Zyxt: A Network Planning Tool for Rural Wireless ISPs

Thomas Pötsch
New York University Abu Dhabi
thomas.poetsch@nyu.edu

Salman Yousaf
New York University Abu Dhabi
salman.yousaf@nyu.edu

Barath Raghavan
ICSI / USC
barath.raghavan@usc.edu

Jay Chen
New York University Abu Dhabi
jchen@cs.nyu.edu

ABSTRACT

Rural areas are home to 45% of the world's population and are neglected in the design and deployment of broadband Internet access. Beyond establishing connectivity, the planning and design of networks in these areas is often challenging because the complexity of planning such networks often exceeds the knowledge and practical expertise of individuals – even individuals that are experts in this field. Yet, proper planning and efficient network designs are critical to the long term viability and scalability of such networks. In this paper, we highlight the challenges that lone operators who lack sufficient technical support and tools must face when designing and deploying their networks. We then present a network planning tool, Zyxt, that aims to help individuals in planning rural wireless networks. We describe the design and implementation of our prototype and evaluate it against synthetic and real-world planning problems. Our evaluation demonstrates that Zyxt is able to solve real-world network planning problems using relatively modest computing resources within a few hours.

CCS CONCEPTS

• **Networks** → **Network manageability**; **Wireless local area networks**;

KEYWORDS

Network planning, Network deployment, WISP networks

ACM Reference Format:
Thomas Pötsch, Salman Yousaf, Barath Raghavan, and Jay Chen. 2018. Zyxt: A Network Planning Tool for Rural Wireless ISPs. In *COMPASS '18: ACM SIGCAS Conference on Computing and Sustainable Societies (COMPASS), June 20–22, 2018, Menlo Park and San Jose, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3209811.3209874

1 INTRODUCTION

Over a decade ago our community set out on a mission to identify and eliminate barriers to the universal adoption of Internet access. As is still true today, we knew that Internet access is as much about economics as it is about technology. Therefore, despite ISPs being unwilling to build expensive infrastructure to serve regions with low user densities, we were certain of our inevitable success so long as cheaper, faster, longer range, and more rugged wireless equipment continued to become available [46].

Today, commodity wireless equipment is cheap, Internet is a basic human right, and major companies have joined the effort. However, despite buzz about high-cost, high-complexity, high-tech solutions to the problem, we have made only slow progress toward universal access. Near highly-connected cities there are communities connecting via dialup and their connections are getting *slower* – now crawling along at 9600 bps. Such neglected rural areas are home to 45% of the world's population.

In this paper we explore how to meet the challenges faced by the lone operator in the vast unconnected frontier. Building basic infrastructure in this frontier, even in wealthy nations, is an enormous endeavor. If universal connectivity is to be achieved, it will be not through the few, large operators connecting the last billions. Instead, we believe connectivity will flow through the thousands wireless ISP (WISP) operators, often one-person outfits, who have a stake in bringing access to their own communities.

The key challenge for these operators is not one of hardware – commodity hardware is widely available and easy to set up – nor is it of management, as there are a number of free systems to aid them once up and running. Instead, it is a mismatch between the skills of to-be operators and the task at hand: planning a WISP network often requires a combination of extensive knowledge and practical expertise seldom found in one individual. As we discuss in Section 2, even for our expert team it was difficult to build such a network quickly, at low cost, and with few missteps; for unskilled lone operators who do not have our resources the difficulty is far greater.

Traditionally, the task of network planning typically falls to large carriers (in the case of backbones) and cloud providers (in the case of datacenters), both of which have the financial and political resources to overcome physical obstacles (e.g. dig trenches, acquire spectrum, build large towers, buy land). In contrast, WISP operators must plan and operate *within* existing constraints and cope with the complex myriad of network planning tasks. No one task dominates others in importance when planning WISP networks, but the accumulation of poorly made decisions can easily bring down a network, leaving users in the dark once again as is the history of numerous rural operators.

In this paper we motivate the problem by describing one of our own experiences planning and deploying a rural WISP network; we highlight the practicalities of building such networks and what distinguishes them from other types of networks and existing approaches in the research literature. We then present our design

COMPASS '18, June 20–22, 2018, Menlo Park and San Jose, CA, USA

ACM ISBN 978-1-4503-5816-3/18/06.
https://doi.org/10.1145/3209811.3209874

and implementation of Zyxt, a semi-automated network planning tool, and evaluate Zyxt's components on synthetic problems as well as holistically on real-world scenarios. We demonstrate that Zyxt reduces the complex problem of network planning to a tractable problem that can be completed in essentially real-time while providing tradeoffs between memory use, error, runtime, and overall cost of the solution.

2 CASE STUDY: A RURAL WISP NETWORK DEPLOYMENT

To make the challenges of planning WISPs apparent, we present a brief case study on a network we deployed in a previously-unconnected region in rural Northern California [18]. Our experience illustrates some of the problems faced by a team of networking professionals when designing and deploying a WISP network infrastructure. While all networks are unique to their circumstances, the challenges are broadly similar, as found when we spoke with many dozens of rural network operators in North America, Asia, and Africa and given our firsthand knowledge of many of these networks. Not all of these challenges will be solved by our proposed planning tool, but our aim is to give the reader a sense of the difficulties faced in WISP design and the operational issues that should be taken into account.

2.1 Case Study Context

We learned of a region, about 50 km by 20 km, that was without broadband Internet connectivity. Local users who wanted Internet access either used a small regional dialup Internet provider or used slow satellite Internet. Figure 1 is a rotated map of the region populated with data of a set of potential customer sites.

The region had no coaxial infrastructure and poorly-distributed twisted-pair copper infrastructure, and thus no cable or DSL service. Cellular coverage was spotty, with no 4G service and unreliable 3G service; incumbent telcos had expressed no interest in improving service to the region, and even left backup generators in disrepair, resulting in frequent outages due to unreliable grid power. Several rivers and creeks cross the main road, which flood frequently cutting off road access. The region as a whole was economically depressed, including a local tribal community, with about a quarter of households living in poverty; however, there were pockets of affluence. Over the past decade at least three other operators have provided service to the region for a time, only to fail due to poor network planning and infrastructure and other challenges, resulting in poor network reliability and performance and leading to eventual business failure. Given this context, our challenge was this: how do we build a cost-effective, performant network to provide connectivity to the population depicted in Figure 1?

2.2 Deployment Process

Our deployment team consisted of several engineers and technicians. Our initial task was to identify a source of upstream bandwidth. No universal map of this information exists, and large telecoms (that are the usual providers of such service) do not publicize locations of their fiber facilities in such regions. After hearing local reports of a facility in the region, we contacted a large provider who, after months of our effort following up with them, confirmed for us that they would be able to sell us upstream bandwidth.

The lack of wireline infrastructure and the cost of building cell infrastructure and acquiring spectrum made microwave links (e.g., directional WiFi) a natural choice [36]. This hardware is cheap, low power, and easy to set up. However, such links require line-of-sight, have distance limitations, and can struggle with reflections, intermittent obstructions (i.e. severe weather), and is spectrum constrained.

Our first challenge was to determine how to distribute connectivity from the upstream gateway site. The telecom rejected our proposal to mount gear at their facility at low cost, leaving us with no option but to trench fiber from their site to another location nearby where this could be distributed. An ideal nearby site was a large, empty hillside near the facility. After another two months of tracking down and negotiating with the reclusive, elderly owner of the empty land, we were told that we could use the hill only for an exorbitant monthly fee. In parallel we considered several other neighboring sites, all of which were further away and none of which had any elevation. After the hillside was eliminated from consideration, we opted to trench fiber further to an alternative, low-lying location, from which we then had to set up backhaul links to a more distant hilltop location we secured, which would serve as a major distribution hub.

The topography of the region – a narrow stretch of land between ocean and mountains that rise 1,000 m – dictated where we could place relay sites. Our constraints were further modulated by additional factors: where we could get power, where line-of-sight existed, where we had access to sites, and where potential users were situated. Existing tools only serve to compute line-of-sight between pairs of nodes, something available in many GIS planning tools. Since such networks have been built for a number of years, we expected that existing tools might be capable of doing semi-automated planning, but we found that the state of the art has scarcely advanced over the last decade. In each area that we aimed to expand connectivity, we first spent many weeks using existing rudimentary planning tools [22] to manually identify multiple locations in concert that had line-of-sight and were located with good proximity to user populations. This was ultimately a guess-and-check approach. Once we had narrowed the list of sites, we then spent additional time to negotiate with land owners, businesses, and civic institutions.

The choice of radio frequencies at our sites was also decided manually and after many considerations. Spectrum contention was commonplace; despite our heavy use of unlicensed 5 GHz spectrum, in which there are numerous non-overlapping channels, we were forced to use other unlicensed bands as well due to contention at major sites.

After over six months of extensive planning, negotiation, and rollout efforts, our modest network consisted of about six sites and provided coverage to perhaps fifty users; it eventually took *years* for our network to expand to serve the majority of the region's userbase. When unthrottled, many subscribers could receive 30-60 Mbps symmetric throughput to the Internet with less than 5 ms latency within our network. At major infrastructure sites we also deploy batteries and networked power monitors, and power all key network devices using Power-over-Ethernet (PoE).

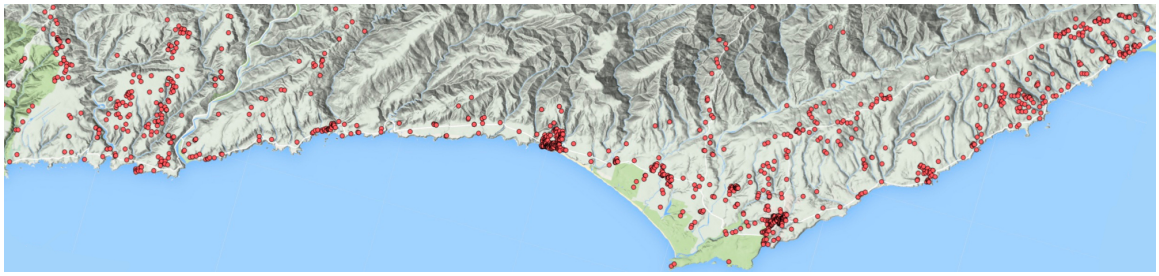


Figure 1: Map of 50 km × 20 km WISP network region and locations of a subset of (potential) customers

Despite being a skilled and experienced group we encountered numerous complex issues in planning, deploying, and managing the network and had to grapple with these issues with few tools at our disposal. As a result, while we were careful to weigh the decisions we made in designing and deploying the network, many decisions were still ad-hoc, and some decisions we made turned out to be mistakes that took time and money to undo.

3 BACKGROUND AND RELATED WORK

3.1 Rural Networks

Networking in developing and rural regions is a topic that has attracted attention over the last decade [7, 9, 32, 44]. This focus has intensified in recent years and researchers have recognized that the challenges presented in these networking regimes are substantially different and require different solutions [3, 10–13, 19, 20, 28, 36, 40, 42, 43, 47, 48, 56]. The various challenges and the approaches described in this body of literature include: connectivity establishment [19, 36, 40], high hardware cost, unreliable or low quality power [49], poor performance [10, 11, 42], and difficulties in operational management [18]. While there has been work on protocols and techniques for faster and more reliable networks, research often does not extend far enough to keep such networks alive after the research is done, as the operational challenges in such networks are quite high [49].

The global rural network coverage has not dramatically increased yet, but we have made progress as many of the above problems now have known solutions that are being incorporated into even mainstream systems [1, 21, 31, 34, 52]. The turning point for rural networking hardware was the growth of cheap, reliable, and fast commodity wireless hardware (built upon SoC boards from Atheros and others, which were mass produced for the 802.11 home router market) which started to become widely available around 2008 from vendors such as Ubiquiti [52] and Mikrotik [31]. More recently large companies such as Google and Facebook have taken an interest in the problem space. New hardware platforms are being developed to deliver middle-mile connectivity [5, 30]. However, the design and deployment of last-mile networks remains an unsolved problem.

3.2 Conventional Network Planning

Historically, designing and managing networks has not been of major research interest to the networking community prior to the popularization of software-defined networking (SDN). Before SDN, the tedious planning and management of wired networks involved a myriad of decisions regarding equipment, configuration policies, and management software. These laborious manual processes slowed innovation, increased complexity, and inflated both the capital and the operational costs of running a network [29]. Today's last-mile networks face a similar bottleneck.

The design and management challenges in rural wireless networks imposed by physical constraints are largely unexplored in the existing research literature. Instead, much of the previous work on wireless networking concentrates on wireless ad-hoc, mesh, and sensor networks [8, 41, 53, 55]. Within wireless, variations of topology planning considered hardware factors such as transmission power and directional antenna [24, 41]. These types of networks are nearly the complete opposite of rural wireless networks. Rural wireless networks, unlike ad-hoc networks, require complex planning to ensure high performance, robustness, and cost efficiency. As such, network operators must invest significant effort in planning, or deal with the consequences later. There are also numerous patents on planning cellular networks [2]. Several approaches have been proposed in cellular network planning for the placement of base stations (e.g. Andrews et al. [4]), but these generally focus on spectrum and interference rather than physical topography. The closest work to ours is an algorithm by Sen and Raman that attempts to minimize the overall network cost by considering tower height, antenna type, and transmit power [45].

Nearly all networking planning research focuses on network complexity problems *internal* to the network (e.g. wiring, cooling, protocols, management, etc.); networking researchers and engineers are typically insulated from the many *external* planning problems (e.g. facility siting, tower siting, fiber path planning, power management, etc.) that other well-resourced teams are responsible for handling in most large organizations (e.g. in datacenter networks, enterprise wireless networks, and regional wireline ISP networks). In rural WISP networks, when all of these problems are borne by a single individual or a very small team, the task becomes overwhelming. Furthermore, WISPs do not have the financial or political capital to mitigate the sources of complexity and therefore must address them directly.

3.3 WISP Network Planning

In contrast to traditionally studied networks, a typical WISP network is deployed across a large and topographically diverse area (e.g. 50 km × 20 km, or 1,000 sqkm). In such an area, considering,

crudely, that sites are typically parcels of rural land on the order of a couple of hectares each, there are about 50,000 potential sites. Even if we immediately aggregate or discard as non-viable 80% of these potential sites using various heuristics, some 10,000 possible site options remain. At each site, the number of constraints to be considered for placement of devices (which are directional, not omnidirectional) is on the order of twenty, including power availability, tree cover, slope of terrain, orientation, type of radio, type of antenna, type of tower or mast, type of hardware, and more.

Across these potential sites, the network only needs on the order of a dozen sites to serve the area, and such sites must be selected jointly, as the best set of sites (and their configurations) out of the thousands of options. This selection of the best small set from a large set of options results in combinatorial explosion, yielding many orders of magnitude greater design complexity than in other types of network design. It is the inability to cope with the combinatorics of the problem that frequently pushes network operators to make many ad-hoc design decisions that result in networks that are unreliable and slow – and thus expensive and short-lived.

When deploying their networks today, rural operators use a mix of incomplete planning tools. Some tools build upon terrain data to estimate line-of-sight between two locations, enabling an operator to perform rudimentary topography planning for relay sites [22], other tools [50, 54] provide tools to manually plan, understand, and deploy wireless networks. Only a few alternative planning models have also been proposed. IncrEase [6] is a planning paradigm that incrementally introduces sets of additional transmission sites. In [25], the authors describe a mathematical model for automated network planning that considers economic and technical constraints. Other systems, such as TowerDB and Celerate, attempt to simplify network management by juxtaposing geographic locations of devices with network information (e.g. IP-address, frequency, and SSID) [18, 51].

4 ZYXT SYSTEM OVERVIEW

Our goal is to enable the semi-automated design of a WISP network through the use of Zyxt, our network design system. We envision a would-be network operator (who may or may not have any network design or management expertise) articulating the geographic locations to be served, policy aims, and other limitations or criteria, and being given a fully-specified network design by Zyxt, including the relay/backhaul locations and the network hardware to deploy, device configurations including spectrum allocation, and physical deployment specifics including elevation and power considerations. Such a design could then be improved through iteration with the design system – for example, as land use is negotiated – and a final design could be used as a blueprint for deployment.

To enable this, Zyxt must translate constraints along with user-specified policies into a cohesive representation that then enables the construction of a network design by a solver. This result is then post-processed to account for other considerations not easily incorporated into the solver representation such as the profitability of providing service to a customer. Eventually, the design generated by the system's solver must be re-represented to enable the operator to refine and converge upon a network design. At the core of this iterative design process is a pre-processing step that combines the operator's design specifications and physical models to produce a problem representation for a solver to then produce a network design. Figure 2 shows the Zyxt system overview. In this paper we focus on the design, implementation, and evaluation of the pre-processing and solver aspects of Zyxt.

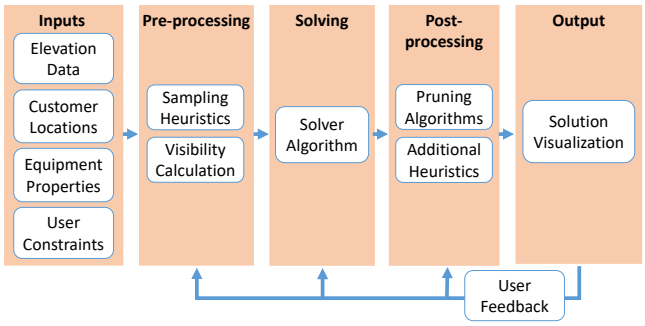


Figure 2: Zyxt system overview

4.1 Pre-processing

The main contributing factor of the site dependencies is site visibility among sites. In this context, we distinguish three different types of sites: customer sites (sinks), potential intermediate towers (relays) and backhaul sites (sources). Given that sites are connected through point-to-point wireless communication equipment, the visibility between locations is largely determined by the elevation of the terrain. Within the region under consideration, we place the sinks and sources and must then determine the set of intermediate tower locations required to interconnect the customer sites. Intermediate tower locations can be at any location within the region including customer sites.

To determine site visiblity, we can use the elevation data to compute the set of *viewsheds* of all locations within the region. These viewsheds describe the visibility (line-of-sight) at each location and can be used to produce a visibility graph that encodes the visibility between all location pairs. This allows us to formulate our design problem as a classical graph theory problem, where vertices represent the source, sink, or relay locations and edges represent the connectivity (line-of-sight) between the vertices. There is an extensive body of research on graph theory and numerous algorithms exist that tackle similar design problems (e.g. [23, 35]).

We assume an undirected graph $G = (V, E)$ with non-negative edge weights c and a subset of vertices $S \subseteq V$ that should be connected using the least amount of remaining vertices, i.e. G that spans S. In order to map our problem to a conventional graph theory problem, each of our potential intermediate tower locations, customer locations and backhaul sites are $v \in V$ and the visibility among the locations are $e \in E$. Further, the edges e that connect the vertices v have assigned edge weights that correspond to the equipment cost of the link. The equipment costs and flow capacities are calculated based on a predefined set of hardware that can be used to cover different transmission ranges and hence correspond to the physical length of the edge. Source and sink vertices are associated with supplies and demands that represent the provided bandwidth to the network and desired bandwidth, respectively.

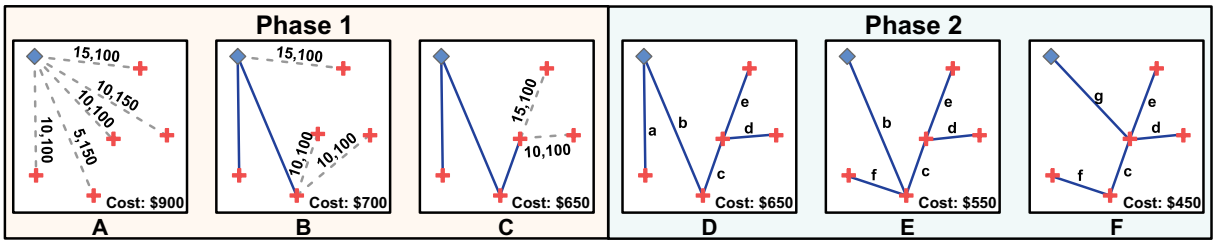


Figure 3: Simplified steps of MCF-RR phase one and two. The blue diamond denotes the source, the red crosses the customer locations, and the dashed edges are annotated with the passed flow and the hardware bandwidth that is chosen after each MCF iteration. Intermediate towers are omitted for clarity.

4.2 Solving

The network planning problem naturally lends itself to a graph optimization representation of connecting the vertices with the minimum total cost edges that provide sufficient flow capacity to satisfy the customers. This problem is a special case of the well-known Fixed-Charge Network Flow (FCNF) problem, which has applications in transportation and communication [17]. The FCNF problem is NP-Hard and could be directly given to a generic solver. However, as we demonstrate, using an off-the-shelf solver is infeasible for our large problems due to memory bottlenecks and long run times.

Given that multiple aspects of the design problem are NP-Hard, such as site selection and spectrum allocation [37], the underlying constraint datasets are rife with error, and the policies expressed by the operator are ambiguous, there is limited room for or value in developing an "optimal" algorithm. Thus, we do not attempt to optimally solve this problem from a theoretical perspective. Instead, we believe it to be sufficient to develop a practical system that is capable of producing a result that is substantially better than the status quo today because the resulting decrease in life-cycle cost over the course of the network's design, deployment, and management will make all the difference for the viability and longevity of WISP networks.

We initially considered several specific algorithms that do not require the use of a general solver that might be suitable starting points to our problem (e.g. Minimum Cost Flow, Steiner Tree, etc.), but these problem formulations do not capture the relevant constraints or objectives of FCNF. Minimum Cost Flow (MCF) bases its optimization on unit costs whereas our costs are fixed if the edge is included (i.e. purchasing a piece of equipment costs the same regardless of the utilization). The Steiner tree algorithm cannot take into account the supplies and demands of the sources and sinks. We initially attempted to post-process the results of these algorithms, but we found that this approach generally yielded network designs that were 2-3 times more expensive than the optimal solution even on small problems. We omit a more extensive comparison due to space constraints and instead focus on comparing solving FCNF using a generic solver against our approach of augmenting MCF with iterative randomized rounding (MCF-RR).

4.2.1 Fixed-Charge Network Flow (FCNF). The Fixed-Charge Network Flow (FCNF) formulation is often used to solve transportation and communication network problems in which a flow has to be distributed from supply nodes through intermediate nodes to demand nodes. For each node, the demand is positive for demand nodes (sinks), negative for supply nodes (sources), and zero for intermediate nodes (relays). The nodes are connected by edges which have a capacity, a unit cost, and a fixed cost. The goal of FCNF is to decide how to ship the demands from the supply nodes to the demand nodes so that the shipping costs are minimized. We map our problem to FCNF as follows: locations are mapped to vertices, visibility between two locations is mapped to an edge, the bandwidths of potential links are mapped to the capacity of the edges, the hardware cost of the links are mapped to the fixed shipping costs, and the desired and offered bandwidth at the node locations are mapped to the demands and supplies of the vertices, respectively.

4.2.2 Minimum Cost Flow with Iterative Randomized Rounding (MCF-RR). In order to overcome the aforementioned unit cost limitations of the MCF algorithm, we approximate our desired algorithm by combining an iterative randomized rounding technique [39] with MCF, which we call MCF-RR.

MCF-RR runs in two phases as illustrated by an example in Figure 3: In the first phase, we run MCF and "keep" edges probabilistically based on their capacity utilization for the next iteration (steps A - C). For example, if the capacity of an edge is 100 Mbps and its utilization is 15 Mbps, then the edge is kept with a probability of 15%. If the edge is kept, its unit cost is set to zero for the next iteration to force MCF to use this edge. Intuitively, keeping edges in this manner causes MCF to push more flow through it in the next iteration (thus taking into consideration the fixed cost of adding edges rather than solely focusing on unit cost). Edges that were not kept in an iteration could potentially be added again in the next iteration. This procedure is repeated until all flows are distributed only over kept edges and the total unit cost converges to zero (step D).

Since the solution of phase one is heavily determined by the probabilistic decisions that are made in the first iterations, we repeat phase one multiple times with different random seeds.[1] The solution with the cheapest total hardware cost is selected as the input for phase two.

In the second phase of MCF-RR, the algorithm tries to further optimize the solution by iteratively backtracking to attempt to remove edges. We first remove edges (one edge at a time) that

[1] In practice, we found that around 10 random seeds were sufficient to find a solution that is a good starting point for phase two.

connect only one customer site from the solution and re-run MCF (e.g. edge a in step D). If MCF brings the same edge back, then we assume that the flow of this edge cannot be easily distributed elsewhere. If the flow of this edge is redistributed to an existing edge (e.g. edge b and f in step E), then the solver compares the total cost of the new solution with the total cost of the previous solution. If the cost of the new solution is lower, the new solution is taken as input to the next iteration. After removing all edges that connect one customer site, we attempt to remove edges that connect two, three, etc. customer sites (e.g. edge b in step E).[2] If the total hardware cost of a solution at any iteration is lower than in the previous iteration, then phase two is restarted from the beginning. The solver terminates phase two after all edges have been removed at least once (step F).

5 IMPLEMENTATION

As a fundamental input for the solver, we use elevation data from the latest SRTM dataset [14] and post-process it using the GRASS GIS library [16]. Based on the locations of the customer and backhaul sites provided by the user, we use several GRASS GIS routines to access, modify and re-scale the elevation data to make them suitable for our model. This includes patching multiple SRTM map tiles together, cropping the maps to the region of interest, re-sampling the raw data to lower the memory footprint and model complexity, and also placing possible intermediate tower locations on the map.

5.1 Visibility Calculation

To obtain the visibility graph, we calculate the visibility from every location to every other location. The visibility between locations is determined by two physical properties: the geographical elevation and the mounting height of the equipment to be placed at that location. While the geographical elevation is defined by the underlying elevation map of the terrain, the mounting height of the equipment can be set by adding the mounting height to the elevations of the pair of locations during the visibility calculation. Two locations are visible to each other if all elevations along the path between the locations are lower than the line-of-sight between the two locations (including mounting height). By introducing mounting heights to both locations, the visibility between the locations can be improved as the line-of-sight could see over obstacles between the two locations.

Depending on the resolution of the elevation map, the total number of viewshed calculations can quickly become very large. Given a map size of n locations, $n^2 - n$ viewshed calculations are required to obtain a full visibility network of all locations considered in the map. When assuming a symmetric visibility of location pairs, i.e. location A can see location B and vice versa, the number of viewshed calculations reduces to $\frac{n^2-n}{2}$. Due to this large number and since the viewshed calculation is an processing intensive task for all points on the map, we implemented the viewshed algorithm in CUDA [33]. As the CUDA architecture allows a calculation of multiple viewsheds in parallel, i.e. we calculate each location's visibility in its own thread, the execution time is significantly reduced.

[2]We order the edge removal in order of ascending flow capacity. Experimentally, we found that randomly ordering candidate edges for removal converges approximately twice as quickly, but increases the total network cost by 5-10%.

Hardware	Bandwidth (Mbps)	Range (km)	Cost ($)
airFiber 5	1200	100	1998
airFiber 2X	500	200	1056
NanoBeam M5-16	150	10	134
PowerBeam M2-400	150	20	158
PowerBeam M5-400	150	25	190
PowerBeam M5-620	150	30	398
PowerBeam 5AC-300	450	20	198
PowerBeam 5AC-620	450	30	458
NanoStation locoM2	150	5	98

Table 1: Equipment list

For example, given a map with 1500 elevation points, the CUDA viewshed implementation is completed within one second using a Quadro M4000 graphics card with 1664 cores, whereas it would take several minutes when done without CUDA. The viewsheds of all locations are stored in binary as an adjacency matrix.

5.2 Solver Algorithms

5.2.1 FCNF. We use a Mixed Integer Programming (MIP) representation of the algorithm to solve the FCNF formulation. We use CPLEX [26], an off-the-shelf generic solver, to create equations for the MIP model.

Formally, a FCNF network is given by $G = (V, E)$ where the demand of each vertex $v \in V$ is d_v, the capacity of each edge $e \in E$ is u_e, the per-unit cost of an edge is c_e, and the fixed cost of an edge is f_e. Further, x_e is the flow through an edge and y_e is a binary variable indicating if a flow is passed through an edge ($y_e = 1$), or $y_e = 0$ otherwise. The FCNF problem can be formulated as follows:

$$\min \quad \sum_{e \in E} c_e x_e + \sum_{e \in E} f_e y_e \tag{1}$$

$$\text{s. t.} \quad \sum_{e \in E(V,v)} x_e - \sum_{e \in E(v,V)} x_e = d_v \qquad v \in V \tag{2}$$

$$0 \le x_e \le u_e y_e \qquad e \in E \tag{3}$$

$$y_e \in \{0, 1\} \qquad e \in E \tag{4}$$

5.2.2 MCF-RR. We implement MCF-RR based on the Min Cost Flow implementation in the Google OR-tools C++ open source library [15]. To assign the unit costs of the edges we base the unit cost on the cost-over-transmission-range of equipment pairs. However, since equipment does not exist for all possible link distances, we approximate the cost-over-transmission-range by fitting a curve to the transmission range vs cost graph (Figure 4) of the set of available equipment. Table 1 shows the equipment that we consider in this paper along with their costs, which represent the cost of a pair of devices.[3] We initially assign the capacity of each edge in our graph to infinity. By setting cost and capacity in this manner, we decouple capacity from hardware cost and assign the actual hardware costs and capacity after iterations of MCF based on the flow through each edge.

[3]We obtained these costs from [52] in September, 2016.

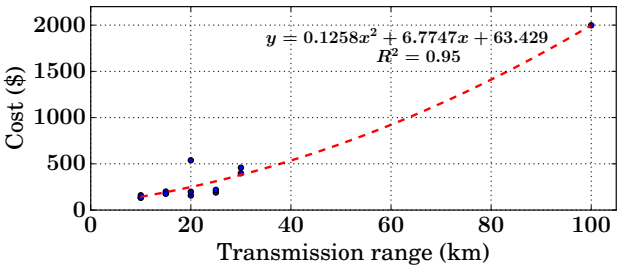


Figure 4: Hardware cost over transmission range

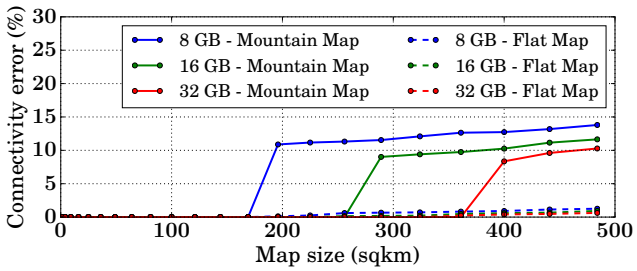


Figure 5: Connectivity error over map size

6 MEMORY BOTTLENECKS

Our network planning problem is first represented and stored in memory as a visibility graph and then within a solver in the solver's internal representation. The large size of these problems and their associated representations can cause memory to be a major bottleneck. In this section we consider the two main memory bottlenecks and show that our MCF-RR implementation can scale to problems that are two orders of magnitude larger than solving FCNF using CPLEX. We then introduce three sampling heuristics that are applied during pre-processing that allows us to solve larger problem sizes and briefly compare them.

6.1 Visibility Graph

The size of the visiblity graph is constrained by the memory available on the host machine. The raw elevation data that we had access to, at its maximum resolution, has a resolution of ~30 m. Thus, the total map resolution can quickly become very large if the region under consideration is large. For example, a 100 sqkm map results in ~135,000 elevation points. Given r elevation points on the map, the memory required by our binary adjacency matrix is $r^2/8$. For maps that do not fit into the memory of the host machine, we must re-scale the map to fit into the memory. A "nearest neighbor" sampling method (supplied by GRASS GIS [16]) may be applied to re-scale the map to the desired size. However, such re-scaling will introduce error to the viewshed calculations due to the averaging of multiple elevation points to one.

We evaluate the error of the visibility graph that is introduced by re-scaling the map resolution. Using our 64 GB machine, we emulated machines with smaller memory so that we were able to calculate the viewsheds of the maps that were not re-scaled, i.e. using the full resolution of the map with no error. The connectivity error is defined for every location pair in the re-scaled map, where we compare its visibility with the visibility of the exact same location pair (latitude and longitude) in the full resolution map. Whenever the connectivity (visible or not visible) differs, it is counted as an error.

Figure 5 shows the relative connectivity error over different map sizes for maps of low and high elevation variance. We picked two 8 km by 8 km regions in rural California with different topology properties, i.e. a flat area with little elevation and a mountainous area with highly varying elevations. The native resolution of these elevation maps was ~30 m. We observe that the connectivity error for a map with little elevation variation is very low and re-scaling of the map does not introduce a large connectivity error. However, on a map with high elevation variation, the connectivity error increases when maps must be downscaled to fit into memory. Re-scaling introduces significant error on non-trivial terrain and should be avoided because it can lead to network topologies that are impossible to connect in reality.

Map Resolution (# of Vertices)	Edges	Customer Sites	Memory (FCNF)	Memory (MCF-RR)
20 × 20 (400)	76,519	80	835 MB	5 MB
40 × 40 (1,600)	1,125,897	320	11,842 MB	113 MB
60 × 60 (3,600)	5,637,153	720	57,372 MB	552 MB
80 × 80 (6,400)	17,955,115	1,280	>64,000 MB	1,642 MB

Table 2: Memory footprint of FCNF and MCF-RR

6.2 FCNF & MCF-RR

To run a solver for FCNF or MCF-RR, the problem representation must fit in memory. We compare the memory footprint of FCNF and MCF-RR using the mountainous region from the previous experiment. We re-scale the region down to four scenarios of different sizes, each with evenly distributed elevation points. We then randomly distribute customer sites within each scenario. In these microbenchmarks we ignore error introduced by re-scaling to focus on the memory footprint of the two algorithms. Table 2 shows the memory required by FCNF and MCF-RR to solve problems of varying number of vertices, edges, and number of customer sites.

We can observe that even a low resolution scenario of 80 × 80 points can completely occupy 64 GB of memory in FNCF. At the native resolution of our elevation data, this corresponds to only a small 2.4 km × 2.4 km region.

The large memory footprint in the CPLEX solver is due to the large system of equations for the vertices and edges in the visibility network. According to the CPLEX documentation [27] and based on extensive benchmarking experiments, we identified that the memory allocation of the MIP model is roughly proportional to the number of edges and can be approximated by the following equation:

$$\text{Memory (MB)} \approx (2 \times \#\text{ of edges} + \#\text{ of vertices}) \times K \quad (5)$$

where K is dependent on the number of threads in use by CPLEX. We experimentally determined that for our 8-threaded machine, K ≈ 0.005. In contrast, the memory MCF-RR requires less than 1% of the memory that the FCNF formulation requires in CPLEX.

6.3 Distributed Parallel Optimization

Based on our benchmarks, a natural question is whether the CPLEX memory bottleneck may be mitigated using a distributed solver. The CPLEX implementation also allows solving a MIP in a distributed environment. We setup a distributed CPLEX environment with 1 powerful master node (Intel Xeon CPU @ 3.50GHz, 64 GB RAM) and 8 worker nodes (Intel Xeon CPU @ 3.00GHz, 16 GB RAM). We conducted a number of the experiments with the goal of being able to execute scenarios that exceeded the memory bottleneck on a single machine. However, during these experiments, we found that while most scenarios were solved more quickly when distributed, the memory bottleneck remains at the master node.

6.4 Pre-processing Sampling Heuristics

As described in the previous section, the number of potential intermediate tower locations in the visibility network depends on the map resolution. The number of edges generally increases significantly with each additional vertex; in the worst case, each additional vertex, n, adds $n-1$ additional edges to the model. Each additional vertex and edge that is added to the model increases the solving time and also the memory footprint. From Equation 5, a large number of edges quickly consumes main memory in CPLEX and as we show later, also significantly increases the solving time. To reduce the model complexity we introduce a few heuristics to remove a large number of vertices (and consequently the number of edges) from the visibility graph.

If we consider scenarios where the map resolution is high and neighboring raster points are in close proximity and elevation, intuitively, many of these locations would have roughly the same visibility and would likely be unnecessarily redundant. Following this observation, we would ideally like to remove a large percentage of vertices to sparsify the visibility graph. It is important to note that unlike re-scaling, this removal process does not introduce any connectivity error to the result from the solver since the visibility between the locations are still calculated at their native resolution. However, vertex removal effectively removes potential tower locations from consideration and can drastically worsen the output if important locations are removed from consideration.

- *Random Removal:* This heuristic uniformly removes random locations from consideration.
- *Elevation-based Removal:* This heuristic takes both elevation and connectivity properties of the vertices into account. The map is first divided into buckets of contiguous points that have similar elevation (e.g. an elevation step size of 100 m). Then, for each bucket the k points that collectively have the greatest visibility across the entire map are kept, where k is defined by the desired number of points to keep.
- *Cluster-based Removal:* Similar to the elevation-based removal, this heuristic divides the map into buckets of fixed elevation intervals. To obtain a better distribution of the k selected points within each bucket, each bucket is further divided into clusters based on spatial clustering and then k points are selected from each cluster who's union has the greatest visibility to parent buckets (i.e. buckets with higher elevation). Ties are broken by the k points with the greatest visibility across the rest of the map.

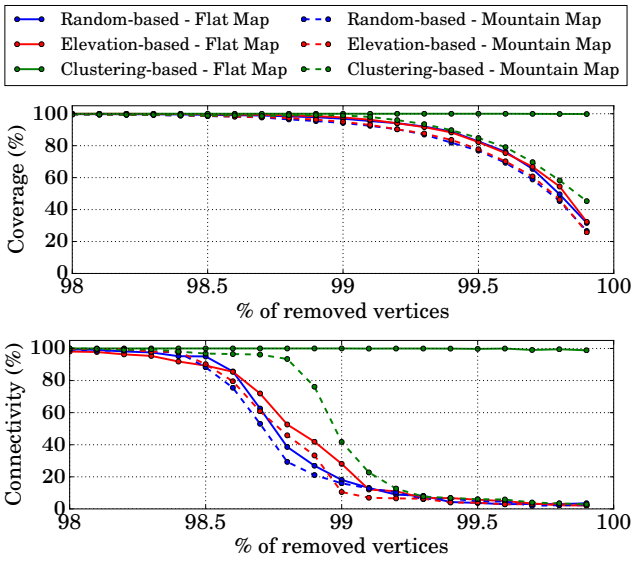


Figure 6: Coverage and Connectivity comparison of the three vertex removal techniques

We introduce two metrics to evaluate the properties of the visibility graph after applying our vertex removal heuristics:

- Coverage: The percentage of original vertex locations that can still be seen from at least one of the remaining vertices after vertex removal. This metric is meant to ensure that potential customer sites located anywhere on the map will be visible by at least one tower.
- Connectivity: The percentage of vertices after vertex removal that are reachable from every other vertex. This metric captures the idea that it is possible to connect all vertices being considered into a single network.

Figure 6 shows the coverage and vertex connectivity of the three vertex removal techniques on the same 8 km by 8 km low and high elevation variance regions we used in our previous experiment. Since the map size is very small compared to the transmission range of state-of-the-art equipment, we artificially limit the maximum edge length of all edges to 300 m. We observe from both figures that the coverage and connectivity does not significantly degrade until 98% of the vertices are removed. When 99% of the vertices are removed, the coverage is still above 95%, but the connectivity significantly decreases. This means, that up to 98% of the vertices can be safely removed without degrading the overall connectivity properties of the visibility graph or introducing any artificial lack of coverage or connectivity. Note that these two metrics do not account for potential increases in final network *cost* due to highly desirable site locations being removed from consideration.

Map Resolution (# of Vertices)	Edges	Customer Sites	Solving Time (FCNF)	Solving Time (MCF-RR)
20 × 20 (400)	76,519	20	10 s	9 s
		80	17 s	77 s
40 × 40 (1,600)	1,125,897	20	305 s	162 s
		320	663 s	880 s
60 × 60 (3,600)	5,637,153	20	2,205 s	690 s
		720	9,696 s	8,577 s
80 × 80 (6,400)	17,955,115	20	NA	2,552 s
		1,280	NA	>86,400 s

Table 3: Solving time of FCNF and MCF-RR

7 SOLVING TIME

Other than memory, the other major performance bottleneck of solving FCNF using CPLEX is its long running time. We conduct several microbenchmarks using the same set of small problems from the previous section to compare the solving time of FCNF and MCF-RR. For each scenario, we also vary the number of customers.

Table 3 shows a comparison of the solving time for FCNF as compared to MCF-RR. We observe that both algorithms take longer to solve larger problems. The number of vertices under consideration, visibility edges, and customer sites all contribute to the size of the problem. Overall, as the problem size increases, FCNF grows more quickly than MCF-RR. In the 6,400 vertex scenario, FCNF was unable to run due to insufficient memory (see Table 2). We also note that the solving time of MCF-RR is significantly larger for models with more customer sites because all possibilities of redistributing flow must be considered in phase two of our algorithm. This is a potential area for future improvement.

7.1 FCNF: Feasible Solutions

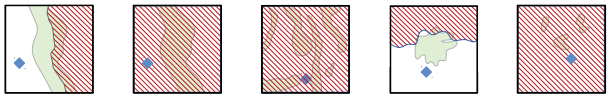
Experimentally, we observed that the CPLEX solver often finds a feasible solution that is close to the optimal solution after a few iterations. CPLEX tries to prove the optimality of the feasible solution, but this takes much longer than finding a feasible solution. Through a wide range of experiments we also found that in 80% of the cases, a feasible solution that equals the optimal solution was found after four iterations and in 99% of the cases after five iterations – typically in scenarios with a small number of customer sites. Simply accepting a feasible solution after five iterations considerably reduces the solving time without a substantial loss in the optimality of the result. The time it takes for CPLEX to produce good feasible solutions varies depending on problem size from a few minutes to over 24 hours.

8 EVALUATION

We evaluated MCF-RR on a variety of synthetic and real-world scenarios and compared our results against FCNF and man-made network designs where possible. Table 4 summarizes the properties of these seven scenarios. For comparison we include estimates of the number of resulting edges and the memory required for the native elevation data resolution. In these evaluations we use the equipment in Table 1.

Scenario	Map Size (sqkm)	Map Resolution (# of Vertices)	Estimated Edges	Customer Sites
Map A	400	0.58M	$1.7*10^{11}$	30
Map B	400	0.58M	$1.7*10^{11}$	30
Map C	400	0.58M	$1.7*10^{11}$	30
Map D	400	0.58M	$1.7*10^{11}$	30
Map E	400	0.58M	$1.7*10^{11}$	30
Medium	2,530	2.8M	$3.9*10^{12}$	18
Large	43,000	47.7M	$1.1*10^{15}$	1,200

Table 4: Estimated scenario dimensions and required memory at the native map resolution



(a) Map A **(b) Map B** **(c) Map C** **(d) Map D** **(e) Map E**

Figure 7: Pictorial sketches of the five synthetic scenarios. The green shapes represent mountainous areas, the red shaded areas contain the customer sites, and the blue diamond is the source.

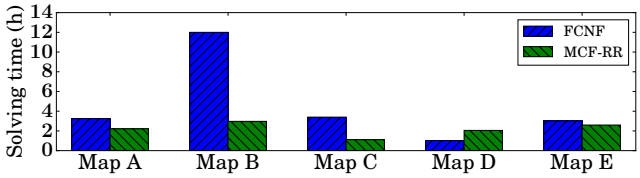
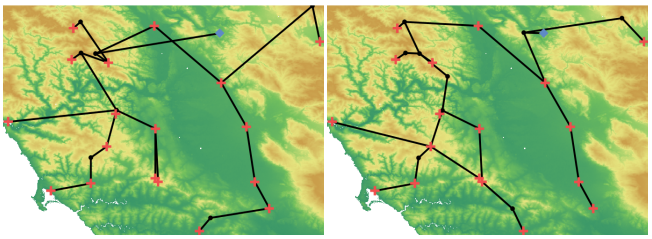


Figure 8: Solving times where FCNF produce feasible solutions whose costs are within 5% of MCF-RR

8.1 Synthetic Scenarios

We created five 20 km × 20 km synthetic scenarios with distinct topographic features to produce what we considered interesting planning requirements. Figure 7 includes pictorial representations of the five synthetic scenarios: Map A contains of a ridge that separates the source from the customer sites; Map B has the same terrain as Map A, but the source and customer sites are randomly distributed across the map; Map C is a mountainous map with high elevation variation (similar to the map described in Section 6); Map D includes a mountain that rises 1.7 km in the center of the map between the source and customer sites; Map E is a relatively flat map that contains multiple smaller terrain features. Each scenario consists of one source and 30 customer sites each with 5 Mbps bandwidth demand. For each scenario we apply the cluster-based vertex removal heuristic in the pre-processing phase to remove 99.5% of the vertices.

Figure 8 compares the solving times of FCNF and MCF-RR for the five synthetic scenarios. Since FCNF could not produce an optimal solution after 12 hours, we show the solving times of FCNF when it finds a feasible solution that has a cost within 5% of the MCF-RR solution. Except scenario D, we observe that MCF-RR solves the problems significantly faster than FCNF. In scenario B, FCNF was not able to find a feasible solution with similar cost to MCF-RR after 12 hours.



(a) FCNF (12 h, $3,230) (b) MCF-RR (10 min, $3,414)

Figure 9: Medium real-world scenario solutions

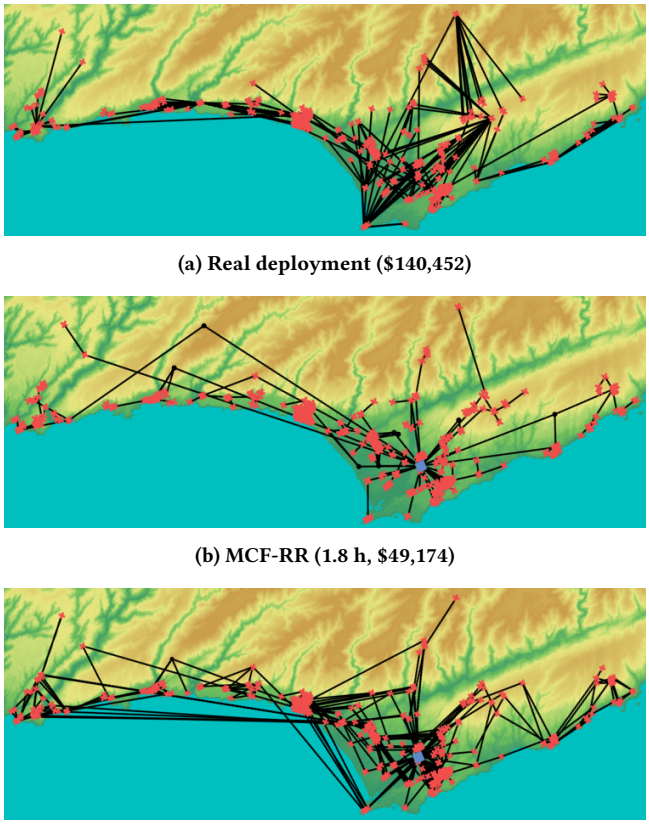
8.2 Real-world Scenarios

Our two real-world scenarios are regions in rural California where actual WISP networks have been deployed. We have the customer site locations for these two scenarios. For the Medium scenario we do not have the actual network topology or overall cost available so we can only compare MCF-RR with FCNF. The Large scenario is from the WISP network that we deployed as described in Section 2, for which we have the network topology as well as the overall hardware cost.

8.2.1 Medium Scenario. Figure 9 shows the solutions of FCNF and MCF-RR for the Medium scenario that only consists of 18 customer sites and we applied the cluster-based vertex removal with 99.95%. Both algorithms produce very similar hardware cost, but MCF-RR finds a solution in under 1.3% of the FCNF's optimal solving time (we stopped the FCNF solver after 12 hours). However, FCNF does produce a feasible solution with 5% of the MCF-RR solution after around 20 minutes.

8.2.2 Large Scenario. Using the subscriber data from our real deployment, we extracted the location and bandwidth requirements of the customers. Due to the large map size and the large number of customers, we applied our cluster-based vertex removal heuristic with 99.7% and also a spatial clustering of customer locations to replace groups of customers (within 500 m clusters) with a single site. The bandwidth demand at each of these sites was assigned the aggregate demand of their constituent customer sites. After clustering, we obtained 509 customer sites and we added one source site to the scenario.

Figure 10 shows the solutions produced by FCNF and MCF-RR after 12 hours and 1.8 hours, respectively. The total hardware cost of the network produced by FCNF is $46,430 and the network produced by MCF-RR is $49,174. Each of these costs are nearly 3x smaller than the hardware cost of $140,452 that we calculated from the dataset of the real deployment. As with the Medium scenario, FCNF produces a feasible solution within 5% of the MCF-RR solution after 3.5 hours. However, our results here come with a few caveats. First, the total hardware cost does not include hardware that would be required to connect the individual customer locations within each of the clustered customer locations. Second, the real network has multiple source locations where we considered only a single source. Third, the real network contains overprovisioned network links that increase the overall cost. Despite these caveats, it is clear that Zyxt using MCF-RR is able to produce comparably low-cost network designs in only a couple of hours.



(a) Real deployment ($140,452)

(b) MCF-RR (1.8 h, $49,174)

(c) FCNF (12 h, $46,430)

Figure 10: Large real-world scenario solutions

9 CONCLUSIONS

Much of the focus on rural Internet access has been on establishing cheap connectivity. As commodity hardware costs continue to fall, planning and designing frontier networks is the next major challenge toward universal access. The planning problem is incredibly difficult for would-be operators due to the combination of high problem complexity and the lack of automated tools. In this paper, we described the difficulties that operators face and presented our approach, Zyxt, to the problem of WISP network planning. We show through a synthetic and real-world scenarios that not only is Zyxt able to solve large scale network planning problems more quickly and easily than manual planning, but that our MCF-RR algorithm also scales better and produces good solutions faster than an off-the-shelf solver. Zyxt is an open source project [38].

REFERENCES

[1] Victor Agababov, Michael Buettner, Victor Chudnovsky, Mark Cogan, Ben Greenstein, Shane McDaniel, Michael Piatek, Colin Scott, Matt Welsh, and Bolian Yin. 2015. Flywheel: Google's Data Compression Proxy for the Mobile Web. In *Proceedings of the 12th USENIX Conference on Networked Systems Design and Implementation (NSDI '15)*. Berkeley, CA, USA.

[2] Edoardo Amaldi, Antonio Capone, and Federico Malucelli. 2003. Planning UMTS base station location: Optimization models with power control and algorithms. *IEEE Transactions on wireless Communications* 2, 5 (2003), 939–952.

[3] Abhinav Anand, Veljko Pejovic, Elizabeth M Belding, and David L Johnson. 2012. VillageCell: cost effective cellular connectivity in rural areas. In *Proceedings of ACM ICTD*.

[4] Jeffrey G Andrews, François Baccelli, and Radha Krishna Ganti. 2011. A tractable approach to coverage and rate in cellular networks. *IEEE Transactions on Communications* 59, 11 (2011), 3122–3134.

[5] aquila 2018. Facebook Aquila. https://en.wikipedia.org/wiki/Facebook_Aquila. (2018).

[6] G. Bernardi, M. K. Marina, F. Talamona, and D. Rykovanov. 2011. IncrEase: A tool for incremental planning of rural fixed Broadband Wireless Access networks. In *2011 IEEE GLOBECOM Workshops (GC Wkshps)*.

[7] Mehrab Bin Morshed, Michaelanne Dye, Syed Ishtiaque Ahmed, and Neha Kumar. 2017. When the Internet Goes Down in Bangladesh. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing (CSCW '17)*. ACM, New York, NY, USA, 1591–1604.

[8] Sandro Bosio, Antonio Capone, and Matteo Cesana. 2007. Radio planning of wireless local area networks. *IEEE/ACM Transactions on Networking* 15, 6 (2007), 1414–1427.

[9] E. Brewer, M. Demmer, B. Du, M. Ho, M. Kam, S. Nedevschi, J. Pal, R. Patra, S. Surana, and K. Fall. 2005. The case for technology in developing regions. *IEEE Computer* 38, 6 (2005).

[10] Jay Chen and Lakshmi Subramanian. 2013. Interactive Web Caching for Slow or Intermittent Networks. In *Proceedings of the 4th Annual Symposium on Computing for Development (ACM DEV '13)*. ACM, New York, NY, USA, Article 5.

[11] Jay Chen, Lakshmi Subramanian, Janardhan Iyengar, and Bryan Ford. 2014. TAQ: enhancing fairness and performance predictability in small packet regimes. In *Proceedings of ACM EuroSys*.

[12] Jay Chen, Lakshminarayanan Subramanian, and Jinyang Li. 2009. RuralCafe: web search in the rural developing world. In *Proceedings of the 18th international conference on World wide web*. ACM, 411–420.

[13] Marshini Chetty, Srikanth Sundaresan, Sachit Muckaden, Nick Feamster, and Enrico Calandro. 2013. Measuring broadband performance in South Africa. In *Proceedings of ACM DEV*.

[14] Earth Resources Observation and Science (EROS) Center. 2013. U.S. Releases Enhanced Shuttle Land Elevation Data. Available at http://eros.usgs.gov/. (2013). Accessed: 2018-04-21.

[15] Google Developers. 2018. Google Optimization Tools (OR-Tools). Available at http://developers.google.com/optimization/. (2018). Accessed: 2018-04-21.

[16] GRASS Development Team. 2017. *Geographic Resources Analysis Support System (GRASS GIS) Software, Version 7.2*. Open Source Geospatial Foundation. http://grass.osgeo.org

[17] G. M. Guisewite and P. M. Pardalos. 1990. Minimum concave-cost network flow problems: Applications, complexity, and algorithms. *Annals of Operations Research* 25, 1 (01 Dec 1990), 75–99.

[18] Shaddi Hasan, Yahel Ben-David, Max Bittman, and Barath Raghavan. 2015. The Challenges of Scaling WISPs. In *Proceedings of ACM DEV*.

[19] Kurtis Heimerl, Kashif Ali, Joshua Evan Blumenstock, Brian Gawalt, and Eric A Brewer. 2013. Expanding Rural Cellular Networks with Virtual Coverage.. In *Proceedings of USENIX/ACM NSDI*.

[20] Kurtis Heimerl and Eric Brewer. 2010. The Village Base Station. In *Proceedings of the 4th ACM Workshop on Networked Systems for Developing Regions (NSDR '10)*. ACM, New York, NY, USA, Article 14, 2 pages.

[21] Kurtis Heimerl, Anuvind Menon, Shaddi Hasan, Kashif Ali, Eric Brewer, and Tapan Parikh. 2015. Analysis of Smartphone Adoption and Usage in a Rural Community Cellular Network. In *Proceedings of the Seventh International Conference on Information and Communication Technologies and Development (ICTD '15)*. ACM, New York, NY, USA, Article 40, 4 pages.

[22] heywhatsthat.com 2017. Heywhatsthat. http://heywhatsthat.com/. (2017).

[23] Michael Holzhauser, Sven O. Krumke, and Clemens Thielen. 2016. Budget-constrained minimum cost flows. *Journal of Combinatorial Optimization* 31, 4 (01 May 2016), 1720–1745.

[24] Zhuochuan Huang, Chien-Chung Shen, Chavalit Srisathapornphat, and Chaiporn Jaikaeo. 2002. Topology control for ad hoc networks with directional antennas. In *Computer Communications and Networks, 2002. Proceedings. Eleventh International Conference on*. IEEE, 16–21.

[25] Stephen Hurley, Stuart Allen, Desmond Ryan, and Richard Taplin. 2010. Modelling and Planning Fixed Wireless Networks. *Wireless Networks* 16, 3 (April 2010), 577–592.

[26] IBM. *IBM ILOG CPLEX Optimization Studio V12.8*. IBM.

[27] IBM. 2018. Guidelines for estimating CPLEX memory requirements based on problem size. Available at http://www-01.ibm.com/support/docview.wss?uid=swg21399933. (2018). Accessed: 2018-02-27.

[28] David L Johnson, Elizabeth M Belding, Kevin Almeroth, and Gertjan van Stam. 2010. Internet usage and performance analysis of a rural wireless network in Macha, Zambia. In *Proceedings of ACM DEV*.

[29] Teemu Koponen, Scott Shenker, Hari Balakrishnan, Nick Feamster, Igor Ganichev, Ali Ghodsi, P Godfrey, Nick McKeown, Guru Parulkar, Barath Raghavan, and others. 2011. Architecting for innovation. *ACM SIGCOMM Computer Communication Review* 41, 3 (2011).

[30] loon 2018. Google Loon. https://x.company/loon/. (2018).

[31] mikrotik 2018. MikroTik. http://www.mikrotik.com/. (2018).

[32] Shridhar Mubaraq Mishra, John Hwang, Dick Filippini, Tom Du, Reza Moazzami, and Lakshminarayanan Subramanian. 2005. Economic Analysis of Networking Technologies for Rural Developing Regions. www.news.cs.nyu.edu/~lakshmi/Pubs/EconomicAnalysisofNetworkingTechnologiesforRuralDevelopingRegions.pdf. In *1st Workshop on Internet and Network Economics, Dec 2005*.

[33] Nvidia. 2018. CUDA Parallel Computing. Available at http://developer.nvidia.com/cuda-zone. (2018). Accessed: 2018-02-28.

[34] operamini 2018. Opera Mini. http://www.opera.com/mobile/. (2018).

[35] Dimitris C Paraskevopoulos, Tolga Bektaş, Teodor Gabriel Crainic, and Chris N Potts. 2016. A cycle-based evolutionary algorithm for the fixed-charge capacitated multi-commodity network design problem. *European Journal of Operational Research* 253, 2 (2016), 265–279.

[36] Rabin K Patra, Sergiu Nedevschi, Sonesh Surana, Anmol Sheth, Lakshminarayanan Subramanian, and Eric A Brewer. 2007. WiLDNet: Design and Implementation of High Performance WiFi Based Long Distance Networks.. In *Proceedings of USENIX/ACM NSDI*.

[37] Chunyi Peng, Haitao Zheng, and Ben Y Zhao. 2006. Utilization and fairness in spectrum assignment for opportunistic spectrum access. *Mobile Networks and Applications* 11, 4 (2006), 555–576.

[38] Thomas Pötsch, Salman Yousaf, and Jay Chen. 2018. Zyxt - WISP Planning Tool. https://github.com/thp-comnets/zyxt. (2018).

[39] Prabhakar Raghavan and Clark D. Tompson. 1987. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica* 7, 4 (01 Dec 1987), 365–374.

[40] Bhaskaran Raman and Kameswari Chebrolu. 2005. Design and evaluation of a new MAC protocol for long-distance 802.11 mesh networks.

[41] Ram Ramanathan and Regina Rosales-Hain. 2000. Topology control of multihop wireless networks using transmit power adjustment. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, Vol. 2. IEEE, 404–413.

[42] Ali Raza, Yasir Zaki, Thomas Pötsch, Jay Chen, and Lakshmi Subramanian. 2017. xCache: Rethinking Edge Caching for Developing Regions. In *Proceedings of the Ninth International Conference on Information and Communication Technologies and Development (ICTD '17)*. ACM, New York, NY, USA, Article 5, 11 pages.

[43] Carlos Rey-Moreno, Zukile Roro, William D. Tucker, Masbulele Jay Siya, Nicola J. Bidwell, and Javier Simo-Reigadas. 2013. Experiences, Challenges and Lessons from Rolling out a Rural WiFi Mesh Network. In *Proceedings of the 3rd ACM Symposium on Computing for Development (ACM DEV '13)*. ACM, New York, NY, USA, Article 11, 10 pages.

[44] Rijurekha Sen, Hasnain Ali Pirzada, Amreesh Phokeer, Zaid Ahmed Farooq, Satadal Sengupta, David Choffnes, and Krishna P. Gummadi. 2016. On the Free Bridge Across the Digital Divide: Assessing the Quality of Facebook's Free Basics Service. In *Proceedings of the 2016 ACM on Internet Measurement Conference (IMC '16)*. ACM, New York, NY, USA.

[45] Sayandeep Sen and Bhaskaran Raman. 2007. Long distance wireless mesh network planning: problem formulation and solution. In *Proceedings of the 16th international conference on World Wide Web*. ACM, 893–902.

[46] Lakshminarayan Subramanian, Sonesh Surana, Rabin Patra, Sergiu Nedevschi, Melissa Ho, Eric Brewer, and Anmol Sheth. 2006. Rethinking wireless for the developing world. In *Proceedings of ACM SIGCOMM HotNets*.

[47] Srikanth Sundaresan, Nazanin Magharei, Nick Feamster, and Renata Teixeira. 2012. Accelerating Last-mile Web Performance with Popularity-based Prefetching. In *Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM '12)*. ACM, New York, NY, USA.

[48] Sonesh Surana, Rabin K Patra, Sergiu Nedevschi, Manuel Ramos, Lakshminarayanan Subramanian, Yahel Ben-David, and Eric A Brewer. 2008. Beyond Pilots: Keeping Rural Wireless Networks Alive.. In *Proceedings of USENIX/ACM NSDI*.

[49] Sonesh Surana, Rabin K Patra, Sergiu Nedevschi, Manuel Ramos, Lakshminarayanan Subramanian, Yahel Ben-David, and Eric A Brewer. 2008. Beyond Pilots: Keeping Rural Wireless Networks Alive. In *Proceedings of USENIX/ACM NSDI*.

[50] swiftfox 2016. Swiftfox. http://www.swiftfox.net/. (2016).

[51] towerdb 2017. Tower DB. https://github.com/inveneo/poundcake. (2017).

[52] ubiquiti 2018. Ubiquiti Networks. http://www.ubnt.com/. (2018).

[53] You-Chiun Wang, Chun-Chi Hu, and Yu-Chee Tseng. 2008. Efficient placement and dispatch of sensors in a wireless sensor network. *Mobile Computing, IEEE Transactions on* 7, 2 (2008), 262–274.

[54] wisptools 2018. WISPTools. http://wisptools.net/. (2018).

[55] Yunnan Wu, Philip A Chou, Qian Zhang, Kamal Jain, Wenwu Zhu, and Sun-Yuan Kung. 2005. Network planning in wireless ad hoc networks: a cross-layer approach. *IEEE Journal on Selected Areas in Communications* 23, 1 (2005), 136–150.

[56] Yasir Zaki, Jay Chen, Thomas Pötsch, Talal Ahmad, and Lakshminarayanan Subramanian. 2014. Dissecting Web Latency in Ghana. In *Proceedings of the ACM Internet Measurement Conference (IMC '14)*. Vancouver, BC, Canada.