RESEARCH-ARTICLE

Centralized Traffic Engineering for Networked Farm Applications

AMMAR TAHIR, University of Illinois Urbana-Champaign, Urbana, IL, United States

YUESHEN LI, University of Illinois Urbana-Champaign, Urbana, IL, United States

JIANLI JIN, University of Illinois Urbana-Champaign, Urbana, IL, United States

DANIEL MOON, University of Illinois Urbana-Champaign, Urbana, IL, United States

CHANGXIN ZHANG, University of Illinois Urbana-Champaign, Urbana, IL, United States

AGANZE MIHIGO, University of Illinois Urbana-Champaign, Urbana, IL, United States

View all

Open Access Support provided by:

University of Illinois Urbana-Champaign

Published: 03 December 2025

Citation in BibTeX format

SEC '25: Tenth ACM/IEEE Symposium on Edge Computing
December 3 - 6, 2025
VA, Arlington, USA

Conference Sponsors:
SIGMOBILE

https://doi.org/10.1145/3769102.3770615
ISBN: 9798400722387

Centralized Traffic Engineering for Networked Farm Applications

Ammar Tahir, Yueshen Li, Jianli Jin, Daniel Moon, Changxin Zhang, Aganze Mihigo, Muhammad Taimoor Tariq, Radhika Mittal, Deepak Vasisht
University of Illinois Urbana-Champaign
Urbana, Illinois, United States

Abstract

Emerging farming techniques rely on smart devices such as multi-spectral cameras that collect fine-grained data, and tele-operated robots that perform tasks such as de-weeding, berry-picking, etc. These networked farm applications (requiring 10s of Mbps of throughput per device to the edge servers, with tens to hundreds of devices in a typical farm) must be supported on a wireless mesh network with limited capacity. In this work, we use these networked farm applications as a compelling case-study to design FarmNetes, a centralized traffic engineering (TE) system for wireless mesh networks. FarmNetes leverages explicit control over farm workloads to make centralized TE decisions (temporal flow schedules, sending rates, load-aware routes, and channel configurations) from an edge server, so as to best meet task requirements. FarmNetes' centralized TE decisions enable it to work with commodity devices and control how the network is shared across flows based on the desired policies (prioritization and fairness) irrespective of the underlying MAC layer link sharing mechanisms. This further enables MAC-agnostic reasoning of wireless network behavior when making TE decisions. Our evaluation, using testbeds in a farm and trace-driven simulations, shows how FarmNetes achieves 3 × higher end-end network throughput and better meets application demands, compared to status-quo wireless mesh strategies.

CCS Concepts

• **Networks** → **Control path algorithms**; **Traffic engineering algorithms**; **Wireless access networks**.

SEC '25, Arlington, VA, USA

ACM ISBN 979-8-4007-2238-7/25/12
https://doi.org/10.1145/3769102.3770615

ACM Reference Format:
Ammar Tahir, Yueshen Li, Jianli Jin, Daniel Moon, Changxin Zhang, Aganze Mihigo, Muhammad Taimoor Tariq, Radhika Mittal, Deepak Vasisht. 2025. Centralized Traffic Engineering for Networked Farm Applications. In *The Tenth ACM/IEEE Symposium on Edge Computing (SEC '25), December 3–6, 2025, Arlington, VA, USA.* The Tenth ACM/IEEE Symposium on Edge Computing (SEC '25) , 16 pages. https://doi.org/10.1145/3769102.3770615

1 Introduction

Emerging autonomous farming techniques, relying on smart networked devices such as robots, drones, and hyperspectral cameras, promise to increase farm yields, reduce wastage, and improve sustainability [17, 19, 33, 46, 53]. As we detail in §2, networked farm applications can be divided into two categories: (i) real-time tele-operation of robots (e.g. from edge servers), and (ii) periodic collection of large amounts of data generated by RGB, thermal, and hyperspectral cameras for offline analysis at the edge. A typical to large farm may require supporting tens to hundreds of such devices. Given that devices in the middle of a field cannot be connected with Ethernet cables, they must rely on wireless connectivity (commonly, a multi-hop WiFi mesh network [35, 41]). The capacity of such a wireless network is fundamentally limited by the available radio spectrum (i.e. it cannot be scaled indefinitely by simply adding more routers). Moreover, many farm devices must operate from under the crop canopy, where network throughput is further degraded (as we show in §3). In this work, we study how to best meet the requirements of networked farm applications at scale on such limited capacity wireless network. To that end, we develop FarmNetes, a centralized traffic engineering (TE) system for on-farm wireless mesh networks. [1]

We first begin with motivating the use of centralized TE in this context, before diving into the ensuing challenges. Farm applications require differential treatment of network traffic, where more urgent real-time remote operation tasks must be prioritized over data collection tasks. Such policies are inherently difficult to realize in a multi-hop wireless network due to interference among independently operating nodes. For instance, we must ensure that the low priority flows use only the spare capacity – not only on the nodes in their direct

[1] FarmNetes is short for Farm Network Traffic Engineering System.

paths, but also on the neighboring nodes along the path due to interference. While wireless MAC layer mechanisms have been proposed for flow prioritization under such settings, they only work partially (not being able to deal with interference from hidden terminals) and/or probabilistically (relying on stochastic back-offs) [57]. Some of the more recent (and yet to be realized) MAC-layer proposals on wireless time-sensitive networking [10, 11] make restrictive assumptions about per-packet timings and require inter-node time synchronization. More importantly, commodity WiFi devices neither support such sophisticated MAC layer mechanisms nor provide an interface to implement them.

Centralized traffic engineering (TE) provides a platform for enforcing such flow-sharing policies using simple commodity devices. In particular, it enables centrally computing flow timings, sending rates, and routes as per the desired policy using global network-wide view (e.g. the timings, rates, and routes for low priority flows can be computed such that they only use the network capacity spared by the high priority flows). This benefit of centralized TE has been well exploited in wired networks (e.g. in datacenters and private WANs [25, 26, 29, 34]). In contrast, centralized systems for wireless networks have largely been restricted to centrally computing routes based on simpler policies (e.g. shortest paths) on a mesh with fixed topology (i.e. fixed distance between nodes), and given input traffic [2, 43]. It has excluded other crucial *end* TE decisions that must be made as per the desired flow-sharing policy (such as flow timings, sending rates, and access point selection). These end TE decisions determine the input network traffic incident on the wireless mesh, and are thereby inter-twined with the mesh routing decisions. Due to uncontrollable workloads and unpredictable access link characteristics in traditional wireless mesh networks in homes and enterprises, these end TE decisions have fundamentally relied on ad hoc distributed protocols (that, as mentioned earlier, fail to support rich network-sharing policies).

FarmNetes is the first centralized TE system for wireless networks that comprehensively performs policy-driven flow scheduling, sending rate enforcement, access point selection, mesh routing, and channel allocations. The comprehensive TE decisions in FarmNetes are enabled by two observations about wireless mesh networks in farms, that makes them differ from those in traditional home or enterprise settings: (1) Farm workloads are known a priori and can be explicitly controlled (e.g. the flow timings and sending rates can be adjusted to prioritize more urgent remote operation tasks with more stringent demands). (2) Farm network characteristics are more predictable, given the relatively homogeneous setting, and the link characteristics can be profiled to enable informed TE decisions. FarmNetes leverages these observations to make comprehensive centralized TE decisions at an edge server, remotely configuring on-field devices (commodity WiFi routers and endhosts).

We must address several challenges to make TE decisions in wireless networks. The basis of any TE system is its ability to answer the following question: "if a flow sends data at rate X bits/s along route P, how much network resources will it consume?" The response to this question allows the TE system to reason about different choices (flow timings, routes, and rates) in a policy-driven manner. While the response is often straight-forward in a wired network (a flow sending at rate X bits/s along a route P would consume X bits/s of the network capacity on each link in the path P), it is non-trivial in a wireless network for multiple reasons.

First, the network resources (in terms of channel bandwidth in MHz) consumed to support a flow over a wireless link is not only a function of its sending rate but also the distance between the sender and receiver, and the environment (e.g. whether the transmission is blocked by crop canopy, or has a clear line of sight to the receiver). FarmNetes models these effects by exploiting the relatively stable and homogeneous farm environment to periodically collect sparse measurements of network throughput under different settings, and use the measurement results to translate a flow's datarate into the corresponding amount of WiFi channel bandwidth it consumes (as described in §4).

Second, the flow's transmission would consume resources not only on hops in its direct path, but also on neighboring devices at which it interferes. Computing a flow's wireless resource consumption is very tricky, since the effects of contention and interference on flow shares are dictated by ad hoc MAC layer wireless mechanisms which often change across generations [3, 11, 54], and are notoriously hard to reason about [21–23, 42]. FarmNetes sidesteps this difficulty by exploiting a unique vantage point enabled by its centralized TE decisions. More specifically, FarmNetes' centralized enforcement of flow sending rates allows it to cleanly abstract and reason about wireless network behavior agnostic of the underlying MAC layer mechanisms. The broad assumption we make, and empirically validate, is that if the incident demand on a node (as dictated by the enforced sending rates) is less than the node's capacity (in terms of its channel bandwidth), the underlying MAC and physical layer mechanisms are able to meet those demands. Driven by this observation, FarmNetes centrally computes and enforces flow sending rates so as to ensure that the cumulative amount of resources consumed at a node (either on a flow's direct path or due to interference) never exceed the node's capacity, and the network resources are divided as per the desired flow-sharing policies. This allows FarmNetes to directly control and reason about how the wireless network capacity is shared across flows. This is in contrast to prior work on modeling wireless performance [21–23, 28, 38, 42] that model specific MAC layer mechanisms, or make restrictive assumptions about them, to capture the effects of interference.

A third challenge stems from the fact that while the TE decisions depend on the resource consumption of flows on their direct paths and at their interfering nodes, the set of feasible paths and interfering nodes (that share the same wireless channel on the same frequency band) depends on channel allocation decisions that are themselves part of FarmNetes' TE design space. FarmNetes resolves this inter-dependency by splitting its decisions into multiple distinct phases: (1) It first makes use of high-level farm network characteristics to guide the WiFi mesh topology design, and its decoupled use of 2.4GHz and 5GHz WiFi bands (for under-canopy and over-canopy transmissions respectively), as detailed in §3. (2) It then leverages coarse-grained knowledge of device locations and flow demands to assign access points to sending devices and assign channels for each interface of the WiFi routers so as to maximize spatial reuse of network capacity, as detailed in §5. (3) It finally uses finer-grained link characteristics and flow demands to make finer-grained TE decisions (selecting routes that can best accommodate flow demands and adjusting flow sending rates), as also detailed in §5.

We evaluate FarmNetes (§7) using a small-scale testbed deployed on a cornfield and larger scale trace-driven simulations, showing how it achieves 3.5 × higher network throughput and 6 × better performance for real-time tasks (in terms of throughput normalized by flow demand) when compared with status-quo wireless mesh techniques used today (that select nearest AP, do not support policy-driven rate enforcement, and typically rely on shortest hop count based routing). We individually evaluate the impact of different components of FarmNetes' TE over this baseline. Our trace-driven simulations also compares FarmNetes's routing with other proposed wireless routing schemes (namely Roofnet's ETT [1, 15]), both with and without our policy-driven rate enforcement, revealing up to 2 × improvement in real-time flows' performance and 30% higher overall throughput. Our real-world testbed evaluation and microbenchmarks highlight how FarmNetes' explicit rate enforcement allows it to cleanly reason about wireless resource usage and enforce desired flow-sharing policies, despite the adhoc MAC layer link sharing with CSMA/CA (that neither achieves fair sharing of channel bandwidth nor of datarates).

FarmNetes' source code and collected datasets are publicly available at: https://github.com/ammartahir24/farmernetes.

2 Understanding Farm Workloads

We surveyed 12 agronomists spanning two institutions, who are conducting cutting-edge research in autonomous farming techniques. We used the survey responses, in combination with an extensive literature review [5–7, 17, 19, 31, 33, 44, 46, 48, 50, 53, 56], to characterize networked farm workload.

Workload Categories. We divide it into two categories:

(a) Real-time Tasks. These tasks include targeted bug-spraying [56], fruit picking [7, 31, 50], mechanical weeding [37, 58], and cover crop planting [5] by autonomous under-canopy robots. These autonomous plant manipulation tasks need to leverage edge servers (typically located in farmer's home/office) to run complex models for controlling the robots. This requires streaming sensor data (e.g. RGB images from multiple cameras) in real-time from under-canopy robots to edge servers, with the throughput requirements consequently ranging from 5-20Mbps per device. Human tele-operation of robots and drones [48] similarly requires an upstream throughput of 5-10Mbps. At scale, a typical farm spanning a few hundred acres may require tens of robots to operate simultaneously.

(d) Data collection. Emerging precision agriculture techniques additionally rely on sensors such as RGB, thermal, and multispectral cameras that generate large amounts of data (10s to 100s of MBs per acre, with a typical farm spanning a few hundred acres). In many cases, such data must be collected multiple times a day, to catch potential issues faster and take appropriate measures. There can be tens to hundreds of such sensors and devices streaming data to the edge servers in a large farm.

Survey Takeaways. Our survey participants are actively engaged in research spanning the two application categories described above. Our survey asked pointed questions about their networking needs. We summarize the takeaways below:

- All participants (N=12/12) feel that a farm network that can deliver high throughput will make it easier to deploy smart agriculture solutions and enable innovation in the space.
- The requirements of participants included network support for real-time tasks (N=4/12), e.g. for remotely controlling autonomous robots, and processing data collected by on-field sensors within a few minutes (N=1/12), a few hours (N=3/12), or a few days (N=3/12).
- Data collection applications use various sensors (RGB, thermal, hyperspectral, LIDAR, thermometer, pH, and spectroradiometers). RGB images were most common (N= 11/12), followed by hyperspectral (N=6/12), and LIDAR (N=4/12).
- Majority of our participants required processing the sensed data in edge servers (N=8/12), with a few indicating need for cloud offloads (N=3/12).

Key Challenges.

- *Incast to edge.* Multiple on-field devices simultaneously streaming data to the edge servers (located in farmer's home/office) can lead to bottlenecks near the network edge.
- *Under-canopy operation.* Many applications involve streaming data from under the canopy, where the transmitting device is completely surrounded by crops, leading to degraded network performance (as shown in §3).
- *Scale.* Even tens of simultaneously active devices can overwhelm the limited wireless capacity on a farm. As such techniques become more popular, a typical to large farm may require many tens to hundreds of simultaneously active devices.

3 Farm Network Design and Characteristics

The challenge of enabling high-bandwidth connectivity in large farms is well-recognized (even by government bodies [19, 52]). Due to the sparse population of farmlands, there is limited deployment of infrastructure-heavy solutions like cellular and fiber networks. Even where fiber endpoints are available, they can only serve as backhauls and must be extended to devices on the field through wireless connectivity. We focus on WiFi (one of the oldest and most popular wireless connectivity options for rural farms [9, 35, 39, 41]), discussing other wireless alternatives in §9. WiFi offers relatively high bandwidth (of the order of few tens of Mbps) but over short ranges (tens of meters). This is in contrast to, say, LoRaWAN [8] that can support a long range of up to 10 Km, but only a few Kbps of bandwidth that is not sufficient to support emerging farm workload described above. The short range of WiFi requires deploying a multi-hop mesh of WiFi routers to ensure coverage over the farm.

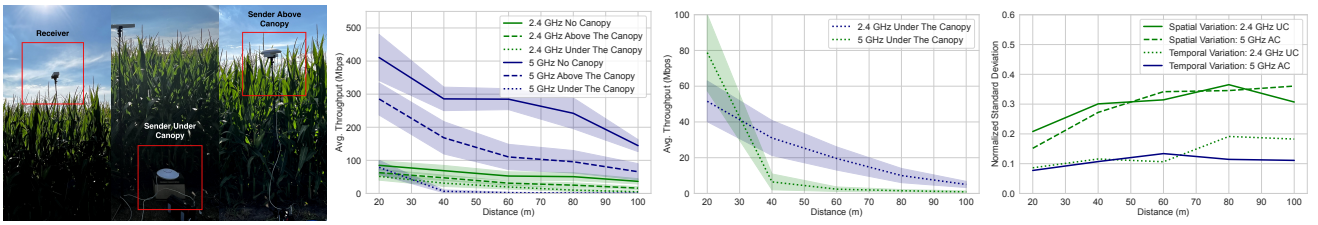
Despite WiFi's popularity, there are no systematic studies on how to best design such a WiFi network for farms, and the performance we can expect to achieve from such a network. We conduct extensive WiFi measurements on an 80-acre cornfield with 6 feet high crops to characterize WiFi performance (§3.1), and use these insights to guide our WiFi mesh network design (§3.2) on which our TE system is built (§5).

3.1 On-Field WiFi Measurements

Measurement Setup:.
(1) Sender: a Unifi LR-6 router that supports Wifi 6 over 5GHz and 2.4GHz bands [27], connected over Ethernet to a Linux device that sends a UDP flow using iperf.
(2) Receiver: a TP-Link RE505X WiFi 6 receiver, that receives data from the sender over WiFi, and is connected to a macOS device that runs the iperf server and logs the throughput.
(3) Channel configuration: We used maximum channel widths (160 MHz for 5GHz and 40 MHz for 2.4GHz).
(4) Transmission Height: We placed the receiver on a 2 m high pole (above the crop canopy level), as shown in Figure 1a. We placed the sender at two different heights:
(i) Under-canopy (<0.5m), mimicking sensors and robots streaming data from under the crop canopy.
(ii) Above-canopy (2m), mimicking relay routers mounted on poles above the canopy height.
(iii) We also conduct control experiments on a path where there is no crop canopy between the sender and the receiver, using a transmission height of 2m.
(5) Transmission Distance: We vary the distance between the sender and receiver from 20m to 100m across experiments.
(6) Experiment Duration: For each setting, we conducted 5 experiment runs. Each run transmitted data for 30 seconds and recorded the resulting throughput. We manually moved the sender by few centimeters between different runs for a given setting, to capture performance deviations arising from finer-grained spatial variations. We repeated these measurements every week from mid-June - August end.
Overall Results. Figure 1b shows the average throughput (aggregated across all the measurements we conducted over the growing seasons) achieved over the 2.4GHz band and 5GHz WiFi bands under different settings. The shaded areas represents the standard deviation over different experiment runs. We have the following key takeaways.
(i) Presence of crop canopy degrades throughput. Even when the sender and the receiver are in line of sight above the crop canopy, the WiFi throughput achieved on either bands is 1.2-2 $\times$ lower than when there is no crop canopy.
(ii) 5GHz achieves higher throughput above canopy due to wider 160MHz channel (vs the 40MHz wide 2.4GHz channel).
(iii) 2.4GHz is more robust under canopy. Figure 1c re-plots the data for the under-canopy setting, for a more direct comparison between the 2.4GHz and 5GHz bands. The throughput achieved by 5GHz, in spite of using a wider channel, drops to nearly zero beyond 40m. On the other hand, the 2.4GHz band can achieve 5-10Mbps even at a distance of 80 meters.
(iv) Low temoral variation. We notice very small temporal variation of throughput (10% standard deviation) on a stationary device over the 30s duration of each experiment run, owing to lack of interference from external sources in a farm (Figure 1d).
(v) Higher spatial variation. Changing the sender's position by a few centimeters resulted in up to 30% standard deviation in throughput (Figure 1d). Our evaluation in §7 models this.
(vi) Variations over growing season. The overall trends above remained consistent over weeks during the growing season, with occasional variations in absolute throughput values, owing to factors such as weather and crop height (result omitted for brevity).

3.2 Network Design Takeaways

These measurement results guide our network design.
1. Above-Canopy WiFi Mesh. Given the short-range of WiFi (upto 100m), and its better performance above the crop canopy (as compared to underneath the crops), our network consists of a multi-hop mesh of WiFi routers placed above the crop canopy (e.g. mounted on poles) in a grid topology at a distance of 80-100m from one another, as shown in Fig 2.
2. Tiered use of 2.4GHz and 5GHz. Our measurement results also indicate a clear division of responsibility between the 2.4GHz and the 5GHz bands: *(i) 2.4GHz under-canopy:* where the routers communicate with under-canopy sensors and robots over their 2.4GHz interface that acts as access point, and *(ii) 5GHz above-canopy:* where the routers use their 5GHz interface to route traffic across the mesh above the crop canopy in order to communicate with the edge servers (or



(a) Measurement Setup (b) Wifi throughput with varying distance (c) Comparison of 2.4 GHz and 5GHz UC (d) Temporal and spatial variations in throughput.

Figure 1: Our Wifi throughput measurement setup and summary of results (shaded area shows standard deviations across experiments run over 12 weeks)

with other on-field devices). Such a two-tiered architecture is needed because 5 GHz does not work under-canopy beyond 40m, restricting majority of under-canopy transmissions to 2.4GHz. Additionally due to the scarcity of channels in the 2.4 GHz band it is not prudent to use it above the canopy. We empirically observed that longer range of transmission above canopy for 2.4 GHz can lead to higher interference, resulting in reduced throughput for the first hop (under-canopy) links.

3. Alleviating edge bottlenecks. Bottlenecks emerge at the WiFi routers at the network edge (connected to edge servers in the farmer's home/office) due to the incast traffic pattern. However, it is often feasible to directly connect one or more of such edge routers to the servers (and to each-other) using wired (Ethernet) connectivity as shown in Fig 2, with the WiFi mesh then extending network coverage to the field. Increasing the number of such wired edge routers (from say, one to even three) can significantly improve overall WiFi throughput by allowing greater spatial reuse of WiFi capacity.

4. Need for judicious traffic management. Our measurements highlight the limited capacity of farm WiFi networks. The 100Mbps of overall throughput on an above-canopy 5GHz link effectively translates to only 30Mbps when we account for resources consumed by simultaneous listening and transmission at a forwarding hop, and interference within such multi-hop transmissions (even in the absence of other competing traffic). The under-canopy network capacity is even lower to begin with, and, with only three non-overlapping channels in the 2.4GHz band, the network can quickly get overwhelmed with just three to four neighboring under-canopy transmissions. Finally, unlike wired networks, throughput is fundamentally limited by the available spectrum bandwidth in wireless, and cannot be improved by adding more routers. Our traffic engineering system (described in §5) is designed to extract maximal performance for farm applications from such limited capacity network.

4 Computing Wireless Resource Usage

Reasoning about TE decisions requires computing the amount of wireless resources consumed by a flow sending

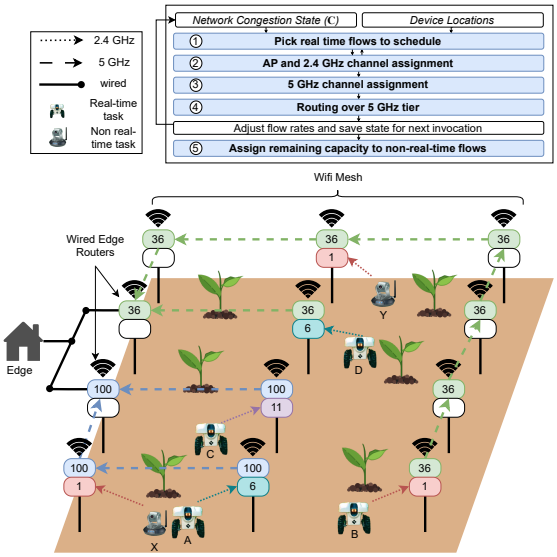


Figure 2: Overall workflow of FarmNetes.

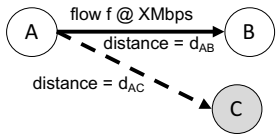
at a certain enforced datarate (both on its direct path and at interfering nodes). We first detail how we do that agnostic of the underlying MAC layer link sharing mechanisms. This is enabled by our TE system's explicit rate control which ensures that the sending rates do not exceed the network capacity on any device. Our resource usage formulation further errs on the side of over-estimation to reinforce that. The broad assumption we make (and validate in §7.1) is that if the incident demand at a node (including direct transmissions and interference) is less than its capacity, the underlying wireless link is able to meet those demands.

Resource Unit (RU). We use the abstraction of *resource units* to capture the normalized amount of capacity at each interface of a device (end-device or router). One resource unit corresponds to the standard bandwidth of 20 MHz on a 2.4GHz channel, and 160 MHz on a 5GHz channel.

Data-driven Model. We use a data-driven model to compute the amount of RUs consumed by a flow. As we observe in §3.1, for a given band (2.4GHz under-canopy or 5GHz

above-canopy) and at a given point in time, the maximum achievable throughput (in the absence of contention or interference from other flows) primarily depends on distance, given the farm setting is relatively static and homogeneous. Our measurement trace from §3.1 provides throughput values at discrete distances in both the 2.4GHz tier and the 5GHz tier. We then apply a logarithmic regression to predict throughput for arbitrary distances in each tier. In practice, such a trace can be periodically updated by profiling the throughput of above-canopy links between routers as part of daily bootstrapping process, and opportunistically measuring the under-canopy throughput based on current device locations. Such periodic updates can help account for the effects changing crop height over the growing season and weather conditions. We further discuss how FarmNetes can handle deviations in expected network behavior in §5.7.

RUs consumed due to direct transmission. Consider a flow f transmitted from A to B with datarate X Mbps over distance d_{AB}, as shown in the figure below. We use $f_{A\to B}$ to denote the one-hop transmission of flow f from A to B (a multi-hop flow would consist of multiple such transmissions). Let $throughput(d_{AB})$ denote the maximum datarate a transmission between A and B could have achieved if it was using the entire channel bandwidth (i.e. one resource unit) at A and at B (this value is available from the measurement traces mentioned above). A transmission at a lower datarate would consume proportionally smaller amount of resource units. Therefore, the amount of resource units (or fraction of channel bandwidth) that gets consumed at A and B by $f_{A\to B}$ can be denoted by:



$$r(f_{A\to B},A) = r(f_{A\to B},B) = \frac{X}{throughput(d_{AB})} \quad (1)$$

where $r(f_{A\to B},A)$ and $r(f_{A\to B},B)$ are resource units consumed by $f_{A\to B}$ at A and B respectively.

RUs consumed on interfering devices. We approximate the amount of RUs consumed by $f_{A\to B}$ due to interference on a neighboring device C at a distance of d_{AC} from A, as:

$$r(f_{A\to B},C) = r(f_{A\to B},A) \times \delta(f_{A\to B},C) \quad (2a)$$

$$= r(f_{A\to B},A) \times min(1, \frac{throughput(d_{AC})}{throughput(D)}) \quad (2b)$$

where D is the distance between two routers in our grid topology. Here, $r(f_{A\to B},A)$ indicates how often (or how much of) the channel is accessed by transmission $f_{A\to B}$, and $\delta(f_{A\to B},C)$ captures the impact of interference on the portion of channel accessed by $f_{A\to B}$. Our formulation captures the decrease in amount of interference with increasing distance between A and C. When $d_{AC} \le D$, we conservatively assume that the interference effect is so high that no part of the spectrum used by $f_{A\to B}$ is usable by C. In such a setting, $throughput(d_{AC}) \ge throughput(D)$, and $r(f_{A\to B},C) = r(f_{A\to B},A)$. As d_{AC} increases beyond D (indicating C is not in the immediate vicinity of A) the interference effect materializes as degradation in signal quality. This means that the portion of channel impacted by $f_{A\to B}$ is still usable by C, but not very effectively ($\delta(f_{A\to B},C) < 1$, captures this effect). We give detailed analysis based on the Shannon–Hartley theorem [47] in §A.

5 FarmNeTES: Farm Network TE System

We now describe FarmNetes, our centralized TE system for managing traffic on farm WiFi mesh network.

5.1 Overview

Inputs: FarmNetes takes as input: (i) Locations of on-field devices. (ii) Network requirements of real-time task that include (a) its duration (typically tens of minutes) (b) the deadline by when it should start, and (c) the *desired* upstream throughput for real-time streaming (often between 5-20 Mbps). [2] Requirements for non-real-time (data collection) task include the amount of data it needs to transmit and its deadline (typically few hours or days).

Goals: (i) Meet the scheduling constraints and desired datarates of real-time tasks (to the best possible extent). (ii) Maximize overall network throughput by filling up the *spare* network capacity with lower priority data collection tasks.

Overall Workflow: Figure 2 summarizes our TE workflow. FarmNetes invokes its TE engine periodically (every few minutes), taking the following steps at each invocation:

• *(1) Picking real-time flows to schedule:* We first pick urgent real-time tasks that must be scheduled in that invocation, i.e. with $slack \le 0$, where $slack = deadline - current\ time$. We then pick as many additional real-time tasks (in increasing order of their slack) as can be accommodated. Specifically, we check that the cumulative demand of the scheduled tasks does not exceed the estimated capacity of 5GHz tier to support the incast traffic to the edge. [3]

• *(2) Assigning APs and 2.4GHz channels:* We sort the selected real-time tasks by their deadlines and greedily assign them APs and channels in the 2.4GHz tier one flow at a time (as described in §5.2). Our algorithm favors APs that can fully meet the datarate demand of the given flow, picking the choice that minimizes overall resource consumption on 2.4GHz tier when there are multiple such options. If no AP can meet the flow's demand, it picks the one that fulfills highest proportion of the demand. This step works jointly with Step 1 above. While urgent real-time flows are always scheduled (even if

[2] This can be pre-estimated from the sensing modality (RGB vs LiDAR) and the data quality requirements of the algorithms running at the edge.

[3] This is a coarse-grained estimate of capacity, that scales with the number of wired edge routers, and can be computed accordingly.

their datarate demands cannot be fully met), a non-urgent real-time flow (with slack > 0) is de-selected if its demand cannot be fully met, and retried for scheduling at a later invocation, opening up room to select another flow.

• *(3) Assigning 5GHz channels:* Once the load on each AP has been determined, we allocate channels for the 5GHz interfaces, as described in §5.3. We must assign channels in 5GHz tier in such a way that any router can reach at least one wired edge router. In doing so, we must adhere to the following constraints: (i) only one channel per interface (with most commodity routers supporting only a single 5GHz interface, this implies only one 5GHz channel per router) and (ii) if one router forwards traffic to another over WiFi, their corresponding interfaces must operate on the same channel. This means that we must have a path from any router to a wired edge router, where all routers along that path (including the edge router) use the same 5GHz channel. Thus our 5GHz channel allocation algorithm effectively partitions the wireless network by the number of wired edge routers, such that each partition uses a different channel in the 5GHz tier. [4] We do this partitioning such that the workload (approximated from the scheduled real-time flows) is divided equally across partitions. Fig 2 shows a toy example where we have two partitions operating over channel 36 (in green) and 100 (in blue). Connecting more edge routers to over wire allows using more channels in the mesh, thereby increasing overall capacity by reducing interference levels. 5GHz band provides a choice of up to three 160MHz channels, that we can spatially re-use when there are more than three wired edge routers.

• *(4) Assigning 5GHz routes:* Once the 5GHz channels are assigned, we assign routes over the 5 GHz tier for each task from their selected AP to the corresponding wired edge router that is connected via the same channel partition. As detailed in §5.4, the route selection for a flow favors paths that consume fewest amount of network resources (after accounting for both hop count and interference effects), among those that have sufficient capacity to accommodate the flow's demand. If no path has sufficient capacity, we favor paths that can accommodate the largest proportion of the flow's demand, even if these are not the shortest paths by hop count (e.g. device B in Fig 2 takes a longer path).

• *(5) Adjusting flow rates and saving state:* The initial set of supported tasks was based on a crude estimate of overall network capacity. Once the channel configurations, and the flow routes are finalized, we make finer-grain adjustments to the flow sending rates as described in §5.5, ensuring that we do not exceed the capacity at any node. At this point, we save the network state to be used as input for the next invocation, ensuring we retain the AP, channel, and routing decisions for scheduled real-time flows to the best extent.

• *(6) Filling spare capacity with data collection tasks:* We finally use a water-filling algorithm to pick routes and rates for the non real-time data collection tasks with lax deadlines, only assigning them network resources spared by the real-time flows (§5.6). We promote such a task to real-time if it approaches its deadlines, but remains unfinished.

5.2 Selecting Access Point and 2.4GHz Channel

Given a flow f originating from an under-canopy sender S that has not yet been assigned an AP, we compute a cost associated with each router R, and select the router, R^*, that has the lowest cost as S's AP, as per the cost function defined below. If a router R has already been assigned a 2.4GHz channel (due to previously scheduled flows), we stick to that channel configuration when evaluating its cost as f's AP. If R has not yet been assigned a channel, we individually evaluate R's cost function under the three different channel configuration options supported by 2.4GHz WiFi, and pick the channel that results in the lowest cost (using that cost as the cost associated with R). If R is eventually selected as the AP for the flow, we assign this lowest cost channel to R, and account for it in the subsequent iterations of the algorithm for other flows.

With channel configured for router R as described above, we define the cost of selecting it as S's AP as follows:

$$cost(f, S \rightarrow R) = cost_{RU}(f, S \rightarrow R) + \lambda cost_{CT}(f, S \rightarrow R)$$

It consists of two cost components:

(i) Resource unit cost:. $cost_{RU}(f, S \rightarrow R)$, denotes the total amount of resource units consumed due to f's transmission from S to R. We define it as:

$$cost_{RU}(f, S \rightarrow R) = \sum_{R_j \in Neighbors(S \rightarrow R) \bigcup S} r(f_{S \rightarrow R}, R_j)$$

Here, $r(f_{S \rightarrow R}, R_j)$ are the resource units consumed at device R_j by f's transmission from S to R (computed as described in §4) – transmission to a more distant router R will consume more resources. $Neighbors(S \rightarrow R)$ denotes the set of devices (routers and end-devices) that are impacted by S's transmission to R, defined formally as all $R_j \in Devices$ with (a) the same channel as S and $\frac{r(f_{S \rightarrow R}, S)}{r(f_{S \rightarrow R}, R_j)} < \alpha$, where we set $\alpha = 10$ in our experiments. We set the channel at S to be the same as that of the AP R under consideration. [5]

(ii) Contention cost:. The second component, $cost_{CT}(f, S \rightarrow R)$, computes whether there is sufficient capacity to accommodate f (after accounting for all flows scheduled so far) if R

[4] Devices in two different wireless partitions can still talk to one another via the wired edge routers.

[5] We ignore devices that have not yet been assigned a channel (later iterations account for them, as and when flows are scheduled on them).

is chosen as its AP. We define it as:

$$cost_{CT}(f, S \rightarrow R) = \max_{R_j \in Neighbors(S) \bigcup S} (r(f_{S \rightarrow R}, R_j) + \mathbf{C}[R_j] - 1)^+$$

Where, $v^+ = \max(0, v)$. The term $\mathbf{C}[R]$ denoted the amount of RUs already consumed at router R due by previously scheduled flows (or transmissions), i.e. $\mathbf{C}[R] = \sum_{f' \in F'} r(f', R)$, where F' is the set of previously scheduled transmissions. This cost function returns 0 whenever there is sufficient capacity available at R and other neighbors of S to accommodate $f_{S \rightarrow R}$. Whenever any of these devices are congested it returns a value greater than 0, representing the excess demand that cannot be accommodated.

Combining the two components:. The final cost is a weighted sum of the resource unit cost and the contention cost, where λ, denoting the weight of contention cost, is set to a very large value. This ensures that we prioritize picking an AP, such that there is enough capacity to support flow f's demand fully (when $cost_{CT} = 0$). If no such router exists, we prioritize the one that can meet the highest proportion of f's demand (has lowest value of $cost_{CT}$) for urgent flows, or omit scheduling f if it is not urgent. If $cost_{CT}$ is the same for multiple routers, we prioritize picking a router that consumes the least amount of resources (i.e. has the least $cost_{RU}$).

5.3 5GHz Channel Assignment

As mentioned in §5.1, our 5GHz channel assignment algorithm effectively partitions the wireless mesh network by the number of wired edge routers n, trying to equally distribute load across each partition. One strategy is to create partitions with equal number of routers. However, the number of routers is not a good proxy for the workload. We therefore run the channel assignment algorithm after assigning APs to the real-time flows that we need to schedule. We then use the datarate that each router (AP) needs in order to support these flows, as a coarse-grained approximation of the workload.

Before discussing the details of our algorithm, we state two properties it must satisfy: (1) Each router must be in exactly one partition (since, with just one 5GHz interface, it can be assigned only one channel). (2) Any router R in a partition P_i must be able to reach the wired edge router $E[P_i]$ associated with its partition by at least one path p such that (2a) all routers in the path are also in P_i (since wireless transmissions must send and listen on the same channel), and (2b) all edges in the path are either of the four lowest cost neighbors (vertical or horizontal) on Manhattan grid (to ensure high transmission throughput as discussed in §5.4).

With n wired edge routers, we initialize our algorithm with a random set of n partitions, $P_1, ..., P_n$, that satisfy the two properties above. Let W_i denote the load at partition P_i. We then run an iterative algorithm, where in each iteration, we pick the partition P_i with maximum load W_i and try to move a router from P_i to another partition P_k, such that the move does not make $W_k > W_i$. We consider a router R_j eligible for transitioning from partition P_i to another partition only if it satisfies the following properties: (i) it must be at the border of P_i, with at least one Manhattan neighbor in a different partition, and (ii) removing R_j does not split P_i into two unreachable parts. When further transitions from the most loaded partition are no longer possible (either because no more of its routers are eligible for transitioning, or because its neighboring partitions have reached equal load), we move to transitioning from the second most loaded partition and so on. Our algorithm concludes when all partitions are of roughly the same size, or there are no more routers eligible for transitioning. Once the real-time workload between partitions is balanced, we attempt to balance the number of remaining routers (which helps when assigning spare capacity to data collection tasks).

When n is larger than the number of available 5GHz channels (up to 3×160 MHz), we assign channels such that each wired edge router has different channel than its two neighbors to minimize interference. In case we do not have at least 3×160 MHz channels (due to regional restrictions or DFS event), we can also pick different width channels for different partitions (e.g. 1×160 MHz and 2×80 MHz). In such cases, we scale fair load per partition based on channel sizes (e.g. 160 MHz channel carries $2 \times$ more traffic than an 80 MHz channel).

5.4 Above-Canopy Routing on 5GHz tier

Once we assign 5GHz channels, we can determine feasible routes and the set of nodes that would interfere with another. We compute shortest cost path for each flow (greedily considering one flow at a time, sorted by their deadlines). The cost of a path for a flow is defined using a logic similar to our AP selection in §5.2. Specifically, when considering path P (composed of an ordered list of router hops) for flow f, we compute its resource unit cost as $pathcost_{RU}(f, P) = \sum_{R_j \in P} cost_{RU}(f, R_{j-1} \rightarrow R_j)$, i.e. the total amount of resource units consumed on the routers along path P, and their neighbors, due to f's transmission along P. We similarly compute the path's contention cost as $pathcost_{CT}(f, P) = \max_{R_j \in P}(cost_{CT}(f, R_{j-1} \rightarrow R_j))$, measuring the degree of overload on the most contended router along path P. Here, $cost_{RU}(.)$ and $cost_{CT}(.)$ are defined as in §5.2, but now considering the resources in the 5GHz band. The final cost of path P is weighted sum of the two components $(pathcost_{RU}(f, P) + \lambda pathcost_{CT}(f, P))$.

We cannot trivially use Djikstra's algorithm [16] to find the lowest cost path due to the mixed nature of our cost function, where the resource unit cost is added over the hops, but the contention cost is taken as max across hops. We modify Djikstra's algorithm to handle this (we skip providing these

modification details for brevity). Our algorithm restricts the next hop from a router to one of its eight neighbors in the Manhattan grid (as throughput for direct transmissions drops significantly beyond this distance).

We summarize a few interesting observations about FarmNetes' route selection: (i) The route selection favors Manhattan paths, i.e. with vertical or horizontal neighbors as next hop (and not diagonal ones). Even though this may increase the number of hops, it ends up consuming fewer resource units due to shorter transmission distance. (ii) Paths of same length (even along Manhattan paths) may still consume different amount of resource units due to different levels of interference. (iii) A longer path may be more favorable if it avoids consuming resources on congested routers.

5.5 Adjusting Flow Rates

While our 2.4GHz AP and 5GHz route selection logic strives to avoid overloading devices, it may not always be possible to do so (especially if the demands of urgent real-time tasks exceeds available capacity). We run the following rate allocation algorithm to divide the wireless capacity fairly between the scheduled real-time flows. We pick the most congested router interface, i.e. $\max_{R \in Routers} \mathbf{C}[R]$, and divide its capacity (1 RU) equally between all flows passing through that router interface. We then pick the next most congested router interface and repeat the process until no router interface has $\mathbf{C}[R] > 1$. Each flow is assigned the sending rate that we get at the conclusion of this algorithm.

5.6 Scheduling Non-realtime Flows

Finally, we assign any spare capacity to non-real-time flows. For each non-real-time flow, we set their starting rate to 1 Mbps and try to assign 2.4GHz AP and a route over 5 GHz tier to them one flow at a time, as described in §5.2 and §5.4. If $cost_{CT}$ of both these operations is zero for a given flow, it means there is capacity for at least 1 Mbps of this flow, and we commit this assignment. We maintain a list of such successes. Once we have iterated through all non-realtime flows in this manner, we then iterate through the list of successfully scheduled flows attempting to push an additional 1 Mbps for each flow along the selected path. If at any point, the $cost_{CT}$ becomes greater than 0, that flow is removed from the list and is assigned its current AP, path, and rate. We continue this process until the list becomes empty. By tracking $cost_{CT}$ (that accounts for resource usage of all flows scheduled so far, both due to direct transmission and interference), we ensure that these non-realtime flows only consume spare resources.

5.7 Handling Deviations in Expected Behavior

Our TE decisions rely on data-driven estimates of on-farm throughput at varying distances (§4). The actual throughput can deviate from these estimates due to various factors. In the case of failures that may result in severe degradation in throughput, e.g. damaged equipment or physical obstructions, the TE system can update its state to route around such devices. For minor variations from throughput profiles (as noted in §3.1 due to spatial changes), we add robustness to our TE system by leaving some headroom when assessing the capacity of a router (10% in our experiments). Another form of deviation can arise from device mobility – since under-canopy robots move at low speed (a few meters per minute), the effects of mobility can be handled by subsequent invocations of the TE engine. We simulate such variations in our experiments (in §7.3) and show that FarmNetes is able to handle them robustly.

6 Implementation

Central Controller. FarmNetes' TE controller runs on a centralized server at the edge and remotely configures on-field routers and devices. Exchange of control information between the server and the devices has low bandwidth requirements that can use an out-of-band LoRa network [8].

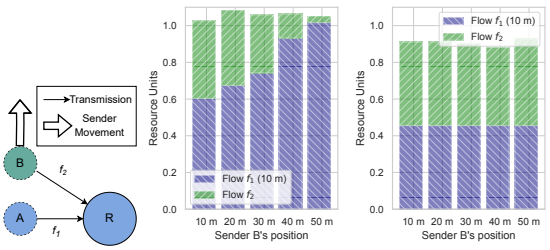
Configuring Routers. Our prototype uses ASUS RT-AC86U WiFi routers updated with Asuswrt Merlin firmware which allows updating IP forwarding rules and changing channels remotely over SSH. In particular, a Raspberry Pi attached to the router receives updates from the central controller and configures the routers over a pre-connected SSH session. We operate routers in ad-hoc mode (Wireless Distribution System) mode and use ip-route to install routing rules for flows based on MPLS tags as set by the central controller.

Configuring End-devices. Each end-device (e.g. Raspberry Pis attached to sensors and robots), runs a FarmNetes daemon that listens for control information from the central controller and accordingly schedules (i.e. starts or pauses) tasks using application-provided interfaces. It uses Linux TC to set the sending rates from the device to the value specified by the controller. The daemon also periodically sends the device location (collected using RTK [49]) to the controller.

7 Evaluation

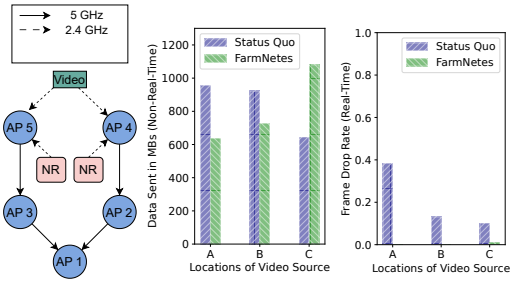
Our evaluation is divided into three phases:

- §7.1: Real-world experiments to validate the effectiveness of centralized rate enforcement in controlling how network resources are shared (irrespective of MAC layer behavior).
- §7.2: Small-scale prototype deployment in a cornfield, showing FarmNetes' performance benefits.



(a) Setup (b) RUs from both flows add up to 1 (c) Fair division of RU between flows

Figure 3: Validating need for explicit rate enforcement.



(a) Setup (b) Data Downloaded (c) Frame Drop-rate

Figure 4: FarmNetes results in improved network management in real-world deployment.

• §7.3: Trace-driven simulations of large scale deployment scenarios, comparing FarmNetes' performance with status quo wireless mesh baselines and prior work [1, 15].

7.1 Real-World Microenchmarks

Validating Resource Unit Abstraction: We start with an experiment to validate our resource unit (RU) abstraction – that a device splits its channel bandwidth (that we abstract as 1 RU) between competing flows. We set up two devices (Raspberry Pi's) *A* and *B* that simultaneously transmit data to an ASUS router *R* (Figure 3a) using UDP(?), with a configured sending rate larger than the link can support. We keep the distance between *A* and *R* fixed, and increase the distance between *B* and *R* across different experiment runs. We measure the datarate achieved by each flow, and compute the corresponding amount of RUs consumed by the flows (using the formulations in §4). Figure 3b shows how the RUs consumed by each flow indeed add up to one – the entire channel bandwidth at *R* is split between the two flows.

Unfairness in the absence of explicit rate enforcement: We further observe the datarate, and the resource consumption, that is naturally achieved by the two flows in the above experiment (where their cumulative sending rate is higher than the capacity at *R*) with CSMA/CA based MAC protocol running on the sending devices. As can be seen in Figure 3b, we find that the flow shares neither correspond to fair use of channel capacity nor to fairness in terms of datarates. The transmission from *B* consumes increasingly smaller amount of RUs as the distance between *B* and *R* increases. This highlights the difficulty of reasoning about MAC layer link sharing mechanisms.

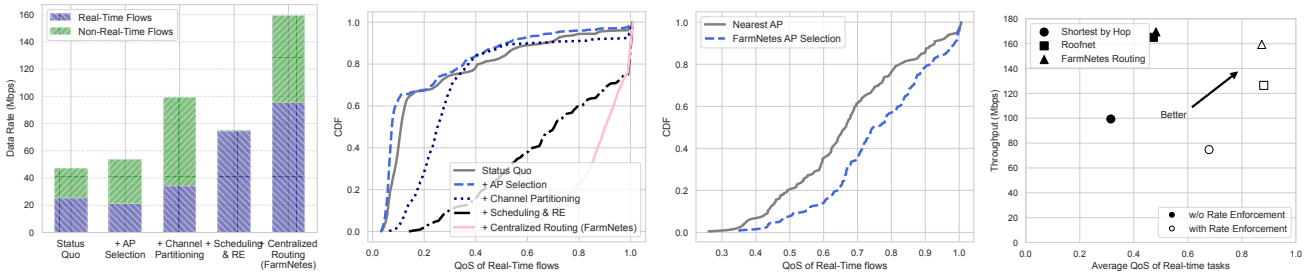
Effectiveness of explicit rate enforcement: We next restrict the sending rates of flows from *A* and *B*, such that the RUs at router *R* get equally divided between them, leaving a 10% headroom. This required us to apply our formulation from §4 (seeded with throughputs measured in §3.1) to compute how much resources a flow sending at a specified rate from the specified distance would consume at *R*. Figure 3c shows that the RUs consumed by each flow (back-computed from the observed datarate each flow achieves). We find that the resource usage indeed matches what we compute and *indirectly* enforce via setting the sending rates. It further highlights how our TE decisions can control the flow shares irrespective of the underlying MAC layer behavior.

7.2 Small-Scale Real-World Testbed

Setup: We mounted five ASUS RT-AC86U routers on poles and placed them at different locations in a cornfield to create a mesh network as shown in Figure 4a. We connected one router to an edge server (a MacBook laptop in our prototype) using Ethernet. We emulated non-real-time (NR) flows on two under-canopy devices (Raspberry Pi's running Linux) that sent data to the edge server using iperf, mimicking large sensor data transfer. These devices connected to the specified routers over 2.4 GHz. Another device (another MacBook in our prototype) streamed an RGB video over UDP at the rate of 15 Mbps to the server (mimicking a real-time remote operation task). We measured the frame drop rate for the video at the edge server for each run. The positions of non-real-time flow sources were fixed across experiments, whereas the video source's location was changed across different experiment runs. We kept these devices stationary for controlled experiments, mimicking scenarios where a remotely-operated robot performs berry-picking or de-weeding.

While in this setup, we used the in-band WiFi connection to configure the devices from the edge server. We separately evaluate the feasibility and overhead of configuring on-field devices over LoRa. It takes 1.3 seconds to send a control message over LoRa, whereas a channel switch can cause an interruption of up to 5.2 seconds at the devices connected to the AP that switches the channel. As we evaluate later, FarmNetes is able to ensure such switches are rare for real-time flows.

Results: Figure 4 shows the performance at different locations of the video flow, comparing FarmNetes with the status quo WiFi mesh baseline that picks the nearest AP at the 2.4GHz



(a) Overall Throughput **(b) QoS of Real-Time Flows** **(c) Effectiveness of FarmNetes' AP selection algorithm** **(d) Comparison with different routing baselines**

Figure 5: Trace-driven simulation results show improved throughput and management of the farm network with FarmNetes.

over a randomly selected channel, and routes along the shortest Manhattan path in the 5GHz tier. Given the relatively small number of routers in our setup, FarmNetes picked the same (optimal) paths for each flow as the status quo. However, FarmNetes made smarter decisions about channel allocation, AP selection, and especially rate assignment (prioritizing video flows, and rate-limiting non-real-time flows to use only the spare capacity). We observe how, with the status quo, video flow suffers from high drop rates as non-real-time flows hogs all the bandwidth. FarmNetes ensures almost zero drop rate (<0.01%) and still ensures high enough throughput for non-real-time flows. In some cases (e.g. when the video source was at location *C*), the overall throughput was better with FarmNetes because of careful channel allocation.

7.3 Trace-Driven Simulations

Our prototype deployment could only test certain aspects of FarmNetes at a small scale. We augment our evaluation with larger scale trace-driven simulations.

Simulator: We build a simulator to simulate FarmNetes' two-tiered mesh network. It can be configured with an input workload, comprising of source devices at specified locations, trajectories for mobile devices (robots) and the corresponding flow demands for both real-time and data collection tasks. It implements FarmNetes' TE techniques, along with other baselines that we compare against. We simulate ideal spectrum sharing based on max-min fairness to split resource units at a device among contending flows (although, in practice, the MAC layer mechanisms may achieve arbitrary flow shares as shown in Figure 3b, which would further degrade the baseline's performance). We use our measurement traces and the formulations described in §4 to compute flow throughput (including the effects of interference). We model spatial variations in throughput by scaling the computed throughput with a scaling factor, sampled from a normal distribution observed in our measurements in §3.1.

Workload: We simulate a square-shaped 450 acre farm that has 225 routers placed in a 15×15 grid where each router is 90 meters apart. Five routers in the first-row are wired edge gateway routers, directly connected to the edge server that is the destination for all flows discussed ahead. We simulate a duration of about 2 hours. Based on §2, our input farm workload over the simulated duration contains a total of: (i) 40 non-real-time flows uniformly spread across the farm with data to send varying between 50 MBs and 500 MBs and a deadline longer than 2 hours (ii) About 200 real-time tasks, each with demands between 5 to 15 Mbps, duration between 5 and 20 minutes. The deadline is such that they can be scheduled 0-10 minutes after they have been requested. Most of these tasks are uniformly distributed across the farm. 25% of the tasks are colocated i.e. the senders are in comparatively close proximity. These tasks simulate scenarios when the farm operator may use multiple robots in the vicinity to complete a job quickly. About 30% of the tasks originate from mobile devices, where a mobile device travels a distance of 5-10 meters per minute along the crop row over the duration of the task. At any time, at most 15 real-time flows are active. We invoke our TE engine once every minute.

Baselines: We start off with a status quo baseline which captures how WiFi meshes typically operate. We do not employ any scheduling or rate enforcement as the network operator typically does not control when a flow should be scheduled and how much data it puts out. All flows, real-time or non-real-time, are scheduled as soon as they arrive and no flow is explicitly rate-limited – the sending rates correspond to the minimum of flow demands and the bottleneck network capacity, computed as per the max-min fair link sharing policy in our simulator. For each flow, we use the standard policy of selecting the nearest router by distance (that has the highest signal strength) as the AP. Similarly, we select the shortest path by hop count when routing over the 5 GHz tier where for each hop, we only consider the 8 immediate neighbors (picking a random choice among multiple equal cost paths). In the 2.4 GHz tier, channels are assigned randomly, whereas

in the 5 GHz tier, all routers operate over the same 160 MHz wide channel. This baseline is labeled as 'Status Quo' in our results in Fig. 5. We also implement a state-of-the-art wireless mesh routing (namely, Roofnet's expected transmission time based routing [1, 15]) as detailed later.

Results: Fig.5a shows the overall network throughput (computed as the total amount of data transmitted to the wired edge divided by the simulation duration), split between real-time and non-real-time flows. Fig. 5b shows the CDF of quality of service (QoS) of real-time flows calculated as throughput achieved by a real-time task normalized by its demand.

• The status quo baseline results in poor overall throughput, as it picks suboptimal paths and APs. Moreover, the QoS of real-time flows is also quite poor (with 80% flows achieving less than half their desired throughput) due to the lack of any rate enforcement for non-real-time flows. We next make incremental improvements over this baseline, adding TE elements of FarmNetes one at a time.

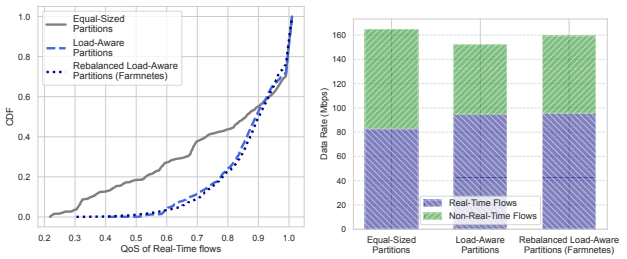
• We first add FarmNetes' AP assignment and 2.4 GHz channel allocation algorithm, as described in §5.2 (ordering flows by their slack time when assigning APs and channel). Doing so improves overall throughput only slightly. The QoS of real-time tasks also remains similar to the status quo, due to lack of explicit rate enforcement, and also because more generally the bottleneck occurs at the 5GHz tier near the edge routers. However, when we isolate scenarios where there are more end-devices in close vicinity of one another, we observe the QoS improvement due to FarmNetes' load-aware AP selection algorithm over the status quo's nearest AP selection (as shown in Figure 5c).

• Next, we add our 5 GHz channel assignment algorithm which partitions the mesh network to operate on different channels, as described in §5.3. This reduces interference in 5GHz tier and leads to significant improvement in overall throughput.

• We next add FarmNetes' scheduling and rate enforcement logic as described in §5.5 and §5.6. This ensures that we prioritize meeting the demands of real-time flows before allocating any resources to non-real-time flows, thereby leading to a huge improvement in QoS of real-time tasks (more than 80% real-time flows meet their desired demands). However the overall throughput reduces because the load-unaware shortest-hop routing leaves limited spare capacity for non-real-time flows.

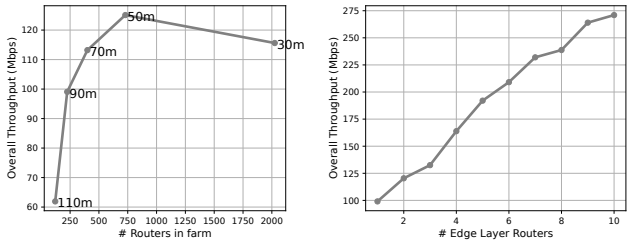
• Finally, we include our load-aware routing algorithm as described in §5.4 to complete the design of FarmNetes. We not only see a significant improvement in QoS of real-time tasks but also see significant gains in overall throughput.

• **Comparison with Roofnet:** In figure 5d, we compare our routing mechanism and the usefulness of rate enforcement with another state-of-the-art wireless routing baseline (Roofnet [1]). Roofnet measures the estimated transmission



(a) QoS of Real-Time Flows **(b) Overall Throughput**

Figure 6: FarmNetes's channel partitioning algorithm ensures high QoS without compromising throughput.



(a) # Mesh Routers **(b) # Wired Edge Routers**

Figure 7: Experiments showing how end-end throughput changes as we increase the number of edge layer routers vs overall routers in WiFi mesh.

time (ETT) for each link in the network using probe packets, and picks the path with the lowest overall ETT. ETT captures the effects of link quality, interference, and (indirectly) congestion. We directly calculate the ETT in our simulations using our RU abstraction and load on the given router (as observed in the previous epoch). We compare FarmNetes routing with Roofnet and shortest-hop-count (as used in the status quo baseline earlier) with and without explicit rate enforcement, while implementing other elements of FarmNetes (load-aware AP and 2.4GHz selection, and 5GHz channel partitioning) for all schemes.

We see that without explicit rate enforcement, none of the routing algorithms can ensure good QoE for real-time tasks (although Roofnet and FarmNetes routing fare better than shortest-hop-count). When we add support for policy-driven rate enforcement, the real-time QoS improves for all schemes, but with some degradation in overall throughput caused by rate-limiting of non real-time flows. FarmNetes achieves the best outcome in this case (higher overall throughput for better/similar QoS than others) because its centralized routing explicitly minimizes congestion, allowing for more spare capacity in the network to accommodate more flows and higher rates. Roofnet's reactive routing is not able to minimize bottlenecks to the same extent.

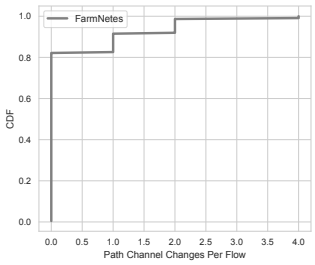


Figure 8: CDF of channel switches per flow.

• **Comparison with other channel partitioning strategies:** We now compare FarmNetes's channel partitioning algorithm with other simpler strategies (Figure 6). If we simply create equal sized partitions, we see a clear improvement in overall throughput. However, we can improve real-time tasks' performance by creating load-aware partitions. This in turn slightly reduces the overall data rate, as partitions can be of quite uneven size and sometimes end up missing out on enough non-real-time workload. By rebalancing load-aware channel partitions, we are able to maintain high QoS, while improving overall datarate.

• **Scaling throughput:** In figure 7 we show the effect of topology on FarmNetes' end-to-end throughput. We show how simply increasing the number of routers in the network does not improve the end-to-end throughput. In figure 7a, we show how using more routers and placing them closer together not only has diminishing returns in terms of overall throughput, but throughput starts dropping off after a point. This is because we are inherently bottlenecked by the available spectrum in a location, where each hop's transmission for a flow consumes a part of this spectrum. The only way to scale the throughput is by using more edge-layer routers. We show this in Figure 7b, where throughput linearly increases as we add more edge layer routers.

• **Number of channel switches:** In figure 8 we show how, over different invocations, FarmNetes' ensures that active real-time flows see minimal channel switches, either on the first hop AP or on the 5GHz path. We see zero channel switches for most flows, whereas > 2 switches for less than 3% flows.

8 Related Work

Farm Networking: Recent work has explored different network connectivity options for farm workloads, e.g.FarmBeats [53] proposes the use of TVWS base-stations talking to on-field devices over WiFi, TVWS-Whisper [13] enables LoRa-like connectivity for IoT devices over TVWS. Our core contribution – centralized TE for a multihop WiFi mesh in farms is orthogonal and complementary to these work. Some of the prior work measure ZigBee [55] and WiFi 2.4GHz performance [9] in a farm at varying transmission height above the crop canopy. Driven by requirements of emerging farming techniques, we conduct the first *under-canopy* measurement of WiFi throughput, which guides our design.

Wireless Mesh: Proposed strategies for wireless mesh networks have typically used decentralized routing protocols [1, 12, 14, 15, 24, 30] and reactive channel assignment algorithms [45]. There have been a few proposals for centralized routing in WiFi mesh with the objective of energy conservation [2] and channel assignment for multi-band routers [43]. None of these work explicitly control the input traffic (i.e. sending rates) – a key feature of FarmNetes that allows it to control flow shares and enforce the desired prioritization policy.

There is ongoing effort towards enabling wireless time-sensitive networking (TSN) [10, 11], that involves using a centralized controller and time synchronization between transmitting devices for frame-by-frame scheduling. These link layer efforts, yet to be realized in practice, are orthogonal and complementary to FarmNetes that directly controls flow shares agnostic of the underlying link sharing mechanisms.

Traffic Engineering: Centralized TE is popularly used in datacenter WANs [25, 26, 29]. TE can be formulated as a multi-commodity flow problem (MCFP) [40]. The decision version of this problem i.e. finding the feasibility of scheduling a set of flows on a network is NP-Complete [18]. However, linear programs exist for a version of MCFP when flows are allowed to be split across multiple paths. Even then such LPs are quite slow and scale poorly [36]. Previous work has looked at faster solutions that trade-off some accuracy for better scalability [32, 36]. While such solutions are appropriate for WAN TE where flows are aggregates of many individual flows and thus splittable, they cannot be used in our context where splitting a real-time flow across multiple paths can result in severe degradation for application performance. SWAN [25] uses demand pinning to route more critical flows along their shortest paths and then solves MCFP on the residual network for larger and less critical flows. We also use a similar approach where we first schedule and route real-time flows, and then use a water-filling algorithm to assign the left-over capacity to non real-time flows.

9 Conclusion and Future Work

In this work, we used networked farm applications as a compelling case-study to design FarmNetes, a centralized TE system for farm WiFi mesh. FarmNetes works with commodity on-field devices, and controls network shares across flows as per the desired policy by selecting appropriate routes and sending rates, agnostic of the underlying MAC layer mechanisms. Several interesting directions remain open for future research. We discuss a few below:

Power availability. We assume on-field routers are solar powered with sufficient battery capacity. Future work can extend FarmNetes to incorporate power efficiency as a goal.

Connectivity beyond WiFi. Other options for high-bandwidth outdoor connectivity in rural farms include TVWS [4] and CBRS [20]. WiFi offers several advantages in comparison. WiFi is easier to set-up and deploy, is globally available, has more spectrum available and is already integrated into most sensors, robots, and cameras. The relatively limited range of WiFi also has a benefit – greater spatial reuse of the available spectrum. Nonetheless, we believe our wireless TE design can also be extended to alternatives such as TVWS and CBRS.

Centralized TE in other wireless contexts. Our TE system can potentially apply to other contexts where wireless mesh networks are used to support predictable and schedulable workloads (e.g. factories, smart cities, mining, etc). It would require profiling and predicting network performance, which might be more challenging in a general (more heterogeneous setting) than in a farm, but yet feasible.

Deployment challenges. Deploying a WiFi mesh and smart devices in farms can be challenging and costly. While FarmNetes works with cheap off the shelf WiFi routers, we acknowledge that deploying and maintaining such network at scale can be cumbersome. Despite these challenges, we are already seeing such deployments in practice (e.g. [41, 51]).

10 Acknowledgements

We would like to thank our shepherd, Ratnadeep Bhattacharya, and SEC'25 reviewers for their helpful feedback. We would also like to thank Naveen Uppalapati, Arun Narenthiran, Matthew Carrol, Asheesh Singh, Soumik Sarkar, Girish Chowdhary, and others at the COALESCE group for their assistance and feedback on this work. This work was supported in parts by USDA NIFA (award number 2021-67021-34418), USDA NIFA Farm of the Future award, and NSF (award numbers 2217144 and 2339758).

References

[1] Daniel Aguayo, John Bicket, Sanjit Biswas, and DD Couto. 2004. MIT Roofnet: Construction of a Community Wireless Network. *Technical Report. MIT Computer Science and Artificial Intelligence* (2004).

[2] Ahmed Amokrane, Rami Langar, Raouf Boutabayz, and Guy Pujolle. 2013. Online flow-based Energy Efficient Management in Wireless Mesh Networks. In *2013 IEEE Global Communications Conference (GLOBECOM)*. doi:10.1109/GLOCOM.2013.6831092

[3] ANSI. 2023. IEEE 802.11-2020: Collision Avoidance in Wireless Networks. https://blog.ansi.org/ieee-802-11-collision-avoidance-wireless-networks/.

[4] Paramvir Bahl, Ranveer Chandra, Thomas Moscibroda, Rohan Murty, and Matt Welsh. 2009. White Space Networking with Wi-Fi like Connectivity. *SIGCOMM Comput. Commun. Rev.* (2009). doi:10.1145/1594977.1592573

[5] Edward Barnes, Gaylon Morgan, Kater Hake, Jon Devine, Ryan Kurtz, Gregory Ibendahl, Ajay Sharda, Glen Rains, John Snider, Joe Mari Maja, et al. 2021. Opportunities for robotic systems and automation in cotton production. *AgriEngineering* 3, 2 (2021).

[6] John Billingsley, Arto Visala, and Mark Dunn. 2008. Robotics in agriculture and forestry. (2008).

[7] Robert Bogue. 2020. Fruit Picking Robots: Has Their Time Come? *Industrial Robot: the international journal of robotics research and application* (2020).

[8] Martin Bor, John Edward Vidler, and Utz Roedig. 2016. LoRa For the Internet of Things. (2016).

[9] James Brinkhoff and John Hornbuckle. 2017. Characterization of WiFi signal Range for Agricultural WSNs. In *2017 23rd Asia-Pacific Conference on Communications (APCC)*. IEEE.

[10] Dave Cavalcanti, Stephen Bush, Marc Illouz, Genio Kronauer, Alon Regev, and Ganesh Venkatesan. 2020. Wireless TSN–Definitions, Use Cases & Standards Roadmap. *Avnu Alliance* (2020), 1–16.

[11] Dave Cavalcanti, Javier Perez-Ramirez, Mohammad Mamunur Rashid, Juan Fang, Mikhail Galeev, and Kevin B. Stanton. 2019. Extending Accurate Time Distribution and Timeliness Capabilities Over the Air to Enable Future Wireless Industrial Automation Systems. *Proc. IEEE* (2019).

[12] Ian D Chakeres and Elizabeth M Belding-Royer. 2004. AODV Routing Protocol Implementation Design. In *24th International Conference on Distributed Computing Systems Workshops, 2004. Proceedings.* IEEE, 698–703.

[13] Tusher Chakraborty, Heping Shi, Zerina Kapetanovic, Bodhi Priyantha, Deepak Vasisht, Binh Vu, Parag Pandit, Prasad Pillai, Yaswant Chabria, Andrew Nelson, et al. 2022. Whisper:IoT in the TV White Space Spectrum. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*. 401–418.

[14] Thomas Clausen and Philippe Jacquet. 2003. *Optimized Link State Routing Protocol (OLSR)*. Technical Report.

[15] Douglas S. J. De Couto, Daniel Aguayo, John Bicket, and Robert Morris. 2003. A High-Throughput Path Metric for Multi-Hop Wireless Routing. In *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking* (San Diego, CA, USA) *(MobiCom '03)*. Association for Computing Machinery, New York, NY, USA. doi:10.1145/938985.939000

[16] Edsger W Dijkstra. 2022. A note on two problems in connexion with graphs. In *Edsger Wybe Dijkstra: his life, work, and legacy*. 287–290.

[17] The Economist. 2016. The Future of Agriculture. https://www.economist.com/technology-quarterly/2016-06-09/factory-fresh.

[18] Shimon Even, Alon Itai, and Adi Shamir. 1975. On the Complexity of Time Table and Multi-commodity Flow Problems. In *16th annual symposium on foundations of computer science (sfcs 1975)*. IEEE.

[19] FCC. 2020. Examining Current and Future Connectivity Demand for Precision Agriculture. https://www.fcc.gov/sites/default/files/precision-ag-connectivity-demand-wg-07222020.pdf.

[20] FCC. 2023. 3.5 GHz Band Overview. https://www.fcc.gov/wireless/bureau-divisions/mobility-division/35-ghz-band/35-ghz-band-overview.

[21] Michele Garetto, Jingpu Shi, and Edward W Knightly. 2005. Modeling media access in embedded two-flow topologies of multi-hop wireless networks. In *Proceedings of the 11th annual international conference on Mobile computing and networking*.

[22] Piyush Gupta and Panganmala R Kumar. 2000. The Capacity of Wireless Networks. *IEEE Transactions on information theory* 46, 2 (2000).

[23] Daniel Halperin, Wenjun Hu, Anmol Sheth, and David Wetherall. 2010. Predictable 802.11 packet delivery from wireless channel measurements. In *Proceedings of the ACM SIGCOMM 2010 Conference* (New Delhi, India) *(SIGCOMM '10)*. Association for Computing Machinery, New York, NY,

USA, 159–170. doi:10.1145/1851182.1851203

[24] Guoyou He. 2002. Destination-sequenced Distance Vector (DSDV) Protocol. *Networking Laboratory, Helsinki University of Technology* 135 (2002).

[25] Chi-Yao Hong, Srikanth Kandula, Ratul Mahajan, Ming Zhang, Vijay Gill, Mohan Nanduri, and Roger Wattenhofer. 2013. Achieving High Utilization with Software-driven WAN. In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*.

[26] Chi-Yao Hong, Subhasree Mandal, Mohammad Al-Fares, Min Zhu, Richard Alimi, Chandan Bhagat, Sourabh Jain, Jay Kaimal, Shiyu Liang, Kirill Mendelev, et al. 2018. B4 and after: managing hierarchy, partitioning, and asymmetry for availability and scale in google's software-defined WAN. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*. 74–87.

[27] Ubiquiti Inc. 2025. Access point U6 long-range. https://store.ui.com/products/u6-lr-us. Accessed: 2025-10-23.

[28] Kamal Jain, Jitendra Padhye, Venkata N Padmanabhan, and Lili Qiu. 2003. Impact of Interference on Multi-hop Wireless Network Performance. In *Proceedings of the 9th annual international conference on Mobile computing and networking*.

[29] Sushant Jain, Alok Kumar, Subhasree Mandal, Joon Ong, Leon Poutievski, Arjun Singh, Subbaiah Venkata, Jim Wanderer, Junlan Zhou, Min Zhu, et al. 2013. B4: Experience with a Globally-deployed Software Defined WAN. *ACM SIGCOMM Computer Communication Review* 43, 4 (2013), 3–14.

[30] David B Johnson, David A Maltz, Josh Broch, et al. 2001. DSR: The dynamic source routing protocol for multi-hop wireless ad hoc networks. *Ad hoc networking* 5, 1 (2001), 139–172.

[31] Shivani Kamtikar, Samhita Marri, Benjamin Thomas Walt, Naveen Kumar Uppalapati, Girish Krishnan, and Girish Chowdhary. 2021. Towards Autonomous Berry Harvesting using Visual Servoing of Soft Continuum Arm. In *AI for Agriculture and Food Systems*. https://openreview.net/forum?id=nmFQlTk6WpV

[32] George Karakostas. 2008. Faster approximation schemes for fractional multicommodity flow problems. *ACM Transactions on Algorithms (TALG)* 4, 1 (2008), 1–17.

[33] Anthony King. 2017. Technology: The Future of Agriculture. *Nature* 544, 7651 (2017).

[34] Alok Kumar, Sushant Jain, Uday Naik, Anand Raghuraman, Nikhil Kasinadhuni, Enrique Cauich Zermeno, C Stephen Gunn, Jing Ai, Björn Carlin, Mihai Amarandei-Stavila, et al. 2015. BwE: Flexible, hierarchical bandwidth allocation for WAN distributed computing. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*. 1–14.

[35] larsklint.com. 2020. Updating Farm Wi-Fi with Ubiquiti Unify. https://larsklint.com/updating-farm-wi-fi-with-ubiquiti-unify/.

[36] Tom Leighton, Clifford Stein, Fillia Makedon, Éva Tardos, Serge Plotkin, and Spyros Tragoudas. 1991. Fast approximation algorithms for multicommodity flow problems. In *Proceedings of the twenty-third annual ACM symposium on Theory of Computing*. 101–111.

[37] Yong Li, Zhiqiang Guo, Feng Shuang, Man Zhang, and Xiuhua Li. 2022. Key technologies of machine vision for weeding robots: A review and benchmark. *Computers and Electronics in Agriculture* 196 (2022).

[38] Yi Li, Lili Qiu, Yin Zhang, Ratul Mahajan, and Eric Rozner. 2008. Predictable performance optimization for wireless networks. In *Proceedings of the ACM SIGCOMM 2008 Conference on Data Communication* (Seattle, WA, USA) *(SIGCOMM '08)*. Association for Computing Machinery, New York, NY, USA, 413–426. doi:10.1145/1402958.1403005

[39] Tamoghna Ojha, Sudip Misra, and Narendra Singh Raghuwanshi. 2015. Wireless sensor networks for agriculture: The state-of-the-art in practice and future challenges. *Computers and electronics in agriculture* 118 (2015).

[40] Adamou Ouorou, Philippe Mahey, and J-Ph Vial. 2000. A survey of algorithms for convex multicommodity flow problems. *Management science* 46, 1 (2000), 126–147.

[41] Ayrstone Productivity. 2025. Outdoor WIFI for farm, ranch, and Rural Home. https://ayrstone.com/www/.

[42] Lili Qiu, Yin Zhang, Feng Wang, Mi Kyung Han, and Ratul Mahajan. 2007. A general model of wireless interference. In *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*.

[43] Ashish Raniwala, Kartik Gopalan, and Tzi-cker Chiueh. 2004. Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks. *ACM SIGMOBILE Mobile Computing and Communications Review* 8, 2 (2004).

[44] David Christian Rose, Jessica Lyon, Auvikki de Boon, Marc Hanheide, and Simon Pearson. 2021. Responsible development of autonomous robotics in agriculture. *Nature Food* 2, 5 (2021).

[45] Eric Rozner, Yogita Mehta, Aditya Akella, and Lili Qiu. 2007. Traffic-aware channel assignment in enterprise wireless LANs. In *2007 IEEE international conference on network protocols*. IEEE.

[46] David Schimmelpfennig. 2016. *Farm Profits and Adoption of Precision Agriculture*. Technical Report.

[47] Claude E Shannon. 1949. Communication in the Presence of Noise. *Proceedings of the IRE* 37, 1 (1949).

[48] Arun Narenthiran Sivakumar, Sahil Modi, Mateus Valverde Gasparino, Che Ellis, Andres Eduardo Baquero Velasquez, Girish Chowdhary, and Saurabh Gupta. 2021. Learned Visual Navigation for Under-Canopy Agricultural Robots. In *Robotics: Science and Systems*.

[49] Global GPS System. 2023. RTK GPS: Understanding Real-Time Kinematic GPS Technology. https://globalgpssystems.com/gnss/rtk-gps-understanding-real-time-kinematic-gps-technology/.

[50] Yunchao Tang, Mingyou Chen, Chenglin Wang, Lufeng Luo, Jinhui Li, Guoping Lian, and Xiangjun Zou. 2020. Recognition and Localization Methods for Vision-based Fruit Picking Robots: A Review. *Frontiers in Plant Science* 11 (2020).

[51] Farm Proven Technology. 2023. On Farm Technology Proving Grounds . https://farmproven.tech/.

[52] United States Department of Agriculture. 2019. The Case for Rural Broadband: Insights on Rural Broadband Infrastructure and Next Generation Precision Agriculture Technologies. https://www.usda.gov/sites/default/files/documents/case-for-rural-broadband.pdf.

[53] Deepak Vasisht, Zerina Kapetanovic, Jongho Won, Xinxin Jin, Ranveer Chandra, Sudipta Sinha, Ashish Kapoor, Madhusudhan Sudarshan, and Sean Stratman. 2017. {FarmBeats}: An {IoT} Platform for {Data-Driven} Agriculture. In *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*. 515–529.

[54] wi fi.org. 2023. Wi-Fi CERTIFIED 6. https://www.wi-fi.org/discover-wi-fi/wi-fi-certified-6.

[55] Huarui Wu, Lihong Zhang, and Yisheng Miao. 2017. The Propagation Characteristics of Radio Frequency Signals for Wireless Sensor Networks in Large-Scale Farmland. *Wirel. Pers. Commun.* 95, 4 (aug 2017). doi:10.1007/s11277-017-4018-5

[56] Denan Xia, Peng Chen, Bing Wang, Jun Zhang, and Chengjun Xie. 2018. Insect Detection and Classification Based on an Improved Convolutional Neural Network. *Sensors* 18, 12 (2018).

[57] Xue Yang and Nitin H. Vaidya. 2002. Priority scheduling in wireless ad hoc networks. In *Proceedings of the 3rd ACM International Symposium on Mobile Ad Hoc Networking & Computing* (Lausanne, Switzerland) *(MobiHoc '02)*. Association for Computing Machinery, New York, NY, USA, 71–79. doi:10.1145/513800.513809

[58] Chen Ziwen, Zhang Chunlong, Li Nan, Sun Zhe, Li Wei, and Zhang Bin. 2015. Study review and analysis of high performance intra-row weeding robot. *Transactions of the Chinese Society of Agricultural Engineering* 31,

5 (2015).

A Appendix: Extended Analysis on Computing Consumed Resource Units

We refer to the figure in §4 to provide further analysis of our formulation for computing the amount of resource units consumed by a flow due to interference.

Analysis for a single source of interference: We now discuss the case where $d_{AC} > d_{AB}$ and the impact of interference is realized through reduced signal-to-noise ratio.

$$r(f_{A\to B},C) = r(f_{A\to B},A) \times \frac{log(ss(d_{AC}))}{log(ss(D)} \tag{3}$$

We get Equation 3 by applying Shannon's Law i.e. $throughput(d) \approx BWlog(ss(d)/N)$ to Equation 2. Here, BW is the total channel bandwidth available for the transmission, $ss(d)$ is the signal strength at distance d (in the absence of any interference), and N is the noise floor (we assume noise floor is constant across devices and ignore it for simplicity). The specific value of $\delta(f_{A\to B},C)$ in our formulation closely approximates the amount of resource units consumed at C when we reason about the interference effects from f in terms of using the leftover resource units at C for another flow f' from C to E at distance $d_{CE} = D$ (which is the inter-router distance in our grid). As per our formulation, the datarate achieved by f' will be its share of channel bandwidth multiplied by its signal strength in dB ($log(ss(d_{CE}))$). This will be equal to:

$$\begin{aligned}
&= (1 - r(f_{A\to B},C))BWlog(ss(d_{CE})) \\
&= \left(1 - r(f_{A\to B},A) \times \frac{log(ss(d_{AC}))}{log(ss(D)}\right) BWlog(ss(d_{CE}) \\
&= (1 - r(f_{A\to B},A))BWlog(ss(d_{CE}) \\
&+ r(f_{A\to B},A)BWlog(ss(d_{CE}) \left(1 - \frac{log(ss(d_{AC}))}{log(ss(D)}\right)
\end{aligned} \tag{4}$$

When $d_{CE} = D$, this is equal to:

$$\begin{aligned}
&= (1 - r(f_{A\to B},A))BWlog(ss(d_{CE}) \\
&+ r(f_{A\to B},A)BWlog\left(\frac{ss(d_{CE})}{ss(d_{AC})}\right)
\end{aligned} \tag{5}$$

This captures the degradation in SINR for the transmission from C to E due to interference power of strength $ss(d_{AC})$ on the portion of channel bandwidth ($r(f_{A\to B},A)$) accessed by the interfering flow f.

Analysis for multiple sources of interference: A device C may see interference from multiple flows $f_1,....f_k$ sourced at $A_1,.....A_k$. As mentioned in §4, we simply add up the amount of resource units consumed by individual interfering flows to approximate the cumulative level of interference. This is a very close approximation when $\delta(f_i,C) = 1$, i.e. when source of interference is so close that simultaneous transmission is not possible. In cases where $\delta(f_i,C) < 1$ (i.e. interference effect is captured by degradation in channel quality), the additive model continues to provide a close approximation when $\sum(r(f_i,A_i)) \leq 1$, where we can assume that interference from different sources impacts different portions of the channel (e.g. at different times). When $\delta(f_i,C) < 1$ and $\sum(r(f_i,A_i)) > 1$, our additive model over-estimates the effects of interference on the portions of channel bandwidth that experience simultaneous interference from multiple sources. In particular, the impact of simultaneous interference from multiple sources on datarate can be approximated as:

$$\begin{aligned}
datarate &= BWlog\left(\frac{ss(d)}{I_1 + I_2 + ...I_n}\right) \\
&= BW(log(ss(d)) - log(\sum(I_i))
\end{aligned} \tag{6}$$

Here, I_i denotes the strength of the interfering signal. Our additive model will instead estimate the datarate associated with the leftover channel-capacity (for the portion of channel that experiences simultaneous interference) as:

$$datarate = BW(log(ss(d)) - \sum(log(I_i)) \tag{7}$$

This directly follows from the reasoning about a single interference source detailed above. Given positive interference signals (in dB), $I_i > 1$ and $\sum(log(I_i)) > log(\sum(I_i))$, resulting in over-estimated impact of interference. This overestimate is restricted to portion of channel bandwidth that experience simultaneous interference (roughly corresponding to $(\sum r(f_i,A_i) - 1)$ resource units). Moreover, its restricted to settings where impact of interference is smaller to begin with $\delta(f_i,C) < 1$.