strategically accepting SDOs instead). When the hourly variance is low (between $1 - 100$KB), spending and rewards largely remain equivalent as without SDOs, with approximately 10% of users experiencing a net increase in rewards upto 15%. Based



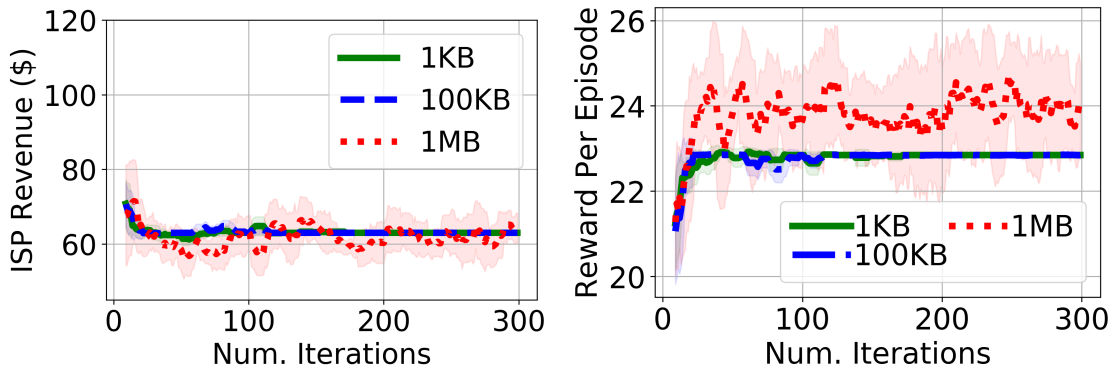
(a) % Change in User Spending with RL wrt Without SDO after Training completes

(b) % Change in Agent Reward with RL wrt Without SDO after Training completes

Figure 3.10: The policy learnt by the RL agents results in (a) equivalent spending as the regime without SDOs for most users with some users experiencing an increase or decrease upto 50%, and (b) largely increases in the realized reward for most users.

on the trace-driven parameters of the 20 users in the simulation, the net effect across users is either equivalent spending or lower spending compared to the regime without SDOs, as seen in Figure 3.9.

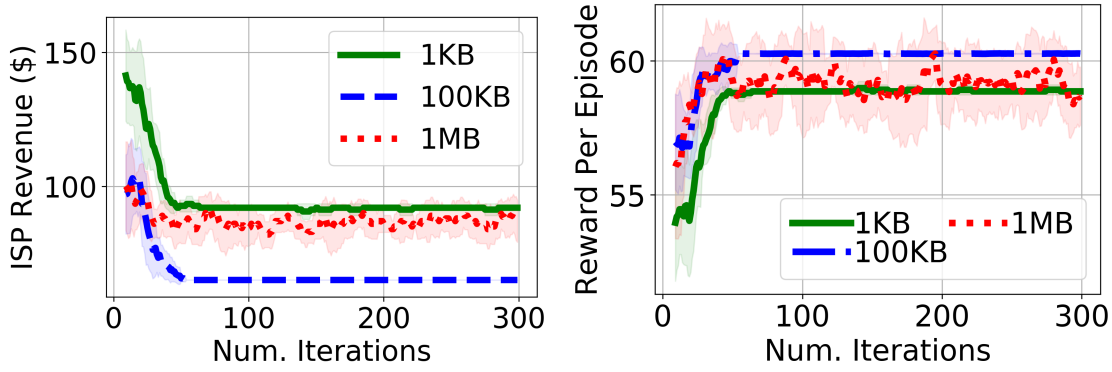
Finally, we show the number of training iterations required for the learning algorithm to converge to an optimal policy that yields stable per-episode spending and reward for two representative users, in Figures 3.11 and 3.12. For the user in Figure 3.11, it takes approximately 10 24-hour episodes with the ISP to converge on a stable policy, while for the user in Figure 3.12, it takes approximately 50. However, note that as described earlier, 384 interactions with the ISP are used to train the model per billing cycle; hence data from 16 rounds of 24-hour episodes with the ISP are used in training in one episode. A convergence time of $10 - 50$ billing cycles observed in Figures 3.11 and 3.12 therefore corresponds to $160 - 800$ billing cycles of data that the agent has trained on before convergence. We observe that across the 20 users in the simulation, heavier data users that incur more overages and costs (even without SDOs) tend to take require more episodes to converge (not shown); the user depicted in Figure 3.12 takes the longest convergence time in our set. We note that while convergence time appears to vary across users based on their data consumption trend over the billing cycle, it does not vary substantially across different values of variance in hourly consumption for a given user.



(a) Evolution of Per-Cycle Spending over Training Iterations

(b) Evolution of Per-Cycle Reward over Training Iterations

Figure 3.11: We show the moving average of (a) ISP revenue (i.e. user spending) and (b) agent reward for a single user in the simulation by greedily exploiting the learnt policy at the end of each training iteration.



(a) Evolution of Per-Cycle Spending over Training Iterations

(b) Evolution of Per-Cycle Reward over Training Iterations

Figure 3.12: We show the moving average of (a) ISP revenue (i.e. user spending) and (b) agent reward for another user in the simulation by greedily exploiting the learnt policy at the end of each training iteration.

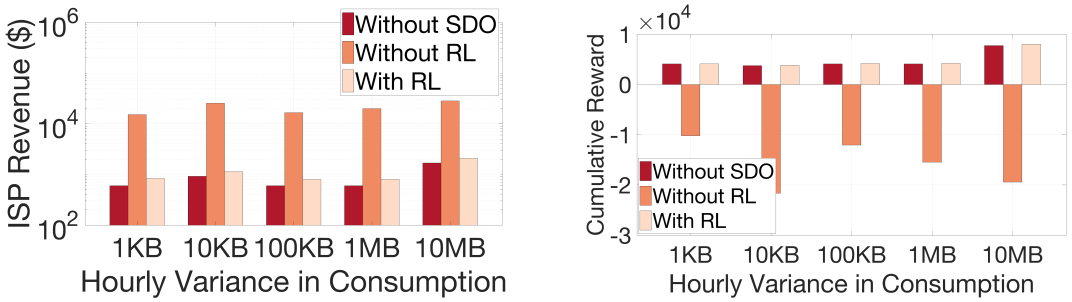
3.6.3 720-hour Billing Cycle

While the previous setup allowed us to compare results based on the RL technique with the ones shown in Section 3.5 based on the overage-averse decision model, we now evaluate whether the RL technique performs equally well on a more realistic setting with a 720-hour billing cycle as is typical. We first find that the simple Double DQN technique used above does not yield promising results here. The state space is now much larger and the agent must learn to associate actions and rewards that are hundreds of billing-cycle hours away from one another. We hence turn to more sophisticated RL approaches that allow us to account for realized gains several timesteps into the future.

Learning Framework. Specifically, we utilize the N-Step Advantage Actor-Critic Model [167] which provides a balance between bootstrapping the value function and using the full Monte-Carlo return by using an N-step trace as the learning signal. Note that with $N = 1$, this simply reduces to a regular Advantage Actor-Critic network (A2C) and with $N = \infty$, this reduces to the episodic policy-gradient based REINFORCE algorithm [167].

The Actor and Critic models are parameterized by two deep neural networks; the Critic estimates the state-value function $V(\mathcal{S})$ that indicates the maximum expected reward realizable from a state until the end of the episode, while the Actor network estimates the optimal policy function and updates the policy distribution in the direction suggested by the Critic. To mitigate the high variance and noise in vanilla policy gradients, gradient updates of the Actor network are scaled by the N-step Advantage values. We minimize the advantage-scaled cross-entropy loss for the Actor model and the root mean squared error for the Critic model.

Evaluation. For the Actor model, we implement a 4-layer neural network which takes as input a state tuple of 6 values. Each layer is a dense linear layer that is initialized using the Variance Scaling method; the 32-unit first layer uses the ReLu activation function and is followed by two 24-unit layers that also use ReLu activations. The fourth layer uses the Softmax activation to output a 3-sized vector indicating the probability with which each of the three actions should be executed. The first three dense layers all employ a 20% dropout rate. The Critic model uses the same architecture as the Actor but outputs just one value indicating the predicted maximum value of being in the input state. Both models use the Adam optimizer with a learning



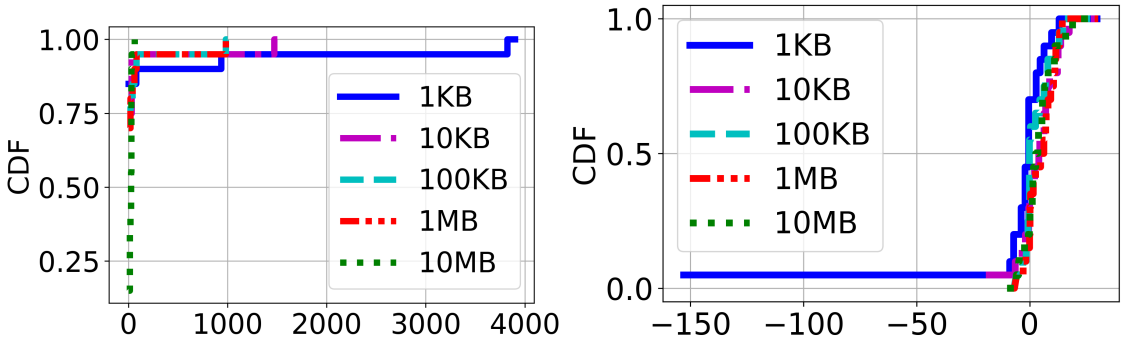
(a) ISP Revenue (aggregated across users) in One Billing Cycle after Training completes

(b) Aggregate Reward across agents in One Billing Cycle after Training completes

Figure 3.13: The policy learnt by the RL agents results in (a) substantially lower user spending than the overage-averse decision model, for (b) equivalent cumulative rewards as the regime without SDOs.

rate of 10^{-3}. We set the discount factor $\mathcal{D}$ to .99. During training time, we follow an ϵ-greedy technique for exploration where the Actor model is used to guide exploitation; we set $\epsilon = .8$ initially and decrease it by $4.5e-3$ every episode. We set N to 200, hence the agent first samples an entire episode of 720 timesteps, then computes 200-step lookahead return for the state transitions seen in that episode. The Critic model is then trained with these 720 states and computed returns over 10 epochs (with shuffling). Note that the state tuples are normalized as earlier. The Actor model is trained once every 5 episodes with the normalized Advantage values. As done previously, we retain a cost scaling factor of $\mathcal{C} = 35$ and a reward scaling factor between 10^{-5} and 10^{-7}.

As earlier, we compare users' performance in terms of their spending and reward with the RL technique against performance without RL (i.e with the hourly decision model being used to decide SDO acceptance) and performance without SDOs (i.e. with the hourly decision model being used to decide between consuming under WiFi and Data Plan). Figure 3.13 depicts the total user spending (i.e. ISP revenue) and rewards in one billing cycle when the RL agents are finished training. We observe that the myopic decision model to decide SDO acceptances results in substantially more spending for the user (for the same net data consumption) compare to the regime without SDOs; in-fact, the increase in suboptimal spending for this case of the 720-hour billing cycle is much larger than the increase observed in Figure 3.9(a) Correspondingly, the cumulative rewards of agents using the hourly decision making policy for SDOs is *negative*. The trained RL agent, on the other hand, eliminates most of the increased spending that the hourly decision model results in; the RL



(a) % Change in User Spending with RL wrt Without SDO after Training completes

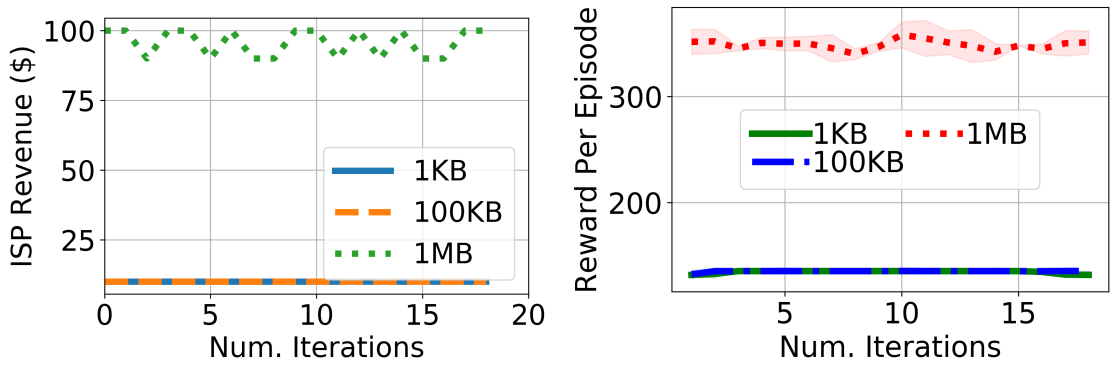
(b) % Change in Agent Reward with RL wrt Without SDO after Training completes

Figure 3.14: The policy learnt by the RL agents results in (a) largely equivalent spending and (b) largely improved reward, as the regime without SDOs.

agent results in marginally higher aggregate spending for users for net equivalent aggregate rewards across them, when compared to the regime without SDOs. This is explained by the cost scaling factor $\mathcal{C}$ that is used in the reward function (see Eq 3.26) to reconcile the difference in units between the data benefit terms and the cost terms involved. This scaling factor essentially determines how sensitive the reward function is to the cost term; since the RL agent here results only in a marginally higher aggregate cost, the cost scaling factor is not large enough for this to result in a substantitive decrease in the reward.

Figure 3.14 depicts the cumulative distribution function of percent change in user spending and reward with RL compared to without SDOs, for different values of variance in hourly user consumption. We observe that the RL agent results in a very significant increase in spending for a very few number of users; most users, however, experience a net small increase in their reward. Since the RL algorithm optimizes the policy it learns based on reward maximization, this indicates that the agents at large learn to choose actions that maximize their reward, despite this resulting in increased spending for a few users.

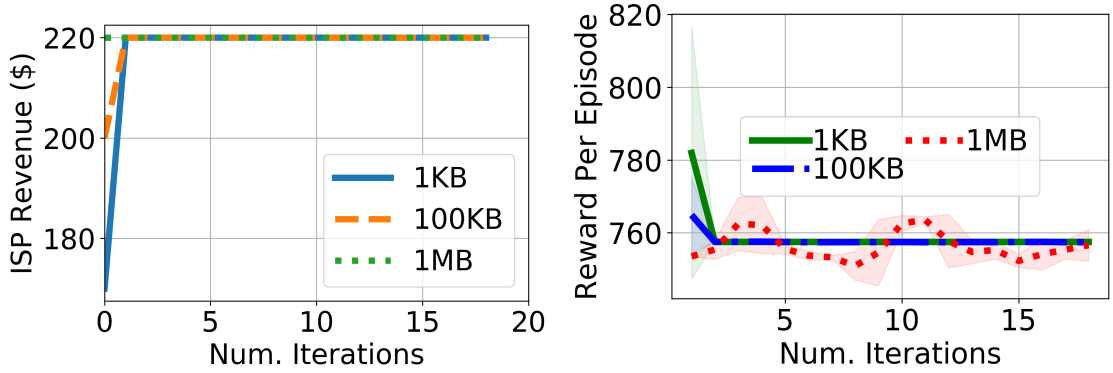
Finally, we show the number of training iterations required for the learning algorithm to converge to an optimal policy that yields stable per-episode spending and reward for the same two representative users as earlier, in Figures 3.15 and 3.16. Note that in this case, we test the learnt policy once every 25 episodes. As seen, the user in Figure 3.15, converges to a stable policy within the first test, i.e. within 25 episodes. For the heavier data user in Figure 3.16 (who required more iterations to converge



(a) Evolution of Per-Cycle Revenue over Training Iterations

(b) Evolution of Per-Cycle Reward over Training Iterations

Figure 3.15: We show the moving average of (a) ISP revenue (i.e. user spending) and (b) agent reward for a single user in the simulation by greedily exploiting the learnt policy at the end of each training iteration.



(a) Evolution of Per-Cycle Revenue over Training Iterations

(b) Evolution of Per-Cycle Reward over Training Iterations

Figure 3.16: We show the moving average of (a) ISP revenue (i.e. user spending) and (b) agent reward for another user in the simulation by greedily exploiting the learnt policy at the end of each training iteration.

for the 24-hour cycle as well), the agent only converges between $50 - 75$ episodes. Similar to the 24-hour case, we see that while convergence time appears to vary across users based on their data consumption trend over the billing cycle, it does not vary substantially across different values of variance in hourly consumption for a given user.

3.7 Discussion

While the reinforcement learning techniques used above are effective in making optimal SDO acceptance decisions for users, convergence requires upto hundred of episodes for the 24-hour as well as 720-hour billing cycles. Note that the use of the N-step advantage function used in the latter also renders the earlier technique of extracting multiple training samples from a state transition (used in the 24-hour billing cycle case) inapplicable here. This lengthy convergence time represents expensive billing costs in the order of thousands of dollars incurred by users in months that they spend exploring the environment, before starting to exploit the discovered optimal solutions. To reduce this duration and make the use of this technique practical, we plan to exploit our paritial model knowledge of how state transitions occur in the environment to pre-train the model. As described in Section 3.6.2, various state transitions can be simulated for a given state and action without actually sampling the real environment, except for the values of y_t, i.e. the ISP offer decision each hour. We can hence use the technique proposed in recent work [170] to first train the model on simulated state transitions with random policies in place for the ISP offer decisions, and then construct a new model during real interaction with the ISP that minimizes KL divergence from the original model. This would allow us to pre-train the model on behavioral priors that are already known (that is, the user's typical consumption patterns and WiFi availability, i.e. x_t, ϕ_t and ψ_t); presumably and as observed in prior work, fewer iterations would then be required in real episodes with the ISP as the only new aspect of the environment that the model has to incorporate now is the real ISP offer pattern (as opposed to the random placeholder policies initially used for the ISP offer decisions).

3.8 Summary

In this work, we analytically and empirically assess the viability of supplemental discount offers from ISPs to their users. We first abstract away from hour-to-hour dynamics to show that most users would accept some SDOs, and that those who consume the most data per SDO would also consume the most data on their cellular data plans. We then build on this framework by developing hourly decision algorithms for users to decide when to accept and ISPs to decide when to make SDOs. We simulate these algorithms over a two-week trace of data usage, empirically establishing that SDOs can increase ISPs' network utilization and revenue. Moreover, ISPs can exploit user uncertainty in when SDOs will be offered to further increase their revenue. Our work captures Verizon's claimed motivation of offering PopData in order to recover usage that would otherwise have been realized on WiFi networks [103], and indeed we find a tradeoff between the ISP maximizing its revenue and its network utilization.

Chapter 4

Network Slicing for Real-time Session Guarantees

We next consider a more user-driven resource procurement approach, wherein the end-device specifies the resource needs for its session to the network at the start of the session. In response, the networks utilizes its radio access network virtualization capabilities to create a slice of resources for the entire session as specified and allocate it to the device. We study the feasibility of such session-oriented slicing and its ability to meet promised performance guarantees, propose mechanisms for the network to elicit truthful resource requests and valuations from users, and discover methods to solve the allocation problem in realtime for real-time applications. Finally, we explore ways to ensure that the user does not exceed their budget constraints and makes optimal decisions in realtime.

4.1 Problem Definition

Cyber-Physical Systems (CPS) and the Internet of Things (IoT) are emerging paradigms for increasingly pervasive and real-time computing environments. Users are coming to rely implicitly on the availability of services like Amazon Alexa and Google Home for their day-to-day tasks, while ambitious next-gen offerings like HoloLens promise to enable new use cases for real-time augmented reality like telepresence. However, the network connectivity that these ubiquitous computing environments rely on is insufficient [186] even for current applications. Proliferate mobile multimedia services such as video conferencing and interactive mobile gaming have specific resource needs

to provide an acceptable Quality of Experience (QoE) for the end-user. Without any means of conveying these needs to the network, the network may not meet them [35, 98] leading, for instance, to Google deploying its network Espresso to contend with its applications' needs.

In this work, **we propose to provide resource guarantees on a per-session basis to QoE-sensitive applications by reconciling available resources with their session needs.** To see the benefits of such guarantees, consider a Skype user starting a video conferencing session for a job interview who cannot procure any guarantees for the call quality and performance. Mechanisms for QoS-aware allocation [69, 119] typically do not model resource consumption at session-level timescales, hence allowing demand spikes to potentially interrupt or degrade the call. Xu *et al.* [186] and others [200] investigate this problem of high variance in cellular resource availability in the context of real-time applications. With buffer-based reactive measures like HTTP DASH infeasible [165, 185], the authors propose a short-term up-ahead estimation of channel conditions to proactively adjust application behavior, thereby reducing perceived delay. Improved channel estimation, however, cannot help sessions that must be preempted altogether due to congestion or spiky traffic, thereby entirely disrupting the call. Hence, channel estimation itself is insufficient to *guarantee call quality or completion.* We instead propose to proactively allocate resources to real-time sessions to guarantee high QoE over their entire duration. Proactive resource provisioning has been studied [69, 187] at the time-scale of packets or transmission time interval (TTI), using dynamic pricing or auction-based methods to allocate limited resources. However, the resulting allocation at one TTI is largely independent of the next; even if allocations at past TTIs were accounted for, e.g. to satisfy a long-term proportional fairness objective, the resulting allocation at some TTI may end up halting the flow temporarily or pre-empting it altogether, limiting their use here. For multimedia applications, while millisecond level network performance affects user perception, user engagement and ultimately QoE occur at the session level. Therefore, the user's QoE depends on the resource allocation throughout the duration of the session, which is on the order of minutes or hours. **Our goal in this work is to proactively provide resource guarantees expressed in terms that a user agent or application can understand and negotiate for, abstracting away the lower-level intricacies of network resource allocation as details left to the network operator.** Similar mechanisms have been proposed for the

Internet backbone [184] and cloud environments [102, 136], and wireless applications are likely to benefit from them even more due to their best-effort nature. However, this also makes it challenging to provide such guarantees in wireless networks. In fact, offering multiple tiers of service guarantees to wireless users is the goal of the emergent *network slicing* paradigm in 5G. Critical components of network slicing are still in their infancy [77], including *RAN slicing*, i.e., designing the wireless radio access network to enforce per-flow performance guarantees, and *slice admission and management*. This work addresses these research challenges [93] that arise in enabling proactive, session-level resource provisioning for wireless networks.

Feasibility: To proactively provision flows for their anticipated duration, available spectrum resources must be quantified and accurately reconciled with session requirements presumably expressed in terms of bitrate, latency and duration. This raises important research questions that we seek to answer in this work. Is this feasible? Even then, are performance guarantees possible despite uncontrollable wireless influences like fast fading? Recent works in RAN slicing [78, 116] facilitate functional slice isolation and empirically analyze various factors to provide probabilistic performance guarantees in cellular networks. We herein employ an admission control algorithm that allows a flow into the network based on a simple resource forecast and reconciliation model that is generalizable to both scheduled and random-access wireless networks. We conduct extensive WiFi experiments and trace-driven LTE simulations with multimedia applications, and find that admitted latency-sensitive as well as bitrate-heavy flows achieve their promised performances and congestion externalities are effectively mitigated. In fact *the network accommodates even more flows by implementing incentivized admission control.* Further, since this may be offered as a *value-added service* that some users may not require, we show reliable guarantees can be made even in the presence of background flows not controlled by our admission algorithm.

Allocation and Incentive Compatibility: Given a forecast of network resource availability and a mechanism to reconcile this with session needs, how should these limited resources be provisioned? The network will likely need to prioritize users with higher resource valuations as it cannot accommodate all session requests. Variation in such valuations could arise from usage context (medium quality for recreational video calls but high for an interview) or device preferences (lower resolution on a smartphone vs a 4K monitor). Further, allocating resources for the duration of a

session is particularly difficult as the operator must account for uncertainty in future needs, and users may strategically misrepresent their needs and valuations. We address these concerns in a novel auction model. The operator offers consecutive auctions throughout the day, and users relay their sessions' resource needs dynamically in a combinatorial bid to the current auction; session durations may span multiple auctions. We show that the spontaneous and real-time nature of sessions can be exploited to reduce the search space of the intractable optimization problem of determining winning bids, thereby facilitating *spontaneous guarantees*. We propose multiple ways for the operator to *incentivize truthful user declarations* even under uncertainty of future bid arrivals and analyze *trade-offs in social welfare, incentive compatibility and operator revenue*.

Usability: For users to procure and benefit from performance guarantees in this system, they must engage in routine auctions by bidding. However, studies have shown [87] that dynamic pricing is challenging for end users who are budget constrained and averse to making real-time network consumption decisions. We consequently address the user-facing challenges of *resource-specification overhead*, *price discovery* and *budget constraints*. We envision that an automated agent will participate in these session-oriented resource auctions on each user's behalf, placing bids using a parameterized utility model and enforcing the user's daily budget. We formulate the distribution of this budget across bids as a dynamic program solved with model-free reinforcement learning, specifically the Monte Carlo policy iteration algorithm [167]. We show via simulation that these agents *maximize user utility* for a given budget within a billing cycle (1 month) *without any loss in revenue to the network operator*.

Overall, *we formulate an end-to-end system for realizing session-level performance guarantees, addressing challenges in the radio access network, incentive mechanisms for resource provisioning, and usability.*

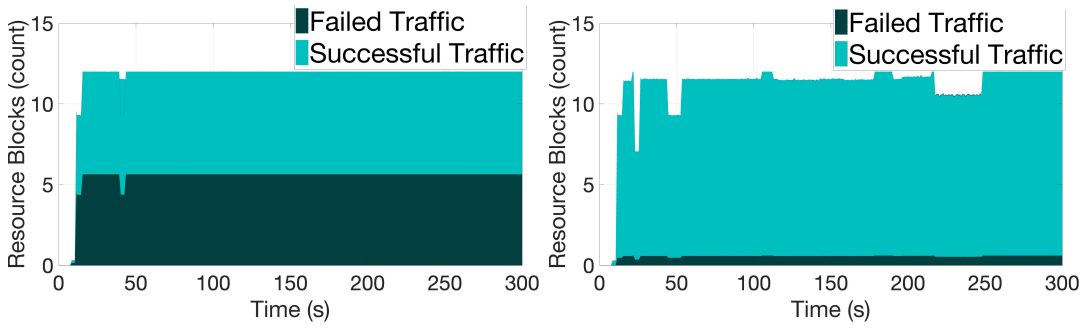
4.2 Related Work

Auctions in wireless networks have been mainly studied in three contexts: spectrum license allocation, secondary cognitive radio allocation, and QoS-aware resource allocation. Auctions for long-term spectrum licenses are held over hours or days with multiple bidding rounds before the auction ends and winners are determined, like the popular simultaneous ascending and combinatorial clock auctions [57, 58]. They do

not account for the faster time scales of session-level allocations for spontaneous application sessions. Further, the combinatorial nature of our auctions presents significant challenges in the context of this prior work. Cognitive radio auctions [196] do not consider session-level app performance, instead availing opportunistic spectrum for much shorter time scales. Auctions have also been employed for QoS-aware real-time channel allocation to primary users in mobile networks. The goals of such approaches [69, 187] differ from ours in their focus on sub-carrier allocation with millisecond granularity and interference mitigation. Using auctions for short-term resource allocation does not guarantee session-level performance, which introduces new combinatorial characteristics that we address. Our work furthers 5G's envisioned network slicing capabilities [77, 93]. We verify the premise of RAN slicing for both LTE and WiFi networks, also studied in parallel by Foukas *et al.* [78] and a few others [116, 137] in the cellular context, and provide incentive-compatible mechanisms for modeling slices and admitting users to them. Other recent works since ours have validated our findings and further proposed techniques for RAN slicing. These proposals [36, 66, 91, 143] for mapping spectrum resources to virtual slices such that slice performance guarantees are closely met further illustrate the practicality of session-level slicing and the relevance of the incentive-compatible auction protocol that we develop for this. We also address budget optimization in the context of repeated auctions, which has been studied in limited settings and even fewer of them combinatorial. Gummadi *et al.* [86] study budget-constrained bidding for sponsored search auctions, but with strong assumptions about the system that guarantee equilibrium. Janssen *et al.* [97] study the combinatorial setting, but their work is limited to the combinatorial clock auction. Almost no work has considered whether reinforcement learning can inform auction bidding strategies as we do here.

4.3 Feasibility of Session-Level Performance Guarantees over LTE

Providing session-level performance guarantees requires an Admission Control (AC) procedure that only admits flows with demands that can be fulfilled for the stated session duration. This reconciliation of available and required resources is then expected to result in admitted flows that are robust to externalities and realize their promised



(a) Without AC (b) With AC

Figure 4.1: (a) Without a resource-aware Admission Control (AC) algorithm, almost half of the LTE network's resources are expended in failed sessions. (b) With the AC in place, this is reduced to ∼5% *while preserving high network utilization.*

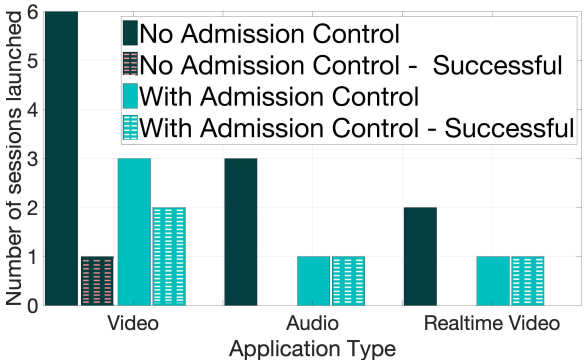


Figure 4.2: With AC, most performance guarantees are met even in the presence of uncontrolled background traffic.

performances. While unpredictable real-time channel fluctuations are inevitable (e.g., fast fading), our premise is that the short timescale of these fluctuations affects session-level QoE less than user competition, which occurs at session-level timescales. We validate this via a proof-of-concept trace-driven simulation of users with different traffic types sharing the resources of an LTE eNodeB. We show session performance can be guaranteed by an AC that 1) accounts for resource competition in flow admission and 2) accommodates for unpredictable wireless externalities when provisioning capacity.

Setup: We use SimuLTE and INET[1] to simulate LTE with TCP/UDP/IP. Our simulation includes one eNodeB with 12 resource blocks and a noisy channel. Users are randomly dispersed in the coverage area, yielding variation in channel qualities. Data usage is modeled from the multimedia activity found in mobile traffic traces

[1] http://simulte.com/, https://inet.omnetpp.org/

of 20 users collected over 10 days. Multimedia content from the traces includes *video streaming*, *audio streaming* and *real-time video conferencing*, parameterized by bitrates and latencies of known applications[2][3][4]. Since the AC algorithm compares these required quantities with the available capacity to determine flow feasibility, it must translate granular frequency-time network blocks into bitrate and latency capacity forecasts. While devising an accurate model for this is a challenging task in itself and out of our scope, we presume a naïve model that is sufficient to indicate general feasibility and benefits of this approach. In essence, the AC procedure maps the 12 resource blocks to a conservative estimate of bitrate capacity (e.g., 10Mbps), thereby allowing a buffer of radio capacity that may be consumed, for example, by an admitted flow(s) with poor signal strength or temporary channel degradation. A flow is admitted after verifying that its required capacity can be accommodated, and the capacity forecast reduced according to the requested bitrate for the specified duration.

Network Performance with AC: We measure link-layer utilization both with and without session-level AC. Figure 4.1(a) shows that without AC, roughly 47% of resource blocks are allocated to sessions that fail (i.e., the stream halts before completion) mainly from excessive resource competition. However, as in Figure 4.1(b), the AC algorithm drastically reduces this wastage (to below 5%) while *preserving nearly full utilization* of available resources. While the AC allocates a conservative 10 Mbps for flow provisioning to guard against externalities, this would presumably leave the network underutilized. In this case, however, congestion between flows and impact from other externalities were almost entirely eliminated while retaining high utilization. These promised performances were achieved with noise and channel quality variation, indicating that session-level guarantees can be provided in wireless cellular networks without the significant cost of network under-provisioning. Even the naïve model of capacity forecasting used by our AC procedure proved sufficient to serve most admitted flows of different application types. This validates our premise that wireless radio resource modeling and reconciliation with session-oriented resource requirements are feasible and can likely provide performance guarantees.

[2] https://support.skype.com/en/faq/FA1417/how-much-bandwidth-does-skype-need, http://download.skype.com/share/business/guides/skype-connect-requirements-guide.pdf

[3] https://help.pandora.com/customer/portal/articles/166391-minimum-specifications-to-run-pandora

[4] https://support.google.com/youtube/answer/2853702

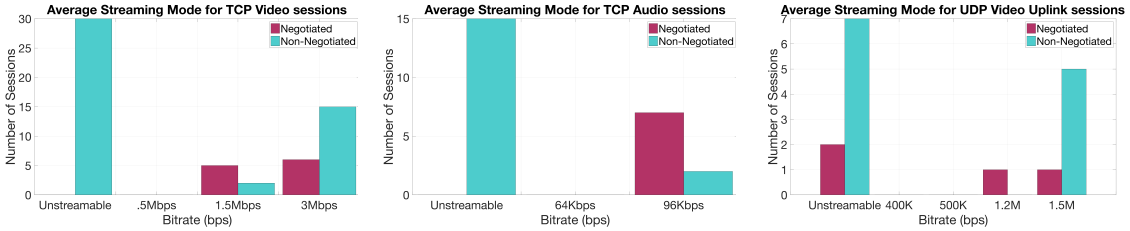
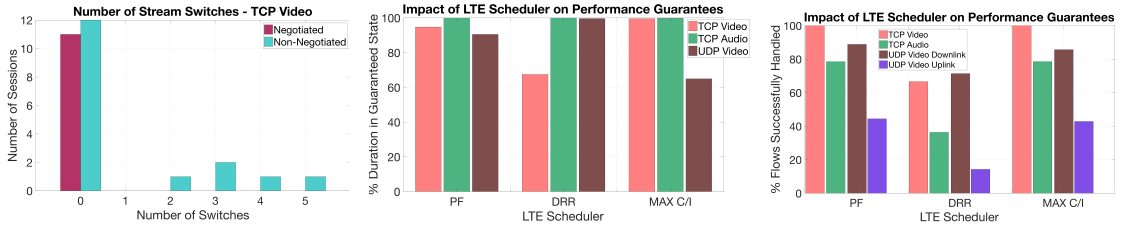


Figure 4.3: Admission Control has significant impact on flow performance for all application types. From (a)-(c), we see reduction in unstreamable flows and improved throughput.



(a) (b) (c)

Figure 4.4: (a) The number of encoding switches during TCP-based video streaming sessions reduces to 0 with negotiated access to the network from upto five without, indicating that the network is able to effectively eliminate congestion externalities faced by these sessions. In (b)-(c), we compare application flow performance for three different scheduling algorithms in terms expectations.

Since AC-admitted sessions may co-exist with unregulated background sessions (of users that do not require performance guarantees), we reserve some network capacity for this traffic and apply AC to the remaining capacity. We employ the MAX C/I scheduling at the MAC layer to prioritize flows with better channel quality [41], and modify the scheduler to further prioritize AC-admitted flows before ranking by channel quality. Figure 4.2 compares the resulting performance of sessions belonging to AC and non-AC traffic. An AC session is deemed *successful* if it streams for its entire duration at its guaranteed bitrate on average; real-time sessions additionally require a packet inter-arrival time below 40 ms. A non-AC session is successful if it achieves *any* resolution supported by its traffic type. As in Figure 4.2, the network is highly congested with non-AC traffic, so only one of eleven non-AC flows succeeds, while four of the five admitted AC flows succeed. Figure 4.2 demonstrates that session-level guarantees can be achieved in the presence of non-AC traffic with appropriate reservations in the capacity forecast and MAC prioritization.

Figure 4.3 shows the numbers of flows achieving a given average bitrate for each traffic type. For audio and video streaming flows in 4.3(a) and (b), the network incurs *no* "unstreamable" sessions with the negotiated mechanism. Only three sessions (constituting real-time flows) over all three applications were unsatisfied with the auction-based admission compared to 100 without. Thus, with AC-based flow admittance, users can reduce uncertainty in their application performance to a large extent and even *deterministically* plan their sessions/usage, relying on stable connectivity. We further quantify this stability of admitted sessions by examining the number of flow *switch events* [132], *i.e.,* the number of times the client on the device switches to a different encoding rate, as a result of network performance. As Figure 4.4(a) depicts, with AC, no TCP-based video streaming flows experience switch events, while five successful flows experience two to five switch events each without negotiation. We see a similar trend for both TCP-based audio and real-time sessions (not shown).

We now study the impact of the MAC scheduling discipline used by the LTE network on the performance of the AC-admitted flows. We compare three scheduling disciplines: MAX C/I, Proportionally Fair (PF) and Deficit Round Robin (DRR) [41]. While Max C/I prioritizes scheduling of flows with better channel quality, PF ensures long-term fair resource allocation among flows, while taking their channel quality into account. DRR is entirely agnostic channel quality, rather allocating resources in a round robin fashion subject to resource availability.

As shown in Figure 4.4(b), Max C/I successfully delivers on all audio and video flow guarantees. However, latency-sensitive traffic is served well only by DRR. Since DRR allocates limited resources to as many flows in its queue as possible, it services low-latency flows like real-time traffic frequently, but higher throughput flows suffer. PF reaches a middle ground between Max C/I and DRR, delivering video guarantees 95% of the time and UDP guarantees 90% of the time. Realtime sessions particularly benefit from PF's long-term fair resource allocation due to their long durations. Figure 4.4(c) shows the total number of successful negotiated flows. Across all schedulers, uplink real-time flows perform the worst due to their inability to re-transmit packets. Amongst the other traffic types, DRR strictly performs the worst in keeping its guarantees to as many nodes as possible. Max C/I and PF perform almost the same. However, all three schedulers still provide 80 to 100% guarantees, indicating that *with additional lower-layer optimization that accounts for users' known flow demands, all performance*

guarantees made to AC-admitted flows can be met by the network.

4.4 Feasibility of Session-Level Performance Guarantees over WiFi

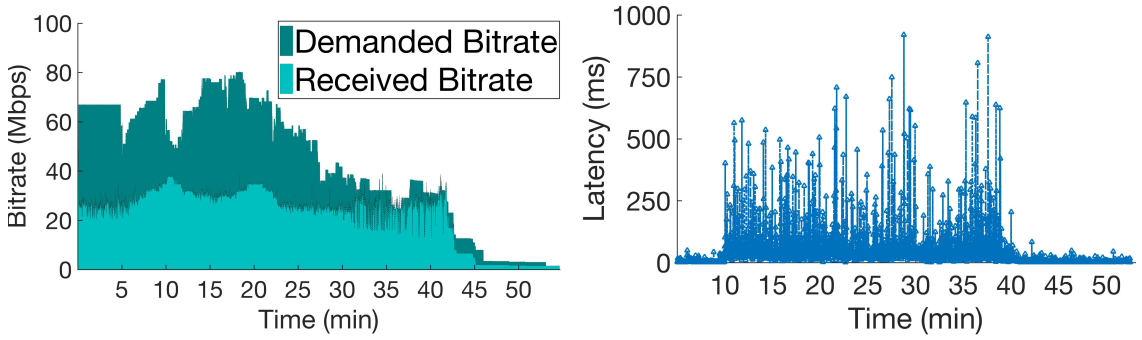
We now assess whether performance guarantees can be made in a random access medium like WiFi. Unlike LTE, WiFi has a short range and operates in unlicensed spectrum, making the channel more susceptible to interference and externalities. A likely use case for performance guarantees, however, is where multiple users engaged with various apps contend for congested resources of a public WiFi network, for instance, in a cafë-like scenario. Measurement studies [24, 82] have shown extensive growth in public hotspot traffic and Access Point (AP) deployment, with WiFi traffic doubling every two years (∼35% video). These experiments verify the feasibility of session-level guarantees in such scenarios.

Setup: We launch 50 iPerf[5] clients in parallel across multiple devices to induce channel quality variations. Clients connect to an 802.11g AP operating at 2.4GHz and launch five sequential sessions over 50 minutes, each comprising a random duration of video streaming, audio streaming or video conferencing (we continue to use resolution rates and corresponding bandwidth requirements that are widely in practice). We further incorporate non-AC web browsing traffic at 50 Kbps, thereby inducing overall activity variance typical of public WiFi.

Network performance without AC: As a baseline, we first engage the 50 clients in their planned mobile activity over this hotspot without the AC algorithm. Clients request the highest supported resolutions for their multimedia sessions (e.g., 1.5 Mbps and 4.5 Mbps for video conferencing and streaming, respectively) since they have no incentive to request lower bitrates due to free hotspot access. An aggregate data demand of up to 80 Mbps is seen in Figure 4.5(a). However, although the 802.11g AP has a theoretical capacity of 54 Mbps, *only half of this is realized by the network*, indicating severe performance degradation from congestion. The network also exhibits high latency and jitter, as in Figure 4.5(b), causing real-time video sessions (requiring ∼100ms) to fail or be lag-ridden.

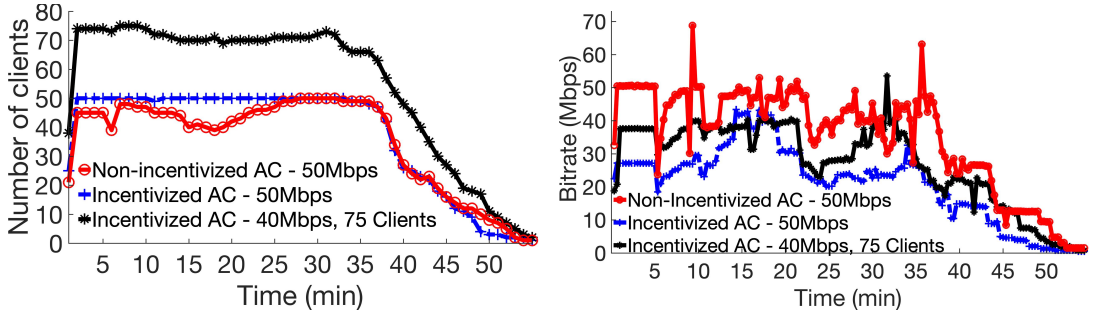
AC Procedure: We now introduce our admission control process. Browsing

[5] https://iperf.fr/



(a) Network Throughput (b) Network Latency

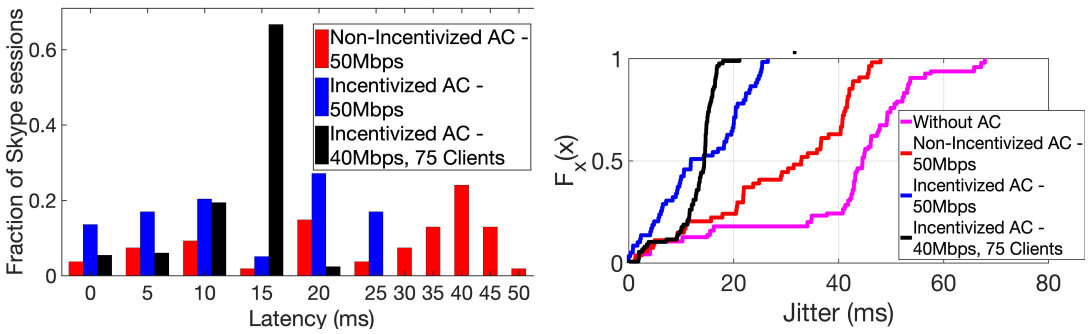
Figure 4.5: (a) Due to high data demand, the network is congested and delivers around 30 Mbps throughput despite 54 Mbps capacity and (b) experiences latency spikes between $250 - 750$ ms and high jitter.



(a) Client Admission (b) Network Throughput under AC

Figure 4.6: (a) The incentive mechanism induces heterogeneity in requested resolution rates that allows the network to admit all initiated sessions, and (b) with AC, network throughput *increases*.

sessions constitute light-weight traffic and always commence upon launch, modeling the background traffic of regular-access users as in the LTE experiments. The AC algorithm simply uses an estimated bitrate capacity as an abstracted representation of available radio resources and permits a session to start only if requisite session bandwidth is available. We herein refer to this as *Non-Incentivized* AC and introduce a corresponding *Incentivized* version. Since Non-Incentivized AC admits flows (subject to feasibility) on a first-come first-serve basis, users always request high bitrates even when they may be content with lower bitrates (e.g., when using a mobile device with low screen resolution). With Incentivized AC, we presume that an incentive mechanism induces clients to request only their value-maximizing resolutions for



(a) Mean latency - Real-time Video

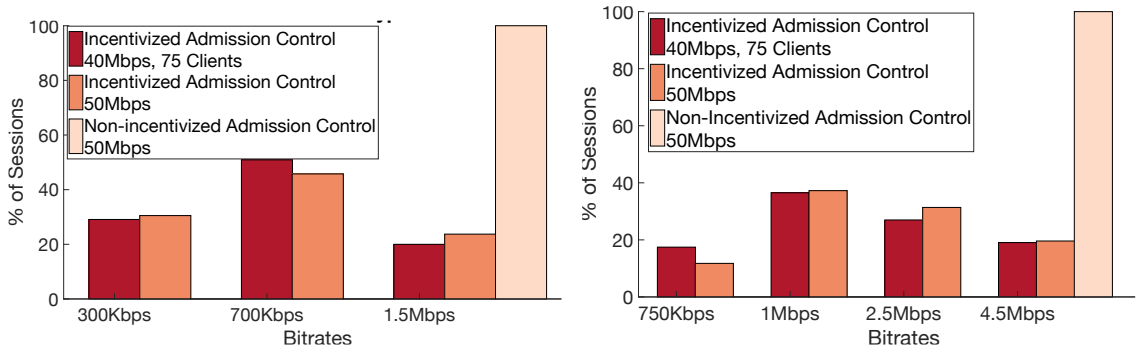
(b) CDF of per-second Jitter

Figure 4.7: With AC, (a) admitted real-time video sessions have latencies within $25-50$ ms and (c) jitter less than 40 ms. Incentivized AC further reduces data demand and improves both jitter and latency.

multimedia sessions. This incentive mechanism may, for instance, be a payment policy that charges admitted clients to persuade them to state only what they need (developed in subsequent sections). Using Incentivized AC, users are thus admitted according to their valuation for the appropriate context. We simulate Incentivized AC with clients streaming multimedia sessions at a resolution that is randomly chosen from the supported ones, simulating the distribution of utilities, preferences and budgets in a population.

Performance With AC: While the AP's theoretical capacity is 54 Mbps, this is rarely realized in practice due to time-varying nature of the wireless channel. Since provisioning based on this capacity may result in poor performance of some flows, the AC procedure is initialized with a capacity of 50 Mbps. Figure 4.6(a) depicts the number of clients admitted into the network under AC. A few clients are consistently rejected for the first half hour due to lack of capacity as aggregate data demand is highest then, Figure 4.5(a). However, with Incentivized AC, *the entire pool of* 50 *clients is admitted into the network* at all times that they initiate multimedia flows. As clients distribute their requesting resolutions in alignment with their true utilities and valuations, aggregate user demand decreases so much that the network has sufficient capacity to now admit *all of them* (in this case), thereby *increasing the utility of the entire set of users*. Even when the provisionable capacity is reduced to 40 Mbps and number of clients increased to 75, the network admits them all with Incentivized AC, thereby increasing net social welfare of users.

With AC in place, *network throughput increases* to almost 50 Mbps, as in Fig-



(a) Histogram of Mean Throughput - Realtime Video

(b) Histogram of Mean Throughput - Video

Figure 4.8: With AC, all admitted sessions of (a) realtime video and (b) video stream at a mean bitrate that meets their performance guarantees.

ure 4.6(b). Due to random access in WiFi, congestion externalities have a severe impact on the network and are almost entirely mitigated with AC. When the Incentivized AC is deployed, as in Figure 4.6(b), only around 30 Mbps is typically required of the network now, indicating that incentivizing users for truthfulness may *allow the network to serve more users* overall. With AC, all real-time sessions stream at mean latencies below 50 ms, as in Figure 4.7(a), while introducing incentives causes lower aggregate demand which further reduces the mean latency to a maximum of 20 ms. In fact, the CDF of per-packet jitter for real-time sessions across all experiments in Figure 4.7(b) indicates that while more than 80% of packets exhibit jitter above 40 ms without AC, ∼80% of packets experience jitter below 40 ms with Non-Incentivized AC and below 20 ms (recommended for real-time video conferencing and gaming) with Incentivized AC. Further, as in Figures 4.8(a) and 4.8(b), we see all admitted multimedia sessions with AC (including audio, not shown), exhibit a mean streaming bitrate that corresponds to a supported encoding rate (i.e. no sessions fail); we also note that these mean bitrates that each session streams at correpsonds to its promised bitrates.

By controlling flow admittance based on bitrate demand, performance guarantees are delivered to latency-sensitive real-time flows as well as other video and audio streams. We thus validate our premise of session-oriented wireless resource provisioning and shift our focus to the design of the incentive mechanism employed by the AC algorithm. Note that given a process for forecasting network resource availability and reconciling it with session demands, our incentive mechanism is agnostic to the Radio

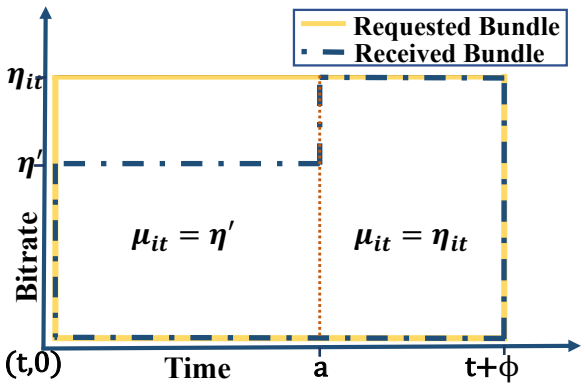


Figure 4.9: A bundle B_{it} received by a user i for request R_{it} may provide a subset of the requested resources. For example, R_{it} specified η_{it} for the duration ϕ_{it} but received a bundle providing $\mu_{ik} = \eta' < \eta_{it}$ for $k < a$ and $\mu_{ik} = \eta_{it}$ for $k \geq a$.

Access Technology (RAT) in use.

4.5 Modeling Auctions for Session-Level Resource Guarantees

We now design the incentive mechanism that determines session admission and cost. We focus on real-time applications that require immediate access, but our model is generalizable to other application types. The network operator discretizes time into a total of T slots per day, and an auction A_t is held in each time slot t. All resource requests are assumed to have a maximum duration of ϕ_{max} time slots; a longer session may simply submit another bid for resources after ϕ_{max} slots. For instance, the network may hold an auction each minute (i.e. $T = 1440$), allowing users to procure resource guarantees for the full duration of their sessions almost spontaneously as they launch them, and set $\phi_{max} = 20$, for resources to be periodically freed up once every 20 minutes. Similarly, the operator supports a discrete set of resource **modes** m_n for $n = 1, \ldots, M$, each corresponding to an operating bandwidth, bitrate, or similar. We define a generic mechanism for the network to define these modes, which allows for a wide range of supported bitrates by common applications, while significantly reducing computational overhead of the auction. To characterize the resources being auctioned, the operator computes a forecast $C_t^{(\tau)}$ for auction A_t of the bandwidth resources that will be available in each time slot $t + \tau, \tau \in \{0, \ldots, \phi_{max}\}$, accounting for resources

reserved in earlier winning bids.

The total number of users submitting bids for A_t is denoted by I_t. For users to express a desired combination of resources that the network can actually serve, they must know the resources available to bid on. We develop a two-round interaction mechanism for this resource discovery. In the first round of each auction A_t, user i expresses a request R_{it} that includes the desired duration $\phi_{it} \leq \phi_{\max}$ and the desired resource mode $\eta_{it} \in \{m_1, \ldots, m_M\}$, along with a corresponding valuation v_{it}. We assume the requested η_{it} is constant over the duration ϕ_{it}, as real-time applications typically have fairly stable resource needs over time. In response to resource requests $R_{it} = (\eta_{it}, \phi_{it})$ from users i, the network operator determines if granting R_{it} is feasible (given projected availability). If not, it generates a set S_{it} of alternate *resource bundles* (based on forecast capacity and adjusting for underlying wireless channel states), where each bundle $B_{it} \in S_{it}$ enumerates the offered resource $\mu_{ik} \leq \eta_{ik}$ for time slots $k = t+1, \ldots, t+\phi_{it}$. Figure 4.9 illustrates an example bundle offered in response to a resource request. Given the set S_{it} of available bundles, user i may bid on a bundle $B_{it} \in S_{it}$ by assigning a new bid value v_{it} on it, yielding a bid $b_{it} = (B_{it}, v_{it})$ for the select bundle $B_{it} \in S_{it}$. Once auction A_t is executed, user i learns the result $x_{it} \in \{0, 1\}$ of the bid and starts consumption if $x_{it} = 1$.

Maximizing social welfare to determine bid winners is desirable since Vickrey-Clarke-Groves (VCG) [108] payments can then be charged to incentivize truthful bidding. The resulting computation, however, is an NP-hard problem, the solution time of which is exponential in bid durations. Thus, in the next section we develop novel reductions to the problem by exploiting the *spontaneous* nature of winning sessions, i.e. that they begin consumption immediately. With this, the network can implement the VCG mechanism in real time, and stating true product valuations v_{it} becomes the dominant strategy of users. This allows users to avoid complex estimation of other bidders' strategies to maximize their own utility and allows the network operator to discover the distribution of true valuations across bidders, indicating the perceived value of network resources and the potential revenue. However, bidders are *multi-parameter agents* [44] in this setting; they state not just their valuations but also the desired mode η and duration ϕ. Incentivizing bidders to truthfully report ϕ requires modifications to the mechanism that account for *temporal correlations* between the decisions taken in different auctions; that is, users with accepted bids are allocated resources for future consumption that are no longer available for subsequent

users to bid for. We develop the resulting allocation and payment schemes and analyze their auction properties.

4.6 Winner Determination

Since offering performance guarantees is a service in addition to users' normal mobile data plans, it will likely represent a small portion of overall operator revenue, and the operator may rather wish to maximize users' welfare. Indeed, for "public utility" goods like network resources that are competitively auctioned, Cramton [58] argues for maximizing social welfare rather than network revenue for the sake of long-term user engagement. As such, we study the winner determination problem with the intent of optimizing social welfare and evaluate the achieved revenue via simulation later. Thus, in this Multi Unit Combinatorial Auction (MUCA), the network maximizes the declared user valuations v_{it} in auction A_t subject to the resource capacity constraints over time slots, yielding the optimization problem:

$$\begin{aligned} \max_{\{x_{it}\in\{0,1\}\}} & \sum_{i=1}^{I_t} v_{it}x_{it} \\ \text{s.t.} & \sum_{i=1}^{I} \mu_{i(t+\tau)}x_{it} \leq C_t^{(\tau)},\ \tau = 1,\ldots,\phi_{\max}. \end{aligned} \tag{4.1}$$

We recognize this MUCA formulation as the NP-hard multi-dimensional knapsack problem (MKP) [81]. The dimensionality stems from the combinatorial nature of the bids, wherein they span multiple time slots (generalizable to multiple base stations and flows needing uplink/downlink capacity). Solving (4.1) is thus prohibitive for real-time network use. Many existing algorithms for fast MUCA winner determination [83, 104, 110] rely on assumptions such as bidder multi-mindedness, sub-modularity and low number of dimensions in the MKP, which do not apply to our auction model. Using approximation algorithms or other heuristics [177] to find a solution could result in significant loss of network revenue when the number of users or the bid duration increases, especially with frequently repeated auctions. More importantly, an exact solution to the MKP is required to incentivize users to bid truthfully in the auction [56]. We instead exploit the nature of *real-time flow* demands to reduce the complexity of (4.1) by considering a series of conditions on bid quantities, durations

and resource availability. We show the simplification of (4.1) for each case, gradually leading up to more realistic and less restrictive conditions. The MKP mostly reduces to the knapsack problem solvable in pseudo-polynomial time [139].

4.6.1 Bundle Generation Policy

Our first task is to define the network operator's policy for generating the set S_{it} of bundles in response to a resource request R_{it}. The operator constructs a bundle B_{it} with resources at time $t+\tau$ given by $\mu_{i(t+\tau)} = \min\left(\eta_{it}, C_t^{(\tau)}\right)$ for $\tau = 1, \ldots, \phi_{it}$, corresponding to the highest possible resource level not exceeding the request η_{it}, based on projected availability. If $\mu_{i(t+\tau)} = 0$ for any τ, no bundle is offered to that user due to severe lack of resource availability. Hence the bundle a user receives comes closest to what the user requested for the specified duration, given capacity constraints. Given this construction of B_{it}, a bundle submitted to the auction may have different resource demands at different time slots, while the original request R_{it} does not. Note that if R_{it} is feasible, then B_{it} perfectly satisfies it by construction above, and gets submitted immediately to the auction A_t. Constructed bundles may have the features defined below.

Definition 1 (Upswitch)**.** *A bundle B_{it} exhibits an **upswitch** if $\exists\, \tau \in [1, \phi_{it} - 1]$ s.t. $\mu_{i(t+\tau)} < \mu_{i(t+\tau+1)}$. The number of upswitches in B_{it} is denoted by $U_{B_{it}}$. Bundle B_{it} has an* **a/b-*upswitch*** *if $\exists\, \tau \in [1, \phi_{it} - 1]$ s.t. $a = \mu_{i(t+\tau)} < \mu_{i(t+\tau+1)} = b$.*

Definition 2 (Downswitch)**.** *A bundle B_{it} exhibits a **downswitch** if $\exists\, \tau \in [1, \phi_{it} - 1]$ s.t. $\mu_{i(t+\tau)} > \mu_{i(t+\tau+1)}$. The number of downswitches in B_{it} is denoted by $D_{B_{it}}$. Further, bundle B_{it} has an* **a/b-*downswitch*** *if $\exists\, \tau \in [1, \phi_{it} - 1]$ s.t. $a = \mu_{i(t+\tau)} > \mu_{i(t+\tau+1)} = b$.*

4.6.2 Reduction to Tractable Optimization Problems

We first consider a network capacity projection that increases monotically over time. That is, if the resource availability projection $C_t^{(\tau)}$ for future timesteps $\tau \in [1, \phi_{\max}]$ shows no decline within that time period, then the auction round A_t satisfies this condition and is said to exhibit property $\mathbb{P}_1(t)$. This would hold, for instance, in any time slot t where no sessions carry over from previous auctions. The entire network's resource capacity is then available equally at t for all future time slots. If A_t exhibits $\mathbb{P}_1(t)$, then there can be no downswitches in any bundles submitted to this auction given our bundle generation policy, i.e., $D_{B_{it}} = 0\ \forall i$. We now show several

simplifications possible to (4.1) when $\mathbb{P}_1(t)$ holds, and also determine conditions under which $\mathbb{P}_1(t)$ is guaranteed to hold.

Definition 3 (Uniform quantity bid)**.** *A bid b_{it} is a **uniform quantity bid** on a bundle B_{it} if the corresponding resource levels μ_{it} are all equal, i.e., if $\forall\tau \in [1, \phi_{it} - 1], \mu_{i(t+\tau)} = \mu_{i(t+\tau+1)}$.*

Theorem 1. *If $\mathbb{P}_1(t)$ holds and each bid to auction A_t is a uniform quantity bid, then the outcome $\{x_{it}\}$ that solves (4.1) with only the capacity constraint at $\tau = 1$ is the solution to (4.1), reducing it to a knapsack problem. Further, $\mathbb{P}_1(t+1)$ is guaranteed to hold.*

Proof. Given $\mathbb{P}_1(t)$ and uniform quantity bids, no time slot is more constrained than the first timeslot, and the occupancy of each bid is the same for its entire duration. Hence, the solution found by the reduced knapsack problem is also the solution to (4.1). Let c_τ be the total resulting consumption of A_t's winnings bids in time-step $t + \tau$ for $\tau = 1 \ldots \phi_{max}$. The resource availabilities of A_t and A_{t+1} are then related as $C_{t+1}^{(\tau)} = C_t^{(\tau+1)} - c_\tau$. Each winning bid i expresses the same μ_i across all its time-slots of consumption, since all submitted bids are uniform by definition. Therefore, the sequence $c_1, \ldots, c_{\phi_{max}}$ decreases monotonically such that $c_k - c_l = \mu_i$ whenever a ϕ_{it} value is reached at k, $k > l$. We know $C_t^{(\tau+1)}$ is monotonically increasing by $\mathbb{P}_1(t)$, and subtracting a monotonically decreasing sequence retains this property for $C_{t+1}^{(\tau)}$. □

While Theorem 1 simplifies the winner determination significantly by reducing the MKP to a single knapsack problem in one time slot, it only applies when all bids are of uniform quantity. Consider the following instance. The network has no active sessions at time t and projects $C_t^\tau = 10$, $\forall\tau$. At t, it admits two flows that consume 3 Mbps for 5 minutes and 5 Mbps for 7 minutes, respectively. At $t + 1$, the projected availability is $C_{t+1}^\tau = 2, \tau = 1, \ldots, 5$ and $C_{t+1}^\tau = 5, \tau = 6, \ldots, \phi_{max}$. A user requesting 4 Mbps for 10 minutes at $t + 1$ would thus receive a bundle granting 2 Mbps for the first 4 minutes and 3 thereafter. This bundle exhibits an upswitch and is therefore not uniform. The network may be able to force the construction of suboptimal uniform bundles if this is a resonable restriction for some applications or use cases. However, upswitches are likely due to varying availability constraints and data consumption patterns.

Lemma 2. *If a bundle B_{it} exhibits an a/b-upswitch at time τ, then any bundle in A_t corresponding to a request $R_{i't}$ with $\eta_{i't} \geq b$ exhibits an a/b'-upswitch with $b' \geq b$ at time τ.*

Proof. Given a bundle B_{it} with $a = \mu_{i(t+\tau)} < \mu_{i(t+\tau+1)} = b$, the bundle generation policy implies $C_t^{(\tau)} = a$, $\eta_{it} \geq b$, $C_t^{(\tau+1)} \geq b$. Hence, for any other bid $b_{i't}$ with $\eta_{i't} \geq b$, we have $\mu_{i'(t+\tau+1)} = \min\left(\eta_{i't}, C_t^{(\tau+1)}\right) \geq b$, $\mu_{i'(t+\tau)} = a$. □

We now derive results that guide the network operator in defining its operating bitrate modes such that the MKP can be simplified even with upswitches. First, if the quantity expressed by each supported mode is equally spaced, we called these *evenly dispersed modes.*

Definition 4 (Evenly dispersed)**.** *A set of modes $\{m_1, \ldots, m_M\}$ is **evenly dispersed** if $\exists y, z \in \mathbb{N}$ s.t. $m_n = z(n + y),\ \forall n \in \{1, M\}$.*

Theorem 2. *If the auction modes are evenly dispersed, then the outcome $\{x_{it}\}$ that solves (4.1) with only the capacity constraint at $\tau = 1$ is the solution to (4.1), reducing it to a knapsack problem. Further, $\mathbb{P}_1(t)$ holds for all t if auction modes are evenly dispersed.*

Proof. We first show that if $\mathbb{P}_1(t)$ holds and modes are evenly dispersed, solving (4.1) in the first time-slot is sufficient to find the optimal solution. Suppose $\{x_{it}\}$ is the optimal solution to the knapsack problem at $\tau = 1$. If none of the accepted bids corresponds to a bundle with an upswitch, i.e., $\nexists B_{it}$ s.t. $x_{it} = 1, U_{B_{it}} > 0$, then all of the bids have uniform quantity and the result reduces to Theorem-1. If, however, an accepted bid corresponds to a upswitched bundle B_{it}, then $x_{i't} = 0$ for all $i' \neq i$, i.e., all other bids are denied. Otherwise, $C_t^{(\tau)} \geq \mu_{it} + m_1$, in which case the excess m_1 would have been provided to the accepted bid. The bundle generation policy ensures that this bid's bundle satisfies the capacity constraints. To show that $\mathbb{P}_1(t)$ always holds when modes are evenly dispersed, we need only show that $\mathbb{P}_1(t)$ holds when bundles with upswitches are present. We note that the existence of bundle upswitches implies $C_t^{(\tau)} < m_M$. Suppose that $C_t^{(1)} = m_n = z(n + y)$ with $n = 1, \ldots, M - 1$, i.e., the

capacity is equal to one of the modes, and that an accepted bundle B_{it} has an upswitch at time slot $t+\tau'$. Then $C_{t+1}^{(\tau)} = 0$ for $\tau \leq \tau'$, as the bundle generation policy assigns the highest possible mode at each time step. Since capacity at any time-slot cannot be negative, $\mathbb{P}_1(t)$ holds for A_{t+1} in this case. Now suppose $C_t^{(1)} = m$ is not equal to one of the modes, i.e., $\nexists n, y, z$ s.t. $m = z(n+y)$. Then, $C_{t+1}^{(\tau)} = \text{argmin}_b\, m - bz$, since upswitch at time slot $t+\tau'$ indicates that the bid will consume the highest possible mode leading up to $t+\tau'$. Since incremental capacity at $t+\tau'$ that causes the upswitch will be a multiple of z since it results from the end in consumption of previously accepted bids, this capacity will continue to reside at $t+\tau'$. Hence, $\mathbb{P}_1(t)$ holds for A_{t+1} in this case as well. Iterative application of this result completes the proof. □

Theorem 2 allows the network operator to support different operating bitrates while solving (4.1) with a single knapsack. Specifically, if the supported modes were evenly dispersed, *e.g.,* 2, 4 and 6 Mbps, then it is sufficient in every auction to solve (4.1) in the first time slot. The operator can choose the exact operating modes by examining those required by target applications and its ability to reserve resources. However, certain real-time applications may not lend themselves to this, e.g., Skype has discrete modes with unevenly dispersed bitrate requirements[6], and the network may therefore offer arbitrary modes to serve these applications. We first note that even in this case, the upswitch count $U_{B_{it}}$ cannot exceed $M-1$ as long as $\mathbb{P}_1(t)$ holds.

Theorem 3. *If $\mathbb{P}_1(t)$ holds, modes are not evenly dispersed, and upswitches occur at time slots $\tau_1, \tau_2, \ldots, \tau_k$ (as in Lemma 2), where $k \leq M-1$, then restricting the capacity constraint in (4.1) to these time slots, along with the first time slot, yields the overall optimal solution. However, $\mathbb{P}_1(t+1)$ need not hold.*

Proof. The result essentially follows from Lemma 2, noting that along with the upswitch time slots, the first time slot must also be checked to satisfy capacity constraints. To show that $\mathbb{P}_1(t+1)$ need not hold any longer, we construct a counterexample. Let m_1 and m_2 correspond to supported bitrates of 1 and 3 Mbps, and suppose a previously scheduled 1 Mbps session is scheduled to end before time slot $t+2$, i.e.,

[6] http://download.skype.com/share/business/guides/skype-connect-requirements-guide.pdf

$C_{t-1}^{(1)} = 2, C_{t-1}^{(2)} = 2, C_{t-1}^{(3)} = 3$. A request at $t-1$ for 3 Mbps with $\phi_{it} = 3$ would yield a bundle with $\mu_{it} = 1, \mu_{i(t+1)} = 1$, $\mu_{i(t+2)} = 3$. Upon acceptance of this bid, the new capacity projection is not monotonic as $C_t^{(1)} = 1, C_t^{(2)} = 0$. □

Theorem 3 shows that even with arbitrarily defined modes, *the complexity of the winner determination scales only with the number of modes supported and not the number of time slots*, as long as the availability projection at t increases monotonically. However, there is no guarantee that capacity projections will continue to be monotonically increasing for future auctions.

Algorithm 2: ComputeConstrainedTimeslots - Pruning time slots under uniform quantity bids ($\mu_{it} = \mu_i$) when $\mathbb{P}_1(t)$ does not hold. **W** is set of bids

```
.
Input  : W, C, t, φ_max
Output : An array of positive integers representing timeslots
slots[0, :] ← [0]
consSlotInInterval ← 0; consValue ← 0;
for τ ← 1, ..., φ_max do
    sumAllAsks ← 0
    endInterval ← 0
    for i ∈ W do
        sumAllAsks ← sumAllAsks + μ_i
        if φ_i == (t + τ) then
            endRegion ← 1
    currSlotConstraint ← sumAllAsks / C_t^(τ)
    if currSlotConstraint > consValue then
        consValue ← currSlotConstraint
        consSlotInInterval ← τ
    if endInterval == 1 then
        slots = [slots, τ]
        consSlotInInterval ← 0
        consValue ← 0
return slots
```

We can simplify the MKP without $\mathbb{P}_1$ if bids are uniform quantity.

Theorem 4. *If $\mathbb{P}_1(t)$ does not hold, but bids are of uniform quantity, solving (4.1) in only the time slots given by Algorithm 1 yields the optimal solution. Further, if all bids are of equal duration, then Algorithm 1 reduces to a single knapsack problem, solved for the time slot with the largest ratio of requested to available capacity (i.e., $\sum_{i:b_{it}\neq\varnothing} \mu_{it}/C_t^{(1)}$).*

Proof Sketch. Let us first consider the special case where the uniform quantity bids are all of the same duration. Uniform quantity bids imply that $\mu_{it} = \mu_i, \forall t, i \in W$. Further since bids have the same duration, the requested resources in each time slot are uniformly equal to the sum of $\sum_{i\in W} \mu_i$. The capacity constraint thus reduces to the single inequality

$$\sum_{i\in W} \mu_i x_{it} \leq \min\{C_t^{(\tau)} : \tau = 1, \ldots, \phi_{\max}\}$$

as the left side of the constraint no longer depends on τ. Hence, the knapsack for any most-constrained time slot (there may be multiple) yields the optimal solution. Algorithm 1 computes the most constrained time slot for each interval with overlapping bids. Suppose bids overlap during $[t, t+\delta_1]$, then sessions expire at $t+\delta_1$, then the remaining bids overlap during $[t+\delta_1+1, t+\delta_2]$, and so on for arbitrary δ_k. The constraint for (4.1) in each interval $[t+\delta_k+1, t+\delta_{k+1}]$ need only be enforced at the most-constrained time slot, yielding optimality of (4.1) *for that interval.* Algorithm 1 computes this time slot for each interval, and hence solving (4.1) restricted to these time slots is optimal. □

Algorithm 1 iterates over each time-slot $\tau \leq t+\phi_{max}$. If a submitted bid(s) is scheduled to finish consumption at τ, it finds the time-slot with the largest ratio of requested to available capacity between τ and the last time-slot when a submitted bid ended. These timeslots are used to solve (4.1). Hence, if bids are uniform quantity, the dimensionality of (4.1) scales with the variance in bid durations, not the number of time slots, leading to relatively fast solutions even for large ϕ_{max}.

4.7 Incentive Compatibility

We have shown several ways for the network to simplify the winner determination in (4.1), making it feasible to optimize for social welfare in real time. This allocation objective, in conjunction with carefully designed payment schemes, can induce strong properties. We first consider a single auction A_t in isolation, and induce a *myopic* notion of truthfulness using the VCG mechanism. We then frame A_t in the context of repeated auctions, where we account for the impact of decisions made in A_t on subsequent auctions. In both cases, we *ensure dominant strategy incentive compatibility while inducing desirable properties that are often challenging to achieve simultaneously, such as revenue monotonicity and ex post individual rationality.*

4.7.1 Myopic Truthfulness

The VCG mechanism has gained wide popularity in its ability to guarantee socially optimal results through dominant strategy incentive compatibility (DSIC); i.e., every bidder's best interest is to bid truthfully, regardless of the strategies of other bidders [108]. Since the network maximizes social welfare, it can implement the VCG mechanism by charging auction winners their *social cost*. The social cost of each bidder i is computed as the difference between the maximum feasible welfare without i and the welfare to others given i's presence, i.e., $\max_{x_{it}\in[0,1]} \sum_{k=1,k\neq i}^{k=I_t} v_{kt}x_{kt} - \sum_{k=1,k\neq i}^{k=_t} v_{kt}x^*_{kt}$, where x^*_{kt} represents the optimal solution with i present.

When applied in combinatorial auctions, however, the VCG mechanism is known to exhibit undesirable failures in *bidder revenue monotonicity* [145], meaning the network's revenue from VCG payments may in fact *decrease* when some bids enter the system. An auction is said to be robust for a set of bidders Δ under VCG payments p_{it} if

$$\forall j \in \Delta \sum_{i\in\Delta} p_{it}(v_{it}, \Delta) \geq \sum_{i\in\Delta\setminus\{j\}} p_{it}(v_{it}, \Delta \setminus \{j\}). \tag{4.2}$$

See Rastegari *et al.* [145] for a more formal treatment of revenue monotonicity. Another type of VCG failure in combinatorial auctions is *goods revenue monotonicity failure* [124], when the operator could increase revenue by not auctioning certain goods (in our case, resource quantities and time slots), hence acquiring an incentive to hide goods from bidders. Most prior work on combinatorial auction frameworks

does not address the issue of VCG-induced monotonicity failures, which are especially challenging to manage in MUCA settings such as ours. We show, however, that under certain conditions applying VCG payments is guaranteed to result in revenue monotonicity. To do this, we rely on the property of *bidder submodularity* [34, 145] which builds on the maximum social welfare $V(\Delta)$ of a set of bidders Δ, corresponding to the objective in (4.1) restricted to Δ. Bidder submodularity holds for bidder sets Δ and Δ' with $\Delta \subseteq \Delta'$ if and only if $\forall i\, V(\Delta \bigcup \{i\}) - V(\Delta) \geq V(\Delta' \bigcup \{i\}) - V(\Delta')$.

Theorem 5. *If (4.1) can be solved in a single time slot t' (e.g., as in Thm 1), and $\mu_{it'} = \mu_{i't'}\, \forall i, i' \in [1, I_t]$, then A_t is guaranteed to be revenue monotonic in bidders under VCG payments.*

Proof. In this scenario, the winner determination problem reduces to choosing the bids with the highest valuations in the first time-slot. Let Δ be a set of bidders. It suffices to show that

$$V(\Delta) - V(\Delta \setminus \{i\}) \geq V(\Delta \cup \{j\}) - V(\Delta \cup \{j\} \setminus \{i\}) \tag{4.3}$$

for bidders i and j; by induction, (4.3) implies the above form of bidder submodularity. Let $A(\Delta)$ be the set of winning bidders, i.e., for a solution $\{x_{it}\}$ to (4.1), $A(\Delta) = \{i : x_{it} = 1\}$. We consider four cases. First, if $j \notin A(\Delta \bigcup \{j\})$ and $j \notin A(\Delta \bigcup \{j\} \setminus \{i\})$, (4.3) holds trivially with equality. Second, if $j \notin A(\Delta \bigcup \{j\})$ and $j \in A(\Delta \bigcup \{j\} \setminus \{i\})$, then $V(\Delta) = V(\Delta \cup \{j\})$ and (4.3) holds if $V(\Delta \setminus \{i\}) \leq V(\Delta \bigcup \{j\} \setminus \{i\})$, which must hold since $A(\Delta \setminus \{i\})$ solves the knapsack problem as solved by $A(\Delta \bigcup \{j\} \setminus \{i\})$. Third, we can never have $j \in A(\Delta \bigcup \{j\})$ and $j \notin A(\Delta \bigcup \{j\} \setminus \{i\})$, by similar reasoning as previous. Fourth, consider $j \in A(\Delta \bigcup \{j\})$ and $j \in A(\Delta \bigcup \{j\} \setminus \{i\})$. First consider the case $v_i > v_j$. Then, $j \in A(\Delta \bigcup \{j\})$ implies $i \in A(\Delta \bigcup \{j\})$, and $j \in A(\Delta \bigcup \{j\} \setminus \{i\})$ implies $i \in A(\Delta)$ since the quantity expressed in i and j are the same by definition. Hence the removal of i in the RHS can, at worst, make no difference if no bids are left to accept, and at best, admit another bid of the same size with the next highest valuation. The LHS has the same choice, hence (4.3) holds since the value of the second highest bid after i is the same in both cases. Next consider $v_i < v_j$. If $i \in A(\Delta)$ and $i \in A(\Delta \bigcup \{j\})$, then equality holds in (4.3). If $i \in A(\Delta)$ and $i \notin A(\Delta \bigcup \{j\})$,

then the RHS of (4.3) is 0, hence satisfying (4.3). Finally, if $i \notin A(\Delta)$, both the LHS and RHS of (4.3) evaluate to 0. Note that (4.3) holds trivially when $v_i = v_j$. □

In this scenario, winner determination is a knapsack problem where bids request the same resources but potentially different valuations. Then, all bids compete equally for capacity, and therefore, removing a bid cannot increase another's social cost, resulting in revenue monotonicity.

Lemma 3. *In our auction, bidder revenue monotonicity implies goods revenue monotonicity.*

In our auction, bidders desire and bid on exactly one bundle, a property referred to as single mindedness. When single-minded bidders exhibit bidder revenue monotonicity, they are goods revenue monotonic as well [124]. Thus, by exploiting the structure of users' real-time resource requests, we have shown that under reasonable conditions, users have an incentive to bid truthfully and the network operator has no incentive to discourage bids from users or hide resources, as doing so will not increase its revenue. However, we also note the following limitation.

Lemma 4. *As long as the auctioneer solves (4.1) for winner determination of A_t and charges winners their social cost, bidders may have an incentive to submit a false session duration ϕ_{it}.*

Consider a case where $\mathbb{P}_1(t)$ holds and all bids in A_t are uniform quantity. The auctioneer then only solves the knapsack problem in the first time-slot (Thm. 1), and users' choice of ϕ_{it} has no impact on their bid allocations or payments. Indeed, as seen earlier, even when $\mathbb{P}_1(t)$ does not hold and results in arbitrary up/downswitches, the winner determination and hence payments depend only on the time slots of these switches. Hence, maximizing the social welfare at A_t only with respect to A_t does not directly incentivize truthfulness in declaration of ϕ_{it}.

4.7.2 Truthfulness Amidst Temporal Correlations

In our setting, the true social cost of a bit b_{it} is not only a function of other bids submitted to the current auction (as discussed earlier). Selecting a bid b_{it} as a winner of auction $A(t)$ directly reduces available capacity in the next ϕ_{it} time slots, which impacts the bids that can be accepted in subsequent auctions A_j, $j \in [t+1, t+\phi_{it}]$. To account for this *temporal correlation* between these periodic auctions, we are faced with the challenge of factoring in the uncertainty in future bids in admitting the present bids. *We thus develop mechanisms to push this uncertainty either to the user or the network, inducing different properties accordingly.*

The *temporal trickle effect* of a winning bid may in fact extend beyond its duration; for instance, allocating resources for a bid b_{it} might preclude allocation for a bid $b_{j(t+\phi_{it}-1)}$, which might in turn allow for the allocation of a bid at $t+\phi_{it}+1$ that would have been infeasible had j's bid been allocated. We argue, however, that it is unreasonable to charge users their social cost beyond the duration ϕ_{it} (unlike the treatment by Parkes *et al.* [135]). First, it is extremely challenging to predict and model the trickle effects starting from allocation of a bid until the last auction in the system, leading to significant computational overhead and possible infeasibility. Second, since mobile network use is dense and diversified, the extended effects of a single bid in the system would arguably be too little to cause a significant impact in the overall social welfare and hence not worth accounting for. Hence we propose to hold user i accountable for their "first-order" social cost with respect to arriving bids during $[t, t+\phi_{\max}]$, hence capturing direct impact during i's consumption and any immediate ripples until $t+\phi_{\max}$.

We now formulate strategies that induce desirable properties despite this temporal correlation amongst auctions and future uncertainty. We first provide definitions of these properties (see [108] for a thorough treatment). **Individual Rationality** is achieved when no bidder receives a negative utility from participating in the auction, i.e., no winning bid is charged more than its reported value v_{it} and no losing bid is charged. The winner determination is **Allocatively Efficient** when social welfare is maximized in the allocation outcomes. If the sum of all payments charged by the auctioneer is non-negative, i.e., the auctioneer does not suffer a net loss, then the mechanism is (weakly) **Budget Balanced**. A property is said to hold **ex ante** if it holds in expectation over the private and unknown information of all bidders, **ex**

interim if it holds when a bidder knows their private information but others only in expectation, and **ex post** if it is guaranteed to hold even when all bidder parameters are revealed. We now develop allocation and payment schemes that navigate trade-offs in these properties by factoring in future bid uncertainty differently.

4.7.2.1 Maximize Expected Social Welfare

Let us consider an allocation strategy alternate to (4.1) to determine winners of auction A_t. Let $\mathbb{O}_t$ be the set of all feasible allocations of A_t. Then

$$o_t^* = \underset{\mathbb{O}_t}{\operatorname{argmax}} \sum_{i=1}^{I_t} v_{it}(o_t) + \sum_{j=t+1}^{t+\phi_{\max}} \mathbb{E}_{D(j)}^{t} \sum_{k=1}^{I_j} [v_{kj}(o_{tj})], \tag{4.4}$$

where I_j is the number of bidders placing bids in the system at time j and $\mathbb{E}_{D(j)}^{t}$ denotes the expectation at time t taken over the distribution $D(j)$ of bids at j (consisting of bid arrivals, requested mode in the bid, duration and valuation). With this allocation rule, the network maximizes the expected social welfare of the next $\phi_{\max}$ time steps in deciding the allocation, rather than maximizing only for the welfare of bidders at A_t. This is implicit in the dependence between o_t and o_{tj}, wherein the latter captures the allocation decision taken at time t *for the timestep* j *in the estimated look-ahead model*. Point estimates derived from, for example, Monte Carlo sampling of outcomes starting from t can be used for unknown parameters of future timesteps [141]. The approach in (4.4) presumes that the network has learned this distribution of bids spanning the next $\phi_{\max}$ time steps. Indeed, computing a deterministic $\phi_{\max}$-step look-ahead model is far more feasible than computing the optimal solution for the multi-stage stochastic programming problem of expected welfare maximization for all remaining auctions [141, 151]. Models requiring computation of the optimal value function at every time-step [135] pose severe feasibility challenges. Further, the time period $\phi_{\max}$ intuitively lends itself as a reasonable look-ahead period since all user allocations starting at t must end by then, providing a standard and relatively short time window for computing prices.

4.7.2.2 Charge Expected Social Cost

We now design payment rules which operate in conjunction with the allocation rule in (4.4) to induce desirable auction properties. First, we consider a rule similar to

above wherein a winning bidder i is charged its expected social welfare cost p_{it} as the difference between maximum welfare without i and welfare to others given i's presence, i.e.,

$$p_{it} = W_t^{-i} - \left(\sum_{l=1, l \neq i}^{I_t} v_{lt}(o_t^*) + \sum_{j=t+1}^{t+\phi_{max}} \sum_{k=1}^{I_j} \mathbb{E}_{D(j)}^t [v_{kj}(o_{tj}^*)] \right), \tag{4.5}$$

where W_t^{-i} represents the welfare without i as

$$W_t^{-i} = \max_{\mathbb{O}_t} \left(\sum_{l=1, l \neq i}^{I_t} v_{lt}(o_t^{-i}) + \sum_{j=t+1}^{t+\phi_{\max}} \sum_{k=1}^{I_j} \mathbb{E}_{D(j)}^t [v_{kj}(o_{tj}^{-i})] \right), \tag{4.6}$$

o_t^{-i} represents an allocation outcome at time t without i in the system, and o_t^* is the optimal solution with i present, with look-ahead model decisions o_{tj}^*. Note that we assume users have quasi-linear utility functions, as ubiquitously done [108].

Theorem 6. *The mechanism implementing the allocation rule in (4.4) and the payment rule in (4.5) is DSIC in all bid parameters. Further, it is ex post individually rational and budget balanced.*

Proof. The user's utility may be written as:

$$\begin{aligned} u_{it} =& v_{it}(o_t^*) + \big(\sum_{l=1, l \neq i}^{I_t} v_{lt}(o_t^*) + \sum_{j=t+1}^{t+\phi_{max}} \sum_{k=1}^{I_j} \mathbb{E}_{D(j)}^t [v_{kj}(o_{tj}^*)] \big) - \\ & \max_{o_t \in \mathbb{O}} \big(\sum_{l=1, l \neq i}^{I_t} v_{lt}(o_t) - \sum_{j=t+1}^{t+\phi_{max}} \sum_{k=1}^{I_j} \mathbb{E}_{D(j)}^t [v_{kj}(o_{tj})] \big) \end{aligned} \tag{4.7}$$

The user wishes to state possibly untrue $\eta'_{it}, \phi'_{it}, v'_{it}$ to maximize u_{it}. The third term in u_{it} can be ignored since it is independent of the user's bid and hence the user has

no influence over it.

$$\begin{aligned}\max_{\eta'_{it},\phi'_{it},v'_{it}} u_{it} &= \max_{\eta'_{it},\phi'_{it},v'_{it}} v_{it}(o_t^*) \\ &\quad + \big(\sum_{l=1,l\neq i}^{I} v_{lt}(o_t^*) + \sum_{j=t+1}^{t+\phi_{max}} \sum_{k=1}^{I_j} \mathbb{E}^t_{D(j)}[v_{kj}(o_{tj}^*)]\big) \\ &= \max\big(\sum_{l=1}^{I_t} v_{lt}(o_t^*) + \sum_{j=t+1}^{t+\phi_{max}} \sum_{k=1}^{I_j} \mathbb{E}^t_{D(j)}[v_{kj}(o_{tj}^*)]\big)\end{aligned} \tag{4.8}$$

However, the outcomes $o_t*, \ldots, o_{tj}^*$ that the user wishes to maximize their utility over *is computed by design to maximize the term in (4.8), as specified in (4.4).* Hence, regardless of the actual distributions of agent bids or their revealed valuations in subsequent timesteps, the user maximizes its utility by revealing its true parameters since the network maximizes on this distribution in its allocation rule. Individual rationality follows trivially. The user's payment will always be less than or equal to their valuation. □

By simply maximizing social welfare in expectation of $[t, t + \phi_{\max}]$ and charging winning bids their expected social cost, the network can not only incentivize dominant strategy truthfulness in η_{it}, ϕ_{it} and v_{it}, but also ensure that no bidders are charged more than what they bid for. We now introduce a new property to evaluate this payment scheme.

Definition 5 (Payment Efficient)**.** *A mechanism is said to exhibit* ***Payment Efficiency*** *if it charges winners their social cost. For instance, the VCG mechanism is payment efficient, since winning bids are charged the difference in social welfare to others due to their presence.*

Lemma 5. *The mechanism implementing the allocation rule in (4.4) and the payment rule in (4.5) is ex ante allocatively efficient and ex ante payment efficient at t for the time-period $[t, t + \phi_{\max}]$.*

While this mechanism maximizes expected social welfare, it may well be the case that the auctioneer under-predicts demand between $[t, t + \phi_{\max}]$ in retrospect which yields the allocation and payment decisions made earlier suboptimal. The auctioneer's

revenue would then be higher if winning bids at t were charged their social cost at $t + \phi_{\max}$ after observing the actual demand.

4.7.2.3 Charge Realized Social Cost

The allocation decision for A_t must be made in real-time for immediate session needs. However, we realize that *payments need not be computed or charged in real time for users to start their sessions.* Consider the payment rule given by the actual after-the-fact difference between welfare without and with i after $t + \phi_{\max}$ time steps have elapsed:

$$p_{it}^{t+\phi_{\max}} = \max_{\mathbb{O}_t} \sum_{j=t}^{t+\phi_{\max}} \sum_{k=1,k\neq i}^{I_j} \left(v_{kj}(o^{-i}) - v_{kj}(o^*) \right), \tag{4.9}$$

where o^* is the optimal allocation of users other than i given by

$$o^* = \operatorname*{argmax}_{\mathbb{O}_{it}} \sum_{j=t}^{t+\phi_{\max}} \sum_{k=1,k\neq i}^{I_j} v_{kj}(o), \tag{4.10}$$

and $\mathbb{O}_{it}$ is the set of all feasible allocations given fixed allocation for i. Admitted users are now charged at $t + \phi_{\max}$, by which time all bids starting consumption at t are guaranteed to end. The auctioneer may now use its retrospective knowledge of bids that came in from t to $t + \phi_{\max}$ to calculate the exact first-order social cost for each bid at t. Before analyzing the unique properties that this rule yields, we first define a modified notion of DSIC.

Definition 6 (DSICE)**.** *Consider bidders that maximize expected future utility. If truthful revelation maximizes the expected future utility of bidders, regardless of the strategy of other bidders, then the mechanism is said to be* ***Dominant Strategy Incentive Compatible in Expectation****.*

Theorem 7. *The mechanism implementing the allocation rule in (4.4) and the payment rule in (4.9) is DSICE in all bid parameters, ex interim individually rational and ex post budget balanced.*

Proof. The user's utility u_{it} *evaluated at* $t + \phi_{max}$ after p_{it} is charged, is:

$$\begin{aligned} u_{it}^{t+\phi_{max}} &= v_{it}(o_t^*) - p_{it} \\ &= v_{it}(o_t^*) + \sum_{j=t}^{t+\phi_{max}} \sum_{k=1,k\neq i}^{I_j} v_{kj}(o') - \\ &\max_{o\in O} \sum_{j=t}^{t+\phi_{max}} \sum_{k=1,k\neq i}^{I_j} v_{kj}(o) \\ &\text{where, } o' = \operatorname*{argmax}_{o\in O} \sum_{j=t}^{t+\phi_{max}} \sum_{k=1}^{I_j} v_{kj}(o) \end{aligned} \tag{4.11}$$

The user wishes to state v'_{it}, η'_{it} and ϕ'_{it} *at time* t to maximize its utility. Since future payments are unknown at current time, the user instead maxmizes expected utility at t.

$$\begin{aligned} \mathbb{E}^t[u_{it}^{t+\phi_{max}}] =& v_{it}(o_t^*) + \mathbb{E}^t[\sum_{j=t}^{t+\phi_{max}} \sum_{k=1,k\neq i}^{I_j} v_{kj}(o')] + \\ & \mathbb{E}^t[\max_{o\in O} \sum_{j=t}^{t+\phi_{max}} \sum_{k=1,k\neq i}^{I_j} v_{kj}(o)] \end{aligned} \tag{4.12}$$

The last term is entirely independent of the agent's bid, hence the maximization of the agent's utility is:

$$\begin{aligned} \max_{\eta'_{it},\phi'_{it},v'_{it}} \mathbb{E}^t[u_{it}^{t+\phi_{max}}] &= \max_{\eta'_{it},\phi'_{it},v'_{it}} v_{it}(o_t^*) + \mathbb{E}^t[\sum_{j=t}^{t+\phi_{max}} \sum_{k=1,k\neq i}^{I_j} v_{kj}(o')] \\ &= \max_{\eta'_{it},\phi'_{it},v'_{it}} v_{it}(o_t^*) + \\ &\sum_{k=1,k\neq i}^{I_t} v_{kt}(o') + \mathbb{E}^t[\sum_{j=t+1}^{t+\phi_{max}} \sum_{k=1}^{I_j} v_{kj}(o')] \\ &= \max_{\eta'_{it},\phi'_{it},v'_{it}} \sum_{k=1}^{I_t} v_{kt}(o_t^*) + \sum_{j=t+1}^{t+\phi_{max}} \sum_{k=1}^{I_j} \mathbb{E}^t[v_{kj}(o_{tj}^*)] \end{aligned} \tag{4.13}$$

where we go from step 2 to 3 by seeing that the expectation at time t of o' is, in-fact, what the network calculates in (4.4) at time t. We see from the final expression that

Payment Scheme	**DSIC**	**Payment Efficiency**	**Individual Rationality**	**Allocative Efficiency**	**Budget Balance**
Expected Social Cost	✓	Ex Ante	Ex Post	Ex Ante	Ex Post
Realized Social Cost	In Expectation	Ex Post	Ex Interim	Ex Ante	Ex Post

Table 4.1: We summarize trade-offs between charging expected social cost at t and realized social cost at $t + \phi_{\max}$.

the o_t^* that the user wishes to induce to maximize collective expected welfare is what the network maximizes as well in (4.4). Hence, for expected-utility maximizers, it is the dominant strategy to reveal their truthful parameters to the network, since the user wishes to maximizes over the same distribution of future bids that the network does in its allocation, and future payment. Since actual payments to user at $t + \phi_{max}$ may hence be more than what they bid, but the user's truthful bidding is nonetheless the best strategy in expectation over the future, the users are *interim individually rational.* □

By charging bidders their true social cost at $t + \phi_{\max}$ based on actual bids that arrived after t, the mechanism essentially *shifts the risk of demand under-prediction* to the bidder. However, winning bidders now bear the risk of being charged more than their bid.

Lemma 6. *The mechanism implementing the allocation rule in (4.4) and payment rule in (4.9) is ex ante allocatively efficient and ex post payment efficient with respect to bids at t for $[t, t + \phi_{\max}]$.*

As we allocate using (4.4), ex ante allocative efficiency holds. By design of (4.9) winning bids pay their true social cost at $t + \phi_{\max}$ and hence the network is also payment efficient. However, if the auctioneer over-predicts resource demand between $[t + 1, t + \phi_{\max}]$, charging users their actual social cost at $t + \phi_{\max}$ yields less revenue than charging them their expected cost at t.

We have proposed three distinct payment mechanisms for our auction model: traditional VCG, paying the expected social cost, and paying the realized social cost. Since the latter two mechanisms require knowledge of future bids, the network would

likely introduce VCG payments first. By evaluating each A_t in isolation (i.e., using (4.1) and VCG payment), the network can learn users' true bid valuations and required bitrates. By estimating bid durations with historical usage, the network may use this distribution of future bid parameters it has learned to offer the latter mechanisms. This would likely increase social welfare in the system, since the network now accounts for the impact of allocation decisions between multiple rounds of auction as in (4.4). In choosing between the two payment rules as in Table 4.1, the network must make a design choice. It may either guarantee individual rationality by choosing (4.5) and assume the risk of under-predicting resource demand, or it may choose (4.9) and ensure payment efficiency while allowing winners to be charged higher than their bids. However, in the latter case, the network now assumes the risk of over-predicting resource demand.

4.8 Usability Constraints

In previous sections, we developed practical allocation and payment strategies for our auctions that achieve spontaneous resource guarantees for real-time applications. We now turn to challenges faced by end users of this system. Most data plans in the US provide known and fixed up-front pricing for the month [198], so engaging spontaneously in auctions may add uncomfortable *expense uncertainty* for the average mobile user. Further, explicitly conveying an app's resource needs and bid parameters every time the user desires guaranteed data access can be a significant deterrent. To address these usability issues, we propose to have automated agents on users' devices that act on their behalf, abstracting them away from resource specification and bidding overhead. We first formulate a user-parameterized utility framework using which agents can discover user valuations transparently for resource guarantees of specific sessions. We then propose a reinforcement learning strategy to enforce users' daily budgets.

4.8.1 Bundle Utility

We determine user i's valuation of a resource bundle $R_{it} = (\eta_{it}, \phi_{it})$. Let ψ_{it} denote the utility per unit of time consumption for the mode η_{it} requested, normalized between $[0, 1]$ across applications and pre-configured by the user. We use an α-fair

model [197] to capture diminishing returns in utility over longer session durations ϕ_{it}. The user's utility associated with request R_{it} is thus given by $U_{it}(R_{it}) = \frac{\phi_{it}^{1-\alpha}}{1-\alpha}\psi_{it}$. If the network responds with an alternate bundle B_{it} that returns a mode below that requested in R_{it}, we impose a penalty to represent the dissatisfaction in receiving a lower mode. We model this penalty for each affected time slot $t+\tau$ as a multiplicative factor $\xi_{i\tau}$, such that a higher penalty corresponds to a smaller mode. We denote $\xi_{i\tau} = 1/(1 + \psi_{it} - \psi^*_{i(t+\tau)})$, where $\psi^*_{i(t+\tau)}$ denotes user i's valuation of the mode corresponding to the offered $\mu_{i(t+\tau)}$. Since users may be especially dissatisfied if their session experiences a downswitch, we further apply a downswitch penalty that grows with the magnitude $(a-b)$ of an a/b-downswitch. We model this penalty for each τ as a multiplicative factor $\zeta_{i\tau} = \rho_i/(\mu_{i(t+\tau)} - \mu_{i(t+\tau+1)})$ when $\mu_{i(t+\tau)} > \mu_{i(t+\tau+1)}$, where $\rho_i \in [0, 1]$ is user-specific. The utility of B_{it} relative to that of R_{it} is then given by

$$U_i(B_{it}|R_{it}) = \frac{\phi_{it}^{1-\alpha}}{1-\alpha} \sum_{\tau=1}^{\phi_{it}} \xi_{i\tau}\zeta_{i\tau}\psi^*_{i(t+\tau)}. \tag{4.14}$$

4.8.2 Budget Constraints

Building on (4.14), we develop an algorithm for the user agent to satisfy a daily budget constraint while placing bids that are proportional to the user's true utility U_i. If agents distribute budgets poorly (as some of the naïve algorithms demonstrated later do), users consistently lose in their auctions of interest, hence forming the false impression that the market rate is prohibitively high and exiting the system. The budget distribution problem relies on a policy to select a valuation v_{it} to declare on a given bundle B_{it} (interchangeably R_{it}) that maximizes the user's total expected future utility, subject to the budget constraint. Without loss of generality, we collapse the distinction between B_{it} and R_{it} by considering a virtual round where the network offers $B_{it} = R_{it}$ if R_{it} is perfectly available. To discover the optimal policy, we model the user environment as a Markov Decision Process (MDP) wherein actions correspond to placing bids, and the user receives a reward equal to the utility $U_i(B_{it}|R_{it})$ if the bid wins and zero otherwise. As seen in Figure 4.10, the state of user i in time slot t is defined as $\sigma_{it} = (t, \beta_{it}, B_{it}, R_{it})$, where β_{it} is the remaining budget at time t (with β_{i1} as the total daily budget). The overall system state determines the probability of winning the bid $P_{\text{win}}(\sigma_{it}, v_{it})$, which also represents the state transition probabilities.

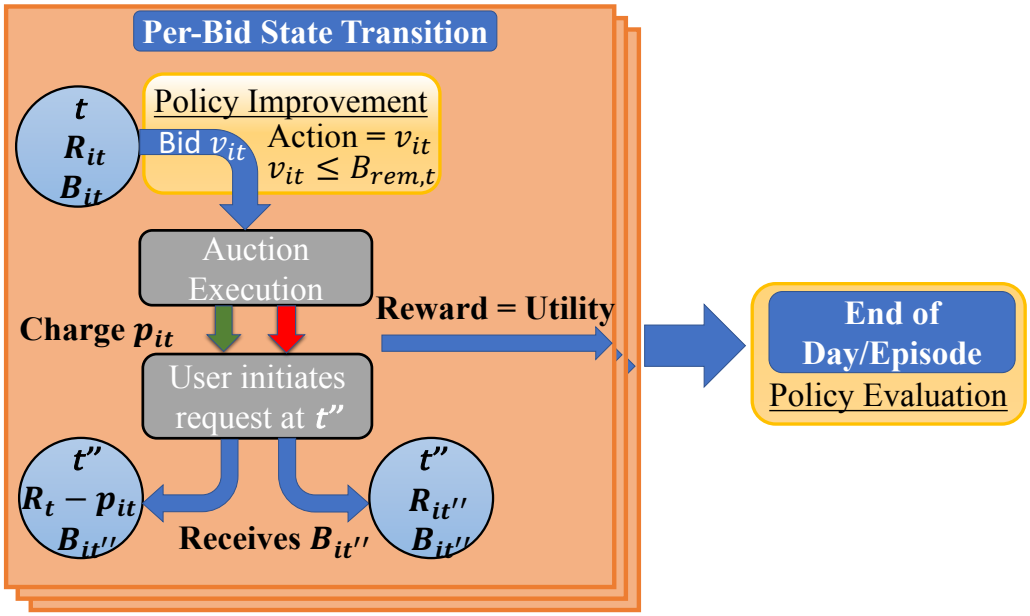


Figure 4.10: The MCPI agent bids based on the current policy. Rewards from the states encountered and the actions taken during the day are used to update the policy end of day.

The optimal budget distribution policy is then given by:

$$\begin{aligned} \max_{b_{it}} \; & U_i(B_{it}|R_{it}) P_{\text{win}}(\sigma_{it}, v_{it}) + \\ & \sum_{k=t+1}^{T} U_i(B_{ik}|R_{ik}) P_{\text{win}}(\sigma_{ik}, v_{ik}) \\ \text{s.t.} & \sum_{k=t}^{T} p_{ik} P_{\text{win}}(\sigma_{ik}, v_{ik}) \leq \beta_{it}. \end{aligned} \tag{4.15}$$

However, users cannot solve (4.15) as the environment is only Partially Observable (yielding a POMDP); they only observe their own actions and rewards and therefore cannot compute the transition probabilities $P_{\text{win}}(\sigma_{it}, v_{it})$. We hence employ a model-free reinforcement learning mechanism to determine the optimal user actions under uncertainty. The offline and episodic Monte Carlo Policy Iteration (MCPI) technique [167] is particularly suitable here as users typically exhibit periodicity in daily mobile activities and resource needs, allowing us to consider a day as an episode. MCPI seeks the optimal bidding policy $\pi^*_{it} = v^*_{it}(\sigma_{it})$ by iteratively evaluating a candidate policy π and updating the action value function $q_\pi(\sigma_{it}, v_{it})$ from episodes sampled from the POMDP. We define $q_\pi(\sigma_{it}, v_{it})$ as the return obtained by placing

bid v_{it} in state σ_{it} and then following policy π, averaged over all future states and actions. The return $G_{it} = \sum_{j=t}^{T} \lambda^j U_i(B_{ij}|R_{ij})$ for a given series of states is the total future discounted reward, where λ is a discount factor representing how much present value a user assigns to future rewards. This captures a degree of uncertainty about the future that stems from the environment as well as the user's estimate of their future session desires. At the end of each episode (e.g., day), the action-value function is updated using

$$q_\pi(\sigma_{it}, v_{it}) \leftarrow q_\pi(\sigma_{it}, v_{it}) + \chi(G_{it} - q_\pi(\sigma_{it}, v_{it})), \tag{4.16}$$

where χ is the learning rate. We follow the well-known ϵ-greedy approach [167] to balance the trade-off between exploring the environment further to know it better (i.e., choose v_{it} randomly with probability ϵ) and exploiting current knowledge of the environment to maximize current returns (i.e., chooses v_{it} to maximize q_π). A bid v_{it} cannot exceed β_{it}, the current available budget. We *anneal* epsilon to eventually always exploit after the environment has been explored sufficiently, which yields the optimal policy if the environment is periodic [167]. We set $\epsilon = 1/N(s)$, where $N(s)$ is the number of times state s is visited, resulting in continuous reduction of exploration from a state as it is visited further and guaranteeing that π_{it} approaches the optimal $v^*_{it}(\sigma_{it})$ as $N(s) \to \infty$. We subsequently show that this exploration leads to convergence in relatively few iterations and study its performance under increasing complexity.

4.9 Evaluation of Budget Distribution Strategy

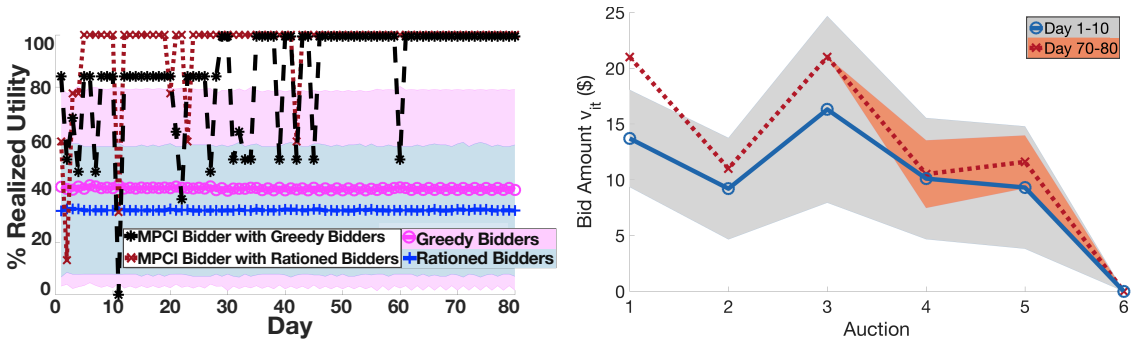
Setup: We simulate a network of 100 users that participate in auctions for performance guarantees over the course of 80 days and have predetermined schedules of resource requests R_{it} to place during the day. To allow all users to have a reasonable chance of acquiring at least some of their desired resources, we set uniform daily budgets as $\beta_{i1} = \beta$. Note that the MCPI strategy can only help each user achieve the maximum utilities for their given budget and is of little help if market rates are prohibitively high (discussed more later).

Baseline strategies: To assess the performance of MCPI-based budget distribution, we introduce two alternative strategies to compare against. Bidders of the *greedy bidding* strategy bid as much of their remaining budget as needed to achieve the

desired utility in each auction as $v_{it} = \min(U_i(B_{it}|R_{it}), \beta_{it})$. Bidders of the *rationed bidding* strategy spread their daily budgets evenly over their (known) daily resource requests, setting v_{it} to the minimum of the bid utility $U_i(B_{it}|R_{it})$ and the session budget. Any residual session budget rolls over to the next session. The maximum realizable daily utility subject to a user's daily budget is

$$U_{i,\max} = \max_{\{y_{it}\in\{0,1\}\}} \sum_{t=1}^{T} U_i(B_{it}|R_{it}) y_{it} \quad \text{s.t.} \sum_{t=1}^{T} \theta_{it} y_{it} \leq \beta_{i1}, \tag{4.17}$$

where θ_{it} is the *critical price* of the auction A_t, *i.e.,* the social cost of admitting the bid in this allocation. Since our auctions incentivize truthfulness, θ_{it} is also the payment charged to user i if their bid wins. Note that users themselves cannot compute $U_{i,\max}$ due to partial observability. We use this value as the theoretical maximum to evaluate the MCPI generated utility against.

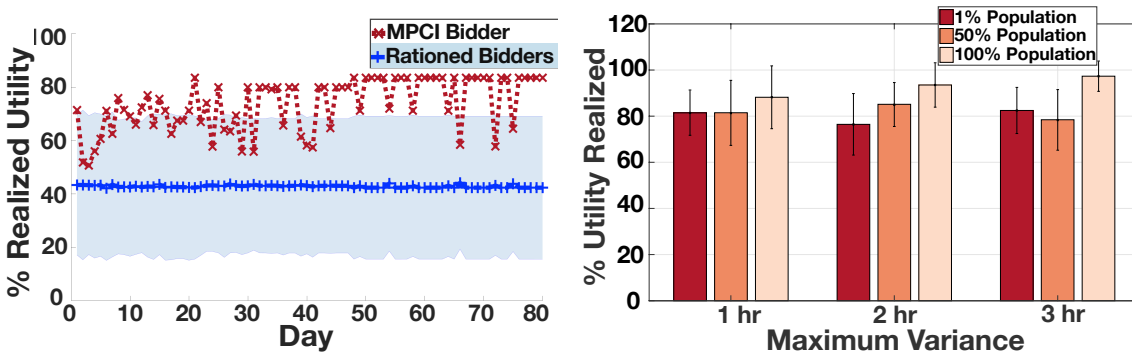


(a) User utility from bidding

(b) MCPI bidding strategy

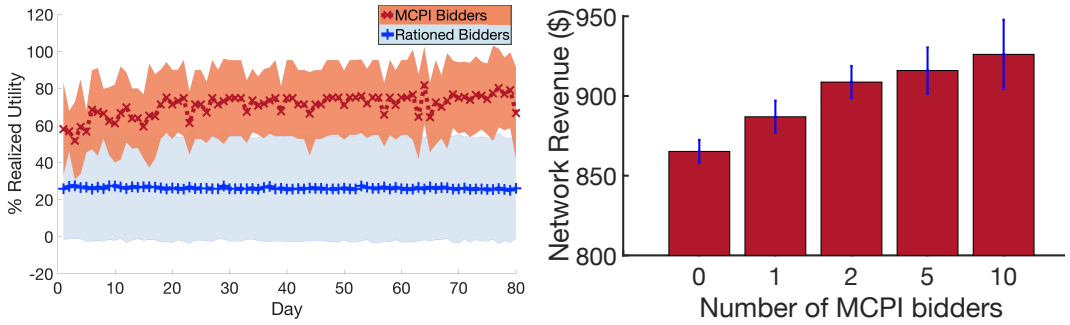
Figure 4.11: (a) MCPI far outperforms naïve strategies, achieving 100% of the maximum utility. (b) After 80 days, there is little deviation in the MCPI bidder's actions.

Performance of MCPI-based bidding: We first study a deterministic setting where we ensure resources are available for every auction, meaning every request elicits a viable bundle. Four scenarios are simulated, with the first two using the greedy and rationed strategies for all users, respectively. In these cases, we measure the fraction of $U_{i,\max}$ that bidders achieves at the end of each day and show the mean and standard deviation across bidders in Figure 4.11(a). In the third and fourth scenarios, we introduce one bidder using MCPI amongst the greedy and rationed users, respectively, then measure the MCPI bidder's U_i. As Figure 4.11(a) shows, the MCPI bidder succeeds at exploring various actions and reaching 100% $U_{i,\max}$ by day



(a) Bundle Variance Impact (b) Schedule Variance Impact

Figure 4.12: (a) MCPI realizes 85% utility despite high uncertainty in resource availability. (b) Temporal variance in resource requests does not significantly degrade performance.



(a) Impact of multiple MCPI agents (b) Revenue Impact

Figure 4.13: (a) With 10% MCPI bidders, $\sim$ 70% of the maximum utility is realized past Day 20. (b) MCPI bidders increase revenue by driving up critical prices in auctions.

40 (even sooner against the rationed bidders), while neither of the naïve strategies achieves more than 40%. The MCPI bidder's bid amounts v_{it} across different auctions during the first 10 and last 10 days of the simulation, Figure 4.11(b), shows heavy exploration during the first 10 days. But the MCPI bids during the last 10 days are much more stable (with some randomness introduced by the ϵ-greedy exploration), and the optimal policy is found. In the presence of MCPI bidders, greedy and rationed bidders continue to have poor mean performance but marginally higher deviation until the MCPI agent converges (not shown). *Hence, a bidder that previously realizes no more than 40% of the maximum utility achievable with their budget (potentially believing that the market rate is prohibitively high), now wins more by bidding per the MCPI-based budget distribution algorithm.* To increase realism, we next consider a congested setting wherein some resource requests may not elicit any bundles. This is done by increasing session durations, which also increases the likelihood of a resource request being turned away due to ongoing consumption of previously admitted flows. Hence, in addition to budget constraints, MCPI learning must implicitly account for resource availability. For instance, if resources are typically unavailable at 6:00PM, then the optimal strategy might be to distribute the budget to other times of day, since the agent will likely not get a chance to express a bid for resource needs at 6:00PM. Figure 4.12(a) shows the resulting increase in the time for the strategy to stabilize. The MCPI bidder still outperforms the naïve bidders and converges, but only 85% of $U_{i,\max}$ is reached by day 30. In this case, exploration has a ripple effect on returned bundles in subsequent auctions, which affects budget changes and slows convergence.

We increase uncertainty in the environment by offsetting resource request schedules by a random time period ω_{it} for fraction f of the user population. Hence a user's resource request times are no longer perfectly periodic. Resource availability as well as critical prices during $[t, t + \phi_{it} + \omega_{it}]$ are affected by this variance, making the MCPI learning more challenging. However, as Figure 4.12(b) shows, the MCPI bidder's utility does not decrease as a function of $\omega_{\max}$ or f, even as these vary from 1-3 hours and 1-100% respectively, indicating that MCPI is beneficial to deploy in realistic network scenarios. We also incorporate non-determinism resulting from simultaneous exploration of multiple MCPI bidders. As the number of MCPI bidders increases, the uncertainty in their models also increases due to *non-stationary* [167], which reduces the likelihood of converging to $U_{i,\max}$. Figure 4.13(a) illustrates the resulting mean and

deviation of the percentage of realized utility across ten MCPI bidders. We observe decreased utility and less convergence with higher variance, but still significantly higher than naïve bidders.

We also study the impact of MCPI bidding on network revenue. Figure 4.13(b) shows the network revenue *increasing* marginally with the number of MCPI bidders (never decreasing). *This is a direct effect of combining our budget distribution strategy with an auction payment and allocation scheme that incentivize truthfulness.* Users, in their best interest, request resources only when needed and have no incentive to misrepresent their valuation (and no value for leftover budget end of the day). MCPI bidding then serves only to spend the budget in ways that simultaneously best represents users' utilities and the chances of winning.

4.10 Discussion

We now discuss practical considerations around deploying our system.

Network Requirements: Our choice of WiFi and LTE for feasibility experiments is motivated by the widespread proliferation of these RATs. The auction model can, in-fact, be executed on any topology where resource availability can be forecasted and reconciled with session needs. Hence, emerging RATs like mmWave (5G) and other WiFi versions are candidate network topologies. The resource modeling mechanism, however, depends on the RAT in use. For instance, in a WiFi network using the MAC-layer point coordination function instead of the distributed coordination function, the access points has more centralized control of flows and therefore may better eliminate wireless externalities. This would likely be factored in the forecast model. In addition, the underlying network must support real-time flow control, e.g., with software-defined networking.

Discovering Session Needs: The app-specific resource needs of a session are best determined by the app itself. Hence, wrapper libraries for network access protocols such as TCP/IP can be used by apps to state the required resources (e.g., bitrate or latency) as they open a new socket connection. The user agent may then be a background process that receives this information. User-specific factors such as intended session duration or daily budget can be explicitly set by the user or estimated by the agent based on historical user activity.

Diversity in Budgets: We have proposed propose multiple methods for the

network to maximize social welfare and incentivize truthfulness in users' resource and valuation specifications. This ensures a certain notion of *competitive fairness* [58]. Since it is not in users' best interest to lie about their needs, only users who truly have the highest value for resources win allocations. However, since this value is expressed in monetary terms, their budgets play a limiting role in their winning chances. In this case, the RL strategy formulated in Section 4.8 will be especially valuable to help more constrained users place bids in auctions with lower critical prices and achieve the best utility possible given their affordability constraints.

4.11 Summary

We design an auction model that captures the market for session-level performance guarantees, decoupling the user-facing auction from network-facing wireless resource management. Our model moves away from radio resource auctions and focuses on application-level provisioning, which is especially useful for emerging real-time multimedia applications. Through trace-driven LTE simulations and extensive WiFi experiments, we verify that not only can wireless externalities be minimized with resource-aware admission control of flows, but more flows can be accommodated by implementing incentive-compatible auction-based admission control. We further analyze the winner determination of our proposed auction model with regard to real-time multimedia applications and show that there are several realistic conditions that render the multi-dimensional knapsack problem solvable in psuedo-polynomial time. These reductions make it feasible to implement the incentive-compatible VCG mechanism and even lead to revenue monotonicity in certain cases. We also analyze the impact of temporal correlations between auctions on incentive compatibility and define novel payment and allocation schemes to handle future bid uncertainty and navigate trade-offs in desirable properties. Finally, we use the Monte Carlo Policy Iteration technique to show that even budget-constrained users can achieve high utility from these auctions.

Chapter 5

Seamless Connectivity without Subscriptions - Authentication and Accounting

We now remove the inherent device-network trust relationship that was assumed in the previous two models by allowing devices to seamlessly authenticate with any access point (without subscriptions) and make real-time payments for consumed data, using public and permissionless blockchains. Here, we first develop secure mechanisms for authentication and bandwidth metering while assuming a simple albeit suboptimal payment solution. In the next chapter, we develop a secure and scalable payment protocol in detail and subsequently illustrate its use in the proposed system. In both these chapters, I address scalability challenges that blockchains pose as well as practical deployment challenges in techniques that involve access point modification.

5.1 Problem Definition

Devices in the IoT may have widely different network requirements and be deployed at locations without dedicated internet access Current approaches to IoT connectivity require device owners to manage individual data contracts and pay separate monthly fees for each IoT device on a given operator's network [70, 169]. Managing such contracts with a LoRaWAN or NB-IoT provider for each device in dense IoT installations, however, appears unscalable, expensive and may be a bottleneck for realizing large-

scale deployments, e.g., in smart cities. These devices have considerable variations in their data needs, which also makes different connectivity models appropriate for different devices [54]. For example, a camera continuously sending a video stream, a temperature sensor sending a single measurement every two hours, and radar sensors on an autonomous vehicle requiring millisecond latencies all upload varying quantities of data at varying frequencies and QoS. Providing for these diverse needs will become even more challenging as IoT deployments grow: the number of smart cities worldwide is expected to grow at a rate of 26% through year 2022 [94], making up a 34 billion USD market in 2019 [19].

The overhead of provisioning dedicated contracts for each device may accelerate as 5G networks are more widely installed: such networks are expected to include dense deployments of multiple access points (AP) of different radio access technologies [27, 28, 193], potentially with different operators, making it even more difficult to pre-specify contracts for individual IoT devices on each nearby operator. In this work, we propose to solve this challenge with Datanet, a system that allows an end-device to *seamlessly and securely connect to a nearby closed network that meets its needs*, with no a-priori trust or identity association, and provide compensation for availed data services in real time. Datanet avails the EAP-TLS authentication mechanism, and is therefore compatible with any access network that supports TLS-based authentication, e.g., WPA2-Enterprise WiFi networks and 5G networks [21].

Datanet has the potential to benefit not only IoT devices in future 5G networks, but also IoT devices and phones/laptops that avail existing network infrastructure. As smart-cities become more widely deployed, cheap WiFi hotspots are expected to be ubiquitous; indeed, a 2016 study from Turin, Italy, found that approximately 50% of streets around a block were within transmission range of in-home WiFi networks [175]. Densely deployed IoT devices, of which a significant portion are expected to be WiFi-equipped, could then realize their variant data needs by accessing readily available (and potentially closed) WiFi hotspots nearby using Datanet. Our analysis presents an even stronger case, showing that an average IoT device in the deployment we analyze is well within range of at least 10 WiFi hotspots. users with typical mobility patterns are within range of several WiFi networks (many of them closed) at any given time. Users of Datanet can then *significantly reduce or even eliminate* their usage of cellular data, instead utilizing one of the network connectivity options around them that are now accessible through Datanet; they can then subscribe to less expensive cellular

data plans.

Research Challenges. While a significant body of literature now exists on heterogeneous networks and optimal network selection [182], practical challenges in seamless association with access networks have prevented many of these proposals from being realized. Though private WiFi networks are often underutilized and can be availed by end-devices for data offloading, it is unclear how end-devices with no prior subscriptions to these hotspots can authenticate with them or be trusted to compensate hotspot owners for the availed data services. These devices may not even be able to discover which closed networks are available to them. Even open hotspots often use captive portals for enforcing sign-up and up-ahead payment charges (e.g., in airports), inhibiting a seamless experience. Though 5G provisions EAP-TLS authentication, this is expected to be coupled with SIM-based identifiers, hence continuing to require long-term subscription contracts and network-specific setup [194]. This continues to exacerbate a fragmented landscape of radio access technologies and network controllers.

We now distill these research challenges. First, Datanet raises **trust and identity issues**. An access point[1] must be able to *securely validate* that an end-device (user or IoT device) is able to pay for its connectivity, with no a-priori association or trust-relationship with the device. The access point must also be able to enforce payment upon providing network connectivity, which generally requires a trusted intermediary to play the role of an adjudicator. However, trusted intermediaries like ISPs typically require fixed-fee subscriptions that lock in the device. Further, the end-device is also not guaranteed that its data needs will be fully met by the access point even after payment since the access point is untrusted. Note that privacy concerns can be addressed by encrypting the session payload at the transport/application layer; though Datanet enables PHY-layer encryption of the traffic between the AP and the device, higher-layer payload encryption protects against other potentially malicious AP behavior such as leaking the PHY encryption key.

Second, fully solving Datanet's trust and identity related challenges requires **trusted metering** and support for **scalable payments**. Though decentralized ledger technologies like blockchains can often act as a proxy for trusted intermediaries, their ability to enforce and adjudicate interactions between the end-device and the

[1]Note that we use the term "access point" to refer to last-mile internet gateways of any Radio Access Technology, including WiFi ruoters, WiFi access points and cellular base stations.

access point requires *the payments and device data usage to be digitally tracked in a tamper-proof manner*. Trusted metering is challenging to achieve in these situations without dedicated trusted hardware. Hence, recent works [45, 144, 153, 160] addressing bandwidth sharing have proposed that end-users make incremental payments for incremental quantities of availed service, such that at most one round of incremental payment or service is wasted if the other party fails to provide the corresponding service or payment. However, these solutions generally require pre-establishing individual payment channels between each device and each access point, and thus scale poorly to scenarios where an end-user may connect to multiple different access points over time. Even so, a metering mechanism of *some* granularity is required to facilitate even the simplest QoS agreements/resource-use contracts essential for most applications, and to form a basis for discerning between unreliable and reliable access points and users.

Finally, Datanet should be **easily deployable:** a generic access point must be able to participate in Datanet *without requiring special-purpose hardware or even software* (e.g., even reflashing the firmware). Key wireless technologies like LoRaWAN for the IoT have faced a slow adoption rate due to the significant infrastructure setup costs they come with [166]. For practical widespread adoption, it is highly desirable to have the solution be general-purpose, agnostic of radio access technology, and require little modification to existing deployed access points. This constraint makes handling remuneration details with a connecting end-device and trusted metering even more difficult, since a generic access point does not have protocols to deal with payment flows. We must also that these operations can be performed without significant resource overhead; the Datanet mechanism must be cheap resource-wise for practical use in battery-constrained smartphones and IoT devices.

Our Contributions. Datanet relies on three core insights to address these challenges. Our system relies on three core insights. First, APs can **validate unknown users' payment ability by analyzing records in tamper-proof and public decentralized ledgers**, where users' balances can be locked in escrow accounts to enforce payments. However, this requires significant AP hardware and software modification to integrate with the blockchain for authentication and processing the micropayments. Our second core insight is that the Authentication, Authorization and Accounting (AAA) mechanism that is widely used for cellular networks and enterprise WiFi solutions can also be used here. A Datanet operator can **modify the cloud-based AAA server to use blockchain-based credentials** to authenticate

and authorize AP access to untrusted end users. A simple configuration change to the AP suffices to offload these operations to the external AAA server, with no hardware or software changes needed. However, this introduces centralization and potential scope for the AAA operator to enforce long-term subscriptions on APs and end-devices for its services. Further, the access point presumably still requires modifications to process micropayments, and more importantly, requiring users to setup dedicated payment channels with sufficient funds for each AP they may interact with imposes significant liquidity and capital constraints on users. To overcome this, we propose a **highly scalable, operator-mediated cryptocurrency payment protocol called *PayPlace*** for large marketplaces like the bandwidth-sharing one here; PayPlace also natively enables **multi-tenancy of AAA Datanet operators**, ensuring that end-devices can connect to any AP and APs can switch AAA operators at any time with no loss of funds. Finally, to address the problem of tamper-proof monitoring/statistic collection of the Datanet network, we design a novel solution based on a **trusted execution environment**, which is widely available in mobile phones and is expected to be deployed in IoT devices [138]. While, as we show, having proprietary software on the access points helps unlock additional and more sophisticated use-cases, a fully functional flow is achieved as is. We demonstrate Datanet's benefits by **designing, prototyping, and evaluating Datanet based on these insights**.

Note that we defer the detailed explanation of the PayPlace payment protocol to subsequent Chapter 6.

5.2 Related Work

Blockchain's potential in displacing the role of centralized ISPs in the device-AP association process has received attention recently. Althea [171] facilitates an incentivized wireless mesh network (with possible gateway to the Internet) by providing Raspberry-Pis running specialized routing and pricing software, which users can plug into their off-the-shelf routers. Althea users make cryptocurrency micropayments to APs to pay incrementally for their data forwarding services, using pre-established payment channels with each router they connect to. This pairwise payment channel model incurs significant scalability challenges as discussed in Section 5.4. Orchid [152] also has similar requirements to facilitate device access to a previously unknown

router; however, it requires routers to be reflashed with special-purpose software. Helium [89] creates a wireless mesh network of proprietary LongFi hotspots that eventually have internet backhaul. These LongFi networks are expected to serve IoT devices and to be deployed by end-users. These protocols thus require APs to install special-purpose hardware or software to accept and validate micropayments that users send over pairwise user-AP channels, posing significant adoption barriers. DataNinja [161] enables smartphone end-users, not APs, to share their cellular data or WiFi connection with another nearby user. However, it does not use any remuneration mechanism, instead relying on altruism. DataNinja meters the bandwidth exchanged but does not perform TEE-based integrity checks to verify these measurements.

Some existing solutions also including mechanisms for bandwidth metering. Ammbr [25] requires the deployment of proprietary routers, which can then be instrumented to be tamper-proof and provide trusted ground-truth measurements; however, this poses infrastructure setup costs for users and is necessarily specific to WiFi. Orchid [152] proposes to impose a high economic penalty (via staking mechanisms) if router fraud is detected. While this may disincentivize routers from setting up "fake" APs to inflate market perception of the network's value, it does not provide a technical solution and it is not clear how router fraud can be efficiently detected. Finally, we note that while all of these proposals facilitate a pay-per-use model between the end-device and access point, they do not, to the best of our knowledge, facilitate practical pay-per-use strategies like utilization-based pricing, which we enable with TEE-attestation network measurements from the end-device.

5.3 Potential Datanet Impact

We first demonstrate the impact of a system like Datanet that allows devices to seamlessly connect to potentially closed access points, without requiring a-priori trust relationships with the APs or an intermediary, as shown in Figure 5.1.

Impact on IoT Devices. To assess potential connectivity benefits afforded to IoT devices, we consider the Array of Things project [20], which currently has a smart-city testbed of 126 IoT devices deployed in the city of Chicago. With Datanet, WiFi-capable IoT devices can utilize any Datanet-enabled WiFi access point that is in range for transmitting the sensed data periodically to the cloud for processing. As shown in Figure 5.1, any WiFi AP can join Datanet as long as the AP supports

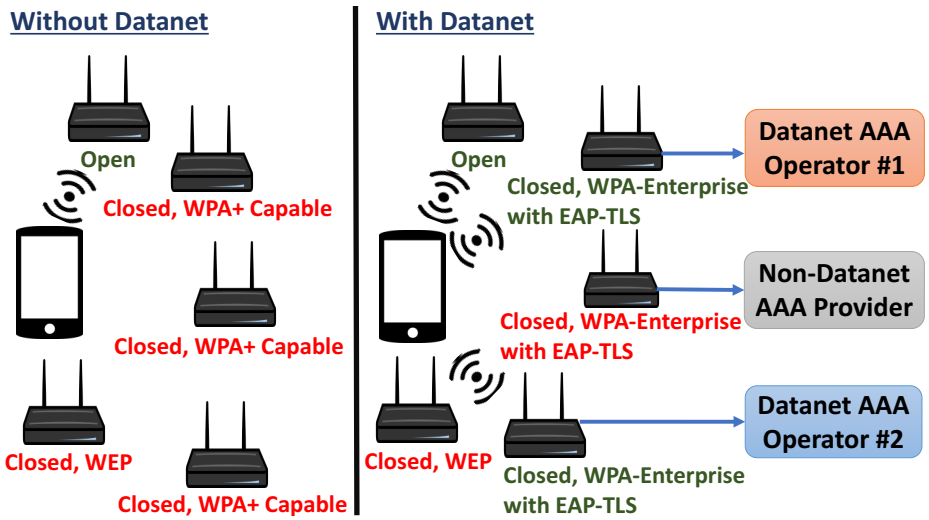


Figure 5.1: With Datanet, end-devices are able to access closed hotspots, as long as the hotspots support authenticating via EAP-TLS (e.g. using WPA-Enterprise) and use a Datanet operator for performing the authentication remotely. Hence, an end-device now has multiple candidate networks to choose between to get internet connectivity.

authenticating with EAP-TLS (e.g. if the AP supports WPA-Enterprise as most do, changing its AAA server to a Datanet operator is a one-click setting change). Using crowdsourced information about WiFi hotspots in the area obtained from WiGLE [183], we correlate the location of each device in the testbed with WiFi APs in transmission range, categorized by the authentication mechanism used by the AP. We consider APs within a $0-50$m radius of the IoT device, based on previous studies on how the WiFi RSSI decays with distance from the WiFi AP [175].

As shown in Figure 5.2, an IoT device in this testbed, on average, can reach approxi-

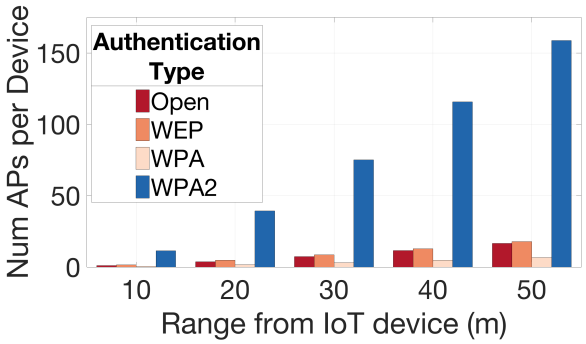


Figure 5.2: Approximately 12 closed Datanet-compatible WiFi access points (WPA/WPA2) are available on average to each deployed IoT device in the testbed, even within a close transmission range of 10m.

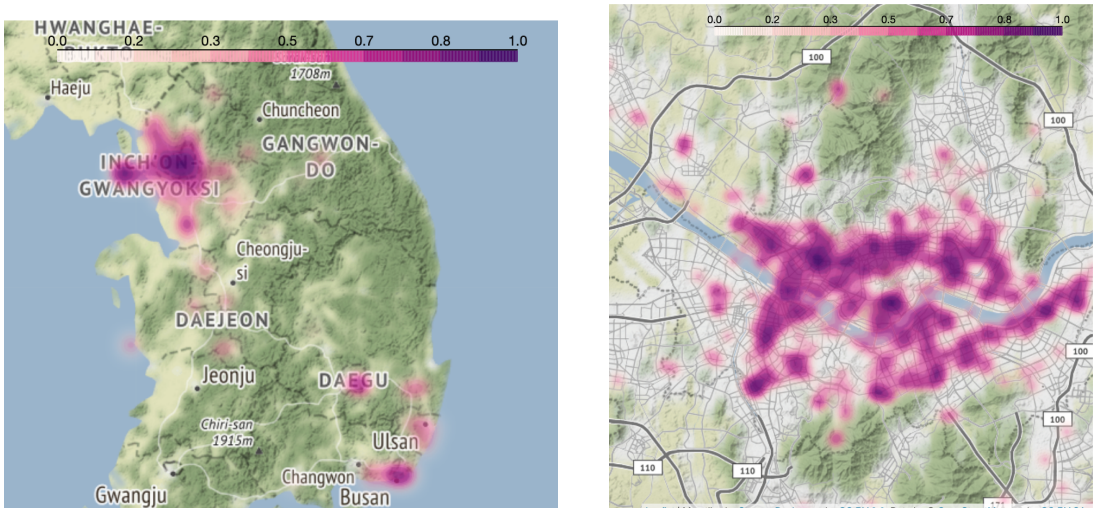
mately 14 hotspots (omitting APs whose network access control mechanisms/employed security suite is unknown) even within a conservative range of 10m that presumably yields strong signal strength. Only 1 of these APs is open on average, while approximately 12 of the 13 closed APs use the WPA1 or WPA2 security suite (hence capable of supporting WPAx-Enterprise standard for remote AAA-based EAP-TLS auth), which make them candidate Datanet APs. Upon widening the acceptable transmission range to 50m, the fraction of open hotspots does not exceed 10%, while the number of closed WPA2 networks increases over tenfold. With Datanet, these private hotspots can become candidate Internet gateways for IoT devices, and be compensated for the occasional data transport services they provide, without requiring prior information.

Impact on End-user Devices. Smartphone users, who must typically utilize a combination of cellular data, open WiFi hotspots (if available) and known private routers (e.g. at home), have significantly more network choices if they join Datanet. Indeed, they may even consider cheaper data plans with lower data limits if Datanet APs are widespread and provide a contract-less means of data access at more competitive rates. To quantify this hypothesized gain from Datanet, we verify whether a dense deployment of currently inaccessible closed APs exists around locations that users typically visit based on their regular mobility patterns. Though Datanet also enables access to EAP-TLS enabled future 5G networks [21], we conservatively limit our analysis to currently deployed WiFi networks.

For this evaluation, we utilize fine-grained mobility traces collected using the LifeMap mobility learning system [48, 49], which track the locations of eight students in Seoul, South Korea once every two minutes for two months. Prior analysis [48, 49] on this dataset showes that students are stationary 85% of the time, indicating that they would not suffer from frequent handoffs between short-range WiFi networks if they were to use Datanet.

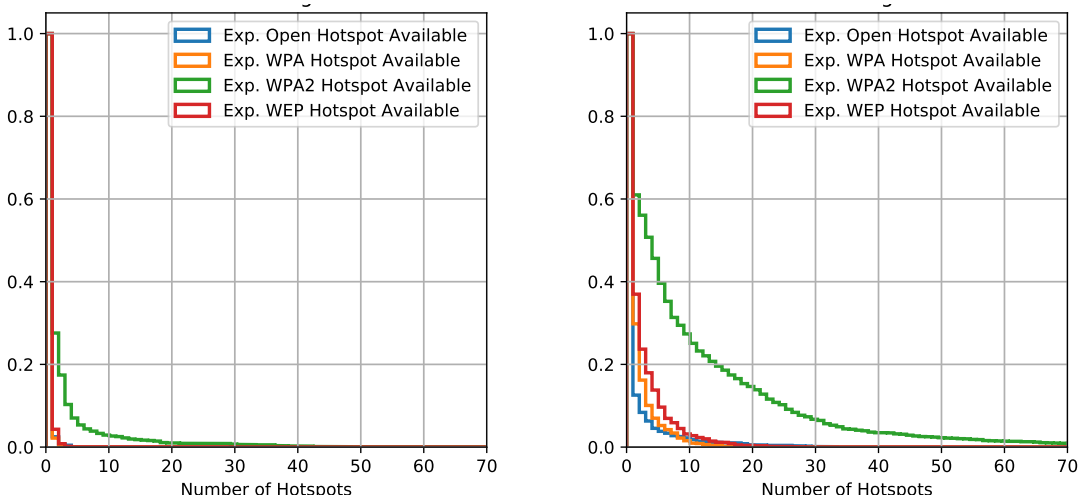
Figure 5.3 illustrates the densities of locations visited by these participants weighted by the duration spent in each, zoomed in on South Korea and Seoul. Some participants also made occasional trips outside South Korea, including locations in the United States. Datanet users who make such visits can particularly benefit from Datanet by avoiding the high international or roaming fees typical of most cellular data plans.

For each unique location in this dataset, we retrieve information about WiFi hotspots found nearby from WiGLE and estimate the average number of accessible APs for different transmission ranges, accounting for user localization errors specified



(a) (b)

Figure 5.3: Heapmap of locations visited by users in the LifeMap mobility dataset [48] over the course of two months, weighted by duration spent in each location. Shown for (a) South Korea and zooming in, (b) Seoul.



(a) Reverse CDF of estimated number of accessible APs within a 10m range of each location, categorized by encryption used

(b) Reverse CDF of estimated number of accessible APs within a 30m range of each location, categorized by encryption used

Figure 5.4: We depict statistics for the number of accessible hotspots for each unique location in the LifeMap mobility dataset [48]

in the mobility trace. Figures 5.4(a) and 5.4(b) depict reverse CDFs of mean hotspot availability corresponding respectively to 10m and 30m radius from each user location

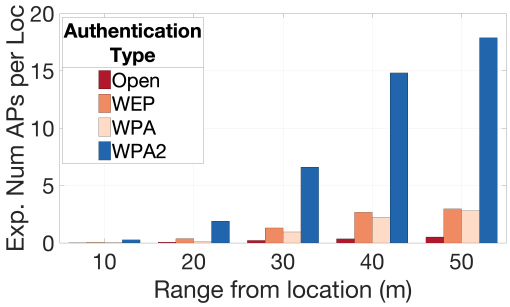


Figure 5.5: Estimated average number of accessible routers across all locations in the LifeMap mobility dataset, weighted by duration spent in that location and categorized by encryption type of routers.

in the trace. There is less than a 5% chance of a user encountering an open hotspot within a 10m radius, compared to a 35% likelihood of finding a closed hotspot in that range (i.e. WPA/WPA2/WEP encryption) and even a 10% chance of encountering upto five WPA2 hotspots. Even after expanding the radius to 30m, there are significantly fewer open hotspots than private ones. There is a 20% probability of encountering atleast 15 closed hotspots in a given location while only a 5% chance of encountering atleast five open hotspots.

Finally, we account for time spent in each location. Figure 5.5 shows the average count of each type of router accessible within different transmission distances from users' locations, weighted by the time users spent in that location. A typical user is within range of $2-17$ closed WPA2 hotspots at any given time, while very few open routers are deployed. Opportunity to utilize these private hotspots thus significantly increases connectivity options for the end-user, demonstrating Datanet's potential benefit.

Impact on Access Points. We next demonstrate that private routers have sufficient idle capacity to serve additional users through Datanet. We analyze the hourly bandwidth utilization of 1,200 home routers from the Measuring Broadband America initiative [154], collected in October 2017.

Figure 5.6 shows a mean network utilization of at most 2 Mbps across routers for all days observed, including peak evening hours. With typical home network capacity of $40-75$ Mbps [96], over 90% of this capacity is unused. Though closed routers in corporate environments may be more heavily utilized, this analysis nevertheless indicates that many private APs would be able to monetize their additional capacity with Datanet.

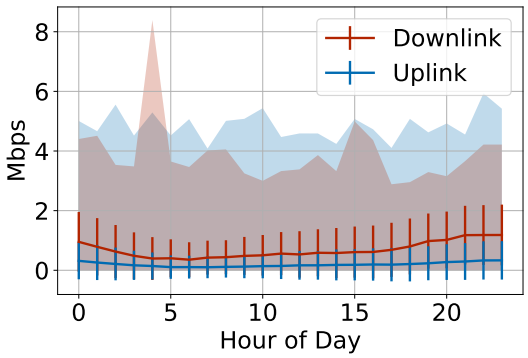


Figure 5.6: The mean downlink and uplink utilization for home routers from the analyzed dataset do not exceed 2Mbps, though the maximum recorded utilizations (shown in shaded region) reaches upto 8Mbps.

To measure these APs' incentive to join Datanet, we estimate the potential AP benefit by correlating government-provided population density traces for Seoul, South Korea [158] with the APs for which they are in transmission range. The population dataset partitions Seoul's total area of 605sq. km. into 19153 regions, and provides hourly measures of the number of people in each region. From the WiGLE database, we find approximately 650000 APs with unique MAC IDs in Seoul. Similar to the trend observed so far, approximately 88% of the APs support WPA2/WPA encryption, while 6% of them are open. Most of Seoul lies within a 50m transmission range of at least one AP (Figure 5.7).

With Datanet, closed APs may seamlessly connect to and serve any interested user. We thus aim to estimate the number of such users for each AP. We pick a representative day for our analysis from the extensive population density traces, and conservatively consider users within a short 10m transmission range. For each closed AP, we estimate the potential number of users it may serve each hour by multiplying the population density in the AP's region for that hour by the transmission coverage area. Figure 5.8(a) shows the resulting mean and standard deviation across the APs, indicating that they could serve over 10 additional users for at least 10 of the busiest hours of the day. Note, however, the user may be in range of multiple such APs (including, for instance, a known router), so APs might compete with each other to attract users. Further, if closed WEP routers do not switch over to WPA-Enterprise authentication (note that only the dated IEEE 802.11a standard does not support it, which few APs still use [130]), then the number of potential users that may connect to closed WPA/WPA2 routers is even higher. Figure 5.8(b) further weights this potential

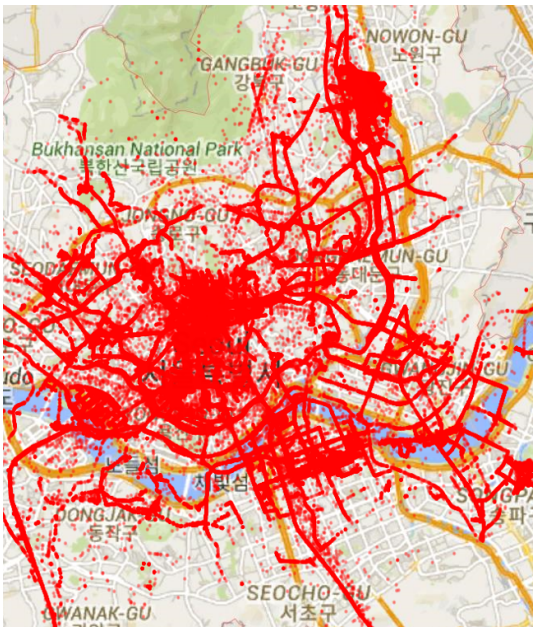
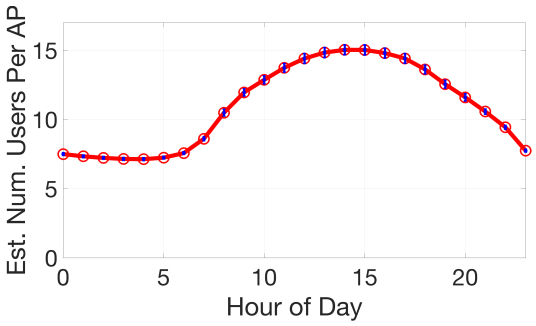
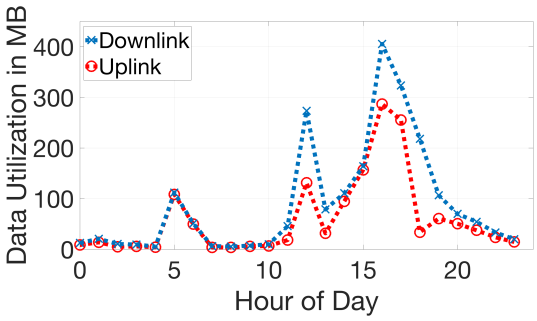


Figure 5.7: Area within a 50m range of an AP with known encryption type in Seoul, retrieved from WiGLE.



(a) Mean and standard deviation of potential number of users that a closed router in Seoul may connect to within 10m range, based on region density for each hour of day



(b) Mean additional downlink and uplink utilization at closed APs from potential Datanet end-devices, by hour of day

Figure 5.8: Depicting the potential benefit to private APs in Seoul by serving data needs of devices accessible through Datanet.

benefit by users' typical data usage over a day, which is estimated from network usage measurements collection on 20 users' smartphones over 10 days. Factoring this in, a closed router may serve up to several hundred megabytes of additional network traffic on average at some hours of the day.

5.4 Blockchain Background

We briefly review background material on micropayments and cryptocurrency payment protocols. While the payment protocol for Datanet, called PayPlace, is developed in the next chapter, we use the well-understood *Custodial Payment Hubs* construction as a simple proxy in this chapter which allows us to develop the rest of the Datanet architecture.

Micropayments. Transactions on the blockchain are known to incur expensive mining fees and confirmation time [180], introducing considerable overhead when small payments (sometimes less than the transaction fee) must be made repeatedly between two entities. The ability to make granular and frequent payments, however, is crucial to Datanet, allowing a device to pay an AP incrementally as bandwidth is consumed and limiting the user's loss to one micropayment if the AP unexpectedly stops serving the user. Hence, the ability to make micropayments allows the untrusted device and AP to transact with minimal risk (reduced to an incremental unit of data service and micropayment) in case the other withholds payment or data service.

To facilitate frequent micropayments at low or no cost, pairwise state channels are commonly used to move micropayments *off the blockchain* [60, 120, 160]. With such channels, the device (i.e. the corresponding user) makes one transaction on the blockchain (incurring fees and latency) to setup the initial payment/state channel with the AP and deposits some amount of cryptocurrency in it. Subsequent payments from the user to the AP can be made entirely outside the blockchain by just sending cryptographically signed payment promises to the AP (upto the value of the funds deposited), without incurring any blockchain-related costs. When the state channel protocol is correctly followed, these cryptographically signed off-chain payment promises can be redeemed for real payments on the root chain at any time.

Challenges with Pairwise Micropayments. While pairwise state channels can reduce micropayment costs as described above, they are still inadequate for the user-AP payments required in Datanet. Datanet users (i.e., end-devices) may purchase

data from multiple nearby APs during their use of the marketplace. Establishing pairwise payment channels with each transient AP then incurs significant overhead since users must commit funds in each channel that they believe will be used later for payments to that AP. Though researchers have proposed to enable off-chain payments between unconnected parties by routing via intermediaries connected by pairwise channels [120, 160], this requires intermediaries to commit their liquidity for others' transactions, a considerable cost in large marketplaces like Datanet. Indeed, recent work has exposed incentive compatibility issues with intermediaries in such routing protocols [37]. Further, fortuitous routes may well not exist between arbitrary user-AP pairs. Another potential solution would be to have an intermediary (e.g., owner/operator of the marketplace) set up pairwise channels with each AP and end-device, for the sole purpose of routing payments between them. However, this operator must then deposit its own capital/liquidity in each individual channel with an AP, which, given the size of the data connectivity market, could be prohibitively large.

Custodial Payment Hubs as a placeholder solution. More expressive *side-chain* models, which allow non-pairwise payment transactions to be conducted off-chain, incur tradeoffs in security. For instance, users in the popular side-chain model Plasma MVP [8] must frequently monitor the side-chain for malicious activity or risk losing their funds. The protocol PayPlace that we subsequently develop in the next chapter is a new side-chain mechanism specifically addressing security issues for large-scale marketplace-based payment scenarios like the bandwidth-sharing one here. For Datanet, PayPlace thus presents an appealing alternative to un-scalable pairwise payment channels. However, since we defer the development of PayPlace, we utilize Custodial Payment Hubs as a temporary solution in this chapter, which allows us to construct the remaining Datanet components without any major modifications. With Custodial Payment Hubs, users/payers deposit funds and establish payment channels with a single custodial intermediary, whom they make instant off-chain payments to for transactions in the marketplace. The intermediary periodically calculates the amounts to be forwarded to each payee (APs in our case) and makes root-chain transactions to withdraw corresponding balances from its channels with consumers and transfer them to the specified APs. This circumvents any liquidity requirements otherwise imposed on such intermediaries in non-custodial solutions.

5.5 Datanet Design

We provide an overview of the challenges in designing Datanet and our approach to addressing them. We then highlight Datanet's key goals and describe the system design in detail.

5.5.1 Approach

Datanet's goal is to enable seamless connectivity between end-devices (i.e., users) and APs, without subscriptions, a-priori device-AP trust relationships, long-term contracts with intermediaries, or complex key sharing and/or key generation schemes. Datanet achieves this by federating access to end-device identity and funds using the blockchain, which also allows funds allotted by users for use in data consumption to not be tied to a single provider. To avoid the significant hardware and software modifications required for APs (of different radio technologies) to integrate with the blockchain, Datanet uses remote AAA servers for blockchain-based user authentication, authorization and payments, allowing LTE, WiFi etc. APs to easily join Datanet.

As explained in Section 5.4, using micropayments eliminates the need for a prior device-AP trust relationship by allowing the end-device to withdraw at the first sign of misbehavior. It also allows the AAA operator to instruct the AP to terminate a user session at the first sign of user misbehavior (e.g. user stops sending micropayments to the operator for the AP's services). To avoid these AAA servers from becoming centralized intermediaries that then impose long-term contracts on end-devices and APs that use them for auth, Datanet's smart-contract has native support for multiple AAA controllers (equivalently, AAA operators) to simultaneously offer data services, fostering a competitive marketplace of these AAA operators on the Datanet smart-contract. This multi-tenancy relies on the PayPlace technique (introduced later), which allows Datanet users to make incremental micropayments *to the AAA operator* that their currently connected access point uses.

Finally, Datanet's novel architecture combines off-the-shelf hardware for Trusted Execution Environments (TEE), available commonly on end-devices today, with smart-contract based decentralized execution rules to measure the amount of data exchanged between Datanet access points and users. While the granularity and reliability of the metered information can vary, e.g., due to limitations imposed by the operating system

on the use of the TEE and TEE availability at the AP, at a minimum Datanet enables enforcement of custom payment schemes based on metered data exchange between the access point and the device, and also provides robustness to market manipulation of the Datanet crypto-token.

5.5.2 Goals and Constraints

We set the following goals for Datanet; the extent to which related work meets these goals is discussed in Section 5.2.

- **Seamlessness:** devices can seamlessly and instantly associate with a (private) AP on Datanet for Internet connectivity without prior identity, association, or payment contract with the AP or an intermediary.
- **No Subscriptions:** the AP serving a device in Datanet should be forced into neither a long-term contract nor a trust relationship with an intermediary.
- **Deployability:** establishing device-AP connectivity should be feasible without requiring special-purpose hardware *or software* at the AP.
- **Trusted Payments:** neither an end-device nor AP should be at risk of significant monetary loss (i.e., the AP failing to provide service after the user pays for it; or a user failing to pay after AP agrees to it) in a Datanet network association.
- **Trusted Metering:** Datanet sessions must result in tamper-proof, auditable statistics of the network traffic and payment for the session.

5.5.3 Datanet Overview

Figure 5.9 illustrates Datanet's core components and the typical end-device flow in Datanet. We define the Datanet **smart-contract** on a blockchain (e.g. Ethereum); the contract implements the PayPlace sidechain (introduced later) for enabling scalable device-AP payments which also supports operator multi-tenancy. The user of an end-device (smartphone, IoT device, etc.) first registers with the Datanet contract (step 1 in Figure 5.9) by depositing some value of cryptocurrency that they wish to later redeem for data connectivity services through Datanet access points. Similar to prior works, we denote this smart-contract's native crypto-token, e.g., an ERC20 token like EOS, by ¢. The registering user's details (public key, amount of ¢ deposited, etc.) are broadcasted by the contract onto the blockchain's event queue (similar to

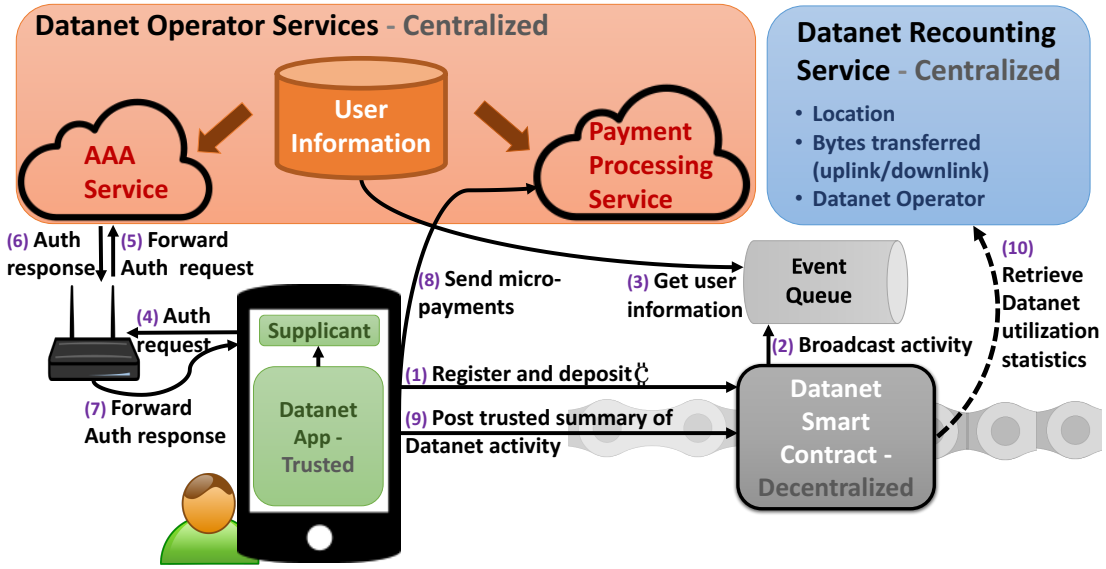


Figure 5.9: We illustrate Datanet's core components and interactions for an end-device to onboard and avail Datanet APs.

Ethereum events [162]) which Datanet operators are subscribed to (steps 2-3). A user's deposit into the Datanet smart-contract essentially establishes a payment channel between the user and an operator(s); users may specify multiple operator that they wish to split their deposit between. The PayPlace protocol specified in the subsequent chapter details how users may later shift their balances to other operators.

A Datanet **operator** hosts cloud applications for (1) performing AAA functions that specify whether an AP should accept connection requests from untrusted end-devices, and (2) processing micro-payments received in real time from end-devices connected to APs that use the operator's AAA service. Remote AAA servers decouple the complex business logic of network admission decisions from the AP, aiding in easy deployability of the blockchain-based mechanism since most radio access technologies can easily integrate remote AAA servers, which are ubiquitously used for authentication in 3GPP networks. Specifically, Datanet AAA servers rely on end-users' blockchain credentials to authenticate them using EAP-TLS certificate-based credential verification. Hence using Datanet AAA servers for authentication also allows for a seamless connectivity experience as end-devices and WiFi hotspots almost ubiquitously support the EAP-TLS standard; with 5G, even cellular networks are expected to support EAP-TLS [21]. We note that the authority signing the end-user's certificate is irrelevant since the Datanet smart-contract on the blockchain specifies

public-keys of valid end-users. Similarly, operators register their credentials with the Datanet smart-contract on the blockchain. End-devices thus retrieve the list of Datanet operators from the blockchain asynchronously and can verify whether the AAA server's credentials belong to a valid Datanet operator during the EAP-TLS handshake.

An **access point** then onboards onto Datanet simply by associating with a Datanet operator of its choice, i.e. directing its auth module (that usually defaults to WPA-Personal for home routers) to the operator's remote AAA service, configurable via the administration interface. We note that the AP may continue to provision WPA-Personal authentication for home users, if needed, using a separate isolated SSID. An end-user then, with their blockchain credentials and the Datanet application installed on their device, has seamless access to such Datanet-enabled APs. *Discovering* these access points is easiest if they can beacon Datanet support; Hotspot 2.0 frames support signalling details like associated Datanet operator, price charged by the access point, QoS capabilities, and others. For APs that do not yet support Hotspot 2.0, the Datanet operators may be queried off-band to retrieve details of surrounding Datanet APs. AP-specific information can always be signalled at the application layer by the corresponding Datanet operator after an end-device establishes a successful connection to the AP. Note that the TLS handshake reveals to the device whether a Datanet operator's AAA server is being used.

An **end-device** initiates association by sending an EAP-TLS authentication request to a Datanet AP using its blockchain credentials (step 4) that is forwarded to the corresponding operator's AAA service (step 5). Once a successful handshake is established with the end-device (steps 6-7), the Datanet application initiates periodic micropayments to the operator identified in the handshake (step 8). For instance, if the advertised charge is .001¢ per minute, the Datanet application makes an incremental payment promise of .001¢ every minute. The remote payment processing service performs continuous authorization of connected users, ensuring that the user has sufficient funds left in their payment channel with the operator to make micropayments and that each received micropayment is valid (i.e. contains a valid signature and assigns the expected amount of funds to the operator in that channel). Processing micropayments remotely through the operator's cloud services allows APs to use Datanet without requiring special-purpose software to handle its users' micropayments. If an active user misses consecutive payments, the payment service notifies the AAA

server, which issues a disconnect command [125] to the AP that terminates the user's session. If, on the other hand, the user finds the AP's service quality poor, the user may halt micropayments and the application searches for an alternate Datanet AP. Through this mechanism, untrusting APs and end-devices engage in incentive-compatible data sessions with negligible loss.

APs periodically receive incremental income due to them from the operator (which receives the micropayments that end-devices make for data services provided by APs and holds on to these funds custodially) based on the PayPlace side-chain mechanism. For illustrative purposes here, we may simply refer to the Custodial Payment Hub model, wherein the custodial operator periodically makes root-chain transactions transferring funds due to payees. Note that this process does not require any modifications on the AP itself to integrate with the blockchain. It may be desirable for AP owners to periodically verify that the income they are assigned by the operator is in keeping with the data services rendered to users, to minimize their trust on the operator and detect any malfeasance immediately. In case a lower payment than expected is received, APs may simply switch the Datanet operator they use for AAA services.

Finally, we instrument the Datanet **application running on end-devices** to generate trusted network utilization measurements that facilitate valuable usage-based micropayment structures (e.g. .001¢ for 1MB of data) and a tamper-proof assessment of the value created by the Datanet system. Since the Datanet application on the end-device is trusted (explained later) and TEE capabilities leveraged to attest that its usage summaries are not tampered with, these periodic measurement reports also guard the ¢ token against market manipulation from incentivized actors, as explained later.

5.5.4 Specialized Micropayment Structures with Tamper-Proof Metering

To limit AP and user losses from misbehavior, the AP must be able to terminate a user's connection upon non-payment. However, since the *Datanet operator* and not the AP receives micropayments (to avoid AP hardware/software modifications, as discussed in Section 5.5.3), the operator must compare the received payments to the services provided and issue a termination if needed. Such remote monitoring requires

tamper-proof metering of provided data services, since end-devices may maliciously attempt to provide a lower compensation than the rate specified by the AP. Note that if APs modified their software to directly collect micropayments, this concern is mitigated (though it results in pairwise payment channels between APs and the user).

Simple micropayment structures [45, 144] that are based only on the duration for which a user is connected to an AP, are easily monitored by a remote Datanet operator. However, more reasonable payment structures likely include some measure of the data service rendered in exchange for micropayments. For instance, the AP may wish to charge .004¢ per MB of data transferred; in this case, the micropayment amount that the operator receives every minute varies based on the amount of data transferred between device and AP. Running traffic monitoring software on the end-device's Trusted Execution Environment (TEE) and reporting the recorded incremental network utilization to the operator would be a straightforward solution. However, popular mobile and embedded operating systems do not typically support running third-party software in their TEE (e.g. ARM TrustZone [190]). Hence, we instead design the Datanet end-device application to be *trusted* (as in Figure 5.9), incorporate network traffic monitoring capability, and require periodic attestations from the device's OS (e.g. with Google's `SafetyNet` API [31]) that Datanet is running on an un-compromised device.

Ensuring that the Datanet application is publicly trusted is done by open-sourcing the codebase and explicitly relating this code to the executable available in App Stores (e.g. Google Play Store's APK) through certified compilation techniques (e.g. [150]) and publicly auditable Continuous Integration servers. Then, for every time period that a micropayment is issued for a usage-based payment structure, the application reports the data usage statistics along with an attestation summary from the OS that may include [31] the calling application, timestamp, unique nonce, and an indication of any known integrity issues (e.g. root capabilities that invalidate OS trust). If the timestamp matches current time, the stated package name of the calling application matches the trusted Datanet application, no integrity issues are indicated, and signature verification of the attestation is successful, then the operator considers the reported utilization as accurate. The operator may further corroborate the reported end-device utilization with coarse per-session network utilization information that it receives from APs (e.g. via RADIUS Accounting packets).

Reliability Metrics with Tamper-proof Metering. With sophisticated resource-

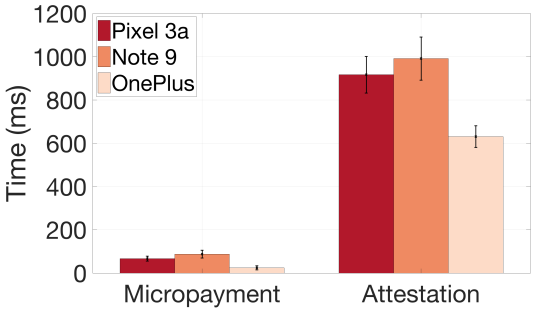
use contracts (e.g., real-time network slice agreements [92]), devices may face steep opportunity costs if contracts are violated, even though their monetary loss is insignificant due to the micropayment structure. For instance, a user may be seriously inconvenienced if she cannot complete a video call due to her connection being prematurely interrupted by the AP. The tamper-proof network utilization readings collected from the trusted Datanet application running on attested end-devices can be used by the operator, smart-contract or any third-party service to assess different APs' and end-devices' reliability, potentially leading to a *reputation system* that informs users' decisions to connect with specific APs and vice-versa. Note that WiFi access points are increasingly equipped with Trusted Platform Module [173]that provide attestation capabilities as well. Such APs can also provide their own trusted measurements to the Datanet operator for computing ground truth about session performance and influencing the reputation mechanism.

5.5.5 Assessing Datanet Utilization with Trusted Activity Summaries

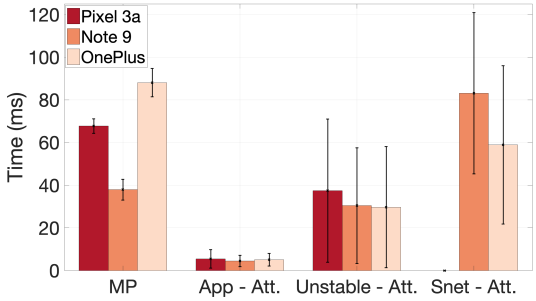
The Datanet application on the end-device also periodically sends a coarse summary of related network activity to the Datanet smart-contract, along with an attestation from the OS. As long as the attestation is verified, the contract broadcasts an event indicating that a Datanet utilization summary was published by an end-device. The summary information can then be cheaply learnt by parsing the block data corresponding to the broadcast event [4] and may be utilized by other services for recounting purposes (step 10 in Figure 5.9) to provide publicly verifiable statistics on Datanet usage. Since Datanet services are availed with special-purpose crypto-tokens ¢, this trusted metering guards against market manipulation of the token value by token-holders, APs and Datanet operators alike, who have strong incentives to inflate the coin value to increase their revenues [152]. As noted above, availing similarly trusted readings from APs, when possible, increases the robustness of this measure.

5.6 Evaluation

While section 5.3 shows that Datanet can benefit generic end-users and IoT devices, We now prototype the proposed Datanet mechanism and assess its overhead.



(a) Execution Time for generating Attestations and Micropayments



(b) CPU utilization as a % of a single CPU for generating Micropayments and Attestation

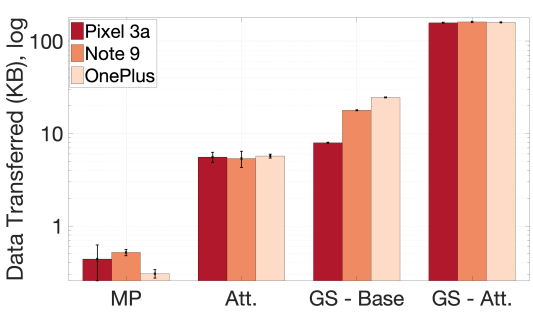
Figure 5.10: We analyze Datanet-related overhead incurred by the end-device in terms of (a) execution times and (b) CPU utilization.

5.6.1 End-device Overhead

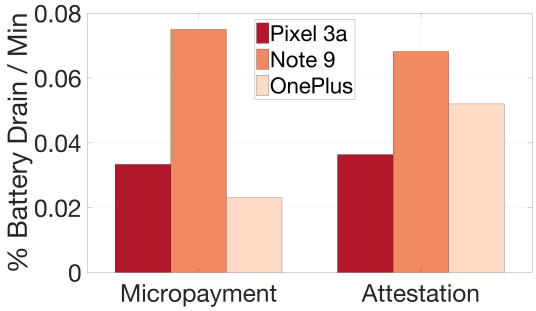
To assess the overhead on end-devices in Datanet, we implement a functional prototype as follows. A Datanet operator is setup on AWS cloud, with a FreeRADIUS instance [80] providing the blockchain-based AAA services and a payment service as described in Section 5.5. We setup a test blockchain network using Ganache [172] including a smart-contract for coordinating user balances and payment transactions. Accounts for the operator and test users are created on Ganache with sample ₵ deposits on the smart-contract. We use an off-the-shelf UniFi AC Pro [174] AP for processing Datanet connection requests from end-devices, configured to use the FreeRADIUS AAA cloud instance.

Next, we develop the Datanet Android application that downloads the list of nearby Datanet APs (resolving to the single UniFi AP in this case), including their MAC address, SSID and price per minute in ₵. The application initiates an EAP-TLS connection to our AP if its (SSID, MAC) tuple is located in the WiFi scan. Upon a successful connection, the application initiates a background service that, for every minute, creates a micropayment transaction specifying the net amount owed by the device to the Datanet operator, including the specified incremental ₵ for the last minute. The user then cryptographically signs this message with credentials that identify them on the root blockchain and sends it to the operator's payment service. The TLS handshake identifies the operator to the application with its public key.

Micropayments and Attestations. We run the Datanet application on three devices – Google Pixel 3a, Samsung Galaxy Note 9 and OnePlus 7 Pro – and assess



(a) Data transferred over the network per micropayment and attestation call, in KB - log scale



(b) Battery drain per minute for micropayment and attestation operations

Figure 5.11: We measure Datanet overhead in terms of (a) network traffic and (b) battery drain by performing attestation and micropayment every minute for 3 hours.

the overhead incurred by each. For attestation calls, we use Android's Safetynet API, which returns a signed response from Google attesting the calling application's package name, version number, timestamp, and device integrity indicator, as described previously [31]. Attestation and micropayment calls are separately repeated every minute at each device for for upto 5 hours. As Figure 5.10(a) depicts, generating a micropayment takes ~ 100 ms on average, and attestation takes ~ 1 s.

We next analyze the CPU utilization time of micropayments and attestation. The `SafetyNet` API's attestation call invokes hidden device processes whose details are not readily accessible publicly; thus the time spent by the calling application's thread in the CPU may not fully capture the call's CPU utilization. Instead, we profile system usage by frequently executing the *top* command, finding that non-negligible CPU utilizations of the `com.android.gms.unstable` and `com.google.android.gms.snet` processes tend to correlate with attestation calls. Figure 5.10(b) depicts the CPU utilization for each of these operations. We label the utilization for the Datanet application, `com.android.gms.unstable` and `com.google.android.gms.snet` as *App-Att*, *Unstable-Att* and *Snet-Att*, respectively. While micropayments consume $40-80\%$ CPU for the few hundred milliseconds that they execute, the attestation processes have wider variance. The OnePlus 7 Pro did not run `com.google.android.gms.snet`.

We next depict the network activity of these operations in Figure 5.11(a). Sending a micropayment to the Datanet operator's payment service consumes less than 500 bytes on average, while sending the attestation command response over the network incurs approximately 10KB. However, the attestation call invocation of hidden Google Play

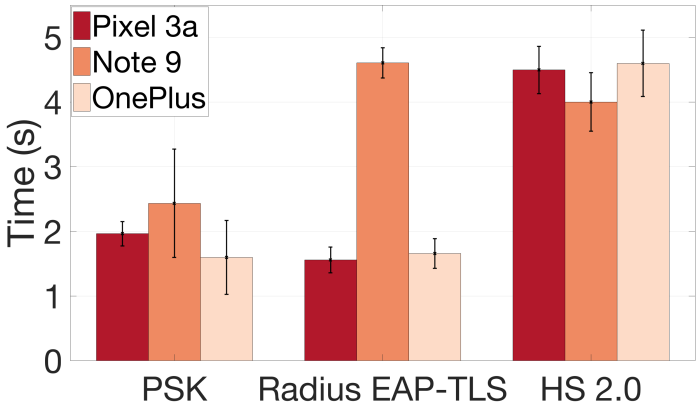


Figure 5.12: Latencies in connecting to a Datanet-enabled AP is seen to be equivalent to connecting with a private shared key or to a Hotspot 2.0 enabled EAP-TTLS AP.

processes may transfer additional information over the network to Google's server. We hence use Android's `BatteryHistorian` and `BatteryProfiler` tools [30] to infer this network traffic and the resulting battery drain. As Figure 5.11(a) shows, the typical network traffic generated by Google Services (measured by `BatteryHistorian`) is around 10KB/minute (GS-Base) but increases to 100KB/minute when the attestation call is performed every minute (GS-Att), indicating a 100KB overhead per `SafetyNet` API call. As seen in Figure 5.11(b), the battery drains at a rate of $.04 - .08\%$ per minute, across attestation and micropayment, indicating *no significant increase in battery consumption* from these operations.

EAP-TLS Negotiation. Since Datanet relies on the EAP-TLS handshake for network authentication and end-devices may frequently associate with different APs, we measure the corresponding network overhead incurred. We configure the Android devices to repeatedly authenticate and then disconnect with the UniFi access point around 100 times. The AP is first setup with WPA-PSK, then with EAP-TLS via the remote RADIUS server, and finally, with Hotspot 2.0-enabled EAP-TTLS (with MSCHAPv2). We compare with Hotspot 2.0 since Datanet can use Hotspot 2.0 to signal relevant AP information like compatibility or operator. For this, we setup the AP to use a popular Hotspot 2.0 provider's remote AAA server and beacon settings, and install the provider's OSU profile on the Android devices to detect and connect to any Hotspot 2.0 networks that beacon this provider's information. Figure 5.12 shows that the EAP-TLS mechanism incurs comparable AP connection latency to PSK and Hotspot 2.0.

5.7 Discussion

The Datanet client application running on end-devices that send micropayments and utilization readings to Datanet operators is *trusted*; that is, the application is open-source, its code verified to be correct. Further, the operator can verify whether they are communicating with this particular application or not through remote attestation like the SafetyNet API. Even though the end-device is not trusted, operators can trust measurements reported by the Datanet client application since the remote attestation API (e.g. SafetyNet) that helps the operator verify the application they are communicating with also attests the integrity of the device that the application is running on. However, this setup does not necessarily port over easily to other operating systems and devices that may not natively offer such attestation APIs out-of-the-box, for instance, IoT devices. Further, our experiments on client-side resource impact of Datanet does not consider IoT devices, which are significantly more battery constrained than smartphones. As future work, we hence plan to implement Datanet client-side operations on a Raspberry Pi (emulating an IoT device). To verify device integrity, we plan on using remote attestation software like Shadow-box[2] that protects the Linux kernel from unauthorized executable file attacks (using Linux IMA) and kernel object attacks using ARM TrustZone capabilities in the Pi 3. For the operator to verify that it is talking to the trusted Datanet client application when it receives utilization readings, we intend to run the application as a *Trusted Application* on the device, using ARM TrustZone and TEE. Finally, we will repeat the power measurement and latency experiments on this Raspberry Pi prototype implementation of Datanet.

5.8 Summary

Though TLS-based authentication is ubiquitously supported in WiFi access points and expected to be deployed in future 5G networks, end-devices generally cannot connect to these access points; they must instead establish dedicated long-term payment contracts with ISPs for Internet access. In this work, we propose Datanet to instead enable seamless connectivity between end-devices and access points without any prior subscriptions using trustless blockchain-federated authentication and authorization.

[2] https://github.com/kkamagui/shadow-box-for-arm

We employ remote AAA servers to perform this blockchain-based auth when unknown devices attempt to connect with untrusting APs, thereby avoiding any hardware or even software modification at access points (which otherwise would considerably impede adoption). To mitigate trusted relationships with the intermediate AAA service providers, we adapt a recently proposed protocol for secure and scalable blockchain-based payments in marketplaces to natively support multiple AAA service providers. This effectively enables a marketplace of Datanet-compatible AAA services whose interactions with devices and APs are federated by the Datanet smart-contract on the blockchain, thereby avoiding AAA operator monopolies. Finally, to enable practically useful payment models like usage-based payments, we design a novel use of trusted execution environments that are available for performing device integrity checks and attestations in the mobile OS, to provide tamper-proof network utilization metering without specialized hardware support. We demonstrate Datanet's potential benefit to IoT devices and end-users alike by correlating the location of deployed IoT devices as well as recorded user mobility patterns with WiFi access points around them. We finally show that Datanet is practical, introducing little overhead to the end-user, access points and the AAA operator alike, by building and evaluating a Datanet prototype.

Chapter 6

Seamless Connectivity without Subscriptions - Billing

We now develop the PayPlace payment protocol for use in the Datanet system to facilitate fast, cheap and operator-mediated payments between large numbers of users and routers. Making cryptocurrency payments in such envisioned bandwidth-sharing marketplaces in non-trivial. The standard solution for off-chain payments, state channels, are optimized for frequent transactions between two entities and impose prohibitive liquidity and capital requirements on payment senders for marketplace transactions. We propose PayPlace, a scalable off-chain protocol for payments between consumers and sellers and show that is has strong security guarantees, leaves a low resource footprint on the blockchain, and is orders of magnitude cheaper than the state-of-the-art cryptocurrency payment system, Zero Knowledge Rollups.

6.1 Problem Definition

Facilitating fast and cheap cryptocurrency payments is important for several marketplace applications that use blockchains, and especially so for large-scale blockchain-based networks like Datanet and others [89, 126, 152] that aim to facilitate sharing of last-mile network resources. There is also increasing interest in enabling well-established two-sided marketplaces like Amazon and Uber on blockchains [75, 76], which requires a scalable mechanism for consumers to make cryptocurrency payments to merchants. Since blockchain transactions are known to be limited by long finality times, low throughput, and high fees [180], off-chain payment mechanisms have come to

be regarded as a promising alternative. However, predominant solutions [64, 121, 160] rely on state channels that are optimized for frequent pairwise payments between two entities (unlike typical marketplace interactions) and impose prohibitively high capital and liquidity requirements on payment senders and intermediaries in the marketplace scenario (more in Section 6.3). Yet other off-chain protocols for broader non-pairwise scenarios [11, 118] rely excessively on the root-chain for securing off-chain funds; the number of blockchain transactions they initiate (and often the associated on-chain computational load) scales linearly in the number of payment transactions (between consumers and merchants), thereby incurring substantial transaction fees and being inherently limited by the throughput of the root-chain.

To the best of our knowledge, no work has yet addressed these practical capital and liquidity challenges in making large quantities of consumer-merchant cryptocurrency payments in limited-throughput, high-cost and resource-constrained blockchains. On the other hand, several proposals [26, 73, 126, 181] have presumed the existence of such a mechanism to design sophisticated blockchain-federated marketplaces, e.g. for crowdsensing. In this work, we develop **PayPlace, a protocol enabling flexible cryptocurrency payment schemes for large-scale marketplace applications**. PayPlace takes advantage of the presence of **marketplace operators** (e.g. Uber/Amazon) that can act as dedicated intermediaries for payment transactions. Hence, PayPlace does not impose excessive capital requirements on consumers; they simply pay the operator for their marketplace orders rather than establish a state channel with dedicated capital with each corresponding merchant. Unlike typical payment intermediary-based routing methods, however, PayPlace does **not impose any liquidity requirements on the operator**. Instead, the PayPlace operator temporarily holds consumers' off-chain payments custodial and periodically makes off-chain payments to corresponding merchants by directly referencing these accrued off-chain funds. For every such holding period, the operator generates a *short commitment* or hash of the aggregate payments to merchants and *notarizes* it on the root-chain.

This operator-mediated temporarily custodial model enables flexible payment schemes, e.g. by allowing marketplace operators to match buyers with sellers asynchronously. For instance, Amazon may decide which of multiple merchants should fulfil an order well after the consumer has paid for it. The custodial holding and periodic forwarding also allows for a natural reduction in the amortized cost per payment

transaction; the operator aggregates off-chain payments for multiple orders received from multiple consumers in that duration and makes just one root-chain transaction to represent the off-chain payments made in-turn to each merchant.

Assuring *safety* of users' funds is challenging in such protocols that involve periodic notarization by an operator of off-chain payment activity on the root-chain [8, 9, 85, 106, 107, 140] (called commit-chains or sidechains). To minimize computational and storage resource expenditure on the root-chain, only a short commitment of the off-chain payment activity (typically an irreversible hash) between users is revealed to the smart-contract. Hence the contract often does not have the ability to assess the validity of the represented transactions and resulting balances. This threatens *safety* of users' funds and is a major source of concern in PayPlace. Indeed, a merchant or operator must not be able to withdraw a larger portion of a consumer's funds than what the consumer has already sent as off-chain payments for marketplace orders to the operator. Similarly, funds once assigned by the operator as payments to merchants must be safeguarded from future tampering as well, including *double-spend attacks* that the operator may launch, wherein the amount assigned by the operator merchants exceeds what the operator has available as off-chain payments from consumers.

Merchants must also be safe from **data availability** attacks [10]. With previously proposed commit-chains [8, 9, 85, 106, 107, 140], the operator could submit a commitment to the root-chain without revealing included transactions (used to generate the hash) to users. Users then cannot verify whether their transactions were included, leaving them unsure of whether the operator has included malicious transactions, whether previously assigned funds are safe, and how much they are eligible to withdraw as of the latest commitment. Neither can the smart-contract verify the validity of off-chain transactions from the (irreversible) hash it receives. This in turn **necessitates that users be online** and monitor the root-chain; if malicious activity like the data availability attack is detected, users are expected to initiate withdrawal of their funds, leading to the well-known problem of **mass exits** [23, 65]. Expecting consumers and merchants to be online, however, significantly limits the practicality of the solution, especially in retail/marketplace settings.

PayPlace solves these challenges with novel constructions tailored to the marketplace context. Figure 6.1 llustrates key aspects of the protocol. First, we provide an easy-to-use view of the system to consumers, wherein they deposit funds in the PayPlace smart-contract and regard this as a **virtual unidirectional payment**

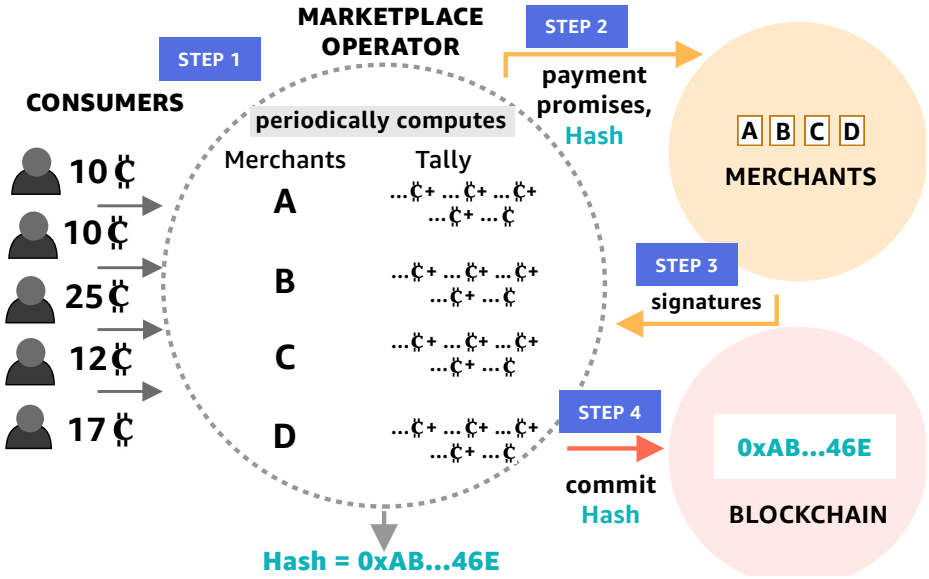


Figure 6.1: The PayPlace operator periodically tallies the accrued consumer payments that it owes to each merchant, acquires their signatures on a representative commitment, and submits it to the root-chain.

channel [1] with the operator. Consumers then make fast payments to the operator off-chain for orders placed in the marketplace without needing to be online to guard their funds. Second, the operator **periodically computes** payments to merchants based on accrued off-chain payments, generates a short commitment or hash of this, broadcasts these computed payments and the commitment to merchants, and also **reveals the off-chain funds accumulated in its virtual channels with consumers to merchants**. Third, online merchants attest their signatures to this commitment if they successfully verify that **no double-spend attacks have been launched** in the operator-generated payments. Fourth, the operator **consolidates signatures** received from merchants on the generated commitment and submits it to the PayPlace smart-contract for notarization. We utilize Boneh–Lynn–Shacham (BLS) signature aggregation [39] to **securely and efficiently combine merchants' signatures of a commitment into a single signature**, avoiding the resource costs of large-scale signature verification. The contract hence stores only an aggregated public key of merchants and notarizes a commitment if the provided aggregate signature is verifiable against the stored aggregate key. Our construction uniquely enables the contract to accept commitments even when some merchants' signatures are missing *and* also protect funds assigned to them in previous notarizations **despite not storing**

merchants' individual public keys or balance. The contract ensures that the total amount withdrawn by a merchant or operator against a consumer's deposit **does not exceed the funds assigned by the consumer to the operator as off-chain payments.** PayPlace is hence resilient to data availability attacks, provides strong merchant safety and **never results in mass exits** since notarized merchant funds are guaranteed to be safe even if the operator later deviates from the protocol.

Our evaluation shows that on-chain computational and monetary costs of PayPlace are orders of magnitude lower than the recently deployed state-of-the-art technique for non-pairwise off-chain payment scaling, Zero Knowledge Rollups.

6.2 Overview

6.2.1 Goals

We refer to a generic consumer by c and a merchant (also referred to as service provider or simply provider) by p. We define *confirmed funds* $f_{c,t}$ and $f_{p,t}$ as the funds available to a consumer c for spending in the marketplace as of time t and the funds available for a merchant p for withdrawal as of time t, respectively. $f_{c,t'}$ equals the deposited amount when a consumer c first joins the system at time t' by depositing funds into the smart-contract, and $f_{p,0} = 0$ for all merchants when the system is starting out at $t = 0$ (i.e. no payments yet). Note that PayPlace allows consumers and merchants to join and leave at largely any time. We say that an honest user (consumer or merchant) is *active* at t, denoted as $\mathcal{A}(u, t) = 1$, if she is "online" listening to smart-contract events and incoming messages at t and follows the protocol in response. Our first property ensures predictable execution time for withdrawals initiated by payment recipients:

Definition 1 (Liveness)**.** *A merchant or an intermediary (involved in relaying consumer to merchant payments) can initiate a withdrawal of their funds at any time t or wait at most a predefined duration to do so. Once initiated, a withdrawal must impose no wait-times and execute to completion immediately, subject to transaction processing latency of the root chain.*

Liveness is not satisfied by existing commit-chain designs [8, 9] that rely on *exit games* where the smart-contract forces users to wait for a significant period of time after they initiate withdrawals in order to prevent potential attacks. Next, it is

important to ensure that neither consumers nor merchants are at risk anytime of having funds already assigned to them stolen, *even if they are arbitrarily inactive.*

Definition 2 (Consumer Safety)**.** *For any $t' > t$, $f_{c,t'} = f_{c,t} - \sum_i \alpha_i$ where $\{\alpha_i\}$ are the values of all payments and withdrawals that consumer c makes in time interval $(t, t']$.*

Definition 3 (Merchant Safety)**.** *For any $t' > t$, $f_{p,t'} \geq f_{p,t} - \sum_i \alpha_i$ where $\{\alpha_i\}$ are the values of all withdrawals that merchant p makes in time interval $(t, t']$.*

Note that merchants may have accrued additional confirmed funds during $(t, t']$ from consumer payments. Indeed, our next property assures that a protocol-compliant merchant is not affected by malicious/colluding merchants.

Definition 4 (Income Certainty)**.** *$\exists \theta > 0, \delta \geq 0$ such that any valid payment initiated at time t by a consumer to a merchant p is available as a part of p's confirmed funds by $t + \theta$ if the merchant and any involved intermediary ω are continuously active during $[t', t' + \delta]$ for some $t' \geq t$ (i.e. $\mathcal{A}(p, t'') = 1$ and $\mathcal{A}(\omega, t'') = 1$ for all $t'' \in [t', t' + \delta]$).*

The next property provides resilience to data availability attacks that are common in commit-chains and sidechains, wherein users are left unsure of their available funds.

Definition 5 (Data Availability)**.** *Merchants and consumers know their confirmed funds $f_{p,t}$ and $f_{c,t}$ at any time t and the information necessary to use them. That is, if $f_{p,t} > 0$, $\exists t' < t$ with $\mathcal{A}(p, t') = 1$ such that p was notified at t' of the value of $f_{p,t}$ and received necessary information to withdraw it. If $f_{c,t} < D_{c,t}$, $\exists t' < t$ with $\mathcal{A}(c, t') = 1$ such that c is notified or aware at t' of the value of $f_{c,t}$ and information to spend it.*

We define liquidity and root-chain footprint requirements.

Definition 6 (Pooled Liquidity)**.** *A consumer can initiate a valid payment at t of value up to $f_{c,t}$ to any merchant.*

Definition 7 (Single-Source Liquidity)**.** *An intermediary involved in relaying consumer payments to recipient merchants does not need to deposit funds in the system.*

In other words, consumers need not partition their funds ahead of time for use with individual merchants and their capital is directly used for finishing initiated payments. We next define additional notation for characterizing transaction efficiency on the root-chain. Let $n_{t,t'}$ and $p_{t,t'}$ be the number of initiated consumer payments and the corresponding number of unique merchant recipients during some time interval $[t, t']$, respectively. Let $r_{t,t'}$ be the number of root-chain transactions required during $[t, t']$ to complete the initiated payments (i.e. to confirm the payments available for

withdrawal/reuse by recipients).

Definition 8 (On-Chain Efficiency)**.** $\exists \beta > 0,\ \delta \in [0, \beta]$ *such that* $\forall k = 0, 1, 2, \ldots$, *the protocol satisfies* $r_{k\beta,(k+1)\beta} = o(n_{k\beta,(k+1)\beta})$ *and* $r_{k\beta,(k+1)\beta} = o(p_{k\beta,(k+1)\beta})$ *as long as merchants are active for* $[k\beta + t, k\beta + t + \delta]$ *for some* $t \in [0, \beta - \delta]$.

6.2.2 Threat Model and Assumptions

Attacker PayPlace aims to satisfy the goals identified in Section 6.2.1. Of these, the security properties are *Consumer Safety*, *Merchant Safety*, *Data Availability*, and *Income Certainty*. Correspondingly, the key attack vectors are:

- A malicious operator may attempt to double-spend consumer payments to multiple merchants or re-assign funds assigned to merchants in previously notarized commitments. The operator may also attempt to withdraw more funds from a consumer's deposit than what has been assigned to the operator through the consumer's off-chain payments. These attacks would violate *Consumer* and *Merchant* Safety. The operator may collude with merchants and may also attempt to impersonate other merchants (e.g. the rogue public-key attack [148]) to launch these attacks. The operator may also withhold information about a submitted commitment and hence violate *Data Availability*.
- Merchants may collude to withdraw more funds from the PayPlace contract than what has been assigned to them, thereby violating both *Consumer* and *Merchant Safety*.
- Merchants may attempt to avoid computational burden (like attesting signatures) when possible, thereby potentially violating *Income Certainty* in PayPlace.
- Malicious consumers may attempt to make invalid off-chain payments to operators or to withdraw funds already assigned to the operator. These attacks violate *Merchant Safety* and often requires violating *Liveness* to guard against.
- Even if some merchants are temporarily inactive (e.g. their communication links with the operator are attacked), their already assigned funds must not be subject to risk, i.e. *Merchant Safety*, and other active merchants must still be able to receive additional income, i.e. *Income Certainty*.

Assumptions We assume that the root chain is secure; in other words the adversary cannot compromise execution of the PayPlace smart-contract on the root-

Protocol	Liveness	Consumer Safety	Merchant Safety	Income Certainty	Data Availability	Pooled Liquidity	Single-Source Liquidity	On-Chain Efficiency
Blockchain Tx.	✓	✓	✓	✓	✓	✓	✓	—
Direct Channels	✓	✓	—	✓	✓	—	✓	✓
PCN	—	✓	—	✓	✓	✓	—	✓
Payment Hubs	—	✓	—	✓	✓	✓	—	✓
Custodial Hubs	✓	✓	✓	✓	✓	✓	✓	—
Plasma-style CC	—	—	—	✓	—	✓	✓	✓
Plasma CC w/ Sign.	✓	—	✓	—	✓	✓	✓	✓
Snappy	✓	✓	✓	✓	✓	✓	✓	—
ZK Rollup	✓	✓	✓	✓	✓	✓	✓	—
PayPlace	✓	✓	✓	✓	✓	✓	✓	✓

Table 6.1: Properties provided by different cryptocurrency payment mechanisms applied to the marketplace context.

chain or impact the consensus process of root-chain miners. We also assume that the root-chain supports BLS signature verification [38, 39, 40]. The BLS signature scheme and associated operations like hashing to the elliptic curve are currently being standardized [22, 59] and popular systems like Ethereum 2.0, Zcash, Chia, and Polkadot already utilize BLS signatures [13, 14, 17, 46]. We assume that users' secret keys are secure (not leaked). Finally, we assume that the root-chain offers an inexpensive mechanism to broadcast messages and write them to logs (like Ethereum Events [16]); the root-chain logs can be traversed to recover messages by clients who missed the broadcast.

6.3 Related Work and Strawman Designs

Before proceeding to explain the PayPlace protocol, we provide an at-a-glance review of how existing solutions perform in terms of meeting the goals stated above. Table 6.1 summarizes this. As a baseline, we first note that directly **using the root chain to make regular crytocurrency payments** to merchants would satisfy almost all identified goals but *On-Chain Efficiency*. Transactions processed on the root chain consume permanent disk space in mining nodes and also incur mining fees that can become prohibitively high during congestion periods. Consider, for instance, the ride-sharing economy that Uber facilitates by matching drivers and riders in the two-sided marketplace. Bitcoin and Ethereum transaction fees for July 2020 average approximately $1.56 and $1.04 respectively [188, 189] , which represents a 6-10% fee for a typical 5 km Uber ride in Switzerland of average cost $13.90 and 44-54% fee for a typical 5 km Uber ride in India of average cost $1.34 [147]. In comparison, credit card fees per transaction is typically 1.5-3%. Since blockchains also have limited throughput,*On-Chain Efficiency* is highly desirable.

In the following review of alternate cryptocurrency payment mechanisms that have been proposed, we find that none simultaneously satisfy *Merchant/Consumer Safety* and *On-Chain Efficiency*; the latter requires moving payment transactions off-chain which then requires users to be *online* atleast periodically to ensure that their funds are not stolen.

First, we consider consumers establishing **direct unidirectional payment channels with each merchant** they transact with for frequent off-chain payments [1]. This crucially fails to enable *Pooled Liquidity* and also violates *Merchant Safety*, though it provides *On-Chain Efficiency*; indeed, since consumers use the root-chain for deposit transactions very infrequently and only after making several off-chain transactions that exhaust the deposited amount, essentially any values of β (cf. Defn.) provides *On-Chain Efficiency*. We next consider **Payment Channels Networks** (PCN) and a specific instance of PCNs called Payment Hubs. With PCNs [64, 121, 160], payment senders rely on non-custodial intermediaries that provide indirect routes (composed of state channels) to the payment recipient. Though this allows consumers to establish state channels (with locked-in funds) with a limited number of intermediaries in order to pay merchants, significant limitations exist with this. First, this is not guaranteed to enable *Pooled Liquidity* since the number of pairwise channels that consumers split

their funds in depends entirely on the network topology. Second, this fails to provide *Single-Source Liquidity* since consumers intrinsically rely on intermediaries' liquidity. In-fact, it has been observed that the resulting rapid fluctuations in intermediaries' link capacities makes it challenging to find routes reliably between payment senders and recipients [142]. Recent empirical analysis of the Lightning Network [37] further confirms that 1) "merchant" nodes [6] receive 80% of the off-chain payments, 2) nodes are hence forced to frequently close and rebalance their channels due to steady depletion of liquidity in the consumer→merchant direction [71] and 3) routing intermediaries have low return on investment on their locked-in funds. These issues are exacerbated in the context of large-scale marketplaces with frequent payments between consumers and arbitrary merchants.

Payment Hubs are PCNs where an intermediary is dedicating to providing a 1-hop route between consumers and merchants. In comparison with PCNs, this facilitates *Pooled Liquidity* by allowing consumers to pool their capital (intended for use in marketplace orders) in a single unidirectional channel with the intermediary, e.g. as with Plasma Debit [12]. However, the other challenges with PCNs carry over. We also consider a **custodial version of Payment Hubs** (also considered as the placeholder payment mechanism in Chapter 5), where the intermediary operator receives consumer payments in dedicated state channels with consumers and periodically initiates root chain transactions to make corresponding payments from to each merchant. However, this violates *On-Chain Efficiency* since the number of on-chain transactions grows with the number of merchants (receiving payments) every period.

Plasma-style commit-chains [8, 9, 107] involve periodic notarization of arbitrary off-chain payment activity on the root-chain by a dedicated intermediary and can be used to alleviate liquidity requirements. The notarization information is simply a short hash of off-chain transactions and does not allow the contract to track and validate individual transactions and resulting balances (by design, to minimize computational and storage resources consumed on the root-chain). This results in violation to both *Consumer* and *Merchant Safety* violated as the operator may simply insert invalid/malicious transactions in a block and use it to withdraw their funds; users can protect their funds only if they are online to detect such activity and withdraw their funds in response. There is no clear definition of *confirmed funds* for users such that these funds are safe even if users are arbitrarily offline. The operator may fail to reveal the set of transactions associated with a published commitment to

users, violating *Data Availability*. We consider a strawman modification to this called **Plasma-style commit-chains with Signature**, wherein merchants' signatures are required by the contract on the commitment to ensure that they have been revealed necessary information about the corresponding Plasma block. While this ensures *Data Availability*, it crucially violates *Income Certainty*. A few merchants withholding signatures maliciously (or even accidentally inactive) lend a commitment unfit for notarization. *Merchant Safety* holds since merchants implicitly agree on the transactions included in a block and the validity of resulting balances by unanimously attesting their signature on a notarized commitment. However, consumers are then subject to collusion attacks by merchants and the operator, where older payments from consumers that have already been withdrawn by receiving merchants may be included again a block. Hence, consumers' signatures are also required on commitments to ensure that they can protect themselves from such attacks (and exit games avoided), which in-turn necessitates that they be *active* to secure their funds; *Consumer Safety* is violated.

We also consider **Snappy** [118], a protocol for marketplace payments that has been recently proposed in parallel to ours. With Snappy, consumers directly send payments to merchants on the root-chain, but are unrestrained by the root-chain's transaction confirmation latency. However, atleast one root-chain transaction is made for each payment; hence Snappy does not provide *On-Chain Efficiency*. Finally, we consider the state-of-the-art solution for off-chain payments that extends beyond pairwise transactions, **Zero Knowledge (ZK) Rollup** [11]. ZK Rollup is advocated by the Ethereum Foundation and have been deployed by multiple companies recently. Unlike Snappy, the number of on-chain transactions required to process initiated payments is typically much smaller than the number of such payments and the number of payment recipients, though not sublinear in growth (i.e. no *On-Chain Efficiency*). By using ZK proofs to periodically assert the validity of several off-chain transactions at once on the root-chain, ZK Rollups simultaneously assure *Safety* and *Liveness*. Consumers deposit their funds in the Rollup contract, and transact with merchants off-chain via a non-custodial operator, which enables *Pooled Liquidity* as well as *Single-Source Liquidity*. Updated account balances are explicitly revealed in the root-chain, ensuring *Data Availability*.

As shown in Table 6.1, **PayPlace** satisfies all identified goals. Using periodic operator-driven notarization on the root-chain, PayPlace ensures non-revocation of

off-chain payments made by the operator to merchants and hence provides *Merchant Safety*. Requiring p's signature on a block for successful notarization also overcomes the *Data Availability* attack and ensures that merchants know and can access their confirmed funds. The operator makes *one* root-chain transaction periodically to assign funds to merchants; the computational costs of this transaction is *constant* in the number of underlying transactions and at-worst, sublinear in then number of payment recipients, thereby guaranteeing *On-Chain Efficiency*.

6.4 PayPlace Architecture

Consumers A consumer c with public key pk_c deposits funds in the PayPlace smart-contract for making payments in the marketplace (intended for any merchant). This design directly results in *Pooled Liquidity*. The consumer may deposit more funds at any time; we use $D_{c,t}$ to denote the total funds deposited by consumer c as of time t. The PayPlace smart-contract is designed to allow consumers to view their deposit as virtually establishing a *unidirectional payment channel* with the operator (with refunds and returns as external to the protocol). That is, consumers make incremental off-chain payments to the operator for each order placed and need not be online to protect their unspent funds, as with unidirectional state channels. The contract's commitment verification and withdrawal modules ensure that the total amount withdrawn by the operator and merchants against a consumer's deposit does not exceed the amount assigned by the consumer to the operator, thereby facilitating this simple view.

An off-chain payment from c for an order consists of a transaction $T = (\mu, pk_\omega, pk_c)$ and the digital signature of the transaction $\sigma = \mathcal{S}(T, sk_c)$. μ is the payment amount and indicates the *total* amount promised by the consumer to the operator as of when T is generated, incorporating the incremental amount the consumer intends to pay for their latest order in the marketplace. pk_ω is the operator's public key (the payment recipient), sk_c is the consumer's private key and $\mathcal{S}(T, sk_c)$ generates a cryptographic signature using sk_c on T. We let $s(T)$ and $\mu(T)$ denote the sending consumer's public key pk_c and the amount μ specified in transaction T respectively. We use $\mu^*_{c,t}$ to denote the total funds spent by c in off-chain payments to the operator as of t. The operator verifies an off-chain payment transaction T received from c at t by evaluating if the sender has sufficient balance to make this transaction (i.e. $\mu(T) \leq D_{c,t}$) and

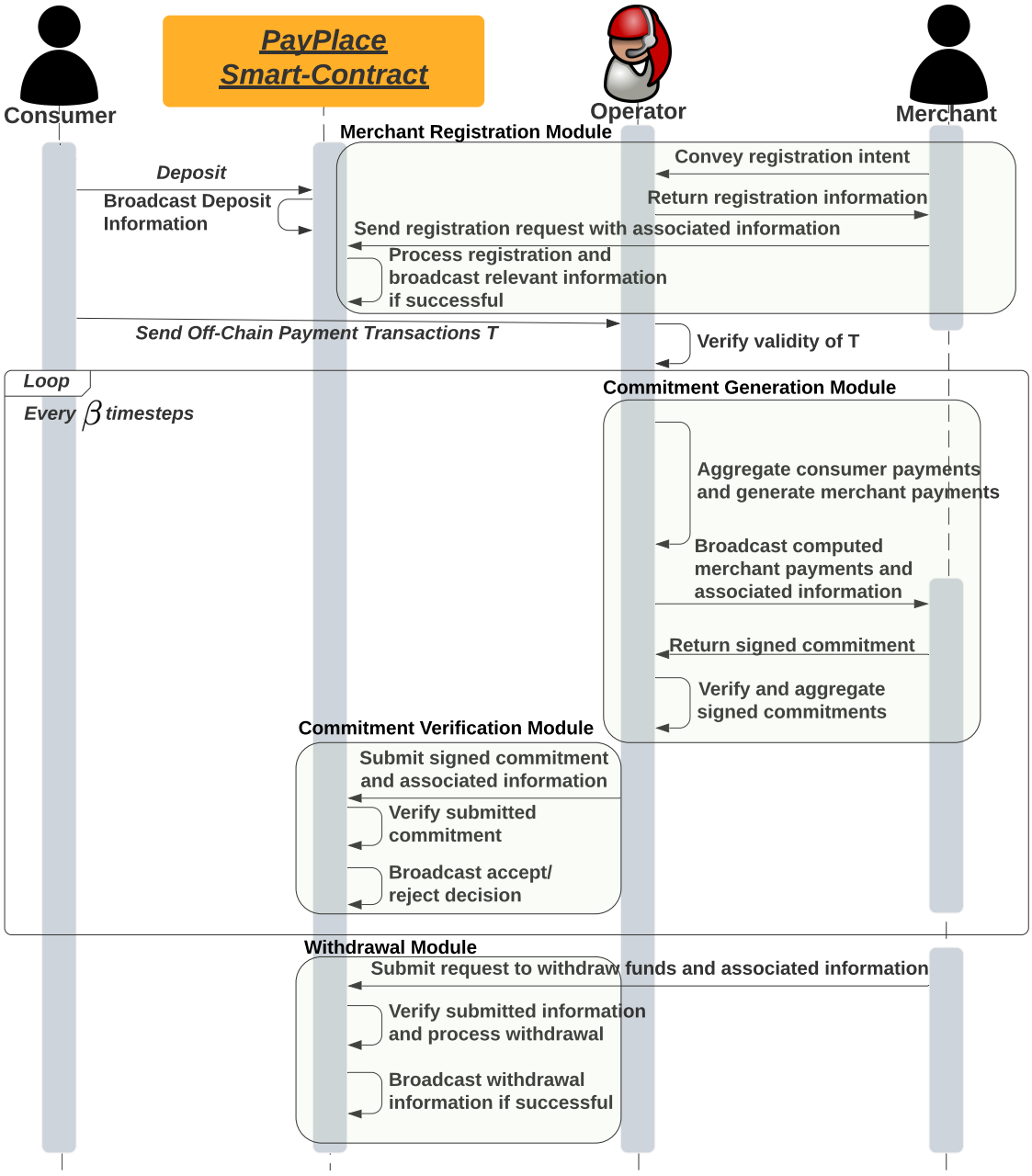


Figure 6.2: Sequence diagram illustrating typical interactions between Consumers, Merchants, the PayPlace contract and the Operator.

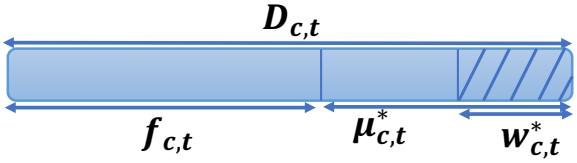


Figure 6.3: State of consumer c's channel with the operator at time t.

ensuring that the operator balance in c's payment channel only increases as a result of T (i.e. $\mu^*_{c,t} \leq \mu(T)$). The operator also verifies the digital signature σ with verification function $\mathcal{V}$; $\mathcal{V}(pk_c, T, \sigma) = 1$ if sk_c was used to sign T to yield σ. We use $\mathbb{C}_t$ to denote the set of the last off-chain transaction received by the operator from each consumer as of time t (reflecting the operator-owned balance in each virtual channel with a consumer as of t).

Consumers are not permitted to withdraw funds already deposited in the channel (e.g. akin to topping up a store card). Then, a consumer's *confirmed funds* at time t is simply their total deposits less off-chain payments, i.e., $f_{c,t} = D_{c,t} - \mu^*_{c,t}$ (Figure 6.3). We let $w^*_{c,t}$ denote the total funds withdrawn (by the operator and merchants) against the operator-owned portion of c's deposit (assigned via off-chain transactions by c to the operator) as of time t. We show in Section 6.5.2 that $w^*_{c,t}$ never exceeds $\mu^*_{c,t}$; i.e. *Consumer Safety* is guaranteed. As shown in Figure 6.2, for convenience, the contract may broadcast (through a mechanism like Ethereum Events) the updated value of $D_{c,t}$ (and $w^*_{c,t}$) when it processes a consumer deposit (or withdrawals against a consumer's funds, correspondingly). Such broadcasted data is also written out to logs.

Operator The operator holds consumers' off-chain payments custodial and forwards them *off-chain* to appropriate merchants every β timesteps. Note that to start receiving payments from the operator, merchants must register first by performing a one-time registration ceremony that involves the operator and the smart-contract. This is depicted as the **Merchant Registration Module** in Figure 6.2 which shows key entities and interactions in PayPlace, and explained in detail in Section 6.5.2. Every β time slots, the operator periodically consolidates payments owed to each registered merchant and generates payment transactions T'. Here, $T' = (\mu', pk_p, pk_c)$ where μ' represents the *total amount* owed by the operator to merchant p based on orders from consumer c since the time p last withdrew her funds on the root-chain. We abuse notation and use $\mu(T')$ and $s(T')$ to denote the payment amount μ' and the referenced source consumer pk_c in T'. Every β timesteps, a merchant p hence receives

an off-chain transaction T' for each consumer whose order(s) p has fulfilled since p last withdrew her funds on the root-chain; we use $\mathbb{T}_p$ to denote these transactions. The PayPlace smart-contract allows merchants to later withdraw funds assigned to them in such off-chain payment transactions T' directly from the deposit of the corresponding consumer $s(T')$, thereby enabling *Single-Source Liquidity*. Note that a successful withdrawal at t by a merchant p transfers all of p's confirmed funds $f_{p,t}$ to p.

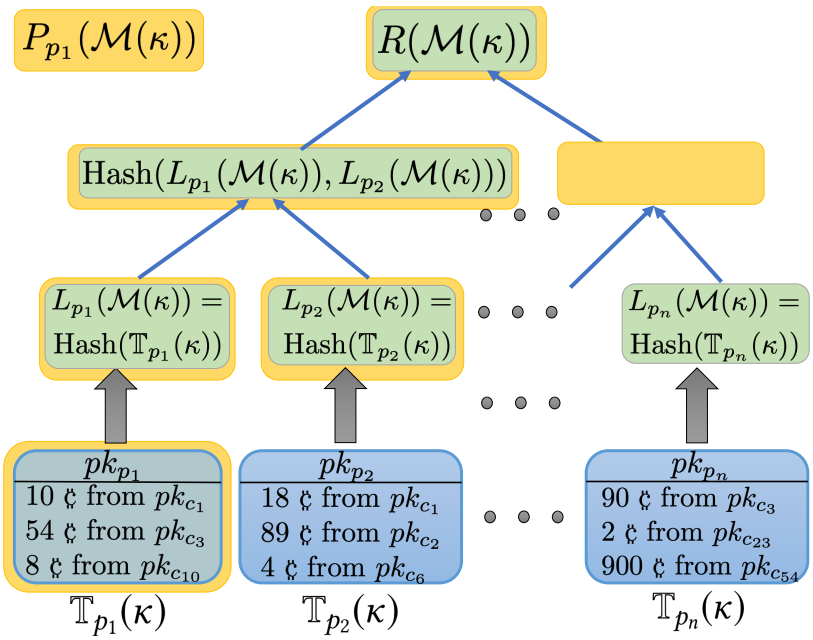


Figure 6.4: Transactions in $\mathbb{T}_p$ reflect payments that p is owed from different consumers. $\mathbb{T}_p$ is hashed to a leaf node $L_p(\mathcal{M})$ in the Merkle tree. A succinct Merkle proof of inclusion $P_p(\mathcal{M}(\kappa))$ for $\mathbb{T}_p(\kappa)$ can be given using the values of nodes with yellow borders.

After computing $\mathbb{T}_p$ for all p, the operator generates a "block" $\kappa = (\mathbb{T}, \mathcal{M})$ that consists of the set $\mathbb{T} = \bigcup_{\forall p} \mathbb{T}_p$ and a Merkle tree $\mathcal{M}$. Note that $\mathbb{T}_p$ is an element of $\mathbb{T}$. We use $\mathbb{T}(\kappa)$ to denote the set $\mathbb{T}$ in κ and $\mathcal{M}(\kappa)$ to denote the Merkle tree $\mathcal{M}$ included in κ; $\mathcal{M}(\kappa)$ is generated from $\mathbb{T}(\kappa)$ and its root is denoted by $R(\mathcal{M}(\kappa))$. Each leaf $L_p(\mathcal{M})$ in the Merkle tree $\mathcal{M}(\kappa)$ corresponds to the hash of the set of payment transactions $\mathbb{T}_p(\kappa)$ for a merchant p (illustration in Figure 6.4). The operator also includes a similar set of transactions $\mathbb{T}_\omega$ assigned to herself, reflecting any commission retained from consumers' payments for providing the PayPlace service. Since $\mathbb{T}_\omega$ is identical to any other $\mathbb{T}_p$, we do not differentiate between the operator and the merchant when referring to the payees of a block, unless required. Finally, we denote the Merkle proof [90] of $\mathbb{T}_p(\kappa)$ by $P_p(\mathcal{M})$, i.e. $P_p(\mathcal{M})$ proves that $\mathbb{T}_p(\kappa)$ corresponds

to $L_p(\mathcal{M})$ and that $L_p(\mathcal{M})$ is a leaf of a Merkle tree with root $R(\mathcal{M}(\kappa))$. The set of merchants that have leaves in $\mathcal{M}(\kappa)$ is denoted by $\mathbb{P}(\kappa)$.

Numerical Example of the PayPlace model. We now provide a numerical example illustrating off-chain payment transactions in PayPlace. To illustrate the consumer and provider transaction models in PayPlace, consider $t = t_1, \ldots, t_8$. Consumers c_1 and c_2 deposit 30₹ and 20₹ respectively into the PayPlace smart-contract at time t_1. Suppose c_1 additionally deposits another 40₹ at t_3. Then, $D_{c_1,t} = 30\ \forall t \in [t_1, t_3)$ and $= 70\ \forall t \in [t_3, t_8]$, and $D_{c_2,t} = 20\ \forall t \in [t_1, t_8]$. To send 10₹ to the operator at t_4, c_1 generates a transaction $T_a = (10, pk_\omega, pk_{c_1})$ and signature $\sigma_{T_a} = \mathcal{S}(T_a, sk_{c_1})$; then $\mu^*_{c_1,t_4} = 10$. Suppose c_2 similarly sends 10₹ to the operator at t_4 via transaction T_b. To send another 10₹ at t_5, c_1 generates a transaction $T_c = (20, pk_\omega, pk_{c_1})$ and the corresponding σ_{T_c}; then $\mu^*_{c_1,t_5} = 20$. At t_5 then, the latest transactions $\mathbb{C}_{t_5} = \{T_b, T_c\}$ and the remaining funds available for consumers c_1 and c_2 to use at t_5 is $f_{c_1,t_5} = 50$ and $f_{c_2,t_5} = 10$ respectively. Suppose two registered providers p_1 and p_2 participate in the system, with no operator fees, and the orders for T_a and T_b are fulfilled by p_1 while T_c is fulfilled by p_2. At t_8, the operator generates the block $\kappa = (\mathbb{T}, \mathcal{M})$, where $\mathbb{T}(\kappa) = \{\{(10, pk_{p_1}, pk_{c_1}), (10, pk_{p_1}, pk_{c_2})\}, \{(10, pk_{p_2}, pk_{c_1})\}\}$ (with $\mathbb{T}_\omega(\kappa) = \emptyset$ since the operator did not deduct any fees in this case). We then have $\mathbb{T}_{p_1}(\kappa) = \{\{(10, pk_{p_1}, pk_{c_1}), (10, pk_{p_1}, pk_{c_2})\}$, $\mathbb{T}_{p_2}(\kappa) = \{(10, pk_{p_2}, pk_{c_1})\}$, and $\mathbb{T}_\omega(\kappa) = \emptyset$. Further, the Merkle tree $\mathcal{M}(\kappa) = H(H(\mathbb{T}_{p_1}(\kappa)), H(\mathbb{T}_{p_2}(\kappa)), H(\mathbb{T}_\omega(\kappa)))$, where H is the one-way irreversible hash function used for generating the Merkle tree and the leaves of $\mathcal{M}$ are $L_p(\mathcal{M}) = H(\mathbb{T}_p(\kappa))$, $p = p_1, p_2, \omega$. Suppose pk_{p_1} withdraws their specified funds at t_{10}. Further, suppose the operator generates another block at t_{16} and neither pk_{p_1} nor pk_{p_2} fulfil any additional consumer orders from t_8 to t_{16}. Then the generated block at t_{16} is $\kappa' = (\mathbb{T}', \mathcal{M}')$, where $\mathbb{T}(\kappa') = \{(10, pk_{p_2}, pk_{c_1})\}\}$ (with $\mathbb{T}_\omega(\kappa)$ and $\mathbb{T}_{p_1}(\kappa)$ as $\emptyset$).

Merchants The operator then broadcasts the generated block κ to merchants along with the set $\mathbb{C}_t$, and the current timestamp s_t. Then, merchants verify the block, attest their (BLS) signatures to its commitment (a hash of $R(\mathcal{M}(\kappa))$ and s_t) and send it to the operator. In doing so, *they protect themselves from double-spend attacks by the operator*; a merchant signs the root only if the operator's payments specified in κ to merchants does not exceed what the operator has been assigned from consumers. This directly also ensures *Data Availability*; indeed, a merchant's *confirmed funds* in PayPlace corresponds to funds assigned to her in the last notarized commitment that

she attested her signature on. The notarization process used by the PayPlace smart-contract makes any incremental income specified in a notarized block inaccessible to a merchant unless her signature on the corresponding Merkle root was provided by the operator during notarization. This design also incentivizes merchants to be periodically *active* and participate in the signing process to receive their incremental income for the last β timesteps. Note that each commitment generated by the operator reflects cumulative payments owed to merchants, hence a merchant that fails to participate in one commitment round (e.g. communication links are down) can simply receive the incremental income by participating in the next round. The operator verifies returned merchant signatures on the commitment, aggregates them into a single one, and submits this to the smart-contract for notarization of the off-chain payments made to merchants in this block. This process of computing merchants' payments and acquiring their signatures is referred to as the **Commitment Generation Module** in Figure 6.2 and explained in detail in Section 6.5.2.

Smart-Contract By submitting a new commitment for notarization, the operator triggers the **Commitment Verification Module** of the smart-contract. We use $\mathbb{K}_t$ to denote blocks that have been notarized as of time t, where $\mathbb{K}_t(-1)$ refers to the last notarized block, $\mathbb{K}_t(-2)$ to the last block and so on. If all registered merchants have signed the submitted commitment, indicating that they have verified the validity of their assigned payments, the commitment is accepted. However, we must allow notarization even when some signatures are missing to ensure *Income Certainty*. Hence, the PayPlace contract *reserves* funds that were assigned to non-signing (or "missing") merchants in the last notarized commitment κ_p which included their signature, i.e. corresponding to $\mathbb{T}_p(\kappa_p)$. To do this, however, it requires the operator to *prove that signing merchants* (i.e. whose signatures are included in the submitted commitment) *are aware of the funds that will be set aside by the contract for non-signing ones*. Combined with the design of the contract's **Withdrawal Module** (explained in Section 6.5.2), this ensures that funds assigned to a merchant through a notarized commitment that contains her signature are secure, even if she subsequently becomes inactive. In other words, a merchant's confirmed funds $f_{p,t}$ corresponds to those assigned in $\mathbb{T}_p(\kappa_p)$. PayPlace is hence impervious to the problem of *mass exits*. The vendor actively participates in commitment generation only to receive *additional income*. If malicious operator actions are detected (e.g. the operator fails to generate a commitment), then the merchant stops fulfilling orders that are handled through this

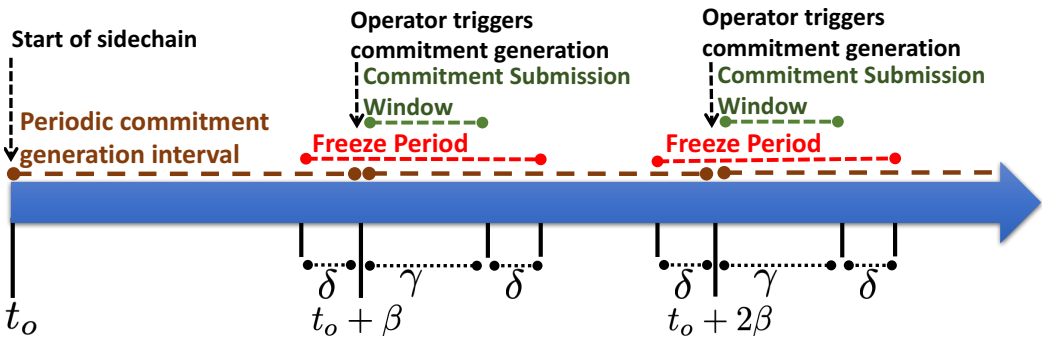


Figure 6.5: Depicting timing events in PayPlace.

operator; a merchant's risk exposure in this semi-custodial model does not exceed the revenue of one notarization period. PayPlace also provides *On-Chain Efficiency*. Only one root-chain transaction per β is required to assign the corresponding payments to merchants regardless of the number of consumer orders and recipients.

Timing Considerations To protect against timing-related attacks on *Consumer* and *Merchant Safety* , we introduce temporal restrictions on merchant and operator actions. Suppose the operator starts the PayPlace system at time t_0; WLOG, we take $t_0 = 0$. Let $\gamma \ll \beta$ denote the (predefined) maximum time required for the operator to generate a new commitment for notarization, and $\delta \ll \beta$ the maximum finality time for the root blockchain. The smart-contract enforces *freeze periods* during which funds may not be withdrawn and new merchants may not register, namely, $[\beta-\delta, \beta+\gamma+\delta]$, $[2\beta-\delta, 2\beta+\gamma+\delta]$, $[3\beta-\delta, 3\beta+\gamma+\delta]$, ... The operator may submit new commitments for notarization only during the *commitment submission window* within each freeze period, given by $[\beta, \beta+\gamma]$, $[2\beta, 2\beta+\gamma]$, $[3\beta, 3\beta+\gamma]$, ... Figure 6.5 depicts the timing of these events, which can be triggered and managed using decentralized oracles [53]. A withdrawal triggered outside a freeze period is immediately processed by the PayPlace smart-contract (without imposing any wait-times) as described in Section 6.5.2, thereby providing *Liveness*.

BLS Signatures PayPlace relies on *signature aggregation* to enable the contract on the root-chain to cheaply verify merchants's signatures on the submitted commitment in constant time. We utilize the BLS (Boneh-Lynn-Shacham) signature scheme [39, 40] based on elliptic curve pairing-based cryptography for this; it provides short signatures that can be securely aggregated. We let $e : \mathbb{G}_0 \times \mathbb{G}_1 \to \mathbb{G}_T$ denote an efficiently computable, non-degenerate pairing where $\mathbb{G}_0$, $\mathbb{G}_1$ and $\mathbb{G}_T$ are groups of prime order q, and g_0, g_1 are generators of $\mathbb{G}_0$, $\mathbb{G}_1$ respectively. Suppose signatures reside in $\mathbb{G}_0$ and

public keys in $\mathbb{G}_1$. H_0 then denotes the hash function that maps from the message space into $\mathbb{G}_0$. $\mathcal{S}(m, sk)$ generates sk's (BLS) signature on m, returning $\sigma = H_0(m)^{sk} \in \mathbb{G}_0$. $\mathcal{V}(pk, m, \sigma)$ verifies if sk signed m to yield σ by evaluating if $e(g_1, \sigma) = e(pk, H_0(m))$, and returning 1 in that case. We use the multiplicative notation for groups, and references to PKI credentials and signatures mean BLS, unless otherwise stated. Using BLS signatures, only *one signature verification* (two bilinear pairings) is required to check whether all required signers (represented by their aggregate public key) have signed the presented aggregate signature. PayPlace's design, however, goes further to allow the smart-contract to determine exactly which merchants have not signed a commitment despite storing only an aggregated public key of registered merchants (as opposed to each key individually, to save on expensive storage resources). Identifying these non-signing merchants enables the contract to safeguard their previously assigned funds from possible misappropriation in the submitted commitment.

We now also briefly describe the rogue public-key attack on BLS signatures which is especially of importance in PayPlace. Consider a set of n keys $K = \{(sk_i, pk_i) : 1 \leq i \leq n\}$ whose public keys and signatures are to be aggregated via BLS. An attacker who knows the public keys in K can choose some $\beta \in \mathbb{Z}_q$ where q is of prime order and compute a false public key $pk_{\text{att}} = g_1^{\beta} * (\prod_{u=1}^{n} pk_u)^{-1}$ where g_1 is a generator for group $\mathbb{G}_1$ of prime order q. The aggregate public key computed by a verifier is $pk_a = pk_{\text{att}} * \prod_{u=1}^{n} pk_u$. The attacker can then declare the signature $\sigma_{a,m} = H_0(m)^{\beta}$ (where H_0 is a random oracle mapping into $\mathbb{G}_0$ which is also of prime order q) and convince the verifier that this has been signed by all n sk_i's as well as pk_{att}'s sk_{att}. To see this, note that verification of $\sigma_{a,m}$ requires checking if $e(g_1, \sigma_{a,m}) = e(pk_{\text{att}} * \prod_{u=1}^{n} pk_u, H_0(m))$ where e is a pre-specified non-degenerate bilinear function ($e : \mathbb{G}_0 \times \mathbb{G}_1 \to \mathbb{G}_T$). But $e(g_1, \sigma_{a,m}) = e(g_1, H_0(m)^{\beta})$ as declared by the attacker to the verifier, and $e(g_1, H_0(m)^{\beta}) = e(g_1^{\beta}, H_0(m))$, and $g_1^{\beta} = pk_{\text{att}} * \prod_{u=1}^{n} pk_u$ by definition of pk_{att}.

6.5 Protocol Details

6.5.1 Smart-Contract State

The PayPlace smart-contract tracks a minimal amount of information, as specified in Table 6.2. Note that we refer to list elements by their indices depending on usage. At

Name	Description
pk_ω	Public key of the operator
apk	Aggregate public key of merchants
apk_a	Aggregate public key of merchants whose signature was included in the last notarization
g	Time that the last commitment generation event was to be triggered by the operator
s	Time that the last notarized commitment was submitted
$R(\mathcal{M}(\mathbb{K}_t(-1)))$	Merkle root of the last notarized commitment as of t
$\mathbb{X}$	Pub keys of merchants who exited/unregistered after the last block was notarized
$\mathbb{B}$	Pub keys of merchants who registered after the last block was notarized
$\mathbb{W}$	Pub keys of merchants who withdrew their funds after the last block was notarized
$\mathbb{M}$	Pub keys of registered merchants whose signatures were not included in the last notarized commitment and the number of consecutive commitments that each of these merchants has missed signing so far
$\mathbb{N}$	Tracks the amount of funds merchants in $\mathbb{M}(-x)$ have been assigned from pk_c, $\forall c$ whom $p \in \mathbb{M}(-x)$ have been assigned funds from in the last notarized block $\mathbb{K}_t(-x+1)$ that had their signature, $\forall x \in [1, \eta]$
$\mathbb{L}$	Hash of payment transactions assigned to missing providers ($p \in \mathbb{M}$) in the κ_p

Table 6.2: State of the PayPlace smart-contract, representing the information it tracks

any time t, the contract stores two aggregate public keys of merchants and only the last notarized block's Merkle root $R(\mathcal{M}(\mathbb{K}_t(-1)))$. Further merchants in $\mathbb{X} \cup \mathbb{W} \cup \mathbb{B}$ have zero confirmed funds and those in $\mathbb{W} \cup \mathbb{B}$ wait until the next notarization to acquire new payments as confirmed funds, if any.

The public keys of registered merchants whose signatures were missing from the last notarized commitment are saved in $\mathbb{M}$. The contract also tracks the number of notarized commitments that these merchants have consecutively missed. We use $\mathbb{M}(-x)$ to refer to merchants whose signatures have been absent since the last $x = 1, 2, \ldots$ commitments. Let η denote the maximum value of x (i.e. the maximum number of consecutive commitments missed by a non-signing merchant in $\mathbb{M}$); note that $\mathbb{M} = \bigcup_{x=1}^{x=\eta} \mathbb{M}(-x)$. Note that the last notarized block κ_p that was signed by a missing merchant $p \in \mathbb{M}(-x)$ is simply $\mathbb{K}_t(-x+1)$ by definition of $\mathbb{M}(-x)$. The contract also tracks the public keys of consumers whom these missing merchants were assigned funds from in the last notarized commitment that they signed (i.e. $\mathbb{Y}_{-x} = \{s(T'), \forall T' \in \mathbb{T}_p(\mathbb{K}_t(-x+1)), \forall p \in \mathbb{M}(-x)\}$) , as well as the total amount that they had been assigned from each consumer (i.e. $\mu_{-x}^{pk_c} = \sum_{p \in \mathbb{M}(-x)} \sum_{T' \in \mathbb{T}_p(\mathbb{K}_t(-x+1))} \mathbb{1}_{s(T')=pk_c} \mu(T')$). We use $\mathbb{N}$ to denote the resulting set of balances $\{\mu_{-x}^{pk_c}, \forall pk_c \in \mathbb{Y}_{-x}, \forall x \in [1, \eta]\}$, and use $\mathbb{N}(-x)^{pk_c}$ to refer to $\mu_{-x}^{pk_c}$. For each merchant $p \in \mathbb{M}$, the contract also stores in list $\mathbb{L}$ the leaf node $L_p(\kappa_p)$ assigned to her in the last notarized block that had their signature. Note that the smart-contract's state is also accessible to anyone traversing the blockchain, and updates to these state values can be broadcast by the contract as well. We illustrate an example usage of states $\mathbb{M}$ and $\mathbb{N}$. Suppose $\mathbb{M}(-1) = \{pk_a\}$, $\mathbb{M}(-2) = \{pk_b, pk_c, pk_d\}$ and that pk_a, pk_b, pk_c, pk_d had 10₿, 30₿, 40₿, 50₿ sourced from pk_{c1},pk_{c5},pk_{c1},pk_{c5} respectively. Then $\mathbb{N}(-1) = \{(pk_{c1}, 10)\}$, $\mathbb{N}(-2) = \{(pk_{c1}, 40), (pk_{c5}, 80)\}$. Further, $\mathbb{N}(-1)^{pk_{c1}} = 10₿$, $\mathbb{N}(-2)^{pk_{c1}} = 40₿$ and $\mathbb{N}(-2)^{pk_{c5}} = 80₿$. Finally, $\mathbb{M} = \{pk_a, pk_b, pk_c, pk_d\}$ in this example.

For ease of protocol description, we assume that consumer information (i.e. pk_c, $D_{c,t}$, $w_{c,t}^*$ for all c) is part of the smart-contract's stored state. However, this information is only required when consumers deposit additional funds into the contract or merchants initiate withdrawal of assigned funds. These events are considerably infrequent in comparison with the periodic notarization events. This information can therefore be moved off-chain and instead represented just by a hash. If storage is significantly more expensive than compute on the underlying DLT, we can optimize for storage at higher computational cost. Similar to the storage mechanism proposed in ZK Rollup [11],

the contract can track just two Merkle roots instead of individual consumer data: one for a Merkle tree A of registered consumers' public keys pk_c; and the other for a Merkle tree B of tuples $(D_{c,t}, \mu'_{c,t}, w^*_{c,t})$, such that pk_c that is stored in the n-th leaf of A corresponds to $(D_{c,t}, \mu'_{c,t}, w^*_{c,t})$ in the n-th leaf of B. Consumer registrations can use the same process as ZK Rollup [11]; however, note that these functions are now more compute-intensive since each requires the contract to verify Merkle proofs. Similarly, provider and operator withdrawals require computation to verify the Merkle proofs for the consumer's balance in B against which they attempt withdrawal. However, all changes to B can be broadcast to others (stored in blockchain's logs), hence all providers and operators can compute the leaves and Merkle proofs in B for any consumer, even if they have been offline.

In exchange, marginally more computational work is expended when consumer information is required (during consumer top-ups or merchant withdrawals).

6.5.2 Detailed Protocol Specification

We now explain the core PayPlace modules (cf. Figure 6.2) in detail and illustrate how they fulfil the security properties identified in Section 6.2.1. We omit the specification of state updates for variables in Table 6.2 that are straightforward. For instance, when a merchant p successfully registers with the PayPlace smart-contract, we do not explicitly state the addition of p to $\mathbb{B}$. $\mathbb{B}$ tracks this information by definition. We ensure that the PayPlace contract is provided sufficient information during registration, notarization and withdrawal processing to ensure that updates to these variables can be correctly executed.

Merchant Registration To register, merchant p first sends a Proof of Possession (PoP) of her credentials to the operator. In other words, the merchant uses her secret key sk_p to sign her public key, generating $\sigma_{p,\text{init}} = \mathcal{S}(pk_p, sk_p)$. If the operator successfully verifies p's signature on the PoP, i.e. $\mathcal{V}(pk_p, pk_p, \sigma_{p,\text{init}}) = 1$, it signs the tuple of p's public key and current timestamp τ_r, i.e., ω generates $\sigma_{\omega,p} = \mathcal{S}(m{=}(pk_p, \tau), sk_\omega)$. The operator returns $\sigma_{\omega,p}$ to the merchant, who provides it along with τ_r to the smart-contract's enrollment function. Algorithm 3 shows the contract's registration processing function; the **Data** field in the Algorithm denotes relevant internal state of the executing entity (the contract in this case). Let t denote the time when the contract receives the registration request (note $t > \tau_r$). Merchant registrations are processed

Algorithm 3: Merchant registration by Contract

Input : $\sigma_{\omega,p}$, τ_r, verified public-key of the caller pk_p
Output : 0 or 1
Data: s, g, $\mathbb{B}$, $\mathbb{X}$, $\mathbb{M}$, $\mathbb{W}$, apk, current time t

1 **if** *not* $(g + \gamma + \delta < t < g + \beta - \delta$ *and* $s < \tau_r)$ **then**
2 **return** 0
3 **if** $pk_p \in \mathbb{B} \cup \mathbb{X} \cup \mathbb{M} \cup \mathbb{W}$ **then**
4 **return** 0
5 **if** $\mathcal{V}(pk_\omega, m{=}(pk_p, \tau_r), \sigma_{\omega,p}) = 1$ **then**
6 $apk = \perp \; ? \; apk = pk_p : apk = apk \cdot pk_p$
7 **return** 1

only when no freeze windows are currently active (cf. Figure 6.5 in Section 6.4) and no commitment has been notarized since the time reflected in the provided timestamp (Lines 1-3). As long p is not known to have already registered (i.e. $p \notin \mathbb{B} \cup \mathbb{M} \cup \mathbb{W}$), or deregistered only since the last notarization (i.e. $p \notin \mathbb{X}$), and the provided signature on pk_p and τ_r is valid, the registration is successful and the contract updates the aggregated public key to include p's key (Lines 3-7). The operator ω must also register using this function to assign any portion of consumer payments that it retains as a fee to itself; it is treated like any other merchant with respect to the notarization and withdrawal of its funds. Note that PayPlace does not rely on the root-chain to support BLS account keys though this specification assumes for readability. If the root-chain does not support BLS account keys (i.e., the keys used for signing blockchain transactions do not support BLS operations), the merchant must explicitly specify their BLS public key pk_p to the smart-contract during registration and provide a PoP for it. This PoP is denoted by $\sigma_{p,BLS} = \mathcal{S}(m{=}(pk_p, pk'_p), sk_p)$, where pk'_p is p's root-chain account key. Since the account credentials on the root chain are different from the ones used for signing in PayPlace, the merchant must provide $\sigma_{p,BLS}$ for any transaction (e.g., withdrawals) with the smart-contract to prove its identity.

Lemma 8. *p can register only once unless ω colludes with p.*

Proof. First, we show that a public key pk_p can be registered only by its owner p (who knows sk_p). Let p's public and private keys on the root chain (i.e. account

keys that are used for signing blockchain transactions) be denoted by pk'_p and sk'_p respectively. Consider $(pk'_p, sk'_p) = (pk_p, sk_p)$, i.e. the root chain supports BLS account keys which can also be used with PayPlace. By design of the underlying blockchain, miners process transactions only if the transaction is signed by the stated sender, in this case, pk_p; hence registering a provider implicitly provides Proof of Possession (PoP) to the blockchain. However, this can be attacked with an oracle [148] $\text{OMSign}(pk_i, msg)$ that returns msg signed by sk_i. In that case, the attacker with a maliciously computed $pk_{\text{att}} = g_1^{\beta} * (\prod_{i=1}^{n} pk_i)$ may provide a signed transaction on the root chain by computing $\sigma_{m,\text{att}} = H_0(m)^{\beta} / \prod_{i=1}^{n} \text{OMSign}(pk_i, m)$ where m is the transaction to be submitted to the blockchain calling the contract's registration function and registering pk_{att} as a vendor, i iterates over the registered providers, and some $\beta \in \mathbb{Z}_q$. In PayPlace, however, this requires that all registered providers (even honest ones) sign this transaction asking for pk_{att}'s enrolment, which they have no reason or incentive to do.

Consider the other case where $(pk'_p, sk'_p) \neq (pk_p, sk_p)$. By design of the underlying blockchain, miners process transactions only if the transaction is signed by the stated sender, in this case, pk'_p; hence registering a provider implicitly provides Proof of Possession (PoP) to the blockchain. It then suffices to show that pk'_p's owner also owns pk_p. First, we note that the provided PoP $\sigma_{p,BLS}$ establishes that pk_p is not a rogue public key. Even if OMSign is available, employing separate hash functions for signing POP messages and other messages [148] guarantees resilience of the provided PoP to the rogue public-key attack. Second, since a provider p generates this PoP by signing the combined hash of their root account public key pk'_p and BLS key pk_p, if $\mathcal{V}(pk_p, H(pk_p, pk'_p), \sigma_{p,BLS}) = 1$, then pk_p's owner is the owner of pk'_p.

Finally, note that the contract rejects the registration unless $\sigma_{\omega,p}$ was generated after the latest freeze period (using the provided timestamp τ_r. Hence, $\sigma_{\omega,p}$ provided by the operator to p for registration is only valid until the next freeze window begins. If p registers successfully with $\sigma_{\omega,p}$, it cannot register again (with $\sigma_{\omega,p}$) even in the current open window since pk_p is added to $\mathbb{B}$. The only way p can register multiple times is if a colluding operator, knowing that p has already registered, waits for the next notarization to succeed (which clears $\mathbb{B}$) and generates $\sigma_{\omega,p}$ with the latest τ_r and p again calls the registration function with this. □

Even if a colluding operator generates a $\sigma_{\omega,p}$ to allow an already registered merchant

to maliciously re-register, the resulting corruption to *apk* does not affect *confirmed funds* of any participant, as we later show.

Commitment Generation The block generation process executed by the operator every β timesteps follows the description in Section 6.5.1 and is shown in Algorithm 5. As shown in the **Data** field of Algorithm 5, the operator keeps track of $\mathbb{K}_t(-1)$, $\mathbb{W}$,$\mathbb{B}$ and the set $\mathbb{R}$ of registered merchants in the system. After generating κ, the operator broadcasts κ, $\mathbb{C}_t$ and the current timestamp τ to merchants. Algorithm 4 details the process used by merchants for verifying the validity of κ and signing it. Note that $\mathbb{T}_m(\kappa_m)$ for all $m \in \mathbb{M}$ is known to all merchants (though not stored by the contract) since it is broadcast by the contract's commitment verification module (explained later). The binary flag hasWithdrawn is 1 if the verifying merchant p withdrew their funds after the last notarization, i.e. $p \in \mathbb{W}$. p verifiess that a commitment window is active (Lines 1-3), and that her confirmed funds does not decrease in this block as long as $p \notin \mathbb{W}$ (Line 4-8). The merchant further ensures that a valid *source transaction*, i.e. an off-chain payment from the consumer to the operator, accompanies each operator-generated payment transaction (Line 9-13). Finally, the merchant guards herself against *double-spend attacks* from the operator by verifying that, for each consumer who is listed as the source for a stated payment to p, the sum of payments promised to other merchants with this consumer as the source does not exceed the operator-owned balance in the consumer's virtual channel (Lines 14-22). If these checks succeed and $\mathcal{M}(\kappa)$ is correctly generated from $\mathbb{T}(\kappa)$ (Lines 23-28), p signs the tuple $(R(\mathcal{M}(\kappa)), \tau)$ and returns it.

Let set $\mathbb{A}_t$ and $\mathbb{M}_t$ respectively denote the public keys (or corresponding indices, based on usage) of registered merchants who return the signed commitment to the operator within timeout duration $\gamma' < \gamma$ and those who do not. Here, γ' is set by the operator such that $\gamma - \gamma'$ is sufficient duration for the operator to perform the remaining steps and submit the commitment to the root-chain. Let set $\mathbb{X}_t$ denote the public keys of previously-registered merchants who had deregistered (i.e. exited) since the last block was notarized (which may be different from $\mathbb{X}$ tracked by the contract, as we see later). After verifying received signatures, the operator computes an *aggregated root signature* $ars_\kappa = \prod_{p \in \mathbb{A}_t} \sigma_{p,\kappa}$, and submits the following commitment to contract for notarization of κ: $R(\mathcal{M}(\kappa))$, τ, $\mathbb{X}_t$, and information on signing and missing merchants. The *signing merchant information* consists of their aggregated public key $apk_{\text{active}} = \prod_{p \in \mathbb{A}_t} pk_p$ and ars_κ. Further, for signing merchants whose

Algorithm 4: Commitment Signing by registered merchant

Input : κ, $\mathbb{C}_t$, τ
Output : $\sigma_{p,\kappa}$ or $\perp$
Data: $\mathbb{T}_p(\kappa_p)$, $\mathbb{M}$, $\mathbb{T}_m(\kappa_m) \forall m \in \mathbb{M}$, $\mathbb{W}$, g,$D_{c,t}$ for each c, $w^*_{c,t}$ for each c, current time t, hasWithdrawn=$\{0, 1\}$

if *not* $g < \tau \leq t < g + \gamma$ **then**
  return $\perp$
consPay, consOpBal= {}
for $T' = (\mu', pk_p, pk_c)) \in \mathbb{T}_p(\kappa)$ **do**
  if *hasWithdrawn*=0 **then**
    T'' =getTransaction$(pk_c, \mathbb{T}_p(\kappa_p))$ ▷ Cf. Alg 5
    if $T''! = \perp$ *and* $\mu(T'') > \mu'$ **then**
      return $\perp$
  T =getSourceTransaction$(pk_c, \mathbb{C}_t)$
  if $T = \perp$ **then**
    return $\perp$
  consOpBal$[pk_c] = \mu(T)$
  consPay$[pk_c] = \mu'$
regMissingMerchants = $\mathbb{M} - \mathbb{W}$
if *not* $[$*regMissingMerchants* $\subseteq \mathbb{P}(\kappa)$ *and* $(\mathbb{T}_m(\kappa_m) \subseteq \mathbb{T}_m(\kappa), \forall m \in$*regMissingMerchants*$)]$ **then**
  return $\perp$
for $T' = (\mu', pk_p, pk_c)) \in (\mathbb{T}(\kappa) \setminus \mathbb{T}_p(\kappa))$.*flatten* **do**
  T =getSourceTransaction$(pk_c, \mathbb{C}_t)$
  if $s(T) \in$ *consPay.keys* **then**
    consPay$[pk_c]$ += $\mu(T)$
if *consPay*$[pk_c] >$ *consOpBal*$[pk_c] - w^*_{c,t}$ **then**
  return $\perp$
$\mathcal{M}' =$ merklize$(\mathbb{T}(\kappa))$
if $\mathcal{M}' = \mathcal{M}(\kappa)$ **then**
  $\sigma_{p,\kappa} = \mathcal{S}(m = (R(\mathcal{M}(\kappa)), \tau), sk_p)$
  return $\sigma_{p,\kappa}$
else
  return $\perp$
Def `getSourceTransaction(`pk_c, $\mathbb{C}_t$`):`
  for $(T, \sigma) \in \mathbb{C}_t$ **do**
    if $s(T) = pk_c$ *and* $\mathcal{V}(pk_c, T, \sigma) = 1$ *and* $\mu(T) \leq D_{c,t}$ **then**
      return T
  return $\perp$

Algorithm 5: Block Generation by the Operator

Input : Set O_t with elements of type (μ', pk_p, pk_c) reflecting the total value μ' of c's orders from the last γ timesteps fulfiled by p

Output : κ

Data: current time t, $\mathbb{K}_t(-1)$, $\mathbb{W}$, $\mathbb{B}$, $\mathbb{R}$

for $p \in \mathbb{R}$ **do**
  if $p \notin \mathbb{W} \cup \mathbb{B}$ **then**
    $\mathbb{T}_p = \mathbb{T}_p(\mathbb{K}_t(-1))$
  else
    $\mathbb{T}_p = \{\}$
  for $(T' = (\mu', pk_p, pk_c)) \in O_t$ **do**
    T'' =getTransaction$(s(T'), \mathbb{T}_p)$ ▷ Pass by ref
    if $T'' = \perp$ **then**
      $\mathbb{T}_p.insert(T')$
    else
      $\mu(T'') = \mu(T'') + \mu(T')$
$\mathbb{T} = \bigcup_{p \in \mathbb{R}} \{\mathbb{T}_p\}$
$\mathcal{M}$ = merklize($\mathbb{T}$) ▷ Generates a Merkle tree whose leaves are hashes of elements in the input set.
$\kappa = (\mathbb{T}, \mathcal{M})$
return κ
Def `getTransaction(`pk_c, $\mathbb{T}_p$`):`
  for $T' \in \mathbb{T}_p$ **do**
    if $s(T') = pk_c$ **then**
      return T' ▷ Pass by ref
  return $\perp$

signatures were missing in the last notarized commitment, i.e. $p \in \mathbb{M}$ and $p \in \mathbb{A}_t$, the operator provides $\mathbb{T}_p(\kappa_p)$. The *missing merchant information* consists of $\mathbb{M}_t$; for each missing merchant who signed the previous notarized block, i.e. $p \in \mathbb{M}_t | \kappa_p = \mathbb{K}_t(-1)$, the operator also includes: $\sigma_{p,\text{init}}$ (collected during registration), $\mathbb{T}_p(\kappa)$, $P_p(\mathcal{M}(\kappa))$, $\mathbb{T}_p(\mathbb{K}_t(-1))$, and $P_p(\mathcal{M}(\mathbb{K}_t(-1))$. If the commitment is not submitted within γ, the operator must wait for the next commitment generation event.

Commitment Verification When the operator submits a new block commitment at t, the smart-contract performs the verification steps described in Algorithm 6, where return values of 0 and 1 indicate commitment rejection and acceptance respectively. For merchants whose signatures were not included in the last notarized block but included in the current one, the contract requires the notarized payment transactions that they had last signed for (Lines 3-6). Since these merchants must be removed from $\mathbb{M}$, $\mathbb{N}$ and $\mathbb{L}$, this provides the contract with necessary information to correctly update the state. If all registered merchants have signed the tuple of the provided Merkle root and timestamp τ, the verification immediately succeeds (Lines 9-10) since their signature conveys that their double-spend and safety checks on the generated commitment succeeded (cf. Algorithm 4). If, however, only a subset of registered merchants have signed the commitment, then extra steps (Lines 11-26) are needed to ensure merchant safety and data availability for those whose signatures are not included.

First, merchants who recently exited must be removed from *apk*. Though the contract hence stores their public keys in $\mathbb{X}$, the contract cannot determine which exited merchants were registered. Since the contract neither tracks individual keys of registered merchants nor validates the full Merkle tree corresponding to a committed root, the operator may well assign payments even to unregistered merchants in a generated Merkle tree (they are now guaranteed security of their notarized funds or protected from data availability attacks). The contract must update *apk* to remove merchants who have exited while retaining registered merchants; however, requires knowing 1) which exited merchants were registered, and 2) the public keys of remaining merchants to recompute *apk*. Second, registered (non-exited) merchants who have not signed the submitted commitment must be identified so that their previously assigned funds can be secured against any malfeasance in this commitment.This is challenging for similar reasons; the individual keys of merchants are not stored. Algorithm 6 is designed to efficiently overcome this problem.

Algorithm 6: Commitment Verification

Input : $R(\mathcal{M}(\kappa))$, τ, $\mathbb{X}_t$, apk_{active}, ars_κ, $\mathbb{M}_t$, $(\sigma_{p,\text{init}}, \mathbb{T}_p(\kappa), P_p(\mathcal{M}(\kappa))$, $\mathbb{T}_p(\mathbb{K}_t(-1)), P_p(\mathcal{M}(\mathbb{K}_t(-1))))$ for $p \in \mathbb{M}_t | \kappa_p = \mathbb{K}_t(-1), \mathbb{T}_p(\kappa_p)$ for $(p \in \mathbb{A}_t$ and $p \in \mathbb{M})$

Output : 1 or 0

Data: apk, apk_a, $\mathbb{M}$, $\mathbb{N}$, $\mathbb{X}$, $\mathbb{L}$, $\mathbb{B}$, $\mathbb{W}$, $R(\mathcal{M}(\mathbb{K}_t(-1)))$, current time t

if *not* $g'_t < s_t \leq t < g'_t + \gamma$ **then**
  return 0
for $p \in \mathbb{M}$ *and* $p \notin \mathbb{M}_t \cup \mathbb{X}_t$ **do**
  ▷ H is the hash func. used by Merklize
  if *not* $H(\mathbb{T}_p(\kappa_p)) = \mathbb{L}_p$ **then**
    return 0
if $\mathbb{X} \cap \mathbb{M}_t \neq \{\}$ **then**
  return 0
if $\mathcal{V}(apk, m = (R(\mathcal{M}(\kappa)), \tau), ars) = 1$ **then**
  return 1
if $|\mathbb{X}_t| > 0$ *or* $\mathbb{M} \neq \mathbb{M}_t$ **then**
  if $apk_{active} \cdot \prod_{v \in \mathbb{M}_t} v \cdot \prod_{p \in \mathbb{X}_t} p \neq apk$ **then**
    return 0
else
  if $apk_{active} \neq apk_a$ **then**
    return 0
if *not* $[\mathbb{X}_t \subseteq \mathbb{X}$ *and* $\mathcal{V}(apk_{active}, (R(\mathcal{M}(\kappa)), \tau), ars_\kappa) = 1]$ **then**
  return 0
for $p \in \mathbb{M}_t - \mathbb{M} - \mathbb{B} - \mathbb{W}$ **do**
  if *not* $\mathcal{V}(pk_p, pk_p, \sigma_{p,init}) = 1$ **then**
    return 0
  ▷ checkMP verifies a Merkle Proof
  if *not* $[checkMP(P_p(\mathcal{M}(\kappa)), R(\mathcal{M}(\kappa)))$ *and* $checkMP(P_p(\mathcal{M}(\kappa)), R(\mathcal{M}(\mathbb{K}_t(-1))))]$ **then**
    return 0
  if *not* $(\mathbb{T}_p(\kappa) \supseteq \mathbb{T}_p(\mathbb{K}_t(-1)))$ **then**
    return 0
if $|\mathbb{X}_t| > 0$ **then**
  $apk = apk_{\text{active}} \cdot \prod_{p \in \mathbb{M}_t} p$
return 1

The contract first verifies that apk matches the key generated from aggregating the provided keys of active, missing, and exited merchants (Lines 11-16). Then, the contract checks the validity of $\mathbb{X}_t$ and the provided ars_κ against the provided apk_{active} (Lines 17-18). At this point, it is not yet guaranteed that all registered merchants have been accounted for in ars_κ; potential attacks against these checks are demonstrated in the proof of subsequent Theorem 9. Additional checks are hence performed. If missing merchant p is in $\mathbb{M}$, then p's credentials have already been verified through the notarization process for a past commitment and its confirmed funds already secure. If missing merchant $p \in \mathbb{B} + \mathbb{W}$, then p's credentials have already been verified though a recent registration or withdrawal event in the root-chain and she has zero confirmed funds to secure as she joined only after the last block was notarized or withdrew all her funds since. For the rest of the missing merchants, the contract checks that a correct PoP has been provided for each, that the provided Merkle proofs are correct, and that the payment transactions assigned to them in the current commitment is at least equivalent to the transactions assigned in the previous commitment (Lines 19-26). In that case, the verification of the submitted commitment succeeds. Further apk is updated to remove deregistered merchants (Line 27-28). Through these checks, the contract can assess whether all registered merchants have been accounted for in the provided commitment *despite no explicit long-term record of their public keys, balances or payment transactions.* If the commitment is accepted, the contract's state variables (in Table 6.2) are updated as necessary. Note that all non-signing merchants of this commitment are identified are tracked appropriately; if a newly missing merchant had been in $\mathbb{B}$ or $\mathbb{W}$, the contract simply stores the $\mathbb{L}_p = H(\{0\})$ for her. The contract broadcasts any updates to apk as well as any additions and deletions of merchants p to $\mathbb{M}$ and their corresponding $\mathbb{T}_p(\kappa_p)$.

Theorem 9. *Suppose p is a registered merchant whose signature is not included in ars_κ. If the commitment is accepted, p was detected as a non-signing merchant, i.e.$pk_p \in \mathbb{M}_t$.*

Proof. For ease of explanation, we set $\mathbb{X}_t = \emptyset$ in the provided commitment (the proof trivially extends to the case where $|\mathbb{X}_t| > 0$). The contract necessitates $apk_{\text{active}} \cdot \prod_{p \in \mathbb{M}_t} = apk$ to proceed with the commitment verification. First note that registration

of a rogue public key is not possible here, as shown in Lemma 8; hence the typical rogue public key attack on BLS signature aggregation is infeasible here.

If the operator omits a registered provider from apk_{active} or $\mathbb{M}_t$, then $apk_{\text{active}} \cdot \prod_{p \in \mathbb{M}_t} \neq apk$. To omit a registered provider while also ensuring $apk_{\text{active}} \cdot \prod_{p \in \mathbb{M}_t} = apk$, the operator generates an unregistered key-pair $(pk_{\text{att}}, sk_{\text{att}})$ and includes $\sigma_{\text{att},\kappa}$ when generating ars_κ (or may simply set $ars_\kappa = \sigma_{\text{att},\kappa}$). Let $\mathbb{A}'_t \subseteq \mathbb{A}_t$ be the subset of active signing providers whose signatures the operator includes in ars_κ along with $\sigma_{\text{att},\kappa}$. The corresponding aggregate public key that will then verify ars_κ successfully is $pk_{\text{att}} \cdot \prod_{p \in \mathbb{A}'_t} pk_p$. Hence, the operator provides $apk_{\text{active}} = pk_{\text{att}} \cdot \prod_{p \in \mathbb{A}'_t} pk_p$ to the contract. To satisfy the contract's requirement, the operator generates $pk_{\text{missing}} = apk \cdot (apk_{\text{active}})^{-1}$ and sets $\mathbb{M}_t = \{pk_{\text{missing}}\}$. This ensures that contract's check $apk_{\text{active}} \cdot \prod_{p \in \mathbb{M}_t} = apk$ passes; however, note that the contract also requires proofs of possession $\sigma_{p,\text{init}}$ from each $p \in \mathbb{M}_t$. To generate $\sigma_{\text{missing,init}}$ requires computing the secret key sk_{missing} given the public key pk_{missing} (which requires violating the Diffie-Hellman assumption), while it can be trivially provided for legitimately missing providers who relayed their $\sigma_{p,\text{init}}$ at the beginning to acquire $\sigma_{\omega,p}$ for registration. Even if the operator includes legitimately missing providers in $\mathbb{M}_t$, the presence of a POP-less pk_{missing} in $\mathbb{M}_t$ is imminent, which renders the attack unsuccessful. □

Theorem 10. *If a commitment is accepted, apk_{active} and $\mathbb{M}_t$ only consist of merchants who had registered with the PayPlace smart-contract at some earlier time, and $\mathbb{X}_t$ only consists of merchants who had registered at some earlier time and deregistered after the last notarization.*

Proof. For ease of explanation, we set $\mathbb{X}_t = \emptyset$ in the provided commitment (the proof trivially extends to the case where $|\mathbb{X}_t| > 0$). We show that inserting credentials that have not been registered with the PayPlace smart-contract in apk_{active} or $\mathbb{M}_t$ will cause the commitment to be rejected. Consider that the operator generates an unregistered key-pair $(pk_{\text{att}}, sk_{\text{att}})$ and includes $\sigma_{\text{att},\kappa}$ in generating ars_κ (or may simply set $ars_\kappa = \sigma_{\text{att},\kappa}$). Let $\mathbb{A}'_t \subseteq \mathbb{A}_t$ be the subset of active signing providers whose signature the operator includes in ars_κ along with $\sigma_{\text{att},\kappa}$. The corresponding aggregate public key that will then verify ars_κ successfully is $pk_{\text{att}} \cdot \prod_{p \in \mathbb{A}'_t} pk_p$. Hence, the operators provides $apk_{\text{active}} = pk_{\text{att}} \cdot \prod_{p \in \mathbb{A}'_t} pk_p$ to the contract. To satisfy the contract's requirement

that $apk_{\text{active}} \cdot \prod_{p \in \mathbb{M}_t} = apk$, the operator generates $pk_{\text{missing}} = apk * (apk_{\text{active}})^{-1}$ and sets $\mathbb{M}_t = \{pk_{\text{missing}}\}$. This ensures that contract's check $apk_{\text{active}} \cdot \prod_{p \in \mathbb{M}_t} = apk$ passes; however, note that the contract also requires proofs of possession $\sigma_{p,\text{init}}$ from each $p \in \mathbb{M}_t$. To generate $\sigma_{\text{missing,init}}$ requires computing the secret key sk_{missing} given the public key pk_{missing} (which requires violating the Diffie-Hellman assumption), while it can be trivially provided for legitimately missing providers who relayed their $\sigma_{p,\text{init}}$ at the beginning to acquire $\sigma_{\omega,p}$ for registration. Even if the operator includes legitimately missing providers in $\mathbb{M}_t$, the presence of a POP-less pk_{missing} in $\mathbb{M}_t$ is imminent, which renders the attack unsuccessful. Finally, consider that the operator generates $ars_\kappa = \prod_{p \in \mathbb{A}_t} \sigma_{p,\kappa}$ and the corresponding $apk_{\text{active}} = \prod_{p \in \mathbb{A}_t} p$ correctly. It is straightforward to see that inserting unregistered key pk_{att} in a correctly generated $\mathbb{M}_t$ will cause the commitment to be rejected since then $apk_{\text{active}} \cdot \prod_{p \in \mathbb{M}_t} \neq apk$.

For credentials in $\mathbb{X}_t$ of an accepted notarization, note that the contract requires $\mathbb{X}_t \subseteq \mathbb{X}$, where $\mathbb{X}$ consists of merchants who have exited or deregistered the system (using the contract's withdrawal module) since the last notarization. Hence $\mathbb{X}_t$ is guaranteed to consist of merchants who had deregistered after the last notarization. If $\mathbb{X}_t$ consisted of merchants who had not registered at an earlier time with the PayPlace smart-contract, then $apk_{\text{active}} \cdot \prod_{p \in \mathbb{M}_t} \prod_{p \in \mathbb{X}_t} \neq apk$ based on the same reasoning above, in which case the contract would reject the commitment. □

Note that this leaves room for 1) a registered merchant to be simultaneously present in apk_{active} and $\mathbb{M}_t$ of an accepted commitment, and 2) for a deregistered merchant in $\mathbb{X}_t$ to be simultaneously present in apk_{active}, and 2) for a deregistered merchant who had deregistered in earlier notarizations (i.e. not tracked in $\mathbb{X}$) to be present in apk_{active} and/or $\mathbb{M}_t$ of subsequently accepted commitments. These states only occur when a merchant has maliciously registered with the PayPlace smart-contract despite being already registered (by colluding with the operator, cf. Lemma 8). However, Lines 7-8 in Algorithm 6 combined with the construction of our withdrawal module ensures that these apparent corruptions are ineffective in violating *Consumer and Merchant Safety*.

Withdrawal Algorithm 7 describes the procedure used by the contract's withdrawal function to determine the amount of funds to be transferred to a merchant (or operator) when invoked. As input, merchant p submits a set of transactions $\mathbb{T}_p$ that assigns payments to her, a set $\mathbb{C}'_p$ of off-chain transactions between each consumer whose funds p has been assigned payments from and the operator. Note that $\mathbb{C}'_p \subseteq \mathbb{C}_{t'}$,

Algorithm 7: Contract processing p's withdrawal

Input : $\mathbb{T}_p$, $\mathbb{C}'_p$, (optional) $\mathcal{P}_p(\mathcal{M})$
Output : Amount of ₲ to transfer to p
Data: apk, $\mathbb{M}$, $\mathbb{N}$, $\mathbb{X}$, $\mathbb{B}$, $\mathbb{W}$,$\mathbb{L}$, $R(\mathcal{M}(\mathbb{K}_t(-1)))$, $D_{c,t}$ for consumers c, $\mu'_{c,t}$, $w^*_{c,t}$, g'_t, current time t

```
1  fundsToTransferToP ← 0
2  blocksPMissed ← 0
3  if not g'_t + γ + δ < t < g'_t + t_a − δ then
4      return 0
5  if p ∈ 𝕏 ∪ 𝔹 ∪ 𝕎 then
6      return 0
7  if p ∈ 𝕄 then
8      if H(𝕋_p)! = 𝕃_p then
9          return 0
10     blocksPMissed ← x|p ∈ 𝕄(−x)
11 if p ∉ 𝕄 and not checkMP(𝒫_p(ℳ)), R(ℳ(𝕂_t(−1))) then
12     return 0
13 for T' = (μ', pk_p, pk_c) ∈ 𝕋_p do
14     T = getSourceTransaction(pk_c, ℂ'_p)          ▷ Cf. Alg 4
15     if T = ⊥ then
16         return 0
17     if μ(T) > μ'_{c,t} then
18         μ'_{c,t} = μ(T)
19     fundsToTransferToP
        += min{max{μ'_{c,t} − Σ_{i=η}^{blocksPMissed−1} ℕ(−i)^{pk_c} − w*_{c,t}, 0}, μ'}
20 return fundsToTransferToP
```

where $\mathbb{C}_{t'}$ denotes the set of off-chain transaction from each consumer to the operator that was revealed to p as a part of some commitment-generation process at time t'. The merchant must also specify whether the withdrawal is a permanent exit (i.e. deregistration). If p's signature was included in $\mathbb{K}_t(-1)$, p also submits a corresponding Merkle proof for $\mathbb{T}_p$.

Note that a merchant who has already withdrawn funds since the last block was notarized (or has registered only since) cannot maliciously initiate a withdrawal since *all of the merchant's confirmed funds* are transferred to her upon a successful withdrawal (Lines 5-6). If the merchant's signature has not been included in the last notarized block, the function expects that the provided transaction set $\mathbb{T}_p$ is equivalent to what she last signed for (i.e. $\mathbb{T}_p(\kappa_p)$) (Lines 7-10). On the other hand, if the merchant's signature was included in the last notarization, then the contract requires a valid Merkle proof for $\mathbb{T}_p$ against $R(\mathcal{M}(\mathbb{K}_t(-1)))$ (Lines 11-12).

The contract then iterates over each payment transaction $T' \in \mathbb{T}_p$ to determine the total funds to be transferred to p (Lines 13-20). First, each T' must be associated with a valid consumer→operator transaction and associated signature (T, σ) in the provided input $\mathbb{C}'_p$. The contract then ensures that no more funds are withdrawn against the source consumer $s(T')$ than what the consumer has assigned as off-chain funds to the operator. $\mu(T') < \mu(T)$ does not suffice as prior merchant withdrawals may have been processed against this consumer $s(T)$. While total withdrawals against a consumer's channel $w^*_{c,t}$ is known to the contract (since withdrawals happen through it), it may not know the most recent value of the operator-owned balance $\mu^*_{c,t}$ in that channel since consumer payments to the operator happen off-chain. We hence use $\mu'_{c,t}$ to represent the highest operator-owned balance in the channel with c *known to the contract* (as revealed by the consumer→operator source transactions submitted by merchants during withdrawals). Note $\mu'_{c,t} \leq \mu^*_{c,t} \leq D_{c,t}$. Finally, the contract must also secure funds of merchants whose signatures have been missing in the last notarization, since double-spend attacks may have been launched against these merchants in that block (by the operator and colluding merchants). The state $\mathbb{N}$ stored by the contract that tracks missing merchants' total funds against each consumer that they have stake in is used for this. After the withdrawal, the contract's state variables (in Table 6.2) are updated as necessary.

Theorem 11. *PayPlace ensures Consumer Safety.*

Proof. A consumer's confirmed funds $f_{c,t}$ at time t is $D_{c,t} - \mu^*_{c,t}$; hence the cumulative funds withdrawn by merchants or the operator against c's deposit should not exceed $\mu^*_{c,t}$ at any time t. In other words, if $w^*_{c,t}$ denotes the total funds withdrawn against c's deposit as of time t, then we need to show $w^*_{c,t} \leq \mu^*_{c,t}$. As described in the withdrawal module, $w^*_{c,t}$ is tracked and updated by the contract every time a withdrawal against c's funds is made successfully. From Algorithm 7, we see that the maximum funds transferred upon a successful withdrawal by provider p is $\min\{\mu'_{c,t} - \sum_{i=\eta}^{x-1} \mathbb{N}(-i)^{pk_c} - w^*_{c,t}, \mu(T')\}$. Let w_p denote this value. Even if no missing merchants exist (i.e. $\mathbb{N} = \emptyset$), a maximum of only $w_p = \mu'_{c,t} - w^*_{c,t}$ is transferred to the merchant; $w^*_{c,t+1}$ is then updated to $w^*_{c,t} + w_p = \mu'_{c,t}$ and $\mu'_{c,t} \leq \mu^*_{c,t}$ by definition. Since $w^*_{c,t+1} = \mu'_{c,t}$, subsequent withdrawals will not transfer any funds out of c. □

Theorem 12. *PayPlace ensures Merchant Safety.*

Proof. To show *Merchant Safety*, we show that a merchant following the PayPlace protocol when signing a commitment is guaranteed safety of funds assigned therein to her even if she becomes arbitrarily unavailable to sign future commitments (or does not wish to, due to detecting malfeasance in subsequent operator-generated commitments). Since the operator is subject to the same rules as a merchant for the registration and withdrawal process (and implicitly attests a commitment by generating it and submitting it to the smart-contract), showing Safety for merchants also secures the Operator's funds in the commitment. We prove this by showing that the following two statements hold:

- Case 1: Suppose p signs a block κ's commitment (i.e. the tuple $(R(\mathcal{M}(\kappa)), \tau)$) and κ's notarization is published to the contract with $\sigma_{p,\kappa}$ included in ars_κ. Then p's funds are safe as long as $\mathbb{K}_t(-1) = \kappa$, i.e. p's funds in κ are fully available for p to withdraw as long as κ is the latest commitment.
- Case 2: Suppose p's signature is not included in the commitment of a block κ' and κ' is published to the contract (i.e. $\mathbb{K}_t(-1) = \kappa'$ at some time t). Define x such that $\mathbb{K}_t(-x) = \kappa_p$, where κ_p is the last notarized block that p signed. Then funds assigned to p in κ_p ($\mathbb{T}_p(\kappa_p)$) are available for withdrawal by p at t.

Case 1. Part a) We first show that p's funds in a notarized block κ that p signed for at t' is safe for any $t > t'$ as long as providers and the operator can only withdraw funds that have been assigned in κ.

Funds assigned to be $\mathbb{T}_p$ are of the form shown in Figure 6.4; for each consumer whose order p fulfiled, it specifies a payment amount to be sourced from that consumer's deposit. WLOG, assume that p has funds assigned from exactly one consumer c in $\mathbb{T}_p$; i.e. $|\mathbb{T}_p = 1|$ and $\mathbb{T}_{\shortmid} = T'$ where $s(T') = pk_c$. Let T be the corresponding source transaction that is revealed to p by the operator as part of $\mathbb{C}_{t'}$ for block verification and signing. Recall that the set of providers that each have a leaf in $\mathcal{M}(\kappa)$ is denoted by $\mathbb{P}(\kappa)$. Let $\mathbb{P}(\kappa)_{-p} = \mathbb{P}(\kappa) \setminus p$. Then it suffices to show that: (1) at time t the contract has at least $\mu(T')$ amount available as c's deposited funds $D_{c,t}$, and (2) that the total funds that can be withdrawn by $\mathbb{P}(\kappa)_{-p}$ cannot exceed $D_{c,t} - \mu(T')$. To prove (1) Note that by definition of PayPlace (Algorithm 4), $\mu(T) \leq D_{c,t'}$ and $\sum_{v \in \mathbb{P}(\kappa)} \sum_{T'' \in \mathbb{T}_v(\kappa)} \mathbb{1}_{s(T'')=c}\ \mu(T'') \leq \mu(T) - w^*_{c,t}$, implying $\mu(T') \leq \mu(T) \leq D_{c,t}$. Note that $D_{c,t}$ is a weakly monotonically increasing function of t since consumer withdrawals are prohibited in PayPlace; hence (1) holds. To prove (2), first note that a merchant can withdraw funds in κ only once as they are then tracked in $\mathbb{X} \cup \mathbb{W}$ and further withdrawals against κ cancelled (Line 5-6 of Algorithm 7) . Let μ_{-p} be the total funds assigned to $\mathbb{P}(\kappa)_{-p}$ with c as the source consumer. By definition, when p signed κ at t', the following held: $\mu_{-p} + \mu(T') \leq \mu(T) \leq D_{c,t'}$. Subtracting $\mu(T')$ from this, we get $\mu_{-p} \leq \mu(T) - \mu(T') \leq D_{c,t} - \mu(T')$. However, this is contradictory if $\mu_{-p} > D_{c,t'} - \mu(T')$. Since $D_{c,t}$ monotonically increases with t as well (weakly), this proves (2).

Part b) To show Case 1 then, it suffices to show that as long as $\mathbb{K}_t(-1) = \kappa$, a withdrawal initiated by any merchant p' corresponds to funds accounted for p' in κ. By design of the withdrawal function (Algorithm 7), if $p' \notin \mathbb{M}$, then p can only withdraw funds assigned to her in $\mathbb{K}_t(-1)$ (i.e. the contract requires a Merkle proof showing that the submitted transaction set $\mathbb{T}_{p'}$ is included in $\mathcal{M}(\mathbb{K}_t(-1)$. If $p' \in \mathbb{M}$, note that the withdrawal function only allows p' to withdraw funds stated in $\mathbb{T}_{p'}(\kappa_{p'})$ (i.e. the last notarized block p' signed). If $p \in \mathbb{M}(-x)$ for $x \geq 2$ (i.e. p' had missed signing $\mathbb{K}_t(-2)$ as well), then Lines 14-16 of Algorithm 4 ensures that p does not sign $\mathbb{K}_t(-1)$ unless $\mathbb{T}_{p'}(\kappa_{p'})$ is included in $\mathbb{K}_t(-1)$. If, instead if $p' \in \mathbb{M}(-1)$, then the commitment verification module that notarized $\mathbb{K}_t(-1)$ enforced that $\mathbb{T}_{p'}(\mathbb{K}_t(-1)) \supseteq \mathbb{T}_{p'}(\mathbb{K}_t(-2)$ (Lines 19-29 in Algorithm 6). Hence this reduces to the proof of Part a) above. Note

that the timing constraints in PayPlace (i.e. the checks in Lines 1-2 of the commitment generation Algorithm 6) ensures that older blocks signed by p cannot be re-notarized by the operator.

Case 2: From Theorem 9, p is detected as missing in any commitment that does not contain p's signature (since p has registered). When the contract processes the commitment for $\mathbb{K}_t(-x+1)$ (i.e. the first commitment submitted without p's signature after the last notarized block containing p's signature), the funds in $\mathbb{T}_p(\mathbb{K}_t(-x))$ are recorded in $\mathbb{N}$ against each source consumer, based on the definition of $\mathbb{N}$. Hence, (1) since p is detected as missing, p's confirmed funds (i.e. funds in the last block $\kappa_p = \mathbb{K}_t(-x)$ signed by p) are tracked in $\mathbb{N}$. Further, note that the contract updates $\mathbb{N}$ to remove release the reservation of these funds only when p is detected as having signed a submitted commitment again (Line $1-6$ of Algorithm 6 ensures that the contract is provided the original transaction set $\mathbb{T}_p(\mathbb{K}_t(-x))$ again in that case to perform the update). Next, (2) no merchant who has signed a commitment that was subsequently published after $\mathbb{K}_t(-x)$ can withdraw funds *locked by* $\mathbb{N}(-y)$ *for* $y \in [x, \eta]$ assigned to merchants whose last signature was on $\mathbb{K}_t(-x)$ or earlier. The maximum allowed withdrawal for a merchant p' in Algorithm 7 Line 19 ensures this: $\min\{\max\{\mu'_{c,t} - \sum_{i=\eta}^{\text{blocksP'Missed}-1} \mathbb{N}(-i)^{pk_c} - w^*_{c,t}, 0\}, \mu'\}$ for each $T' \in \mathbb{T}_{p'}$ with $c = s(T')$. Finally, (3) at the time t' that p signed $\mathbb{K}_t(-x)$, note that funds reserved so far in $\mathbb{N}$, i.e. $\sum_{i=\eta}^{x-1} \mathbb{N}(-i)^{pk_c}, \forall c$ have already been incorporated (accounted for) in $\mathbb{K}_t(-x)$. Lines $15-16$ of Algorithm 4 executed by merchants for commitment signing ensures this. Hence the double-spend verification checks performed by p in Lines 17-20 of Algorithm 4 ensures that funds assigned to p are not double-spent against the funds already reserved in $\mathbb{B}$ (cf. Case 1 above). Note that confirmed funds of a non-signing merchant is reserved in $\mathbb{N}$ only the first time that their signature is detected missing after being present on the last notarization; subsequent commitments with the merchant's signature continuing to be absent does not result in modifications to $\mathbb{N}$. □

Corollary 1. *PayPlace provides Data Availability and Income Certainty.*

Given Theorem 12, *Income Certainty* is straightforward to infer from Algorithm 6 and Algorithm 7. *Data Availability* follows by design of the Withdrawal Function. A

merchant p's confirmed funds $f_{p,t}$ in PayPlace corresponds to the funds assigned to them in the last notarized block they signed, $\mathbb{T}_p(\kappa_p)$. If p signed the last notarized block, Algorithm 7 expects a Merkle proof for $\mathbb{T}_p(\kappa_p)$ against $R(\mathcal{M}(\mathbb{K}_t(-1))$, which p knows since p signed $\mathbb{K}_t(-1)$. If p did not sign the last notarized commitment, then the contract simply expects the transaction set they last signed for, $\mathbb{T}_p(\kappa_p)$. Since p is detected as missing in the first commitment she misses after the last notarization that had her signature (Theorem 9), (Algorithm 6, Lines 25-26 ensures that the contract is provided $\mathbb{L}_p$ by the operator when p is missing to store $H(\mathbb{T}_p(\kappa_p))$ in $\mathbb{L}_p$), as long as p provides $\mathbb{T}_p(\kappa_p)$, the check in Line 8 of Algorithm 7 succeeds.

6.6 Evaluation

We evaluate the computational and monetary costs incurred by PayPlace for its main recurring operations, commitment generation by the operator and commitment verification by the smart-contract. We use ZK Rollup as the baseline in our analysis; ZK Rollup has become a popular solution for non-pairwise off-chain payments and has recently been deployed by multiple teams on Ethereum 2.0 mainnet [7, 117]. Note that it also satisfies almost all properties identified in Table 6.1.

Notation Let n be the number of payments made by consumers during β and p_u the number of unique payment recipients. Consider PayPlace and ZK Rollup notarization executed at the end of β. Let p_r and c_r be the total number of registered merchants and consumers, and c_u the average number of unique consumers that a merchant has been assigned payments from. Let p_m , $p_{m'}$ and p_a respectively be the number of non-signing merchants in the submitted commitment, the subset of these that had not signed the previous notarization either (note $p_{m'} < p_m$), the number of signing merchants who had not signed the previous commitment. Let p_x, p_b and p_w respectively be the number of deregistered merchants , newly registered merchants and the number of registered merchants who had withdrawn their funds since the last notarization. For the zkSNARK circuit used in the ZK Rollup, let g' be the number of gates, w' the number of wires and l' the number of known circuit inputs (for maximum instance size).

Off-Chain Computational Overhead Computing zkSNARK proofs is highly expensive, with bilinear pairings and group exponentiations dominating all other involved operations in cost. We sidestep this in PayPlace by offloading some computation

	Runtime		# Pairing & Exp.	
	PayPlace	ZKR	PayPlace	ZK Rollup
Op.	$O(p_r)$	$O(n)$	$2p_r$	$\frac{n}{z_{\max}}(4g' + w' - l')$
Mer.	$O(c_u \cdot p_r)$	$O(1)$	$2c_u$	0

Table 6.3: Comparing off-chain computational load and runtime.

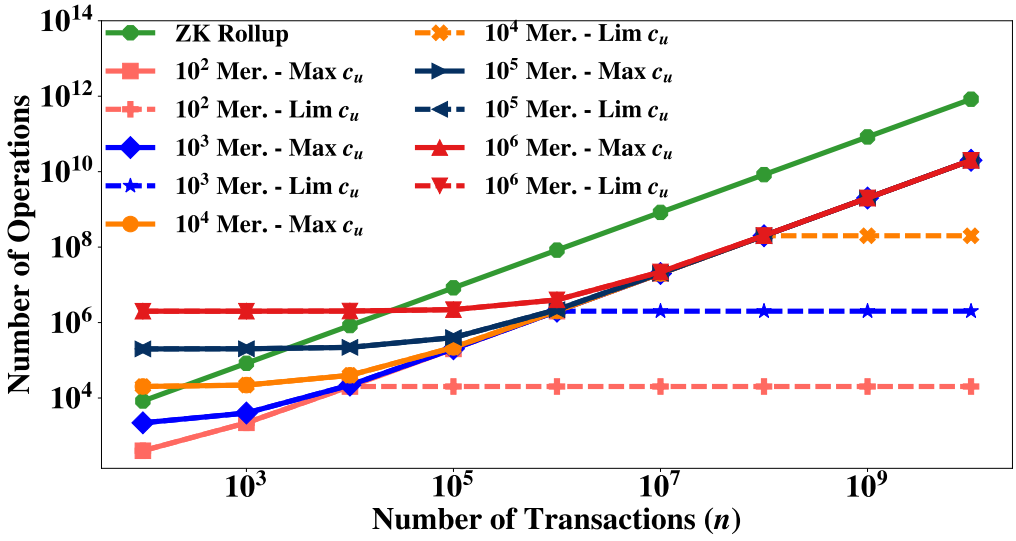


Figure 6.6: The number of pairings and exponentiations scales linearly with n for ZK Rollup and linearly in $p_r + c_u$ for PayPlace. As n increases, PayPlace incurs orders of magnitude lower computational load, even for large marketplaces (high p_r) with maximum c_u.

to each merchant, who protects her assigned funds in a block by the verification steps she performs before signing it. We hence assess the runtime complexity and dominant computational load for both the operator and merchants. For ZK Rollup, we consider Groth16 SNARKs [84], which are in wide use and recently deployed [18, 111, 114] in live ZK Rollup implementations. The number of off-chain payment transactions that can be included in the proof depends on the arithmetic circuit and further optimizations; let this maximum number of transactions be denoted by $z_{\max}$. For n total transactions then, $\lceil n/z_{\max} \rceil$ prover computations need to be performed by the operator. Table 6.3 reflects the corresponding amortized computational load and runtime complexity. For PayPlace, note that Merkle root of a p_r-leaved tree can be computed in $O(log_2(p_r))$ time [168] (given $O(log_2(p_r)$ space). Hence, the operator's runtime complexity for commitment generation is dominated by the signature verification operations (p_r verifications in the computational worst-case when all merchants return signed commitments) while merchants' by the double-spend verification checks they do before signing a block (Algorithm 4). In essence, computational costs for a ZK Rollup operator scales linearly in n while for PayPlace it is primarily a function of p_r and c_u. For both PayPlace and ZK Rollup, the operator and merchant runtimes can be reduced to $O(1)$ with arbitrary space complexity (i.e. these computations are fully parallelizable).

We now empirically study the computational load of the two techniques in practice (in terms of the expensive cryptographic operations - pairings and exponentiations). We vary the number of transactions during β from $100-10$B. To put this in perspective, Amazon is estimated to process roughly 27M order per day and Uber roughly 15M rides per day worldwide [61, 127, 129]. Recent data [111, 115] from ZK Rollup systems indicate a capacity of $2-3$K transactions per proof, hence we conservatively set $r_{\max} = 3000$ and consider a load of only 150K total pairings and exponentiations per SNARK proof computation (in practice, 150K is approximately the number of constraints in the Rollup circuit reported by benchmarks, yielding much higher $4g'+w'-l'$). As Figure 6.6 depicts (log-log scale), the dominant off-chain computational load in ZK Rollupincreases linearly in the number of transactions. From Table 6.3, however, it is evident that the number of such operations is PayPlace is not a function of n but of p_r and c_u. We hence vary p_r from $100-1$M merchants; to put this in perspective, Amazon and Uber have around 2M sellers and drivers respectively. To assess worst-case load, we let c_u equal the number of transactions per merchant.

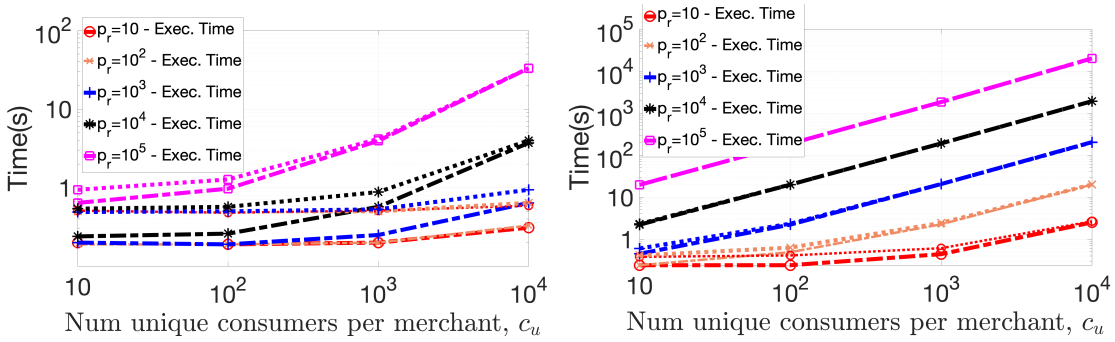
Operation	**Best Case**	**Average Case**	**Worst Case**
Bilinear Pairings	$O(1)$	$O(1)$	$O(p_m - p_{m'} - p_b - p_w)$
Multiplications in $\mathbb{G}_1$	$O(1)$	$O(1)$	$O(p_m + p_x)$
Hashing into $\mathbb{G}_0$	$O(1)$	$O(1)$	$O(p_m - p_{m'} - p_b - p_w)$
Non-$\mathbb{G}_0$ Hashes	$O(1)$	$O(1)$	$O(p_a + (p_m - p_{m'} - p_b - p_w) \cdot log_2(p_r))$

Table 6.4: Best, Worst and Average-case. Runtime Complexity of notarization in PayPlace, categorized by the operation type.

For instance, if $n = 1000$ and $p_r = 100$, then we evenly distribute the orders across merchants as 10 orders per merchant, and assume a worst-case scenario of 1 unique consumer per order, resulting in $c_u = 10$. If $p_r > n$, only n merchants receive orders ($c_u = 1$ for them). In practice, marketplaces may involve recurrent transactions between consumers and merchants (e.g. due to co-location and especially if β spans longer time periods). We hence also consider c_u no greater than p_r to model this. We refer to this as the "Limited" or Lim c_u in Figure 6.6 and the former as the "Maximum" or Max c_u. As we see from Figure 6.6, even as the number of merchants increase exponentially, the computational load across the operator and all merchants in PayPlace is orders of magnitude lower than ZK Rollup as the number of marketplace transactions increase exponentially. In-practice, the PayPlace operator factors in for typical p_r and u_c in the marketplace to estimate the duration γ required to execute these off-chain operations.

Off-Chain Execution Time. We implement the off-chain computations performed by the PayPlace operator and merchants. We use Chia-Network's implementation of BLS signatures [47] that avails the BLS12-381 curve, and execute the operations on a 2015 Macbook Pro with 2.5 GHz Quad-Core Intel Core i7 processor and 16GB RAM.

Figure 6.7(a) depicts the execution (wall clock) time and the total CPU time (both user and kernel mode) for the operator to generate the Merkle tree of merchant payouts (with dashed lines representing CPU times). For each merchant, the operator generates a list of consumers who have sent payments as well as the payment amount. This resulting transaction set for a merchant constitutes one leaf of the Merkle tree. Since the size of this transaction set, and correspondingly the time to hash it into the Merkle tree, depends on the number of consumers that have made payments for



(a) Time taken by the operator to generate the Merkle tree specifying merchant payouts. Dashed lines represent CPU time.

(b) Time taken by merchants to verify and sign the operator-generated Merkle tree. Dashed lines represent CPU time.

Figure 6.7: We depict the CPU and total execution time for recurring off-chain PayPlace operations. Note that the axes of all figures are in log scale.

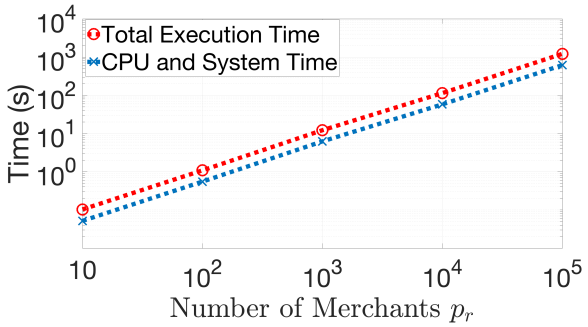


Figure 6.8: Time taken by the operator to verify merchants' signatures on the Merkle root and aggregate them. Note that the axes of all figures are in log scale.

the merchant, we vary this quantity c_u to study its effect on system utilization, along with varying the number of merchants p_r. Even for $100,000$ merchants with $10,000$ paying consumers each, the Merkle tree generation time does not exceed 100 seconds. For a given number of access points, the execution and utilization times increase with the number of user addresses; however, this relationship appears sub-linear until the number of users paying an AP reaches $10,000$.

Figure 6.7(b) shows the cost for the Merkle tree verification and signing operations that merchants perform when the operator publishes the computed set of transactions. For a given p_r, the execution time and CPU load increases linearly with c_u. When $100,000$ merchants are present and $10,000$ paying consumers listed in each leaf, the execution time exceeds 2 hours. This linear overhead manifests in the merchant's verification function, since it scans every leaf and transaction within a leaf, to ensure

that the operator is not staging double-spend attacks through a maliciously computed Merkle tree. Finally, Figure 6.8 depicts the worst-case execution and CPU time taken by the operator to verify the signed Merkle roots. The operator aggregates the signed Merkle roots and performs a single verification of the aggregate signature to see if all merchants have correctly signed it, which is $O(1)$ in p_r. If this fails, the operator checks each signature until the malicious merchant is found. The worst-case time thus scales linearly in the number of merchants.

These results shows that the operator and merchants' recurring PayPlace operations may take a fraction of a second to several hours, depending on the number of merchants and the number of users performing transactions. Further splitting merchants into sets such that the resulting sets of merchants are owed payments from non-overlapping sets of users significantly controls this overhead. For instance, in the Datanet case where merchants are access points and consumers end-devices, a user in the United States would rarely make payments to access points in France. Finally, the periodicities of Merkle tree generation and fund settlement on the root blockchain should be chosen according to these overheads.

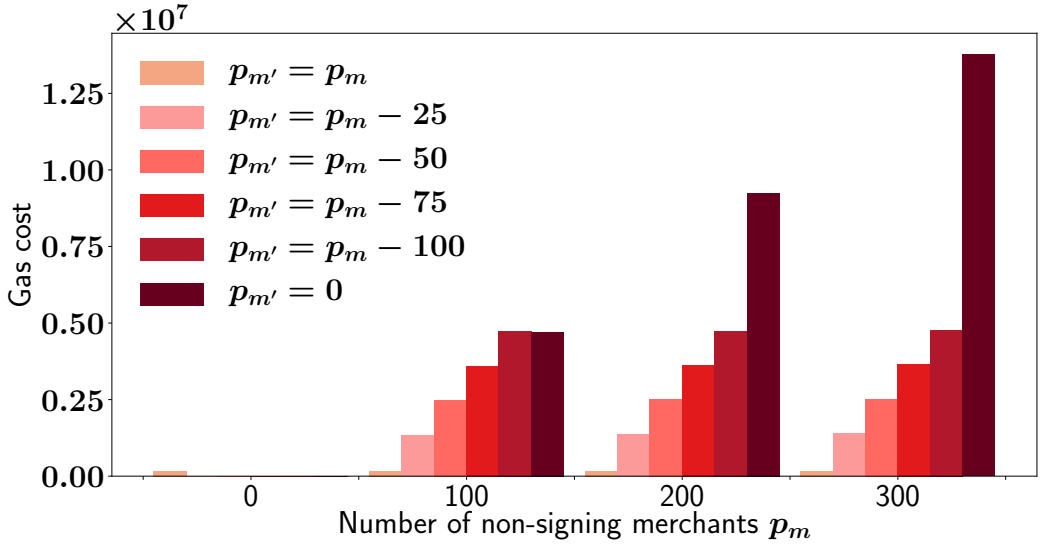
On-Chain Notarization Complexity We next assess the on-chain runtime complexity of PayPlace and ZK Rollup notarizations. For PayPlace, we further categorize this by three scenarios and the core operations involved. We consider the best-case scenario as $p_x = 0$, $p_a = 0$, $p_m = 0$ (i.e. all registered merchants have signed the submitted commitment with no one having recently exited or missed the previous one), and otherwise as the worst-case. We consider the case where all non-signing merchants had missed the previous notarization as well (i.e. $p_{m'} = p_m$), all signing merchants had signed the previous notarization as well (i.e. $p_a = 0$), and no merchants exited since the last notarization (i.e. $p_x = 0$) as the average case. In practice, merchants registering and exiting the system is likely infrequent in comparison with notarization events, especially since PayPlace guarantees safety of their notarized funds and does not incur any exit games.

Table 6.4 specifies the runtime complexities for PayPlace. PayPlace is overwhelmingly $O(1)$ in the number of transactions n and recipient providers p_r (cf. Algorithm 6) except in the worse-case, where the hashes required for verifying Merkle proofs for newly non-signing merchants scales logarithmically with p_r. This directly results in *On-Chain Efficiency* in PayPlace. In the best-case *as well as* average-case scenario, PayPlace is O(1) in all operations. This provides an important insight; the complexity

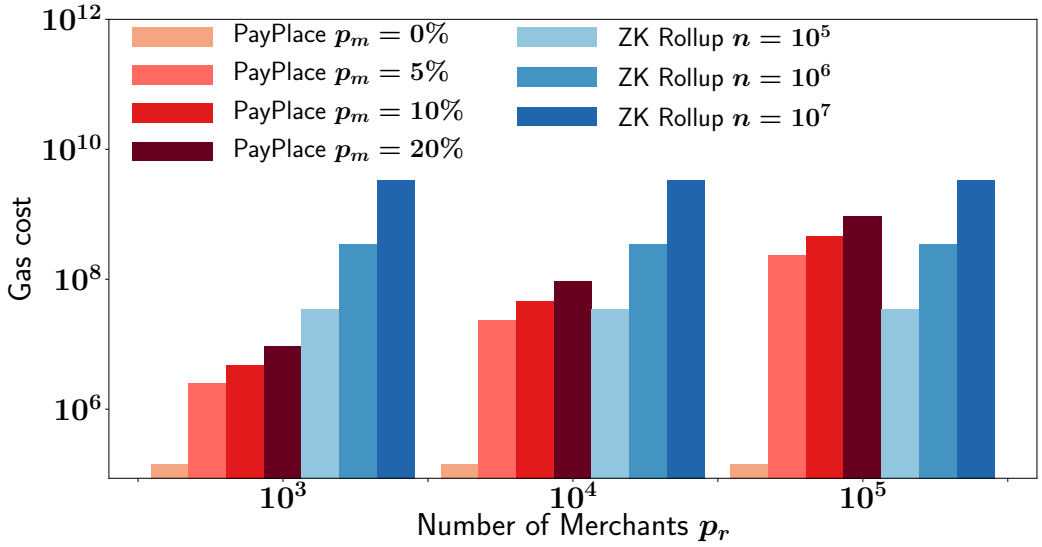
of the smart-contract's notarization module in PayPlace does not increase *even for arbitrarily large quantities of non-signing merchants* as long as these non-signing merchants remain inactive for multiple notarizations once they become inactive. Further, well-known marketplace operators like Amazon and Uber are often atleast semi-trusted and merchants may well opt to only intermittently participate in the signing process to receive additional income, leading often to low $p_m - p_{m'}$ and $p_a = 0$. In the worst case, the most expensive operation, Bilinear Pairings, scales only in $p_m - p_{m'}$. For ZK Rollup, though the Groth16 SNARK verification can be run in constant time, only $z_{\max}$ transactions can be included in one proof. Hence the amortized time complexity is $O(n)$.

On-Chain Notarization Costs To assess the on-chain computational resources required for frequent notarization operations, estimate the gas costs incurred in the Ethereum blockchain for ZK Rollup and PayPlace notarizations. As of the Istanbul network update [72], the SNARK verification for Rollup is estimated to cost approximately 300K gas [115, 199]. Transactions further have to be published on the root chain at least in CALLDATA to ensure data availability [11]. CALLDATA is then a recurring cost of 16 gas [72] per byte and each included transaction is 15 bytes. We add an overhead of 50K gas to account for additional costs, e.g., due to logging, storage slot modifications, etc. as done previously [11]. For PayPlace, we set p_a, p_b and p_x to 0 as these represent negligible overheads. As evident from Table 6.4, the computational complexity is affected significantly by p_m, $p_{m'}$ and p_r; we study those here. While native support for BLS12-381 curve operations (i.e. pre-compiles) is being planned in Ethereum [2, 5, 15], the alt_bn128 curve is mainly used for zkSNARKs and BLS signatures. We hence use gas costs charged by the alt_bn128 pre-compiled contract offered in Ethereum [3] to estimate the cost for BLS signature operations. Parings cost $34\text{K} \cdot \text{num}_{\text{pairings}} + 45\text{K}$ gas, the cost of a key multiplication in $\mathbb{G}_1$ (for public key multiplication) is 150, and the cost per keccak256 hash is 42. We estimate a higher cost of 100 gas for hashing into $\mathbb{G}_0$. Note that verifying x PoPs can be done with $x + 1$ pairings (rather than $2x$) [38, 39]. We assume a fixed overhead of 30K (for addition/assignment operations, broadcasting events) and a variable overhead of 10K in the number of additional non-signing merchants in the submitted commitment (i.e. $p_m - p_{m'}$) to account for the storage and broadcast operations involved.

Figure 6.9 illustrates these estimated gas costs in PayPlace and ZK Rollup notarizations. We set $p_r = 1000$ in Figure 6.9(a), and vary the value of $p_m - p_{m'}$ for



(a) Estimated gas costs for PayPlace notarization with $p_r = 1000$ as a function of p_m and $p_{m'}$



(b) Estimated gas costs for Worst-case PayPlace ($p_{m'} = 0$)) vs ZK Rollup notarizations as a function of n and p_m

Figure 6.9: (a) Gas cost for PayPlace notarization primarily scales with the $p_m - p_{m'}$ rather than with p_m. It increases with *additional* merchants who have not signed the notarization compared to the previous one. (b) Gas cost for ZK Rollup increases with n while worst-case PayPlace is orders of magnitude cheaper when p_m is relatively low wrt n.

different values of p_m. Crucially, we observe that even as the number of non-signing merchants p_m increases, *the gas required is near-constant* as long as $p_m - p_{m'}$ is the

same. This directly corroborates the analysis in Table 6.4, where only multiplications in $\mathbb{G}_1$ scales with p_m corresponding to Line 10 from Algorithm 6. Since bilinear pairings far exceed the rest in cost per operation and scales linearly only with $p_m - p_{m'}$, we see in Figure 6.9(a) that notarization costs predominantly increase only in the number of *additional* non-signing merchants in a commitment and not the *recurring* non-signers. In-fact, very little gas is expended when $p_m = p_{m'}$ irrespective of the value of p_m. In the marketplace context, this implies that notarization costs are high only if merchants tend to oscillate between being active and inactive during consecutive commitment submission windows (resulting frequently in $p_{m'} \gg p_m$) . In practice, we may expect merchants to participate reliably in the notarization process to receive additional income on-time, or to frequently miss the signing process and only sporadically accrue income (i.e. the operator is highly trusted). Essentially, PayPlace is designed so that merchants impose a relatively high cost the first notarization that they miss after being active, but little cost for subsequent ones. Figure 6.9(b) (log scale) compares gas estimates for ZK Rollup vs the worst-case gas estimates for PayPlace (i.e. when $p_{m'} = 0$) for different p_r based on the fraction of the merchant population that is non-signing (i.e. p_m). Unsurprisingly, the former scales linearly in n, as also seen in Table 6.4 and is $O(1)$ in p_m and p_r; however, even with $p_{m'} = 0$, the latter is often orders of magnitude lower in cost (even for large p_m if $p_m \ll n$). It is evident that operators choosing between these two off-chain payment solutions must assess the expected transaction volume and the merchant population. When merchants' devices can reasonably be expected to participate in the signing process (once per day or hour, based on β) or atleast stay in their active or inactive states for extended periods, *PayPlace is highly beneficial by scaling throughput at no marginal gas cost.*

6.7 Integrating PayPlace with Datanet

PayPlace can be seamlessly integrated into Datanet. Indeed, the PayPlace operator can also perform the remote AAA functions that the Datanet operator does. In-fact, as Figure 5.9 in Chapter 5 depicts, the Datanet operator does indeed subsume the functions of the PayPlace operator as well, by receiving the micropayments from devices, verifying them, and periodically using the PayPlace process to assign funds to appropriate access points and notarize this in the root-chain. PayPlace renders Datanet robust to the scalability challenges faced by other micropayment protocols,

and guarantees that the operator cannot manipulate funds owed to their APs once notarized. PayPlace assumes a single operator in the marketplace; correspondingly, consumers' funds deposited for use in the contract are locked to that single operator. In our case, however, multiple Datanet operators exist in the marketplace, and we would like users to effortlessly connect to access points of any AAA operator without requiring separate deposit transactions with multiple smart-contracts. We therefore modify the PayPlace mechanism accordingly. When depositing ¢ into the contract, users specify a split of these funds between different registered AAA operators of their interest (e.g. based on operators often used by nearby APs). The contract then restricts the total amount that can be withdrawn by APs and the operator against this user to the amount specified in this split. At a later time, if the user wishes to shift funds currently tied with one operator for use with another, the contract initiates a *challenge period*, when the original operator or its APs can contest the user's action, by showing proof (via user-signed micropayments) that the user is attempting to shift funds already spent for data access with the current operator. In the absence of such a challenge, the user's specified funds are marked as transferred for use with the specified new operator. This incurs significantly less overhead than the inter-contract coordination and fund transfers that would be required if each operator instead used a separate, dedicated PayPlace contract.

APs receive their due income from the operator based on the PayPlace side-chain mechanism. AP owners attest their signature to the root of the Merkle tree that their Datanet operator periodically generates, assigning income to each AP. Note that a single network controller may own multiple APs onboarded with a Datanet operator; as long as the same blockchain credentials are specified for the underlying APs by the owner, only one signature by the corresponding private key is required on the generated Merkle root. Before signing, AP owners verify that funds being assigned to them by the operator in the Merkle tree have not been double-spent by the operator and that the Merkle tree has been generated correctly. Note that this can be fully automated and run on any device (cloud or personal computer) that has the AP's key available for signing. It may be desirable for AP owners to periodically verify that the income they are assigned by the operator is in keeping with the data services rendered to users, to minimize their trust on the operator and detect any malfeasance immediately. PayPlace guarantees that once the signed Merkle root is notarized on the smart-contract, the funds assigned to APs in the corresponding Merkle tree are

irrevocable and cannot be misappropriated. Hence, APs may switch the Datanet operator they use for AAA services without concerns about loss of prior income.

6.8 Discussion

In practice, most blockchains impose limits on the amount of computations that can be performed as a part of a single transaction. The notarization process executed by the PayPlace smart-contract, however, scales linearly with factors like $p_m - p_{m'}$ in the worst-case analysis from Table 6.4, imposing limits on these factors to stay within the block limit. One way to overcome this is to allow operators to force the exit of such non-signing merchants using the withdrawal module before submitting the notarization. Alternatively, the operator may use a zkSNARK to prove to the contract that funds previously assigned in the last notarization to newly non-signing merchants $p_m - p_{m'}$ have been included in the Merkle tree whose root has been submitted for notarization. Finally, the operator may entirely avoid this linear cost of non-signing merchants by instead submitting an additional aggregate signature of newly joined merchants (since the last notarization) on the previous notarized root. This ensures that these new merchants can detect if the operator has malicious omitted a merchant who was assigned funds in the previous commitment from the current one; hence signing merchants can still protect themselves from double-spend attacks.

6.9 Summary

We develop PayPlace, an off-chain payment protocol optimized for large marketplaces that overcomes liquidity and capital drawbacks of previous solutions while keep the root-chain footprint low. PayPlace takes advantage of the presence of marketplace operators and introduces them as semi-custodial intermediaries in the payment process. Consumers pay the operator *off-chain* during order placement, and the operator periodically forwards the accrued payments *off-chain* without requiring any liquidity. Our construction results in highly usability; consumers are able to view their off-chain payments to the operator as transactions in a unidirectional payment channel, while merchants are guaranteed safety of their notarized funds even if they are arbitrarily offline. We show that, based on how frequently merchants oscillate between being available to sign notarizations and not, PayPlace is potentially orders of magnitude

cheaper in on-chain and off-chain execution costs compared to the state-of-the-art technique for non-pairwise off-chain payments, Zero Knowledge Rollups.

Chapter 7

Summary of Contributions and Future Directions

7.1 Summary of Contributions

In this thesis, we aim to facilitate incentive-compatible and seamless connectivity between unsubscribed devices and wireless networks. The ubiquitous subscription model prevalent today serves a multitude of purposes: 1) it creates a-priori identity relationships between the device and the network that enable authentication, e.g. using SIM or PSK, 2) it subsumes a trust relationship wherein the device trusts the usage metering done by the network in order to impose the terms of the subscription contract and the network trusts the device to make the corresponding payment at the end of the billing cycle, 3) it implicitly provides authorization to the device to connect to the network during the subscription period based on the agreed-upon terms and authorizes the network to levy usage-based charges as per the contract, and 4) it forces the device owner to setup payment details off-band with the network that is then used by the network to charge the appropriate amount at the end of the billing cycle based on the contract and monitored usage during the cycle.

Removing pre-established subscription contracts therefore leads to a variety of research challenges that span across these four core functions of Authentication, Authorization, Accounting and Billing. However, limitations of such a static long-term usage-based contract are already felt today as networks often fail to keep up with applications' resource needs that can be determined only in realtime, and the

subscription model becomes an increasingly poor fit for emerging connectivity use-cases like the IoT. The granularity of typical month-long mobile data plans is such that users must forecast their network usage over a long period of time and assign a single monetary value to its utility. Finer-grained real-time information about user needs does not play a role in resource allocation, though mobile applications are launched and their resources allocated only in real time. This results in unrealized value for both the end-user and the network operator and further limits the user to availing resources that belong only to their subscribed network(s). Further, the rapid and dense deployment of IoT devices (e.g. smart cities) will require device owners to manage increasingly complex usage contracts with operators' LoRaWAN or NB-IoT networks, posing a prohibitively unscalable and expensive bottleneck for large-scale IoT deployments. The overhead of provisioning dedicated contracts for each device may accelerate as 5G networks are more widely installed. Such networks are expected to include multiple access points of different radio access technologies, potentially with different operators, making it even more difficult to pre-specify contracts for individual IoT/user devices with each operator.

The goal of this thesis is to enable seamless and subscription-less connectivity between devices and wireless networks with no a-priori identity or trust relationships, and to facilitate the establishment of user-driven session-oriented usage contracts with the network in realtime. We setup three distinct networking scenarios that retain progressively fewer characteristics of the subscription-based model and address the research challenges in each.

In Chapter 3, we first propose that typical monthly data subscription plans be supplemented with ad-hoc discount offers, wherein users may consume unlimited data for the offered hour for a small fixed fee if they accept the discount offer. This scheme allows users to realize additional resource needs for their sessions in realtime that had not been accurately captured in their monthly forecast when signing up for the subscription; indeed, these simple offers, when available, can be utilized to avoid risking unforeseen overages which are much more expensive in comparison, while the retention of monthly subscriptions continues to provide a predictable source of revenue to the network. We first develop a monthly abstraction of such supplemental discount offers (SDOs) to show that users are generally willing to accept some SDOs, allowing the ISP to increase its revenue and the user to maximize their data consumption utility. We then develop a dynamic model that captures the ad-hoc nature of these

offers, and show that users face a complex hourly decision problem as to which SDOs they should accept over their billing cycles, since they are unaware of their exact future needs or when future SDOs will be made. The ISP faces a similarly challenging problem in deciding when to offer SDOs so as to maximize its revenue, subject to users' decisions. We develop optimal decision criteria for users and ISPs to decide whether to make or accept SDO offers. Our analysis shows that both users and ISPs can benefit from these offers, and we verify this through numerical experiments on a one-week trace of 20 cellular data users. Critically, however, we find that ISPs can exploit user uncertainty in when future SDOs will be made to *maximize its revenue at a net loss to users*. This hence alludes to the challenges that end-users face in practice when attempting to make realtime decisions about their consumption and costs without complete information. To establish dynamic usage contracts between the device and the network in practice, it becomes important to mitigate this. We employ popular reinforcement learning techniques like Double Deep Q Networks and Actor Critic Models for the user to learn the ISPs offer schedule and the variation in their own resource needs over time, and show that the suboptimality from real-time decision making that users incur can almost entirely be eliminated.

In Chapter 4, we consider an even more dynamic user-driven approach to acquiring session-oriented network resources in realtime by proposing a model wherein a slice of network resources is dynamically created and assigned to an end-device based on the session needs it explicitly specifies. By acquiring resources for the entire duration of their session, devices can then reliably estimate their session performance at the onset rather than waiting for SDOs to be offerred to realize realtime needs that were not captured in the monthly subscription contract. Our focus is on real-time multimedia applications such as interactive gaming, live video streaming, and augmented reality that have strict latency and bitrate requirements but are resistant to buffer-based mitigations like HTTP DASH since resource needs are immediate. Recent innovations in network slicing have demonstrated the feasibility of dedicating specified amounts of network resources to individual sessions in the radio access network, and encouraged by this, we propose to reserve network resources for multimedia sessions *in real time* according to their declared needs, thereby providing *ad hoc session-level performance guarantees*. Through WiFi experiments and trace-driven LTE simulations, we show that such session-level resource provisioning is robust to real-time channel fluctuations and congestion externalities over the lifetime of a session. This approach, however,

raises challenges: how can the network ensure that users are honest about their resource needs and optimally allocate its limited resources to users, *under uncertainty in future sessions' resource needs*? We derive a novel Multi-Unit Combinatorial Auction (MUCA) model with a unique structure that can be exploited for fast winner determination and yet incentivize truthful bidding, properties not simultaneously achieved in a generic MUCA but essential to making *real-time* session guarantees. Further, since dynamic bidding in realtime is challenging for end-users who are budget-constrained, we develop a Reinforcement Learning based utility-maximizing strategy to distribute their budget across sessions, and show that it yields high user utility and effectively again mitigates the suboptimalities from real-time decision making (as with the case of SDOs too).

While Chapters 3 and 4 explore the establishment of dynamic usage contracts between devices and networks that capture devices' session-oriented resource needs, utilities and valuations, this addresses only the Authorization stage of device-network connectivity process. After negotiating on the resource allocation and price, the network authorizes the device's connection to begin and is authorized by the device to levy charges based on the agreed terms. However, it is yet unclear how the device can Authenticate with the network in the first place without any prior identity setup, and how the network's measurement of device usage can be relied on as ground truth for enforcing payments when there is no trust relationship between the two. In Chapter 5, we introduce *Datanet*, a system to facilitate seamless authentication with non-custodial credentials and trustworthy utilization metering of data sessions between untrusting devices and wireless networks. Our core insight is that exchange of services and payments can be *trustlessly* enforced by distributed ledger technologies; the credentials that blockchains use for account management can also be used for TLS-based authentication in networks using the widely supported EAP-TLS standard (across WiFi and more recently, Cellular networks as well). However, this raises several challenges: for instance, requiring special-purpose trusted hardware for bandwidth metering that is tamper-free and can be trusted by a smart-contract as ground truth, or even software modifications at the access point to integrate with the blockchain significantly hinders solution adoption. Further, the blockchain's ability to enforce transaction rules is limited by the extent to which the underlying exchange of services is digitally trackable, which is susceptible to manipulation in this case. More importantly, considering the sheer ubiquity and scale of internet-enabled devices and last-mile

access points, the load imposed by such a system on the blockchain if it were to be used to adjudicate every session between devices and networks is prohibitively large. Public and permission-less blockchains, which we must rely on to enable a non-custodial solution, are known to suffer from throughput and latency limitations as is. Using remote authentication servers based on the AAA framework, incremental off-chain micropayments for trust-minimized and incentive-compatible connectivity, and trusted execution environments for establishing usage-based micropayment schemes, we address these challenges to design and build DataNet, a system providing seamless and incentivized connectivity between untrusting end-devices and APs, without significant computation or network overhead.

Finally, in Chapter 6, we develop a scalable and cheap cryptocurrency payment system that seamless integrates with the Datanet system and allows devices to quickly make payments for their ongoing data sessions with APs without requiring any access point modification. Facilitating fast and cheap cryptocurrency payments is important for several marketplace applications that use blockchains, and especially so for large-scale blockchain-based networks like Datanet and others [89, 126, 152] that aim to facilitate sharing of last-mile network resources. The standard solution for off-chain payments, state channels, are optimized for frequent transactions between two entities and impose prohibitive liquidity and capital requirements on payment senders for marketplace transactions. We propose *PayPlace*, a scalable off-chain protocol for payments between consumers and sellers. Using PayPlace, consumers establish a virtual unidirectional payment channel with an intermediary operator to pay for their transactions, thus allowing AAA operators in Datanet to also performs the payment-related functions of a PayPlace operator. Unlike state channels, however, the PayPlace operator can reference the custodial funds accrued off-chain in these channels to in-turn make tamper-proof off-chain payments to merchants, *without locking up corresponding capital in channels with merchants*. Our design ensures that new payments made to merchants are

7.2 Future Directions

I plan to address limitations of current work and explore promising venues of further research based on enabling session-oriented real-time usage contracts between devices and wireless networks without subscriptions.

Modeling Competition between Networks. Without being restricted to connecting only to networks with which devices have a subscription contract established, devices are now faced with a significantly more competitive market of providers at any time that. In this work, however, we have not explicitly considered the impact of this increased real-time competition between networks on prices and resource availability. Both with SDOs as well as UBid, we assume a monopolistic ISP that the device is constrained to connecting to and focus on challenges of incentive-compatibility, revenue maximization and making optimal decisions in realtime. Though the SDO model does consider users' option to consume WiFi instead of cellular data, we do not model competition between ISPs. In a competitive setting, SDOs may attract new users to an ISP by allowing them to supplement their data plans; on the other hand, other ISPs could counter these offers by simply increasing their plans' monthly quotas. Similarly, in case of UBid, devices will likely get pricing quotes from multiple ISPs for their desired session resources and evaluate between them. Though competition from multiple accessible radio access networks has been studied in the context of hetnet selection [32, 67, 105], these models do not consider session-level resource reservations.

Session-oriented Resource Guarantees for Other Broader Scenarios. UBid is a first study of modeling auctions for network resources in a manner that decouples the auction model from wireless-specific scheduling details. In structuring this auction and studying its properties, we ignored concerns of mobility in the dimensionality of the MKP as well as more sophisticated bundle generation policies that might handle buffer-based video streaming as well as take into account concurrent resource requests to structure returned bundles to better schedule users in conjunction. Relaxing each of these restrictions has non-trivial implications on our results, and are interesting venues of future work. One might also consider the repeated nature of the auction not just in budget distribution of users (as we do) but also in availability projected by the network. In other words, the network might have incentive to impose reserve prices or hide a portion of its availability in order to increase its revenue.

Faster Coordinated User-Learning for Real-time Decisions. In this thesis, we show the promise of reinforcement learning techniques in mitigating the suboptimality induced in real-time user decisions due to incomplete information about future prices and consumption. As illustration in Chapters 3 and 4, this suboptimality results in significantly higher spending for a net lower data consumption utility for users, and hence these mechanisms to improve real-time spending dynamics are im-

portant. However, we do not consider explicit user coordination in this process. If the ISP changed recomputed their SDO offer schedule during the billing based on users' current data quota states rather than computing it once at the beginning of the cycle based on historical data, then the environment becomes non-stationary since users (in the current setup) use reinforcement learning to learn independently. Such non-stationarity is also seen with multi-agent RL in UBid, where as the number of agents independently learning increases, the effectiveness of the learning technique drops. A promising direction of future work is to explore coordinated learning in this context. Users may have incentive to share certain information about their states but not others; since each user is self-interested and rational, it is especially important in this case to study the incentive-compatibility of coordinated learning techniques.

Implementing PayPlace. In Chapter 6, we estimate gas costs for PayPlace and ZK Rollup and show that on-chain costs of PayPlace is often orders of magnitude lower. As future work, we plan to implement the PayPlace smart-contract on the Ethereum blockchain. Pre-compiles for BLS signature operations are expected to be added to Ethereum [2, 5, 15]. By deploying PayPlace, we wish to measure the realized computational and gas costs and compare this with reported numbers from live ZK Rollup systems that have recently been deployed on Ethereum 2.0 mainnet [7, 117]. We expect these results to be similar to our estimations in Chapter 6 which are also based on real data and prices. More importantly, this implementation will allow us to study the configurations and impact of hueristic parameters in PayPlace.

For instance, since PayPlace payments are decomposed into consumer→operator and operator→provider transactions, a shorter interval aggregation t_a indicates a lower *counter-operator risk* to providers in receiving compensation for the consumer orders fulfilled during the last t_a. At the same time, notarizations are root-chain transactions that impose monetary fees on the operator. The choice of t_a must reflect these considerations. Providers may even periodically vote on t_a's value to reflect their trust in the operator over time; the fees charged by the operator (i.e., commissions on consumer's payments) can reflect the overhead of notarization costs every t_a. Similarly, as seen in the evaluation presented in Chapter 5, the time taken for generating a block's Merkle tree increases with the number of providers registered in the system, since each provider has a leaf of transactions in the tree. Further, the number of checks performed by a provider before signing the Merkle tree's root is directly proportional to the number of providers as well as the number of consumers sourcing the provider payments.

Limiting γ without limiting the size of the marketplace therefore requires horizontally scaling the PayPlace sidechain in a manner this is transparent to consumers. Most importantly, most blockchains impose limits on the amount of computations that can be performed as a part of a single transaction. In Ethereum, for instance, this corresponds to a limit of 10M gas per block[1]. The notarization process executed by the PayPlace smart-contract, however, scales linearly with factors like $p_m - p_{m'}$ in the worst-case analysis from Table 6.4 in Chapter 6, imposing limits on these factors to stay within the block limit. As noted towards the end of Chapter 6, there are multiple potential ways to overcome this in PayPlace and an analysis of the tradeoffs presented is left as part of the implementation in future.

Extension to Just-in-Time Spectrum Sharing. An important research focus in the multi-year OfCom DSA study conducted in the UK[2] has been the extension of mobile network coverage to rural areas. Mobile operators incur significant capital expenses in setting up base stations and other infrastructures in an area, as well as operating costs. Without sufficient demand in the region, it becomes prohibitively unprofitable for them to provide service there, even if they already have the spectrum licenses to operate there. This has caused wide gaps in coverage in rural areas where cellular data demand is less dense.

Companies like Vanu solutions[3] have entered this space and tried to provide coverage in these area using small cell technologies with innovative energy solutions that make them cheaper to operator. They then sub-license the spectrum owned by the primary operators in that area and use their equipment to provide coverage. In-fact, this is a realization of the idea of carrier-driven offload to long-tail WiFi and other radio access technologies [146], where carriers enter into long-term negotiations with private owners of WiFi routers so that they may initiate user equipment offload to those access points when load on their base station is deemed to high. This becomes especially important during peak load times, like football games and concerts when the stadium capacity routinely over-congests the networks which have not been provisioned with the hardware to scale to such high demand. During such times, if private owners of hotspots could deploy their routers in the area and engage into contracts with operators to allow operator-driven offload to these routers, both end-user and operators

[1] https://etherscan.io/chart/gaslimit

[2] https://www.ofcom.org.uk/research-and-data/technology/general/emerging-tech/decentralised-spectrum-access

[3] https://www.vanu.com

stand to gain significantly.

However, this has been challenging to achieve in practice since all of these rely on establishing prior contracts between the operator and equipment owner, either to sub-license the operator's spectrum (as in the case of Vanu providing rural area coverage) or to allow the operator to drive offload to private hotspots. The operator must be able to audit usage in Vanu to ensure that appropriate commission is paid to them while access point owners need to ensure that operators pay them their dues for each person consuming data on their router. As evident, this requires fine-grained accounting and auditing, and necessitates either a prior trusted relationship between the parties or significant system engineering and integration on both ends. We again see the potential of trust-less and publicly auditable transactions over a blockchain in making this system highly efficient and practically feasible. If Vanu-user transactions were, by design, recorded in a public blockchain, they are auditable by the operator who can ensure the fees they are paid. Similarly, we can apply the AAA-based architecture from Datanet to route users on the offloaded WiFi access points to a blockchain-based AAA server.

Economics of Blockchain-based Wireless Service Provisioning. Blockchain-based resource sharing applications like Datanet make use of native crypto-tokens for payments and rewards on the platform. However, crypo-tokens are a very new offerring, and it is unclear how to design them, both from micro- and macro- economic perspectives. For instance, there are several types of monetary policies that a token may employ (inflationary vs deflationary, policy revealed up-ahead vs withheld) [52] and different pay-out schemes for contributors (miners) that strongly influence their incentives to provide service for the platform [109, 112, 149, 155]. These tokens are also typically traded on the exchange, which make them all the more challenging to model since we must account for market speculation and other forces [43, 55]. Further, we have large degrees of freedom in designing the monetary policy to facilitate certain micro-economic incentive goals. For example, we may provide early users of the network more tokens for being early adopters. Or we may increase the tokens paid to a router over time the longer it is engaged with the system the longer a router is engaged with the system. Some of these broadly economic research challenges are detailed below and pose promising venues of future research to realize systems like PayPlace and Datanet in practice:

- What should be the rate at which new coins are minted? Should this be fixed at

all? Should the mining pool be fixed? While a deflationary model increases the likelihood of growth in coin value over time (provided the network grows in value), the factors that influence the coin minting rate over the span of several years may be impratical to predict at the onset. In-fact, if coins rise in value beyond the underlying market utility of the networks' services due to severe scarcity, the network may well collapse as consumers switch to alternate/competing service providers that offer market-competitive rates and expose them to less market instability.

- How should mining rewards change over time? Initially, contributors are compensated from the unminted pool of coins created by the company. However, as cash flows in from customers in terms of the token, a larger portion of the contributor compensation would come from the fees paid by consumers. We would then expect contributor rewards come from consumer fees. How should the contributor reward be changed over time, accounting for the decrease in unminted pool and increase in consumer demand and hence payments?
- Under what conditions can a "stable" exchange rate emerge for these tokens? What is the expected behavior of contributors and speculators when tokens are no longer a store of future value? In a traditional stock market an asset with zero growth nevertheless has a value and an incentive for shareholders to hold on to it due to fixed dividends that it pays out (as a result of the cash flows sustained at the final growth stage). Since there is no notion of dividends here, what is the expected market behavior (consumers/contributors/speculators/early-spage investors) in this case?

These venues of future work presented above address both limitations in current work and explore new applications of study.

Bibliography

[1] [bitcoin-development] anti dos for tx replacement. https://lists.linuxfoundation.org/pipermail/bitcoin-dev/2013-April/002433.html. Accessed: 2019-12-07. Cited on pages 138 and 143.

[2] BLS12-381 curve and aggregation libraries. https://github.com/ethereum/eth2.0-pm/issues/13. Accessed: 2019-12-07. Cited on pages 178 and 191.

[3] EIP 1108: Reduce alt_bn128 precompile gas costs . https://eips.ethereum.org/EIPS/eip-1108. Accessed: 2020-04-05. Cited on page 178.

[4] Eip 2028: Transaction data gas cost reduction. https://eips.ethereum.org/EIPS/eip-2028. Accessed: 2020-26-03. Cited on page 129.

[5] Eip 2537. https://github.com/matter-labs/EIPs/blob/bls12_381/EIPS/eip-2537.md. Accessed: 2020-04-05. Cited on pages 178 and 191.

[6] Lightning Network Search and Analysis Engine. https://1ml.com/. Accessed: 2020-07-30. Cited on page 144.

[7] Loopring. https://loopring.org/. Accessed: 2020-07-30. Cited on pages 172 and 191.

[8] Minimum Viable Plasma. https://ethresear.ch/t/minimal-viable-plasma/426. Accessed: 2019-12-07. Cited on pages 122, 137, 139, and 144.

[9] More viable plasma. https://ethresear.ch/t/more-viable-plasma/2160. Accessed:2020-07-30. Cited on pages 137, 139, and 144.

[10] A note on data availability and erasure coding. https://github.com/ethereum/research/wiki/A-note-on-data-availability-and-erasure-coding. Accessed: 2019-12-07. Cited on page 137.

[11] On-Chain Scaling to Potentially 500 tx/sec through mass tx validation. https://ethresear.ch/t/on-chain-scaling-to-potentially-500-

tx-sec-through-mass-tx-validation/3477. Accessed: 2019-12-07. Cited on pages 136, 145, 155, 156, and 178.

[12] Plasma Debit: Arbitrary Denomination Payments in Plasma Cash. https://ethresear.ch/t/plasma-debit-arbitrary-denomination-payments-in-plasma-cash/2198. Accessed: 2019-12-07. Cited on page 144.

[13] Polkadot Keys. hhttps://wiki.polkadot.network/docs/en/learn-keys. Accessed: 2020-05-04. Cited on page 142.

[14] Pragmatic Signature Aggregaton with BLS. https://ethresear.ch/t/pragmatic-signature-aggregation-with-bls/2105. Accessed: 2019-12-07. Cited on page 142.

[15] Precompiled SNARK Pairings for BLS Signatures. https://ethresear.ch/t/precompiled-snark-pairing-for-bls-signatures/3196/3. Accessed: 2019-12-07. Cited on pages 178 and 191.

[16] Solidity Docs. https://solidity.readthedocs.io/en/v0.5.13/. Accessed: 2019-12-07. Cited on page 142.

[17] Switch from BN254 to BLS12-381. https://github.com/zcash/zcash/issues/2502. Accessed: 2019-12-07. Cited on page 142.

[18] ZoKrates. https://github.com/Zokrates/ZoKrates. Accessed: 2020-07-30. Cited on page 174.

[19] Statista Dossier on Smart Cities. Technical report, Statista, 2019. https://www.statista.com/study/51495/smart-cities/. Cited on pages 4 and 110.

[20] Array of Things. https://arrayofthings.github.io/, 2020. Cited on page 114.

[21] 3GPP. Security architecture and procedures for 5G System. Technical Report 33.501, 3rd Generation Partnership Project (3GPP), July 2018. https://www.etsi.org/deliver/etsi_ts/133500_133599/133501/15.01.00_60/ts_133501v150100p.pdf version 15.1.0. Cited on pages 110, 116, and 125.

[22] A. Faz-Hernandez, S. Scott, N. Sullivan, R. Wahby, C. Wood. Hashing to Elliptic Curves - draft-irtf-cfrg-hash-to-curve-04. https://tools.ietf.org/html/draft-irtf-cfrg-hash-to-curve-04. Accessed: 2020-05-04. Cited on

page 142.

[23] John Adler and Mikerah Quintyne-Collins. Building scalable decentralized payment systems. *arXiv preprint arXiv:1904.06441*, 2019. Cited on page 137.

[24] M. Afanasyev, T. Chen, G. M. Voelker, and A. C. Snoeren. Usage Patterns in an Urban WiFi Network. *IEEE/ACM Transactions on Networking*, 18(5):1359–1372, Oct 2010. Cited on page 76.

[25] Ammbr. Ammbr Whitepaper. Technical report, Ammbr Foundation, 2018. https://ammbr.com/docs/2018/11/Ammbr_Whitepaper.pdf. Cited on page 114.

[26] Baoyi An, Mingjun Xiao, An Liu, Guoju Gao, and Hui Zhao. Truthful crowdsensed data trading based on reverse auction and blockchain. In *International Conference on Database Systems for Advanced Applications*, pages 292–309. Springer, 2019. Cited on page 136.

[27] Sergey Andreev, Vitaly Petrov, Mischa Dohler, and Halim Yanikomeroglu. Future of ultra-dense networks beyond 5g: harnessing heterogeneous moving cells. *IEEE Communications Magazine*, 57(6):86–92, 2019. Cited on page 110.

[28] Jeffrey G Andrews, Stefano Buzzi, Wan Choi, Stephen V Hanly, Angel Lozano, Anthony CK Soong, and Jianzhong Charlie Zhang. What will 5g be? *IEEE Journal on selected areas in communications*, 32(6):1065–1082, 2014. Cited on pages 2 and 110.

[29] Matthew Andrews, Glenn Bruns, Mustafa Doğru, and Hyoseop Lee. Understanding quota dynamics in wireless networks. *ACM Transactions on Internet Technology*, 14(2-3):14, 2014. Cited on pages 26 and 40.

[30] Android Developer. Profile battery usage with Batterystats and Battery Historian. https://developer.android.com/topic/performance/power/setup-battery-historian, 2020. Cited on page 132.

[31] Android Developer. SafetyNet Attestation API. https://developer.android.com/training/safetynet/attestation, 2020. Cited on pages 128 and 131.

[32] Ehsan Aryafar, Alireza Keshavarz-Haddad, Michael Wang, and Mung Chiang. Rat selection games in hetnets. In *2013 Proceedings IEEE INFOCOM*, pages 998–1006. IEEE, 2013. Cited on page 190.

[33] AT&T. Unlimited data plans, 2017. https://www.att.com/plans/unlimited-

data-plans.html. Cited on page 23.

[34] Lawrence M Ausubel and Paul R Milgrom. Ascending auctions with package bidding. *Advances in Theoretical Economics*, 1(1), 2002. Cited on page 90.

[35] Sabina Baraković and Lea Skorin-Kapov. Survey and Challenges of QoE Management Issues in Wireless Networks. *Journal of Computer Networks and Communications*, 2013, 2013. Cited on page 68.

[36] Dario Bega, Marco Gramaglia, Marco Fiore, Albert Banchs, and Xavier Costa-Perez. Aztec: Anticipatory capacity allocation for zero-touch network slicing. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, pages 794–803. IEEE, 2020. Cited on page 71.

[37] Ferenc Béres, Istvan Andras Seres, and András A Benczúr. A cryptoeconomic traffic analysis of bitcoins lightning network. *arXiv preprint arXiv:1911.09432*, 2019. Cited on pages 122 and 144.

[38] Dan Boneh, Manu Drijvers, and Gregory Neven. Compact multi-signatures for smaller blockchains. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 435–464. Springer, 2018. Cited on pages 142 and 178.

[39] Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 416–432. Springer, 2003. Cited on pages 138, 142, 152, and 178.

[40] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 514–532. Springer, 2001. Cited on pages 142 and 152.

[41] F. Capozzi, G. Piro, L. A. Grieco, G. Boggia, and P. Camarda. Downlink Packet Scheduling in LTE Cellular Networks: Key Design Issues and a Survey. *IEEE Communications Surveys Tutorials*, 15(2):678–700, Second 2013. Cited on pages 74 and 75.

[42] Juan Pablo Carrascal and Karen Church. An in-situ study of mobile app & mobile search interactions. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 2739–2748, 2015. Cited on

pages 3, 7, and 20.

[43] Christian Catalini and Joshua S Gans. Some simple economics of the blockchain. Technical report, National Bureau of Economic Research, 2016. https://www.nber.org/papers/w22952. Cited on page 193.

[44] Shuchi Chawla, Jason D. Hartline, David L. Malec, and Balasubramanian Sivan. Multi-parameter Mechanism Design and Sequential Posted Pricing. In *Proceedings of the Forty-second ACM Symposium on Theory of Computing*, STOC '10, pages 311–320, Cambridge, Massachusetts, USA, 2010. ACM. Cited on page 81.

[45] David Chen, Zhiyue Zhang, Ambrish Krishnan, and Bhaskar Krishnamachari. Payflow: Micropayments for bandwidth reservations in software defined networks. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications Workshops*, pages 26–31. IEEE, 2019. Cited on pages 21, 112, and 128.

[46] Chia Network. BLS signatures in C++, using the relic toolkit. https://github.com/Chia-Network/bls-signatures. Accessed: 2020-05-04. Cited on page 142.

[47] Chia Network. Bls signatures. https://github.com/Chia-Network/bls-signatures, 2020. Cited on page 175.

[48] John Chon and Hojung Cha. Lifemap: A smartphone-based context provider for location-based services. *IEEE Pervasive Computing*, 10(2):58–67, 2011. Cited on pages xix, 5, 116, and 117.

[49] Yohan Chon, Hyojeong Shin, Elmurod Talipov, and Hojung Cha. Evaluating mobility models for temporal prediction with high-granularity mobility data. In *2012 IEEE International Conference on Pervasive Computing and Communications*, pages 206–212. IEEE, 2012. Cited on page 116.

[50] Cisco. Cisco Annual Internet Report (2018–2023) White Paper . Technical report, 2017. Cited on page 2.

[51] Cisco. Cisco Visual Networking Index: Global mobile data traffic forecast update, 2016–2021 white paper, 2017. http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html. Cited on page 23.

[52] Grégory Claeys, Maria Demertzis, and Konstantinos Efstathiou. Cryptocurren-

cies and monetary policy. Technical report, Bruegel Policy Contribution, 2018. Cited on page 193.

[53] Ethereum Alarm Clock. Welcome to Ethereum Alarm Clock's documentation. https://ethereum-alarm-clock.readthedocs.io/en/latest/. Accessed: 2020-05-02. Cited on page 152.

[54] Paolo Collela. 5G and IoT: Ushering in a new era. https://www.ericsson.com/en/about-us/company-facts/ericsson-worldwide/india/authored-articles/5g-and-iot-ushering-in-a-new-era, 2017. Cited on pages 4 and 110.

[55] John P Conley et al. Blockchain and the economics of crypto-tokens and initial coin offerings. *Vanderbilt University Department of Economics Working Papers*, (17-00008), 2017. Cited on page 193.

[56] Vicki M Coppinger, Vernon Smith, and Jon A Titus. Incentives and Behavior in English, Dutch and Sealed-Bid Auctions. *Economic Inquiry*, 18(1):1–22, 1980. Cited on page 82.

[57] Peter Cramton. Simultaneous Ascending Auctions. *Wiley Encyclopedia of Operations Research and Management Science*, 2010. Cited on page 70.

[58] Peter Cramton. Spectrum Auction Design. *Review of Industrial Organization*, 42(2):161–190, 2013. Cited on pages 70, 82, and 107.

[59] D. Boneh, R. Wahby, S. Gorbunov, H. Wee, Z. Zhang. draft-irtf-cfrg-bls-signature-00. https://tools.ietf.org/html/draft-irtf-cfrg-bls-signature-00. Accessed: 2019-12-07. Cited on page 142.

[60] Christian Decker and Roger Wattenhofer. A fast and scalable payment network with bitcoin duplex micropayment channels. In *Symposium on Self-Stabilizing Systems*, pages 3–18. Springer, 2015. Cited on page 121.

[61] DMR. 110 Uber Statistics, Demographics and Facts (2020). https://expandedramblings.com/index.php/uber-statistics/. Accessed: 2020-07-30. Cited on page 174.

[62] Wei Dong, Swati Rallapalli, Rittwik Jana, Lili Qiu, KK Ramakrishnan, Leo Razoumov, Yin Zhang, and Tae Won Cho. ideal: Incentivized dynamic cellular offloading via auctions. *IEEE/ACM Transactions on Networking*, 22(4):1271–1284, 2014. Cited on pages 20 and 26.

[63] Jagadeesh M Dyaberi, Benjamin Parsons, Vijay S Pai, Karthik Kannan, Yih-Farn Robin Chen, Rittwik Jana, Daniel Stern, and Alexander Varshavsky. Managing cellular congestion using incentives. *IEEE Communications Magazine*, 50(11), 2012. Cited on pages 19 and 26.

[64] Stefan Dziembowski, Lisa Eckey, Sebastian Faust, and Daniel Malinowski. Perun: Virtual payment channels over cryptographic currencies. *IACR Cryptology ePrint Archive*, 2017:635, 2017. Cited on pages 136 and 143.

[65] Stefan Dziembowski, Grzegorz Fabianski, Sebastian Faust, and Siavash Riahi. Lower bounds for off-chain protocols: Exploring the limits of plasma. *IACR Cryptol. ePrint Arch.*, 2020:175, 2020. Cited on page 137.

[66] Salvatore D'Oro, Francesco Restuccia, Alessandro Talamonti, and Tommaso Melodia. The slice is served: Enforcing radio access network slicing in virtualized 5g systems. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, pages 442–450. IEEE, 2019. Cited on page 71.

[67] Melhem El Helou, Marc Ibrahim, Samer Lahoud, Kinda Khawam, Dany Mezher, and Bernard Cousin. A network-assisted approach for rat selection in heterogeneous cellular networks. *IEEE Journal on Selected Areas in Communications*, 33(6):1055–1067, 2015. Cited on page 190.

[68] Mohamed El-Sayed, Amit Mukhopadhyay, Carlos Urrutia-Valdés, and Z John Zhao. Mobile data explosion: Monetizing the opportunity through dynamic policies and qos pipes. *Bell Labs Technical Journal*, 16(2):79–99, 2011. Cited on pages 19 and 26.

[69] M. S. ElBamby and K. M. F. Elsayed. An auction approach to resource allocation with interference coordination in LTE-A systems. In *2014 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1885–1890, Istanbul, Turkey, April 2014. Cited on pages 20, 68, and 71.

[70] Digi-Key Electronics. IoT Cellular Data Plans. https://www.digikey.com/en/resources/iot-resource-center/iot-cellular-data-plans, 2020. Cited on page 109.

[71] Felix Engelmann, Henning Kopp, Frank Kargl, Florian Glaser, and Christof Weinhardt. Towards an economic analysis of routing in payment channel networks. In *Proceedings of the 1st Workshop on Scalable and Resilient Infrastructures for*

Distributed Ledgers, pages 1–6, 2017. Cited on page 144.

[72] EthHub. Istanbul roadmap. https://docs.ethhub.io/ethereum-roadmap/istanbul/. Accessed: 2019-12-07. Cited on page 178.

[73] Shaohan Feng, Wenbo Wang, Dusit Niyato, Dong In Kim, and Ping Wang. Competitive data trading in wireless-powered internet of things (iot) crowdsensing systems with blockchain. In *2018 IEEE International Conference on Communication Systems (ICCS)*, pages 289–394. IEEE, 2018. Cited on page 136.

[74] Gerhard P Fettweis. The tactile internet: Applications and challenges. *IEEE Vehicular Technology Magazine*, 9(1):64–70, 2014. Cited on page 2.

[75] Forbes. Amazon moves towards ledgers and blockchain. https://www.forbes.com/sites/davidteich/2019/09/10/amazon-moves-towards-ledgers-and-blockchain/. Accessed: 2019-12-07. Cited on page 135.

[76] Forbes. Hailing Rides Down Crypto Lane: The Future of Ridesharing. https://www.forbes.com/sites/andrewrossow/2018/07/18/hailing-rides-down-crypto-lane-the-future-of-ridesharing. Accessed: 2019-12-07. Cited on page 135.

[77] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina. Network Slicing in 5G: Survey and Challenges. *IEEE Communications Magazine*, 55(5):94–100, May 2017. Cited on pages 12, 69, and 71.

[78] Xenofon Foukas, Mahesh K. Marina, and Kimon Kontovasilis. Orion: RAN Slicing for a Flexible and Cost-Effective Multi-Service Mobile Network Architecture. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking*, MobiCom '17, pages 127–140, Snowbird, Utah, USA, 2017. ACM. Cited on pages 69 and 71.

[79] Xenofon Foukas, Georgios Patounas, Ahmed Elmokashfi, and Mahesh K Marina. Network slicing in 5g: Survey and challenges. *IEEE Communications Magazine*, 55(5):94–100, 2017. Cited on page 3.

[80] FreeRadius. FreeRADIUS - a multi-protocol policy server. https://github.com/FreeRADIUS/freeradius-server, 2020. Cited on page 130.

[81] Arnaud Freville. The multidimensional 0-1 knapsack problem: An overview.

European Journal of Operational Research, 155(1):1–21, 2004. Cited on page 82.

[82] Kensuke Fukuda, Hirochika Asai, and Kenichi Nagami. Tracking the Evolution and Diversity in Network Usage of Smartphones. In *Proceedings of the 2015 Internet Measurement Conference*, IMC '15, pages 253–266, Tokyo, Japan, 2015. ACM. Cited on page 76.

[83] Andrea Giovannucci, J. A. Rodriguez-Aguilar, Jesus Cerquides, and Ulle Endriss. Winner Determination for Mixed Multi-unit Combinatorial Auctions via Petri Nets. In *Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '07, pages 104:1–104:8, Honolulu, Hawaii, 2007. ACM. Cited on page 82.

[84] Jens Groth. On the size of pairing-based non-interactive arguments. In *Annual international conference on the theory and applications of cryptographic techniques*, pages 305–326. Springer, 2016. Cited on page 174.

[85] Lewis Gudgeon, Pedro Moreno-Sanchez, Stefanie Roos, Patrick McCorry, and Arthur Gervais. Sok: Off the chain transactions. *IACR Cryptol. ePrint Arch.*, 2019:360, 2019. Cited on page 137.

[86] Ramakrishna Gummadi, Peter Key, and Alexandre Proutiere. Repeated Auctions under Budget Constraints: Optimal Bidding Strategies and Equilibria. In *the Eighth Ad Auction Workshop*, 2012. Cited on pages 21 and 71.

[87] Sangtae Ha, Soumya Sen, Carlee Joe-Wong, Youngbin Im, and Mung Chiang. Tube: Time-dependent pricing for mobile data. In *Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, SIGCOMM '12, pages 247–258, Helsinki, Finland, 2012. ACM. Cited on pages 8 and 70.

[88] Sangtae Ha, Soumya Sen, Carlee Joe-Wong, Youngbin Im, and Mung Chiang. Tube: time-dependent pricing for mobile data. *ACM SIGCOMM Computer Communication Review*, 42(4):247–258, 2012. Cited on pages 24 and 26.

[89] Amir Haleem, Andrew Allen, Andrew Thompson, Marc Nijdam, and Rahul Garg. Helium: A Decentralized Wireless Network. Technical report, Helium Systems Inc., 2018. http://whitepaper.helium.com/. Cited on pages 21, 114, 135, and 189.

[90] Shai Halevi, Danny Harnik, Benny Pinkas, and Alexandra Shulman-Peleg. Proofs of ownership in remote storage systems. In *Proceedings of the 18th ACM conference on Computer and communications security*, pages 491–500. ACM, 2011. Cited on page 149.

[91] Bin Han, Vincenzo Sciancalepore, Di Feng, Xavier Costa-Perez, and Hans D Schotten. A utility-driven multi-queue admission control solution for network slicing. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, pages 55–63. IEEE, 2019. Cited on page 71.

[92] Madhumitha Harishankar, Sireesha Pilaka, Pragya Sharma, Nagarjun Srinivasan, Carlee Joe-Wong, and Patrick Tague. Procuring spontaneous session-level resource guarantees for real-time applications: An auction approach. *IEEE Journal on Selected Areas in Communications*, 37(7):1534–1548, 2019. Cited on page 129.

[93] Madhumitha Harishankar, Patrick Tague, and Carlee Joe-Wong. Network Slicing as an Ad-Hoc Service: Opportunities and Challenges in Enabling User-Driven Resource Management in 5G. In *Proceedings of 1st International Workshop on Trustworthy and Real-time Edge Computing for Cyber-Physical Systems (TREC4CPS)*, Nashville, TN, USA, Dec 2018. Institute for Software Integrated Systems, Vanderbilt University. Cited on pages 69 and 71.

[94] Kevin Hasley. Smart cities: A more liveable future, today, 2018. Cited on pages 4 and 110.

[95] Junxian Huang, Qiang Xu, Birjodh Tiwana, Z Morley Mao, Ming Zhang, and Paramvir Bahl. Anatomizing application performance differences on smartphones. In *Proc. of ACM MobiSys*, pages 165–178. ACM, 2010. Cited on page 23.

[96] Speedtest Intelligence. Speedtest global index. https://www.speedtest.net/global-index, 2020. Cited on page 118.

[97] Maarten Janssen, Vladimir Karamychev, and Bernhard Kasberger. Budget Constraints in Combinatorial Clock Auctions. Cited on pages 21 and 71.

[98] Michael Jarschel, Daniel Schlosser, Sven Scheuring, and Tobias Hoßfeld. Gaming in the clouds: QoE and the users' perspective. *Mathematical and Computer Modelling*, 57(11-12):2883–2894, 2013. Cited on page 68.

[99] Libin Jiang, Shyam Parekh, and Jean Walrand. Time-dependent network pricing

and bandwidth trading. In *Proc. of IEEE Network Operations and Management Symposium Workshops*, pages 193–200, 2008. Cited on pages 19, 20, and 26.

[100] Carlee Joe-Wong, Sangtae Ha, and Mung Chiang. Sponsoring mobile data: An economic analysis of the impact on users and content providers. In *Proc. of IEEE INFOCOM*, pages 1499–1507. IEEE, 2015. Cited on page 28.

[101] Bumsoo Kang, Inseok Hwang, Jinho Lee, Seungchul Lee, Taegyeong Lee, Youngjae Chang, and Min Kyung Lee. My being to your place, your being to my place: Co-present robotic avatars create illusion of living together. In *Proc. 16th Annual International Conference on Mobile Systems, Applications, and Services*, pages 54–67. ACM, 2018. Cited on page 2.

[102] A. Karamoozian, A. Hafid, M. Boushaba, and M. Afzali. QoS-aware resource allocation for mobile media services in cloud environment. In *2016 13th IEEE Annual Consumer Communications Networking Conference (CCNC)*, pages 732–737, Jan 2016. Cited on page 69.

[103] Kellen. Verizon introduces popdata, 30 and 60 minute all-you-can-stream 4g lte sessions at $2 or $3, 2016. http://www.droid-life.com/2016/10/13/verizon-popdata-cost-available/. Cited on page 65.

[104] Terence Kelly. Generalized Knapsack Solvers for Multi-unit Combinatorial Auctions: Analysis and Application to Computational Resource Allocation. In *Proceedings of the 6th AAMAS International Conference on Agent-Mediated Electronic Commerce: Theories for and Engineering of Distributed Mechanisms and Systems*, AAMAS'04, pages 73–86, New York, NY, 2005. Springer-Verlag. Cited on page 82.

[105] Alireza Keshavarz-Haddad, Ehsan Aryafar, Michael Wang, and Mung Chiang. Hetnets selection by clients: convergence, efficiency, and practicality. *IEEE/ACM Transactions on Networking*, 25(1):406–419, 2016. Cited on page 190.

[106] Rami Khalil, Arthur Gervais, and G Felley. Nocust-a non-custodial 2nd-layer financial intermediary. *IACR Cryptol. ePrint Arch.*, 2018:642, 2018. Cited on page 137.

[107] Ramy Abdelmageed Ebrahim Khalil and Arthur Gervais. System and method for scaling blockchain networks with secure off-chain payment hubs, May 9 2019. US Patent App. 16/183,709. Cited on pages 137 and 144.

[108] Vijay Krishna. *Auction Theory*. Academic press, 2009. Cited on pages 6, 81, 89, 92, and 94.

[109] Joshua A Kroll, Ian C Davey, and Edward W Felten. The economics of bitcoin mining, or bitcoin in the presence of adversaries. In *Proceedings of WEIS*, volume 2013, page 11, 2013. Cited on page 193.

[110] Piotr Krysta, Orestis Telelis, and Carmine Ventre. Mechanisms for Multi-unit Combinatorial Auctions with a Few Distinct Goods. In *Proceedings of the 2013 International Conference on Autonomous Agents and Multi-agent Systems*, AAMAS '13, pages 691–698, St. Paul, MN, USA, 2013. International Foundation for Autonomous Agents and Multiagent Systems. Cited on page 82.

[111] Matter Labs. Introducing Matter Testnet. https://medium.com/matter-labs/introducing-matter-testnet-502fab5a6f17. Accessed: 2020-07-30. Cited on page 174.

[112] Zongxi Li, A Max Reppen, and Ronnie Sircar. A mean field games model for cryptocurrency mining. *arXiv preprint arXiv:1912.01952*, 2019. Cited on page 193.

[113] Patrick Loiseau, Galina Schwartz, John Musacchio, Saurabh Amin, and S Shankar Sastry. Incentive mechanisms for internet congestion management: Fixed-budget rebate versus time-of-day pricing. *IEEE/ACM Transactions on Networking*, 22(2):647–661, 2014. Cited on page 26.

[114] Loopring. Loopring 2020 Development Roadmap. https://medium.com/loopring-protocol/loopring-2020-development-roadmap-d660b93563e3. Accessed: 2020-07-30. Cited on page 174.

[115] Loopring. Loopring Testing Phase 1: Data Recap. https://medium.com/loopring-protocol/loopring-testing-phase-1-data-recap-ed0c67396870. Accessed: 2020-07-30. Cited on pages 174 and 178.

[116] Cristina Marquez, Marco Gramaglia, Marco Fiore, Albert Banchs, and Xavier Costa-Perez. How Should I Slice My Network?: A Multi-Service Empirical Evaluation of Resource Sharing Efficiency. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*, MobiCom '18, pages 191–206, New Delhi, India, 2018. ACM. Cited on pages 69 and 71.

[117] Matter Labs. Matter Labs - Trustless Scalability and Privacy. https://matter-

labs.io/. Accessed: 2020-07-30. Cited on pages 172 and 191.

[118] Vasilios Mavroudis, Karl Wüst, Aritra Dhar, Kari Kostiainen, and Srdjan Capkun. Snappy: Fast on-chain payments with practical collaterals. *arXiv preprint arXiv:2001.01278*, 2020. Cited on pages 136 and 145.

[119] W. Miao, G. Min, Y. Jiang, X. Jin, and H. Wang. QoS-aware resource allocation for LTE-A systems with carrier aggregation. In *2014 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1403–1408, April 2014. Cited on page 68.

[120] Andrew Miller, Iddo Bentov, Surya Bakshi, Ranjit Kumaresan, and Patrick McCorry. Sprites and state channels: Payment networks that go faster than lightning. In *International Conference on Financial Cryptography and Data Security*, pages 508–526. Springer, 2019. Cited on pages 121 and 122.

[121] Andrew Miller, Iddo Bentov, Ranjit Kumaresan, Christopher Cordi, and Patrick McCorry. Sprites and state channels: Payment networks that go faster than lightning. *arXiv preprint arXiv:1702.05812*, 2017. Cited on pages 136 and 143.

[122] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013. Cited on page 54.

[123] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015. Cited on page 54.

[124] Nozomu Muto and Shirata Yasuhiro. Goods Revenue Monotonicity in Combinatorial Auctions. Discussion Papers 2013-13, Graduate School of Economics, Hitotsubashi University, October 2013. Cited on pages 89 and 91.

[125] Network Working Group. RFC 5176 - Dynamic Authorization Extensions to Remote Authentication Dial In User Service (RADIUS). https://tools.ietf.org/html/rfc5176, 2008. Cited on page 127.

[126] Nodle. The Nodle Network: A New Economic Model to Free the Mobile Internet. Technical report, Nodle. https://docsend.com/view/gjtn4jc. Cited on pages 135, 136, and 189.

[127] Oberlo. 10 Amazon Statistics You Need to Know in 2020. https://www.oberlo.in/blog/amazon-statistics. Accessed: 2020-07-30. Cited on page 174.

[128] Andrew Odlyzko. Will smart pricing finally take off? In Soumya Sen, Carlee Joe-Wong, Sangtae Ha, and Mung Chiang, editors, *Smart Data Pricing*, pages 1–33. Wiley Online Library. Cited on pages 7 and 24.

[129] Business of Apps. Uber Revenue and Usage Statistics (2020). https://www.businessofapps.com/data/uber-statistics/. Accessed: 2020-07-30. Cited on page 174.

[130] Abner Olivieri. Which devices support WPA2-Enterprise? https://support.screen.cloud/hc/en-gb/articles/360000262918-Which-Devices-Support-WPA2-Enterprise, 2020. Cited on page 119.

[131] Orange. EE monthly price plans, 2017. http://ee.co.uk/content/dam/ee-help/help-pdfs/ee-pay-monthly-price-plans-may-2017.pdf. Cited on page 23.

[132] Ozgur Oyman and Sarabjot Singh. Quality of experience for http adaptive streaming services. *IEEE Communications Magazine*, 50(4), 2012. Cited on page 75.

[133] Peter Palensky and Dietmar Dietrich. Demand side management: Demand response, intelligent energy systems, and smart loads. *IEEE transactions on industrial informatics*, 7(3):381–388, 2011. Cited on page 26.

[134] Hanjin Park, Youngmi Jin, Jooho Yoon, and Yung Yi. On the economic effects of user-oriented delayed wi-fi offloading. *IEEE Trans. on Wireless Communications*, 15(4):2684–2697, 2016. Cited on pages 20, 24, and 26.

[135] David C. Parkes and Satinder Singh. An MDP-based Approach to Online Mechanism Design. In *Proceedings of the 16th International Conference on Neural Information Processing Systems*, NIPS'03, pages 791–798, Whistler, British Columbia, Canada, 2003. MIT Press. Cited on pages 92 and 93.

[136] Yuyang Peng, Dong-Ki Kang, Fawaz Al-Hazemi, and Chan-Hyun Youn. Energy and QoS Aware Resource Allocation for Heterogeneous Sustainable Cloud Datacenters. *Opt. Switch. Netw.*, 23(P3):225–240, January 2017. Cited on page 69.

[137] V. Petrov, M. A. Lema, M. Gapeyenko, K. Antonakoglou, D. Moltchanov,

F. Sardis, A. Samuylov, S. Andreev, Y. Koucheryavy, and M. Dohler. Achieving End-to-End Reliability of Mission-Critical Traffic in Softwarized 5G Networks. *IEEE Journal on Selected Areas in Communications*, 36(3):485–501, March 2018. Cited on page 71.

[138] Sandro Pinto, Tiago Gomes, Jorge Pereira, Jorge Cabral, and Adriano Tavares. IIoTEED: an enhanced, trusted execution environment for industrial IoT edge devices. *IEEE Internet Computing*, 21(1):40–47, 2017. Cited on page 113.

[139] David Pisinger. Where Are the Hard Knapsack Problems? *Comput. Oper. Res.*, 32(9):2271–2284, September 2005. Cited on page 83.

[140] Joseph Poon and Vitalik Buterin. Plasma: Scalable autonomous smart contracts. *White paper*, pages 1–47, 2017. Cited on page 137.

[141] Warren B Powell. Clearing the Jungle of Stochastic Optimization. In *Bridging data and decisions*, pages 109–137. Informs, 2014. Cited on page 93.

[142] Pavel Prihodko, Slava Zhigulin, Mykola Sahno, Aleksei Ostrovskiy, and Olaoluwa Osuntokun. Flare: An approach to routing in lightning network. *White Paper*, 2016. Cited on page 144.

[143] Qiaofeng Qin, Nakjung Choi, Muntasir Raihan Rahman, Marina Thottan, and Leandros Tassiulas. Network slicing in heterogeneous software-defined rans. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, pages 2371–2380. IEEE, 2020. Cited on page 71.

[144] Rahul Radhakrishnan, Gowri Sankar Ramachandran, and Bhaskar Krishnamachari. Sdpp: Streaming data payment protocol for data economy. In *2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pages 17–18. IEEE, 2019. Cited on pages 21, 112, and 128.

[145] Baharak Rastegari, Anne Condon, and Kevin Leyton-Brown. Revenue Monotonicity in Combinatorial Auctions. *SIGecom Exch.*, 7(1):45–47, dec 2007. Cited on pages 89 and 90.

[146] Filippo Rebecchi, Marcelo Dias De Amorim, Vania Conan, Andrea Passarella, Raffaele Bruno, and Marco Conti. Data offloading techniques in cellular networks: A survey. *IEEE Communications Surveys & Tutorials*, 17(2):580–603, 2014. Cited on page 192.

[147] RideGuru. How much does an Uber cost? A 2019 price analysis

around the world. https://ride.guru/content/newsroom/how-much-does-an-uber-cost-a-2019-price-analysis-around-the-world. Accessed: 2020-07-30. Cited on page 143.

[148] T. Ristenpart and S. Yilek. The power of proofs-of-possession: Securing multi-party signatures against rogue-key attacks. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 228–245. Springer, 2007. Cited on pages 141 and 158.

[149] Fabian Ritz and Alf Zugenmaier. The impact of uncle rewards on selfish mining in ethereum. In *2018 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pages 50–57. IEEE, 2018. Cited on page 193.

[150] Xavier Rival. Symbolic transfer function-based approaches to certified compilation. *ACM SIGPLAN Notices*, 39(1):1–13, 2004. Cited on page 128.

[151] Michael Römer and Taïeb Mellouli. Future Demand Uncertainty In Personnel Scheduling: Investigating Deterministic Lookahead Policies Using Optimization And Simulation. In *ECMS*, pages 502–507, Regensburg, Germany, 2016. Cited on page 93.

[152] Jake S. Cannell, Justin Sheek, Jay Freeman, Greg Hazel, Jennifer Rodriguez-Mueller, Eric Hou, and Brian J. Fox. Orchid: A Decentralized Network Routing Market. Technical report, Orchid Labs, 2019. https://www.orchid.com/assets/whitepaper/whitepaper.pdf. Cited on pages 21, 113, 114, 129, 135, and 189.

[153] David L Salamon, Gustav Simonsson, Jay Freeman, and Brian J Fox. Orchid: Enabling decentralized network formation and probabilistic micro-payments, 2018. Cited on page 112.

[154] SamKnows. Measuring Broadband America. https://www.measuringbroadbandamerica.com/, 2017. Cited on page 118.

[155] Okke Schrijvers, Joseph Bonneau, Dan Boneh, and Tim Roughgarden. Incentive compatibility of bitcoin mining pool reward functions. In *International Conference on Financial Cryptography and Data Security*, pages 477–498. Springer, 2016. Cited on page 193.

[156] Soumya Sen, Carlee Joe-Wong, Sangtae Ha, Jasika Bawa, and Mung Chiang. When the price is right: enabling time-dependent pricing of broadband data.

In *Proc. of ACM SIGCHI*, pages 2477–2486. ACM, 2013. Cited on pages 24 and 26.

[157] Soumya Sen, Carlee Joe-Wong, Sangtae Ha, and Mung Chiang. Incentivizing time-shifting of data: a survey of time-dependent pricing for internet access. *IEEE Communications Magazine*, 50(11), 2012. Cited on pages 7 and 24.

[158] Seoul Metropolitan City Office. Seoul Living Population. http://data.seoul.go.kr/dataVisual/seoul/seoulLivingPopulation.do, 2020. Cited on page 119.

[159] Muhammad Zubair Shafiq, Jeffrey Erman, Lusheng Ji, Alex X Liu, Jeffrey Pang, and Jia Wang. Understanding the impact of network dynamics on mobile video user engagement. *ACM SIGMETRICS Performance Evaluation Review*, 42(1):367–379, 2014. Cited on pages 3, 7, and 20.

[160] Vibhaalakshmi Sivaraman, Shaileshh Bojja Venkatakrishnan, Mohammad Alizadeh, Giulia Fanti, and Pramod Viswanath. Routing cryptocurrency with the spider network. In *Proceedings of the 17th ACM Workshop on Hot Topics in Networks*, pages 29–35, 2018. Cited on pages 112, 121, 122, 136, and 143.

[161] Smartiply. DataNinja. http://www.getdataninja.com/, 2020. Cited on page 114.

[162] Solidity. Contracts. https://solidity.readthedocs.io/en/v0.4.24/contracts.html, 2020. Cited on page 125.

[163] Mike Sorrentino. Verizon may give you unlimited data, up to an hour at a time. CNet, 2017. https://www.cnet.com/news/verizon-popdata-let-you-have-unlimited-data-but-only-for-an-hour/. Cited on page 46.

[164] Sprint. Unlimited cell phone plan, 2017. https://www.sprint.com/en/shop/plans/unlimited-cell-phone-plan.html. Cited on page 23.

[165] Thomas Stockhammer. Dynamic Adaptive Streaming over HTTP –: Standards and Design Principles. In *Proceedings of the Second Annual ACM Conference on Multimedia Systems*, MMSys '11, pages 133–144, San Jose, CA, USA, 2011. ACM. Cited on page 68.

[166] Jothi Prasanna Shanmuga Sundaram, Wan Du, and Zhiwei Zhao. A survey on lora networking: Research problems, current solutions and open issues. *IEEE Communications Surveys & Tutorials*, 2019. Cited on page 112.

[167] Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT press, 2018. Cited on pages 53, 60, 70, 101, 102, and 105.

[168] Michael Szydlo. Merkle tree traversal in log space and time. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 541–554. Springer, 2004. Cited on page 174.

[169] T-Mobile. Network pricing for the Internet of Things. https://www.t-mobile.com/business/iot/pricing, 2020. Cited on page 109.

[170] Dhruva Tirumala, Alexandre Galashov, Hyeonwoo Noh, Leonard Hasenclever, Razvan Pascanu, Jonathan Schwarz, Guillaume Desjardins, Wojciech Marian Czarnecki, Arun Ahuja, Yee Whye Teh, et al. Behavior priors for efficient reinforcement learning. *arXiv preprint arXiv:2010.14274*, 2020. Cited on page 64.

[171] Jehan Tremback, Justin Kilpatrick, Deborah Simpier, and Ben Wang. Althea whitepaper. Technical report, Hawk Networks, 2019. https://althea.net/whitepaper. Cited on pages 21 and 113.

[172] TruffleSuite. Ganache: One click blockchain. https://www.trufflesuite.com/ganache, 2020. Cited on page 130.

[173] Trusted Computing Group. How to Secure Network Equipment Against Attack. Technical report, 2018. https://trustedcomputinggroup.org/wp-content/uploads/TCG-on-Securing-Network-Equipment.pdf. Cited on page 129.

[174] Unifi. 802.11ac PRO Access Point. https://www.ui.com/unifi/unifi-ap-ac-pro/, 2020. Cited on page 130.

[175] Andrea Valenzano, Dario Mana, Claudio Borean, and Antonio Servetti. Mapping wifi measurements on openstreetmap data for wireless street coverage analysis. In *Free and Open Source Software for Geospatial (FOSS4G) Conference Proceedings*, volume 16, page 5, 2016. Cited on pages 5, 110, and 115.

[176] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. *arXiv preprint arXiv:1509.06461*, 2015. Cited on page 54.

[177] M Jalali Varnamkhasti. Overview of the algorithms for solving the multidimensional knapsack problems. *Advanced Studies in Biology*, 4(1):37–47, 2012. Cited on pages 20 and 82.

[178] Verizon Wireless. PopData FAQs, 2017. https://www.verizonwireless.com/support/popdata-faqs/. Cited on pages 14 and 24.

[179] Verizon Wireless. Verizon plan faqs, 2017. https://www.verizonwireless.com/support/the-verizon-plan-faqs/. Cited on page 38.

[180] Marko Vukolić. The quest for scalable blockchain fabric: Proof-of-work vs. bft replication. In *International workshop on open problems in network security*, pages 112–125. Springer, 2015. Cited on pages 121 and 135.

[181] Jingzhong Wang, Mengru Li, Yunhua He, Hong Li, Ke Xiao, and Chao Wang. A blockchain based privacy-preserving incentive mechanism in crowdsensing applications. *IEEE Access*, 6:17545–17556, 2018. Cited on page 136.

[182] Tong Wang, Pengcheng Li, Xibo Wang, Yunfeng Wang, Tianhao Guo, and Yue Cao. A comprehensive survey on mobile data offloading in heterogeneous network. *Wireless Networks*, 25(2):573–584, 2019. Cited on page 111.

[183] WIGLE. WIGLE.net. https://wigle.net/, 2020. Cited on page 115.

[184] Tilman Wolf, James Griffioen, Kenneth L. Calvert, Rudra Dutta, George N. Rouskas, Ilya Baldin, and Anna Nagurney. ChoiceNet: Toward an Economy Plane for the Internet. *ACM SIGCOMM Comput. Commun. Rev.*, 44(3):58–65, jul 2014. Cited on page 69.

[185] Mengbai Xiao, Chao Zhou, Viswanathan Swaminathan, Yao Liu, and Songqing Chen. Bas-360: Exploring spatial and temporal adaptability in 360-degree videos over http/2. IEEE, 2018. Cited on page 68.

[186] Qiang Xu, Sanjeev Mehrotra, Zhuoqing Mao, and Jin Li. Proteus: Network Performance Forecast for Real-time, Interactive Mobile Applications. In *Proceeding of the 11th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '13, pages 347–360, Taipei, Taiwan, 2013. ACM. Cited on pages 67 and 68.

[187] K. Yang, N. Prasad, and X. Wang. An Auction Approach to Resource Allocation in Uplink OFDMA Systems. *IEEE Transactions on Signal Processing*, 57(11):4482–4496, Nov 2009. Cited on pages 20, 68, and 71.

[188] YCharts. Bitcoin Average Transaction Fee. Accessed: 2020-07-30. Cited on page 143.

[189] YCharts. Ethereum Average Transaction Fee. https://ycharts.com/

indicators/ethereum_average_transaction_fee. Accessed: 2020-07-30. Cited on page 143.

[190] Kailiang Ying, Amit Ahlawat, Bilal Alsharifi, Yuexin Jiang, Priyank Thavai, and Wenliang Du. Truz-droid: Integrating trustzone with mobile operating system. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*, pages 14–27, 2018. Cited on page 128.

[191] Andrea Zanella, Nicola Bui, Angelo Castellani, Lorenzo Vangelista, and Michele Zorzi. Internet of things for smart cities. *IEEE Internet of Things journal*, 1(1):22–32, 2014. Cited on page 2.

[192] Cheng Zhang, Bo Gu, Sugang Xu, Kyoko Yamoriy, and Yoshiaki Tanaka. Time-dependent pricing for revenue maximization of network service providers considering users preference. In *Proc. of IEEE APNOMS*, pages 1–6, 2013. Cited on pages 19 and 26.

[193] Haijun Zhang, Na Liu, Xiaoli Chu, Keping Long, Abdol-Hamid Aghvami, and Victor CM Leung. Network slicing based 5g and future mobile networks: mobility, resource management, and challenges. *IEEE Communications Magazine*, 55(8):138–145, 2017. Cited on page 110.

[194] Jingjing Zhang, Lin Yang, Weipeng Cao, and Qiang Wang. Formal analysis of 5g eap-tls authentication protocol using proverif. *IEEE Access*, 8:23674–23688, 2020. Cited on pages 21 and 111.

[195] Liang Zhang, Weijie Wu, and Dan Wang. Time dependent pricing in wireless data networks: Flat-rate vs. usage-based schemes. In *Proc. of IEEE INFOCOM*, pages 700–708. IEEE, 2014. Cited on pages 19 and 26.

[196] Y. Zhang, D. Niyato, P. Wang, and E. Hossain. Auction-based Resource Allocation in Cognitive Radio Systems. *IEEE Communications Magazine*, 50(11):108–120, November 2012. Cited on page 71.

[197] L. Zheng, C. Joe-Wong, J. Chen, C. G. Brinton, C. W. Tan, and M. Chiang. Economic viability of a virtual ISP. In *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, pages 1–9, Atlanta, GA, USA, May 2017. Cited on pages 28 and 100.

[198] Liang Zheng, Carlee Joe-Wong, Matthew Andrews, and Mung Chiang. Optimizing data plans: Usage dynamics in mobile data networks. In *IEEE INFOCOM*

2018-IEEE Conference on Computer Communications, pages 2474–2482, Atlanta, GA, USA, 2018. IEEE. Cited on page 99.

[199] zkSync. zkSync: Tokens and Fees. https://zksync.io/faq/tokens.html. Accessed: 2020-07-30. Cited on page 178.

[200] Xuan Kelvin Zou, Jeffrey Erman, Vijay Gopalakrishnan, Emir Halepovic, Rittwik Jana, Xin Jin, Jennifer Rexford, and Rakesh K Sinha. Can accurate predictions improve video streaming in cellular networks? In *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*, pages 57–62. ACM, 2015. Cited on pages 3 and 68.