Scalability and Satisfiability of Quality-of-Information in Wireless Networks

Scott T. Rager* Ertugrul N. Ciftcioglu† Ram Ramanathan‡ Thomas F. La Porta* Ramesh Govindan§
*The Pennsylvania State University, University Park, PA 16802
†IBM Research, Yorktown Heights, NY 10598
‡Raytheon BBN Technologies, Cambridge, MA 02138
§University of Southern California, Los Angeles, CA 90089
Email: rager@psu.edu, enciftci@us.ibm.com , ramanath@bbn.com, tlp@cse.psu.edu, ramesh@usc.edu

***Abstract*—Quality of Information (QoI) provides a context-dependent measure of the utility that a network delivers to its users by incorporating non-traditional information attributes. Quickly and easily predicting performance and limitations of a network using QoI metrics is a valuable tool for network design. Even more useful is an understanding of how network components like topology, bandwidth, protocols, etc. impact these limitations. In this paper, we develop a QoI-based framework that can provide accurate estimates for limitations on network size and achievable QoI requirements, focusing on completeness and timeliness. We extend this framework to model competing flows and data loads as random variables to capture the stochastic nature of real networks. We show that our framework can provide a characterization of delays for satisfied queries to further analyze performance when some late arrivals are acceptable. Analysis shows that large tradeoffs exist between network parameters, such as QoI requirements, topology, and network size. Simulation results also provide evidence that the developed framework can estimate network limits and delays with high accuracy. Finally, this work also introduces *scalably feasible QoI regions*, which provide upper bounds on QoI requirements that can be supported for certain network applications.**

I. Introduction

Traditional approaches to studying network scalability and performance limitations have been focused on finding theoretical limits on throughput and delay. In many applications, however, the relationship between these metrics and the effectiveness of the network is highly non-linear. Therefore, having a framework to evaluate network scalability with respect to achievable *Quality of Information* (QoI) requirements is necessary.

Additionally, while theoretical, asymptotic analysis of individual network topologies, protocols, etc. is important, such analysis lacks the ability to quickly obtain an accurate estimate of a projected network's abilities when these individual components are pieced together. Often, researchers must turn to extensive simulation or experimentation testbeds to test proposed network setups. While the increasing availability of open source simulation frameworks like ns-3 [2] and experimental testbeds, such as IoT-LAB [3], Indriya [4], and Wisebed [5], have reduced the barriers to run simulations and experiments, significant time to familiarize oneself with the platform and implement the desired scenario is required. Physical testbeds also require time-sharing and have a limit on the maximum number of nodes that can be tested. Finally, once a scenario is implemented, testing a range of values for more than one or two independent variables will be time and computing intensive because the number of trials that must be run can grow quite large.

Research was sponsored by the U.S. Army Research Laboratory under the Network Science Collaborative Technology Alliance, Agreement Number W911NF-09-2-0053.

This work was presented in part at IEEE Wireless Communications and Networking Conference (WCNC) 2016, April 2016 [1].

As an example of a scenario in which we are interested, imagine being given the task of deploying a wireless sensor network for a particular application. Given a proposed network with a defined size, topology, parameters, and protocols, what is the level of QoI requirements it can support? Now, consider the converse: Given a certain QoI that is desired by users of a network, what is the maximum number of nodes that the network can support? Which has a bigger impact on this scalability: the imposed information requirements or the strict timeliness requirements?

Our main contribution in this paper is a novel framework that can predict scalability and performance of a network with respect to QoI requirements for answering such questions. We explain this framework in detail in Section V and provide example applications in Section VI. As a second contribution we extend this framework in Section VII, capturing the stochastic nature of query sources and destinations as well as data requirements, and show that it can be used to characterize query delays. In both cases, we provide results from realistic implementations in the ns-3 network simulation environment.

We also present several pieces of supporting work. First we provide an example of an application that relies on QoI to highlight the difference in QoI and traditional metrics in Section IV. We show in Section VIII that our framework is also quite useful in quickly and easily understanding the impact of parameters and design choices, providing a secondary benefit to network designers of allowing them to compare networks and identify tradeoffs. Finally, we show how the framework can also provide bounds on QoI capacity in some applications in Section IX.

II. Related Work

The scalability model derived in this work is inspired by the symptotic scalability framework outlined in [6], which has been previously applied to content-agnostic static networks [7] and mobile networks [8]. We provide several differences and additional analysis here compared to these works, which mainly stem from our use of QoI requirements instead of static data rates. The first difference is that we are able to evaluate performance and scalability under timeliness constraints, which is not possible under any of these prior works. Second, we illustrate the effects on an application's performance, which is not linear with respect to data rates in most cases. Additionally, we provide a formulation that includes parameters characterized by random variables. This improved modeling allows us to characterize expected delays with a probability distribution, which the previous works do not provide.

Other works characterize the capacity of wireless networks, like [9], [10], but all do so by considering how networks scale asymptotically or by analyzing specific network instances instead of developing a general model. Experimental techniques, like Response Surface Methodology [11], discrete-event simulators [2], and real wireless network testbeds [3]–[5] may be applied to solve the problem we do, and, in fact, after finalizing a network design, implementing one of these methods to further verify capabilities is desirable. The anticipated applications of this work, however, are different in that either there is not enough time to develop a realistic simulation or implementation, or the goal is to evaluate a number of combinations of design choices, such that implementing each possible combination and/or running trials over the sets of independent parameters would be too time-intensive.

A large number of works provide definitions for QoI and frameworks that utilize it. We will address only the most relevant ones here. Primarily, QoI has been considered from a number of angles including routing [12], scheduling/rate control [13], [14], and impact on usage of network resources [15]. Our focus is on a broader scale here, though, modeling scalability and limitations of an entire network.

A framework called Operational Information Content Capacity (OICC) is outlined in [16], which describes the obtainable region of QoI, a notion similar to the *scalably feasible QoI region* developed here. OICC differs, though, in the fact that it does not provide any method for determining the possible size of the network or impact of network design choices like medium access protocols. We also note that a notion similar to QoI satisfiablility was considered in [17] which addresses resource allocation for long-term average QoI outage satisfaction. However, the focus of [17] is energy-efficient scheduling and power allocation in a single-hop three-node network rather than scalability.

In Section IV, we use similarity-based image collection as an example of an application that is best evaluated using QoI. This application has previously been considered in [18] and [19]. Our scope is greater than that of [18], which does not consider attributes of timeliness, nor the consideration of transmission rates and network topology. We use the same similiary-based image selection algorithm as in [19], but provide new methods of quantifying QoI.

III. Network Model

We develop the framework in this paper with an aim for application in ad hoc networks or in Device-to-Device (D2D) networks [20], in which nodes can act as both clients and servers, issuing queries and serving responding with data. Additionally, nodes serve as routers, forwarding traffic for the rest of the network when it lies on a given path between a source-destination pair.

We assume that the network has an achievable channel rate, W, which is the minimum effective channel rate in the network including the effects of protocols and errors. Specifically, since wireless channels are often lossy, some method of automatic repeat request (ARQ) protocol is likely to be implemented in the data link layer of the network. As in [21], given a probability of packet loss, calculating the expected number of retransmissions for each packet is straightforward. For example, from [21], the expected number of retransmissions, $E(N)$, for a packet with a probability of loss p_l is $E(N) = \frac{1}{1-p_l}$. Using this value, we assume we can use the error-free channel rate, which we will call W^*, and derive an *effective* channel rate to use in determining expected scalability and performance:

$$W = W^* \cdot (1 - p_l) \tag{1}$$

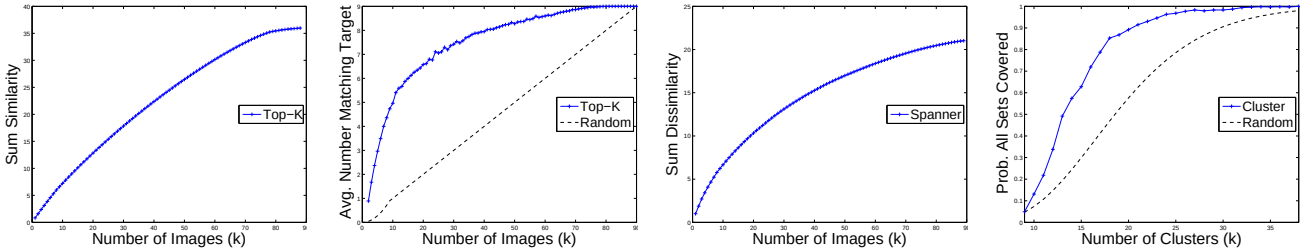
Here, an estimate of the probability of losing a packet must be known. However, as we will show, this framework creates expressions that are easy to compute, so testing a range of possible loss probabilities is also feasible if an accurate estimate is unkown.

In addition to retransmission protocols, coding techniques to detect or correct errors, such as cyclic redundancy check (CRC) codes or forward error correcting (FEC) codes, respectively, can be easily included in our framework. When these schemes are used in the network, the expected data size is simply adjusted accordingly. We discuss data sizes and elaborate on this point in Section V since they rely on QoI definitions in Section IV.

For a traffic model, we adopt a model in which each node produces queries according to a Poisson distribution with an average rate of λ and send the query to another node chosen at random. This model provides a general example of expected traffic with randomness that we can use to provide an example of applying our framework in Section VI, but the framework is in no way limited to this particular traffic scenario. As a second example, we provide a different network application in Section IX while introducing the concept of defining QoI capacity regions.

A. Defining Scalability

We determine the effective scalability of the network by two methods. First, we assume that nodes have finite queues sizes, so we say that the network is no longer scalable when the



(a) Top-K: Sum Similarity (b) Top-K: Avg. Match Target (c) Spanner: Sum Dissimilarity (d) Clustering: Cover All Sets

Fig. 1. Completeness metrics for the three image selection algorithms. Each exhibits a diminishing return as more images are added.

expected traffic rate is greater than the expected service rate, since such a scenario will cause queues to become overloaded and drop traffic. The second definition of scalability is related directly to QoI. As explained in more detail in Section IV, we adopt a timeliness constraint for all query responses. When the network is no longer capable of meeting this timeliness constraint for all queries with a probability of $(1-\epsilon)$, where ϵ can be set as close to zero as desired, then it is not considered scalable.

IV. QoI Model

QoI is a multi-dimensional metric that can be defined for an application to give a more meaningful measure of the value of information. It consists of attributes such as timeliness, freshness, completeness, accuracy, precision, etc. For example, information that contributes to a decision-making process may only be useful if it arrives before the decision must be made, or it may have varying usefulness based on how similar or dissimilar it is to other data already collected.

The specific details of which attributes are considered and how they contribute to QoI is application-dependent. Chosen QoI metrics are stored as a vector associated with a data item. Here, as in [22], we specify a vector of minimum values for each QoI metric, and information is evaluated based on whether it satisfies all of the QoI requirements or not. We use this approach to establish the edges of QoI satisfiability for the vector of metrics, which defines the boundaries of maximum achievable QoI regions in the metric space.

We choose to use two QoI attributes, one that is time-based and one that is information-content-based. The first attribute is timeliness, T, of data. For the second attribute, we present a notion of *completeness*, C, which we show can be defined multiple ways, depending on the application and context. Together, a QoI requirement of $\mathbf{q} = \{C, T\}$ specifies a quantity of data that must be delivered as well as a deadline by which it must arrive to be useful. Since completeness is a rather new concept, we explain an example image selection algorithm and show how it can be evaluated with completeness.

A. Example Application: Similarity-based Image Retrieval

As a motivating example, we choose a network in which nodes store photographs that are to be exchanged or collected at one or more data sinks. This example covers surveillance missions of military tactical networks or camera sensor networks. In this model, nodes can act as both clients and servers, issuing queries and serving images in response. Therefore, as the network size increases, the amount of traffic is also likely to increase, but possibly disproportionately. This fact exemplifies why the problem of characterizing the scalability limits for an instance is not straightforward.

We give a very brief overview of an example application that relies on QoI. Details about the algorithms and metrics are in Appendix A.

Our example application allows a user to issue queries to collect images based on scores of similarity or dissimilarity, which provide contextually valuable information. An algorithm called **Top-K** provides the k most similar images to a given target image. For example, if a user has a picture of an unknown suspicious person entering a building, but the person is not identifiable from that image, it would be useful to collect more images that are similar to that one with the possibility that another picture may have a better view of the person in question that can be used for identification or more context. We use a metric called *Sum Similarity*

In contrast, given the set of all photographs available in the network, we might want to return the set of k images that exhibits the most diversity, ideally providing a user with a good sampling, or *complete view*, of available images. For instance, such a result would be quite useful in a surveillance mission. We present two query algorithms that can be used to achieve this goal, **Spanner** and **Clustering**. The Spanner algorithm has an associated metric called *Sum Dissimilarity*, and the Clustering algorithm can be evaluated by the probability that the returned set covers all distinct locations of interest.

B. Experimental Results

To provide example values of these completeness metric definitions, experiments applying each query algorithm were run on a set of pictures taken at $n = 9$ different settings around the Penn State campus. Each of these 9 settings is of a pictorially different area, e.g. a particular building, a downtown street, or a lawn setting, and over 20 images of each was taken. Then, for individual trials, 10 images from each set were randomly selected to create an image pool of 90 pictures. The three algorithms were run over these 90 images, with the target image being randomly selected in the case of Top-K. Results for each of the different completeness metrics were averaged over $1,000$ trials are shown in Figure 1.

Figure 1(a) shows the average sum similarity of images returned by the Top-K algorithm. Figure 1(b) provides the second definition of completeness for the Top-K algorithm, the number of images matching the set that the target image was randomly chosen from. Figure 1(c) depicts the average Sum Dissimilarity returned by the Spanner algorithm, and Figure 1(d) represents the empirical probability of all 9 sets being represented in the k returned images. For reference, we also include expected values for the metrics in Figures 1(b) and 1(d) if the images were selected from the entire image pool at random, i.e., without regard for image similarity or dissimilarity. Details on these expected values given random selection are in Appendix B.

These figures exhibit the diminishing returns of completeness as more images are collected. This effect visually shows how QoI differs from throughput. As seen in these graphs, transmission of successive images does not result in a linear gain in completeness. For example, in Figure 1(b), it is evident that a value of only $k \approx 10$ is needed to collect 5 images matching the target content, while collecting an additional 2 from the same set usually requires collecting over twice that number of pictures. Similarly, Figure 1(d) shows that jumping from $k = 10$ to $k = 20$, the likelihood of capturing at least one image of every setting grows substantially from just over 10% to approximately 90%. To approach probabilities close to gaining that final 10%, however, requires a jump to $k \approx 30$.

The relationship between the number of images and completeness in each of these graphs also shows that obtaining a certain value of QoI or completeness requires a different number of images depending on the set available and their similarities. We can denote the number of images required to achieve a level of completeness, C, as $k_{req} = Q(C)$. This relationship will be useful later in determining capacity and scalability limits.

C. Further Discussion of QoI

We have defined and provided examples for a number of ways that completeness can be defined and used to obtain a concrete data requirement from a contextual QoI requirement. Throughout the rest of the paper, we use sum similarity and the probability of covering all sets using clustering as completeness metrics, but we note that any of the definitions of completeness used here, or any other QoI requirement that can be translated into a data requirement, for that matter, can be used.

V. QoI Scalability

Given the nonlinear returns of completeness and importance of timeliness outlined in the previous section, we contend that establishing limits of throughput is insufficient. A non-QoI-aware approach ignores very useful information such as how changes in achievable data rates affect the actual utility of information at its destination as well as how changes in one QoI requirement can affect satisfiability of other requirements.

For these reasons, we set the goal of determining the capacity of a network, and relatedly, the scalability achievable with respect to QoI requirements. Previous methods that predict scalability do so by singling out a single bottleneck node and determining when it will be overloaded and drop packets. Here, we are forced to look at end to end flows and determine all contributors to delay, including MAC access, queuing delays, and multi-hop propagation to find the point at which timeliness requirements are violated or traffic load must be reduced thus adversely affecting completeness, leading to a more comprehensive analysis.

A. QoI Satisfiability Framework

In order to establish the framework, we examine an arbitrary flow, F_1, in the network that has a QoI requirement of $\mathbf{q} = \{C, T\}$, where C is the minimum required completeness metric of choice, such as sum similarity as explained above, and T is the required timeliness. This flow will have a data size requirement, which is given by a chosen QoI function $Q(C)$. Using the example application from IV, for example, $Q(C)$ can return the number of images, k_{req}, required to achieve the requested completeness C. For simplicity, in this section, we use a fixed value for the data size requirement, but we expand our consideration to use a distribution for the data size requirement in our applications in Section VI.

Assuming each image has a size of I_S, then we can also use B to describe the total number of bits required by the flow, $B = k_{req} * I_S$. To match realistic network implications, we assume this data will be transmitted in a series of packets with size P_S bits each. This packet size should include any bits necessary to implement techniques of error detection or correction that the network is utilizing. The number of packets per flow, then, is simply $P_N = \lceil B/P_S \rceil$. We assume that each node in the network can transmit at W bits per second when it is allocated media access.

Our goal is to establish the limits at which this arbitrary flow can no longer be completed with the QoI requirements satisfied. We build and explain our model for achieving this goal by working through an example TDMA line network, a portion of which is shown in Figures 2 and 3. In this network, we assume a simple 3-slot TDMA scheme, which allows each node equal time access to the medium and, in this explanation, we remove any potential interference or hidden terminal issues. Recall that packet losses and retransmissions can be captured in this framework by using an *effective* channel rate based on the physical channel rate and loss probability. Also, we discuss addressing non-TDMA networks, such as those that use a CSMA protocol, in Section X.

The TDMA Slot assignments are labeled above each node in Figures 2 and 3. For simplicity, we will assume that one slot is appropriately sized to transmit a single packet, i.e., $T_{slot} = P_S/W$ and that packets use static routes calculated beforehand such that the overhead is not a consideration here.

When calculating overall delay, four sources of delay are generally considered: processing, queuing, transmission, and propagation. We focus on transmission and queuing delay here and ignore processing and propagation delay, because the latter two are negligible compared to the former two for the wireless

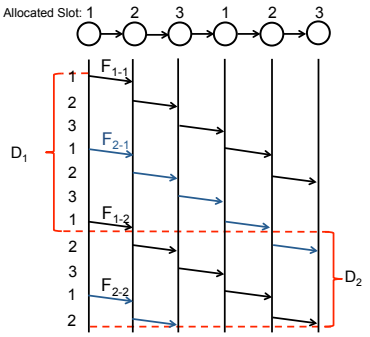


Fig. 2. Example of line network using TDMA highlighting source of delays, D_1 and D_2. (Node labels are slot assignments.) Here, $TF = 2$, $CF = 3$, and $DF = 1$.

channel rates we are targeting in this work (on the order of MBs/sec or lower). If a higher level of precision is desired, however, propagation and processing delays can be added to the time to transmit each packet, which we adopt as the time duration of a time slot, T_{slot} below, without changing the formulation of the framework. Furthermore, we do not incorporate consideration of excessive queuing delays since, as we stated in the model, our aim is to evaluate stable network traffic, which implies relatively small queue backlogs.

In developing our framework, we find it more intuitive to split the total delay into two functional parts, which we call D_1 and D_2. These two components do not separate queuing and transmission delays, but considering parts of each simultaneously. The first contributor, D_1, is the end to end delay incurred by sending the B bits across the designated path in the network. To quantify this delay, we must consider several factors beyond just the available bandwidth and number of packets in a flow. First, each node must share the channel with its one-hop neighbors, creating a factor of delay inversely proportional to the percentage of time access that it is allocated for a channel. We call this factor of delay the Channel Factor, which in this case is given by $CF = N_{frame}/N_i$, where N_i is the number of slots allocated to node i and N_{frame} is the total number of slots in each frame. In this case, $N_i = 1, \forall i$ and $N_{frame} = 3$.

The second factor to consider for D_1 is the fraction of allocated slots that are utilized by the node to serve flows other than F_1 that are either originating in or being routed through nodes on the path of flow F_1. We call the total number of flows competing at node i the Traffic Factor, TF_i, of that node. For any flow, the maximum contributor to delay is the node along the path with the maximum TF_i, which we will just call TF here. More detail about determining this TF is outlined in Appendix C. Incorporating these considerations into a calculation for D_1, we achieve the following expression:

$$D_1 = T_{slot} \cdot P_N \cdot CF \cdot TF \tag{2}$$

Figure 2 depicts the delay of D_1 in a simple case of only two flows, F_1 and F_2, being present, in which case $TF = 2$. In this example we use flows that consist of only 2 (F_{i-j} is packet j in flow i) packets to portray the delay. In most

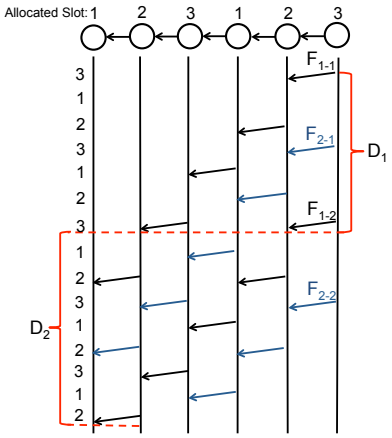


Fig. 3. In the opposite direction, $DF = 2$. (TF and CF are unchanged here.)

real applications, P_N will be much larger, making D_1 a good approximation of this delay component.

The second delay that exists is due to the packets travesing multi-hop paths. This delay is simply the time for a single packet to traverse the path length. Note that this delay is not necessarily just the path length multiplied by T_{slot}, because of possible channel contention and/or ordering constraints. We show here how ordering constraints impact this TDMA network. A node cannot forward a packet from the flow until it receives that packet from the previous hop. In our line network example, when the direction of the flow matches the nodes' schedule of slots $1-2-3-1-2-3$, as in Figure 2, each successive node receives a packet on the time slot before it is scheduled, resulting in no extra delay. For a flow in the opposite direction, though, where nodes are scheduled $3-2-1-3-2-1$, as in Figure 3, the first slot 1 is not utilized, because the first node scheduled in time slot 1 has not yet received a packet in the flow. Every other slot is wasted, on average, for the initialization of the flow, resulting in approximately twice the delay. We will use a term that we call the *Delay Factor*, or DF, to account for this effect where it exists. The multi-hop queuing delay, then, is modeled by

$$D_2 = T_{slot} \cdot DF \cdot (PL - 1) \tag{3}$$

where PL is the average path length.

We note several points about this delay factor. First, in a loaded network, the nodes can and will serve other flows while awaiting the arrival of packets in this flow of focus. That utilized bandwidth does not, however, preclude this DF impact on delay for the flow of interest, F_1. Any node cannot serve F_1 until a packet from that flow has been received. Second, this delay is only accounted for once per flow because all other packets are pipelined. All other packets' delay is captured by the end to end delay, D_1. This effect is best illustrated by examining the difference between D_2 in Figures 2 and 3. Here, we see the multi-hop propagation requires twice the number of slots because every other slot is unused in F_1's propagation. In a TDMA model, the average delay factor is

	CF	μ_{TF}	σ_{TF}	DF	PL_{max}
Clique	$N-1$	1	0	1	1
Star	$N-1$	$N-2$	$\sqrt{N-2}$	$\frac{N}{2}$	2
Line	3	$\frac{N-1}{2}$	$\sqrt{\frac{N-1}{2}}$	1.5	$N-1$
Grid	5	$\sqrt{N}$	$N^{\frac{1}{4}}$	2.5	$2\cdot\sqrt{N}$
NSFNET	4	2.15	1.47	2	4

TABLE I
CF, TF, DF, AND PL VALUES FOR EXAMPLE TOPOLOGIES

	Equation
Clique	$W\cdot T - I_S\cdot k_{req}\cdot(N-1)\geq 0$
Star	$W\cdot T-(N-1)\cdot I_S\cdot k_{req}\cdot(1+\frac{-\ln(\epsilon)}{2(N-1)(N-2)}\pm\sqrt{\frac{\ln(\epsilon)^2}{4(N-1)^2(N-2)^2}-2\frac{\ln(\epsilon)}{(N-1)(N-2)}})\cdot((N-1)(N-2))-P_S\cdot N\geq 0$
Line	$W\cdot T-3\cdot I_S\cdot k_{req}\cdot(1+\frac{-\ln(\epsilon)}{N-1}\pm\sqrt{\frac{\ln(\epsilon)^2}{(N-1)^2}-4\frac{\ln(\epsilon)}{N-1}})\cdot\frac{N-1}{2}-P_S\cdot 1.5\cdot(N-1)\geq 0$
Grid	$W\cdot T-5\cdot I_S\cdot k_{req}\cdot(1+\frac{-\ln(\epsilon)}{2\sqrt{N}}\pm\sqrt{\frac{\ln(\epsilon)^2}{4\cdot N}-2\frac{\ln(\epsilon)}{\sqrt{N}}})\cdot\sqrt{N}-P_S\cdot 2.5\cdot(2\cdot\sqrt{N}-1)\geq 0$
NSFNET	$W\cdot T-4\cdot I_S\cdot k_{req}\cdot(1+(\frac{-\ln(\epsilon)}{2\cdot 2.15}\pm\sqrt{\frac{\ln(\epsilon)^2}{4\cdot 2.15^2}-2\frac{\ln(\epsilon)}{2.15}})\cdot 1.47)-P_S\cdot 2\cdot 4\geq 0$

TABLE II
SCALABILITY EQUATIONS

the average of all possible waiting times, which is equal to the number of slots in the schedule divided by 2.

By adding the two components of delay, we can give a relation for a network that will successfully achieve an average flow's data and timeliness requirements:

$$T_{slot}\cdot P_N\cdot CF\cdot TF + T_{slot}\cdot DF\cdot(PL-1)\leq T$$

Recalling that the time of a slot is determined by $T_{slot} = P_S/W$ and that the total number of bits required for a flow $P_S\cdot P_N = k_{req}\cdot I_S$ (where k_{req} is given by a function of required QoI), we can get the following expression:

$$W\cdot T - k_{req}\cdot I_S\cdot CF\cdot TF - P_S\cdot DF\cdot(PL-1)\geq 0 \quad (4)$$

Every network will have its own set of parameters that can be substituted into this general formula, providing a tool to approximate limitations and sensitivity to changes in specific parameters. To provide concrete examples of how the framework is used, next we give derived values of factors for a TDMA-based wireless network with a number of different topologies.

VI. EXAMPLE OF APPLYING FRAMEWORK

We illustrate the application of Equation (4) using an N-node TDMA network with five different topologies: clique, star (a topology common in D2D networks), line, grid (also known as a mesh), and the NSFNET network [23]. Discussion of other network control protocols and topologies are addressed in Section X. We adopt a traffic model that uses Top-K queries as an example application. We assume that all nodes have a set of collected images that are used to respond to Top-K queries. Each node produces queries according to a Poisson process with exponential interarrival times with parameter λ, each with a target image and target QoI, $\mathbf{q} = \{C, T\}$, describing the required completeness (here, we use sum similarity) and timeliness, and sends it to another node chosen at random. Here, the actual size of the response fulfilling the query can be given by a distribution that incorporates the number of images required, k_{req}, and the size of each image. The details of this distribution can be given from empirical observations of experiments like those in Section IV.

A. Scalability Equations

We outline details on how to derive an expression for the Traffic Factor (TF) of a network in Appendix C. The result from that process is a mean and standard deviation of a distribution for the traffic factor. In the case of uniform network structures that are scalable, these expressions depend on the network size N. For fixed-size networks with a given topology, such as the NSFNET network in Figure 9 (also in Appendix C), the distribution is given with values calculated using the topology and traffic pattern.

Once an expression for the Traffic Factor (TF) is derived, we can use it along with expressions for the Channel Factor (CF), Delay Factor (DF), and Path Length (PL) for the network to create specific instances of Equation (4) that estimate the scalability and QoI-satisfiability limits of the particular network of interest. Since our goal is to determine the point at which the system is unable to support the offered traffic load within the timeliness constraints, we use maximum values for these factors where applicable, specifically TF_{max} and PL_{max}.

The PL_{max} is usually quickly determined by an examination of the topology, such as $PL_{max} = N-1$ for a line network and $PL_{max} = 2\cdot\sqrt{N}$ for a grid network. In some cases, such as the example in Section IX, a direct value of TF_{max} may be clear from the topology and application. When traffic is stochastic, though, as it is in this case, we can utilize the distribution of TF_b to capture the expected maximum value of TF with a probability of $(1-\epsilon)$. If we use a normal distribution for TF, then $TF_{max} = \mu_{TF} + \eta\cdot\sigma_{TF}$ where η is chosen to satisfy the desired value of ϵ. This notion is analogous to the z-score of a standard normal distribution. As an example, using $\eta = 3.5$ captures approximately 99.7% of the maximum of the TF distribution, providing an estimate within $\epsilon = 0.03$ of the maximum probability. When the Poisson distribution is used to approximate TF, we can use a

Chernoff bound to provide the TF_{max} expression. Here, we use the following multiplicative Chernoff bound (proof of this bound is in Appendix D): Assuming $\eta > 1$,

$$P(TF \geq (1+\eta)\mu_{TF}) \leq e^{\frac{-\eta^2}{2+\eta}\mu_{TF}} \tag{5}$$

Setting the RHS side of (5) equal to ϵ and solving for η, we get the following expression, which should have real, non-negative roots for small ϵ values:

$$\eta = \frac{-\ln(\epsilon)}{2\mu_{TF}} \pm \sqrt{\frac{\ln(\epsilon)^2}{4\mu_{TF}^2} - \frac{2\ln(\epsilon)}{\mu_{TF}}} \tag{6}$$

Table I shows expressions for factors representing general clique, star, line, and grid networks as well as the NSFNET network as derived in Appendix C and in [6]. Then, substituting the factors into equation 4, we achieve the scalability equations for each topology in Table II.

To find the limitation of a particular parameter or QoI component, the scalability equation can be solved for the variable of interest. Then all known values can be substituted to get the limit of the variable of interest. For example, given a network size and completeness requirement, we can determine a clique network's minimum sustainable timeliness with the equation $T \geq \frac{I_S \cdot k_{req} \cdot (N-1)}{W}$, where k_{req} is given by the completeness function $Q(C)$. In practice, solutions for these equations will most likely need to be made numerically, but doing so is rather straightforward using any number of commonly available tools and is much faster and simpler than developing and running separate simulations to determine scalability. Additionally, as we will show in Section VIII, these equations can also be easily used to determine the impact of other network parameters on this timeliness limit.

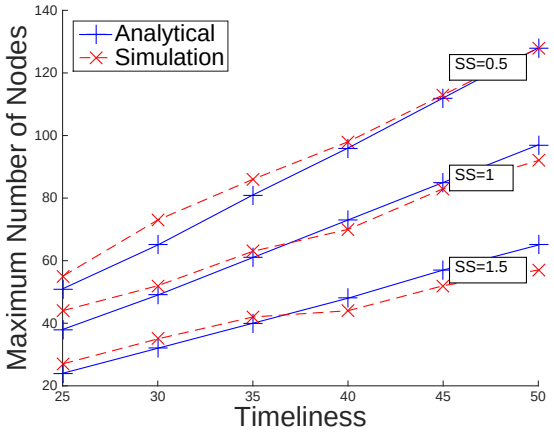
B. Minimum Timeliness/Maximum Query Rate

Solving the Scalability equations for T reveals the minimum satisfiable timeliness value for which all queries are expected to complete within the deadline constraint, which we will call T_{min}. Consequently, this minimum satisfiable timeliness value also correlates to the maximum traffic rate that can be served by the network, $\lambda_{max} = \frac{1}{T_{min}}$. If the average rate of queries, λ, is greater than $\frac{1}{T_{min}}$, then the traffic will exceed the network capacity and the number of active queries in the system will grow without bound, causing packets to be dropped and/or delays to grow without bound. Therefore, the maximum query rate per node is $\lambda_{max} = \frac{1}{T_{min}}$, and, consequently, the minimum timeliness for which *all* flows can be expected to complete before the deadline is T_{min}.

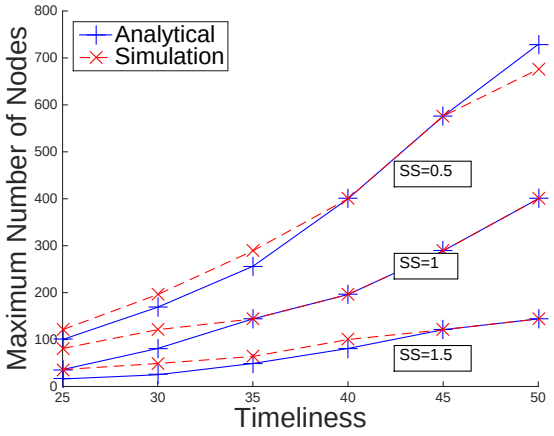
In some applications, having a certain amount of queries not complete by the timeliness requirement may be acceptable. As we show in Section VII, we can develop a more detailed characterization of the delay equation than the Scalability Equations here for these applications.

C. Validation of Scalability Equations

To show how effective estimates using this framework can be, we simulated two of the network topologies and traffic



(a) Line Network, $I_S = 18KB$



(b) Grid Network, $I_S = 48KB$

Fig. 4. Empirical results match analytical results closely for all tests.

described above in Section VI in the ns3 network simulator [2], comparing empirical results to those generated analytically with this framework, labeled *Analytical*. The results of these simulations are shown in Figure 4. We use a channel rate of $W = 2Mbps$, packet sizes of $P_s = 1500$ bytes, and image sizes of 18 and 48 Kbytes. As above, the correlation between Sum Similarity and k_{req} is taken from the actual observed relation in Figure 1(a). All values of parameters (SS, T, I_S, etc.) were chosen to test a variety of network sizes and QoI requirements while remaining within realistic network sizes, both with respect to real-world deployments and simulations with feasible run-times.

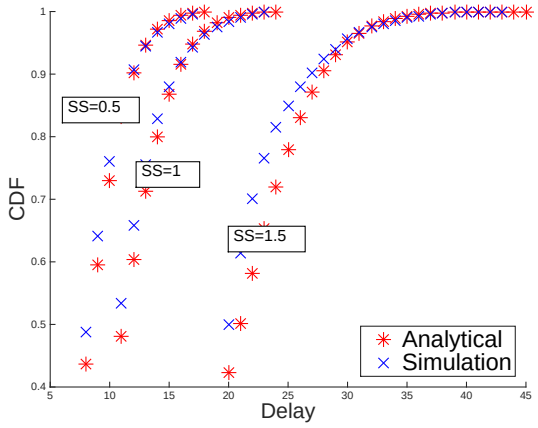
VII. Finding Limits and Characterizing Delay

As explained in Section IV, delay of a flow can be expressed as

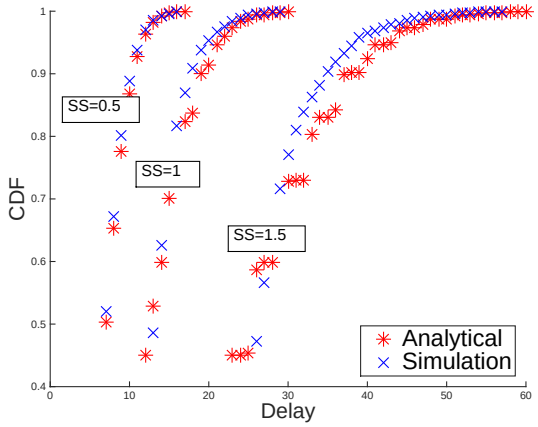
$$D = \frac{k_{req} \cdot I_S \cdot CF \cdot TF}{W} + \frac{P_S \cdot DF \cdot (PL-1)}{W} \tag{7}$$

Let $PL()$ be a function that provides the path length between i and j, and let TF_i^j be a random variable of the Traffic Factor for the bottleneck node between i and j, i.e. the node along

$$F_D(d) = \frac{1}{N} \cdot \sum_{i=1}^{N} \sum_{j \neq i} \sum_{tf=1}^{tf_{max}} F_{P_N}(\frac{d - C_2 \cdot PL(i,j)}{C_1 \cdot p_N}) \cdot f_{TF_{i|j}}(tf) \cdot p(j) \quad (10)$$



(a) Line Network, $N = 40$, $I_S = 36KB$



(b) Grid Network, $N = 49$, $I_S = 72KB$

Fig. 5. Characterization of delay using framework follows distribution of empirical results in most cases.

the path from i to j with the highest u_x. Finally, let P_N be a random variable that describes the number of packets in a given request, capturing both the possible randomness of k_{req} and I_S. Then, building on the equation for delay above and making some substitutions, we can get the following equation to describe the delay from a node i given a destination of j:

$$D_i^j = \frac{P_S \cdot CF \cdot P_N \cdot TF_i^j}{W} + \frac{P_S \cdot DF \cdot (PL(i,j) - 1)}{W} \quad (8)$$

Defining two constants to simplify the expression,

$$C_1 = \frac{P_S \cdot CF}{W}$$
$$C_2 = \frac{P_S \cdot DF}{W}$$

we can express the delay as

$$D_i^j = C_1 \cdot P_N \cdot TF_i^j + C_2 \cdot PL(i,j) \quad (9)$$

We can develop an expression for a distribution of delay as follows. First, we define the cumulative distribution of a source-destination pair (i, j):

$$P(D_i^j \leq d) = P(C_1 \cdot P_N \cdot TF_i^j + C_2 \cdot PL(i,j) \leq d)$$
$$= P(P_N \cdot TF_i^j \leq \frac{d - C_2 \cdot PL(i,j)}{C_1})$$

Next, conditioning over all possible values of TF, we get

$$P(D_i^j \leq d) =$$
$$\sum_{\tau=1}^{\tau_{max}} P(P_N \cdot TF \leq \frac{d - C_2 \cdot PL(i,j)}{C_1} | TF = \tau) \cdot f_{TF_i^j}(\tau)$$
$$= \sum_{\tau=1}^{\tau_{max}} P(P_N \leq \frac{d - C_2 \cdot PL(i,j)}{C_1 \cdot \tau}) \cdot f_{TF_i^j}(\tau)$$

Substituting the cumulative distribution representing the data load, F_{P_N}:

$$F_{D_i^j}(d) = \sum_{\tau=1}^{\tau_{max}} F_{P_N}(\frac{d - C_2 \cdot PL(i,j)}{C_1 \cdot \tau}) \cdot f_{TF_i^j}(\tau)$$

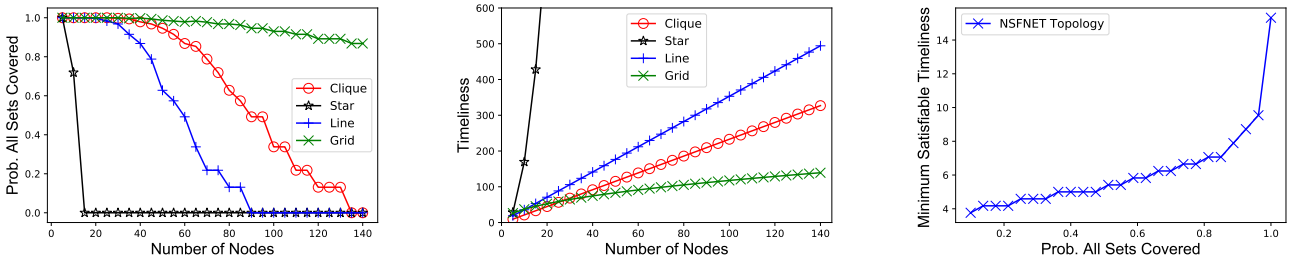
Then, we can generalize the expression to give a distribution for a flow originating in node i with an unknown destination by conditioning over all possible destinations, j.

$$F_{D_i} = \sum_{j \neq i} [\sum_{\tau=1}^{\tau_{max}} F_{P_N}(\frac{d - C_2 \cdot PL(i,j)}{C_1 \cdot \tau}) \cdot f_{TF_i^j}(\tau)] \cdot p(j)$$

Finally, we can get an average distribution of all flows' delays by summing over all sources and dividing by the the number of sources. This average delay distribution is in Equation (10).

A. Probability of Timeliness Satisfiability

While the minimum timeliness at which all flows are expected to complete before their deadlines can be determined by the scalability equations in Section VI, some applications may benefit from an understanding of what the probability of completing within the timeliness constraints for those below the minimum fully satisfiable timeliness. For example, if a mission issues a number of queries for information to support decision-making, receiving 80% or 90% of the responses may be sufficient for making the decision. The question of importance, then, is "How far can we reduce the timeliness constraint and still expect to receive x% of the queries in time?" Or, equivalently, we may pose the question, "When the network is operating at the edge of capacity, what is the expected delay for x% of queries to be completed?" Since equation 10 provides the distribution of delays, it provides quality estimates to answer these questions.



(a) Prob. All Sets Covered vs. N (Timeliness = 50) (b) N vs. Timeliness (Prob. All Sets Covered = 0.8) (c) Prob. All Sets Covered vs. Timeliness (N = 14)

Fig. 6. One can quickly and easily find QoI limits and how these limits scale as network size increases for scalable networks (a) and (b). For a fixed-size network, the framework shows limits and tradeoffs in competing QoI requirements, such as in for the NSFNET topology in (c).

B. Validation of Delay Characterization

Figure 5 shows expected delay distributions from 10 alongside distributions of delays recorded in ns3 simulations of the same networks. We argue that minimum QoI requirements for most applications tend to be over 50%, and therefore focus on the top half of the delay distribution. In all cases here, analytical predictions of satisfying the timeliness requirement are within about 10% of empirical results for probabilities above 0.5.

These delay distributions also provide much more information about the expected delays of all queries in addition to the minimum satisfiable timeliness. For each completeness requirement of a minimum expected Sum Similarity, the maximum delay at the top of the distribution provides the minimum satisfiable timeliness. Examining the distributions, though, we see that many of the queries finish well before that maximum delay. For example, focusing on the delay distribution when the Sum Similarity requirement is 1.5 in Figure 5(a), we note that the maximum delay is approximately 45 seconds, but 80% of the queries on average finish in almost half of that delay.

VIII. Impact on Network Design

Now that we have established a model for QoI satisfiability and scalability and shown its accuracy in predicting estimated limits, we show how it can also provide quick, but useful intuition about the impact of various network parameters. Once an equation for a network's limitations is formulated as was done for equations in Table II, it can be solved for a single parameter to gain an understanding of its reliance on the other factors. This analysis can be done very quickly and easily by approximating when appropriate, or by using symbolic and/or numerical solvers.

To exemplify this application in network design, we have applied this process to the equations in Table II using the probability that all settings of interest will be represented, or covered, by the set of images returned by the clustering algorithm explained in Section IV as the completeness metric. Recall that the highly non-linear relationship between the number of images collected and probability of covering all sets, of which there are 9 in our experimental setup, is given in the empirical results of Figure 1(d). We present several figures that provide insight into network limitations and tradeoffs.

Figure 6(a) visualizes the maximum expected probability that the images returned will create a complete cover of locations for a query given a fixed timeliness constraint of $T = 50$ seconds. While it may be intuitive that a grid network's ability to deliver valuable content does not degrade as much as the star, line, and clique networks, what is not intuitive is *how much* of a difference we should expect. Here, we see that as the star, line, and clique networks become completely overloaded, the grid network can scale up to over 140 nodes while supporting flows with a 90% probability of delivering a complete cover.

If given a topology and application with predetermined requirements of $\mathbf{q}$, we may be interested in how large our network can grow before its capacity to deliver the desired QoI per flow is no longer possible. An example of an answer to this question is in Figure 6(b). This graphs shows the major impact timeliness requirements can have on maximum network size.

Finally, for networks of a fixed size, we can use the scalability equations to determine limits and tradeoffs in the other parameters in the network, especially various QoI requirements. Figure 6(c) shows the timeliness constraints that can be satisfied for a desired probability of collecting images that cover all areas of interest. This detailed understanding of the relationship between competing QoI requirements is not only of great utility to a network designer, but would also be of use to applications looking to optimize tradeoffs without wasting precious resources while trying to learn how these parameters affect each other.

IX. Scalably Feasible QoI Regions

In some applications, the data requirements to achieve a certain level of QoI may include a minimum number of nodes or sources in addition to the size of the data load that must be served. For example, a system may require that each image must come from a different node, perhaps because each node only has one image or maybe by design to provide increased credibility through corroboration, another contextual metric of interest in QoI-aware networks. We will use this example to illustrate an interesting observation that arises in such scenarios.

Let us assume that we are designing a surveillance network with nodes arranged in a grid and one data sink in the corner

of the network. This data sink node issues queries with a set of QoI requirements that include completeness and timeliness, $\mathbf{q} = \{C, T\}$, but this completeness must be met while only allowing one image per source node. With this traffic model, we can derive a maximum traffic factor and create a scalability equation for this particular scenario.

Here, the data sink node will have two neighbors through which all queries will be forwarded, so clearly we can choose either of these to consider as the bottleneck node. Even if the network balances traffic evenly over both of these nodes, each must forward at least $\frac{N-1}{2} - 1$ since there are $N-1$ sources, half of which must be forwarded through each of the data sink's two neighbors, excluding the 1 image from each of these neighbors themselves. If we assume these queries may be simultaneously served in the worst case, then we can simply use this expression for the maximum Traffic Factor. Then, using the same values derived for a grid network in Table I in place of the other factors in Equation (4), we get the following scalability equation:

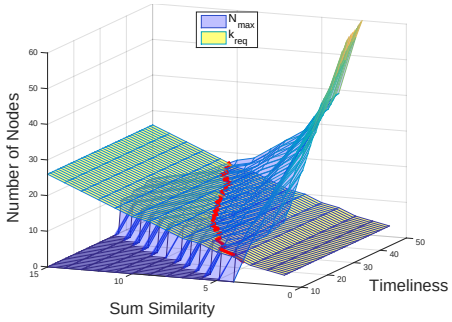
$$W \cdot T - 5 \cdot I_s \cdot (\frac{N-1}{2} - 1) - 2.5 \cdot P_s \cdot 2\sqrt{N} \geq 0 \quad (11)$$

From this equation, we can determine the maximum network size, which we will call N_{max} here. However, the specified restriction that each node can contribute only one image to k_{req} implies a minimum network size of $N \geq k_{req}$ in order to achieve the completeness outlined in the QoI requirements.

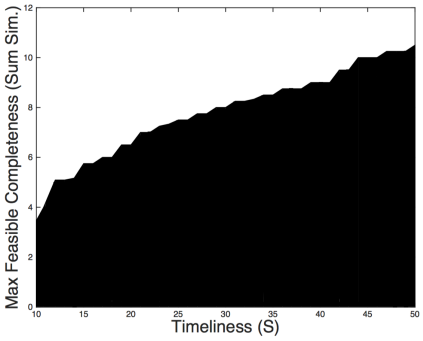
These two facts are important, because when $N_{max} < k_{req}$, then it is not possible to provide the QoI level $\mathbf{q}$. Hence, this set of QoI requirements is infeasible, or *scalably infeasible*. This phenomenon defines the concept of a *Scalably Feasible QoI Region*, which refers to the region in which sets of QoI pairs can be supported with the given network signature.

Figure 7(a) provides a visual representation of this region for a grid network that institutes the given traffic model. Here, N_{max}, calculated for QoI requirements of Sum Similarity over the range $[1, 15]$ and Timeliness over the range $[10, 50]$, is shown in the graph by the blue surface. On the same graph is the number of images required, k_{req}, for each Sum Similarity requirement, shown with the yellow surface on the graph. The intersection of these two surfaces, displayed with a red line, provides the edge of the scalably feasible QoI region. In this example, only QoI pairs to the right of this line, i.e. the region where $N_{max} > k_{req}$, are scalably feasible. Figure 7(b) shows this same scalably feasible QoI region in two dimensions, projecting the region onto the x-y plane. In this figure, the shaded region represents all valid pairs of completeness and timeliness requirements.

In general, regardless of how many images a node can contribute to a query response, it is possible to analyze the trade-off between different QoI attributes for a fixed value of maximum network size, N_{max}. Specifically, by fixing N in Table II, one can obtain T and k_{req} (and, hence, C) resulting in the set of supportable $\{C, T\}$ pairs defining a feasible region



(a) The curved plane represents maximum scalability, and the flat plane represents minimum required images. Therefore, all (Sum Sim., Timeliness) pairs to the right of the intersecting red line are within the scalably feasible QoI region.



(b) The shaded region represents feasible values of QoI (Completeness, Timeliness pairs) - a 2-d projection from Figure 7(a)

Fig. 7. Graphical representations of the *Scalably Feasible QoI Region*

for QoI. This region can be visualized by intersecting the blue scalability curved plane with a flat surface fixed at N_{max} instead of the yellow surface in Figure 7. The scenario given in this section goes one step further and also takes into account the network size actually required to generate/support a given QoI requirement (using the non-flat yellow surface derived from experimental results in Figure 7(a)).

X. Discussion

We note that although TDMA is primarily used in this work, the same approach can be taken to derive QoI-based relations in networks that use other MAC layer protocols. In these cases, the appropriate Delay Factor would need to be derived for each protocol. To examine an 802.11 network, for example, the DF would capture queuing delays and could be determined by extending a delay model such as in [24], and the CF could be estimated through studies of channel utilization like [25].

Similarly, although we only address the regular topologies of clique, line, and grid networks here, we believe the framework can be applied to more complex, irregular network topologies. In fact, some of the simple models presented here may already be quite useful for some of the topologies addressed. As shown in [26], for example, a dense random network may be closely approximated by a clique network, or

random networks' capabilities can be approximately bounded by the limits of clique and grid networks, since these networks can be viewed of as examples of dense and sparse random networks, respectively.

XI. Conclusion

This work provides several contributions to the field of QoI-aware wireless networks. First, we motivated the use of completeness and timeliness as QoI attributes, providing an example application and several different ways to measure completeness. Next, we developed a framework that can be used to predict QoI and network size limits for a specific network as well as predict expected probabilities of satisfying timeliness constraints beyond the point in which all queries are satisfied. We validated these models' accuracy by comparing analytical results with simulations performed in the ns3 network simulator in both cases. Examples of the impact of different network parameters were shown, providing concrete examples of the framework's usefulness in real-world applications. In addition, the concept of scalably feasible QoI regions was introduced. For future work, we plan to make generalizations of factors that will allow for easy application of this framework to any non-regular network topology and expand this framework to include consideration of more complex network control actions, such as caching and/or data compression or fusion, which are all of interest in QoI-aware networking.

Appendix A
Details on Image Retrieval Algorithms

To satisfy completeness of a query, we utilize measurements of the similarity or dissimilarity between pairs of images as explained in the rest of this section. To get a similarity measurement, we use the same choice as was shown to be effective in [19]. A technique called Color and Edge Directivity Descriptor (CEDD) [27] provides a 54-byte vector of qualities inherent to a photograph like lightness, contrast, and color. The similarity between two images can then be given as a scalar by calculating the *Tanimoto Similarity* [28] between their CEDD vectors. Dissimilarity is simply defined as 1 minus the similarity.

A. Selecting Similar Images

The first type of query we introduce occurs when a user already has one image of a particular area or object of interest and would like to obtain similar images to get a more complete view of that specific scene or object. In this scenario, we use the **Top-K** algorithm for this application, which returns the k images with the most similarity with respect to the target image.

We can evaluate the completeness of the result in one of two ways. First, we can use the similarity of the images as a value representing each image's effectiveness in providing a more complete view of the target scene. If we sum the similarity of all k images returned by the algorithm, we get a representation of completeness, which we naturally call *Sum Similarity*. While this measure of completeness is abstract, it can be refined in an actual implementation through testing and evaluating. This definition of completeness is useful, though, because it can be applied without any predetermined knowledge of the environment or pool of images.

Often, though, we can partition the environment in which the network operates into a number, n, of distinct settings or areas. In those cases, we can utilize a second method of quantifying completeness. Assume that each image belongs to one of these n sets, related to the setting it depicts. Naturally, then, when executing a Top-K query, the goal is for the algorithm to return images from the same set as the target image. Completeness can then be given by the fraction of images returned that are in the same set as the target image.

B. Selecting Diverse Images

One query that provides diverse images is known as the **Spanner** of the set of known photographs. For the Spanner algorithm, we employ a greedy algorithm similar to that in [19]. Here, the algorithm simply chooses images that provide the greatest minimum distance from all images already chosen. This minimum distance can be added to a running sum to provide a completeness metric of *Sum Dissimilarity*. This value represents a measure of completeness because a higher level of dissimilarity provides a more complete view of the feature space.

The other query that can achieve a complete view over all images is **Clustering**. In the Clustering algorithm, all images are separated into k clusters based on their pairwise distances using any version of a k-means clustering algorithm, where k is given by the user. Then, the most central image from each cluster is returned. Here, assuming that the photographs of the same settings or objects of interest exhibit similar characteristics, Clustering also provides a complete view of the network's environment.

Both Spanner and Clustering algorithms can also be evaluated using a model assuming the environment is split into n sets. With this model, we can define completeness as either the number of sets represented by at least one of the k images returned or the probability of all n sets being represented by at least one image when k are returned. Here, though, we only show results for the second definition.

Appendix B
Explanation of Expected QoI with Random Image Selection

A. Top-K

First, we explain the expected number of images that are from the same set as the target image in the Top-K algorithm when images are selected from the entire image pool at random. We define the following:

- n = total number of images (summed over all sets)
- S = number of sets
- S_k = set of target image
- k = number of images selected

- N_S = number of images in each set (for simplicity, assumed to be the same for all sets)
- x = number of images returned from set S_k

$$P(X=x|k) = \begin{cases} \frac{\binom{k}{x} * \binom{n-k}{k-x}}{\binom{n}{k}} & \text{if } k \leq N_S, \\ \frac{\binom{N_S}{x} * \binom{n-N_S}{k-x}}{\binom{n}{k}} & \text{if } N_S < k \leq n - N_S \end{cases} \tag{12}$$

Equation 12 provides the probability that x of the k selected images will be from the target set. When $k \leq N_S$, the total number of possible combinations of choosing x from the target set and $k-x$ from the $n - N_{S_k}$ remaining images over the entire sample space (n choose k). When $N_S < k < n - N_S$, then we consider the possible combinations of choosing x images from the target set and $k-x$ images from the remaining $n - N_S$ images. This probability formula can then be used to derive the expected values of x displayed in Figure 1(b).

B. Clustering

For Clustering, we want to determine the probability that we will cover each of the S sets with at least one of the k chosen images if we had chosen them randomly. We will call X_i the random variable that represents the number of images from set i in the results. We use the following expression:

$$P(X_i > 0, \forall i|k) = (1 - P(X_i = 0))^S \tag{13}$$

where X_i is given by a multivariate hypergeometric distribution, which gives us the following:

$$P(X_i = 0|k) = \frac{\binom{n-N_s}{k}}{\binom{n}{k}} \tag{14}$$

This probability expression is plotted directly against the percentage of trials in which all sets were covered in experiments using the Clustering algorithm in Figure 1(d).

Appendix C
Derivation of Traffic Factor for Example Topologies

The Traffic Factor will depend on the traffic pattern and routing scheme used in the network, but here we outline a general formula for modeling it with a random variable based on the traffic model described in Section III. For this derivation, we assume that the network is operating in a steady state with the query rate λ that is arbitrarily close to the the maximum satisfiable traffic rate since we are estimating the upper-most scalability limits and characterizing delay for when the network is operating at its maximum capacity.

Let $S_{(i,j)}$ be the set of nodes that make up the shortest path between node i and j. For a node x, let $\rho(x)$ be the number of shortest paths in which x is present. Now we assume that at a given time, the probability of node i currently sending data to node j is given by p_f. We will let χ_{ij} be a Bernoulli trial that represents the existence of this flow, i.e., χ_{ij} will equal 1 with probability p_f and 0 otherwise. Then, the traffic factor of node x, TF_x, would be the sum of χ_{ij} for all pairs (i,j) in which $x \in P_{ij}$. Since p_f is i.i.d. for all (i,j) pairs such that $i \neq j$, then TF_x is Binomial random variable with $n = \rho(x)$ and $p = p_f$.

Since working with the binomial distribution can be tedious, we note that for most cases TF_x can be approximated. In some cases, namely when p_f does not approach 0 or 1 as n increases, the de Moivre-Laplace theorem states that the normal distribution can be used as an approximation. When p_f does approach 0 and n is sufficiently large, then the Poisson distribution is a more appropriate approximation as we show in examples in Appendix C.

With a goal of determining maximum scalability and QoI-satisfiability limits, we focus on the bottleneck node, b, which is the node that results in the largest mean μ_{TF_x} value. For topologies with regular patterns like the line and grid networks, finding the bottleneck node and expressions to capture the expected traffic through it is intuitive. When handling non-uniform topologies, like the NSFNET topology, we use some straightforward processing on the graph to determine the bottleneck and appropriate values for the traffic factor.

A. Clique Network Traffic Factor

In a clique network, all nodes have a direct link to every other node, so no node is required to forward other nodes' traffic. Therefore, we can simply set $\mu_{TF} = 1$ and $\sigma_{TF} = 0$.

B. Star Network Traffic Factor

In a star network, also called a hub-spoke network, one node acts as the central hub or access point for the network. Naturally, this central hub node is the bottleneck node since all communication flows through it. Since there are $N-1$ non-hub nodes and each of those have $N-2$ possible non-hub node destinations, the number of paths that go through the hub node is $\rho(b) = (N-1)(N-2)$. Then, since there are N flows and $N*(N-1)$ total paths in the network, we can approximate the probability of each path containing a flow as $p_f = \frac{1}{N-1}$. Therefore, p approaches zero as the network size, N, and thus, n, increases, so we use a Poisson approximation for TF_b, giving the following distribution:

$$f_{TF_b}(t) = e^{-(N-2)} \frac{(N-1)^t}{t!}$$

C. Line Network Traffic Factor

For a line network, first, let us look at the number of paths that go through the bottleneck node, which is the center node for a line network. We will assume that N is odd here, for simplicity of notation, but the logic is the same for even values of N. Since there are $\frac{N-1}{2}$ nodes on each side the center node, the total number of paths that go through it is

$$\rho(b) = 2(\frac{N-1}{2})^2$$

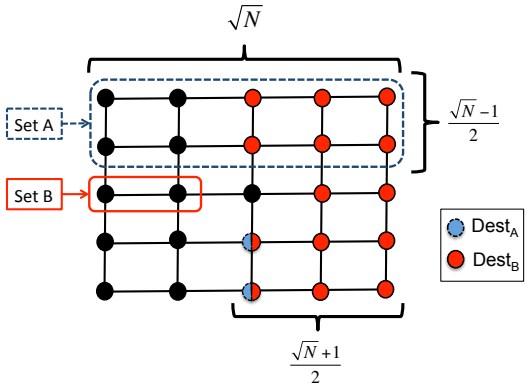


Fig. 8. Sources and destinations used in proving TF for grid networks

Here, p_f is the same as in the star network, $p_f = \frac{1}{N-1}$. Then, TF_x will be a Binomial random variable with $n = \rho(b)$ and $p = \frac{1}{N-1}$, and the Poisson approximation would be:

$$f_{TF_b}(t) = e^{-(\frac{N-1}{2})}\frac{(\frac{N-1}{2})^t}{t!}$$

D. Grid Network Traffic Factor

Again, the bottleneck node, i.e., the node with the highest number of paths going through it is the center node, and we give the derivation for when $\sqrt{N}$ is odd, but the logic follows similarly for even values. As proved in [29], the most optimal routing scheme for maximum capacity is "Row-First, Column-Second" routing, so we assume paths follow this approach. Again, we adopt a traffic pattern in which each node is the source of exactly one flow and that the destination is uniformly chosen from all other $N-1$ nodes. For each source node, we can determine the number of destinations that route through the center. We separate nodes into two categories for this counting.

First, we consider the nodes circled in set A in Figure 8, of which there are $\sqrt{N} \cdot \frac{\sqrt{N}-1}{2}$. Through manual inspection, one can deduce that the only destination nodes in the figure that result in a path that is relayed by the center node are the two bottom nodes in the center column in the figure, marked with blue. There are $\frac{\sqrt{N}-1}{2}$ of these destination nodes for the nodes in set A, so the total number of paths from the nodes in set A is $\sqrt{N} \cdot (\frac{\sqrt{N}-1}{2})^2$. Now, if we also consider the reverse, i.e. imagine the figure rotated vertically, then we can give the total number of paths from nodes not in the same row as the center node as $2 \cdot \sqrt{N} \cdot (\frac{\sqrt{N}-1}{2})^2$.

Next, we consider the nodes in the same row as the center node, which we call set B. Here, all destinations on the "opposite" side of the center as well as those in the same column of the center require being routed through the center node when originating from any nodes in set B. Just as above, we can count the number of paths from the nodes in set B that route through the center and double it to count the reverse. The resulting number of paths is $2 \cdot (\sqrt{N} \cdot \frac{\sqrt{N}+1}{2} - 1) \cdot (\frac{\sqrt{N}-1}{2})$.

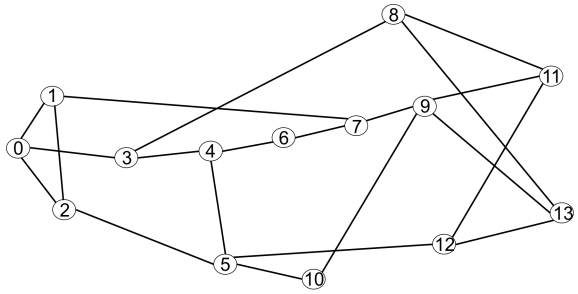


Fig. 9. Topology of NSFNET network. Node 5 is on the highest number of shortest paths between all pairs, so it is the bottleneck node.

Adding together these paths and simplifying gives us the following expression for the total number of paths that go through the center node:

$$\rho(b) = \sqrt{N} \cdot (N-2) + 1 \tag{15}$$

Just as with line networks, the probability of each path containing a flow is $p_f = \frac{1}{N-1}$, so the traffic factor for the center node of a grid network is approximated with a Poisson distribution:

$$f_{TF_b}(t) = e^{-(\frac{\sqrt{N}\cdot(N-2)+1}{N-1})}\frac{(\frac{\sqrt{N}\cdot(N-2)+1}{N-1})^t}{t!}$$

which can be approximated by the following for large values of N.

$$f_{TF_b}(t) = e^{-(\sqrt{N})}\frac{(\sqrt{N})^t}{t!}$$

E. NSF Network

We can also determine the traffic factor for an arbitrary network like the NSFNET topology [23] shown in Figure 9. In cases such as this, the bottleneck may not be immediately evident, but finding it is simply a matter of using any standard shortest path algorithm to find the number of shortest paths on which a node x is present, $\rho(x)$, for all nodes. Then, the bottleneck node is the node with the maximum $\rho()$ value: $b = \arg\max_x \rho(x)$. In the NSFNET topology, node 5 is the bottleneck with $\rho(5) = 28$. As above, the probability of each path currently being used to serve a flow is $p_f = \frac{1}{N-1} = \frac{1}{13} = 0.077$. Since this probability is close to zero, the Poisson approximation with parameter $\rho(b) * p_f = 28 * 0.077 \approx 2.15$ is appropriate for this traffic factor distribution:

$$f_{TF_b}(t) = e^{-2.15}\frac{2.15^t}{t!}$$

Appendix D
Proof of Multiplicative Chernoff Bound

The authors in [30] prove a different bound for $0 < \eta \leq 1$. We follow the same logic to prove the following bound for $\eta \geq 1$.

Theorem 1: Let X be the sum of independent Poisson trials and $\mu = \mathbb{E}[X]$. The following bound holds for $\eta \geq 1$:

$$P(TF \geq (1+\eta)\mu_{TF}) \leq e^{\frac{-\eta^2}{2+\eta}\mu_{TF}} \tag{16}$$

Proof: First, we reiterate the following relative Chernoff Bound given in [30]: For any $\eta > 0$,

$$P(X \geq (1+\eta)\mu) < (\frac{e^{\eta}}{(1+\eta)^{(1+\eta)}})^{\mu} \tag{17}$$

Using (17), to prove (16), we must show that the following holds for $\eta \geq 1$:

$$\frac{e^{\eta}}{(1+\eta)^{(1+\eta)}} \leq e^{\frac{-\eta^2}{2+\eta}} \tag{18}$$

Taking the natural logarithm and rearranging, we get

$$f(\eta) = \eta - (1+\eta)\ln(1+\eta) + \frac{\eta^2}{2+\eta} \leq 0 \tag{19}$$

Then, the first and second derivates of $f(\eta)$ with respect to η are

$$f'(\eta) = -\ln(1+\eta) + \frac{\eta(\eta+4)}{(\eta+2)^2} \tag{20}$$

$$f''(\eta) = -\frac{1}{1+\eta} + \frac{8}{(\eta+2)^3} \tag{21}$$

Note that $f(1) = -0.053$. Additionally, for all $\eta \geq 1$, $f'(\eta) < 0$ and $f''(\eta) < 0$. Therefore, $f(\eta)$ is decreasing and concave down for $\eta \geq 1$. This fact, along with the fact that $f(1) < 0$, implies that $f(\eta)$ stays negative for $\eta \geq 1$. Hence, (18) is satisfied, completing our proof. ■

REFERENCES

[1] E. N. Ciftcioglu S. Rager, R. Ramanathan, T. F. La Porta, and R. Govindan. Scalability and satisfiability of quality-of-information in wireless networks. In *Proc. IEEE WCNC*, Doha, Qatar, April 4-6, 2016 2016.

[2] ns-3: Open-source event-driven network simulator. https://www.nsnam.org.

[3] Georgios Z Papadopoulos, Julien Beaudaux, Antoine Gallais, Thomas Noel, and Guillaume Schreiner. Adding value to wsn simulation using the iot-lab experimental platform. In *Wireless and Mobile Computing, Networking and Communications (WiMob), 2013 IEEE 9th International Conference on*, pages 485–490. IEEE, 2013.

[4] Manjunath Doddavenkatappa, Mun Choon Chan, and Akkihebbal L Ananda. Indriya: A low-cost, 3d wireless sensor network testbed. In *International Conference on Testbeds and Research Infrastructures*, pages 302–316. Springer, 2011.

[5] Geoff Coulson, Barry Porter, Ioannis Chatzigiannakis, Christos Koninis, Stefan Fischer, Dennis Pfisterer, Daniel Bimschas, Torsten Braun, Philipp Hurni, Markus Anwander, Gerald Wagenknecht, Sándor P. Fekete, Alexander Kröller, and Tobias Baumgartner. Flexible experimentation in wireless sensor networks. *Commun. ACM*, 55(1):82–90, January 2012.

[6] R. Ramanathan, E. Ciftcioglu, A. Samanta, R. Urgaonkar, and T. La Porta. Symptotics: a framework for estimating the scalability of real-world wireless networks. *Wireless Networks*, 2016. to appear, available online at http://link.springer.com/article/10.1007/s11276-016-1204-4.

[7] R. Ramanathan, A. Samanta, and T. La Porta. Symptotics: A framework for analyzing the scalability of real-world wireless networks. In *Proceedings of the 9th ACM symposium on PE-WASUN*, pages 31–38. ACM, 2012.

[8] E.N. Ciftcioglu, R. Ramanathan, and T.F. La Porta. Scalability analysis of tactical mobility patterns. In *MILCOM 2013*, pages 1888–1893. IEEE, 2013.

[9] J. Li, C. Blake, D. SJ De Couto, Hu Imm Lee, and R. Morris. Capacity of ad hoc wireless networks. In *Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 61–69. ACM, 2001.

[10] P. Gupta and P.R. Kumar. The capacity of wireless networks. *Information Theory, IEEE Transactions on*, 46(2):388–404, 2000.

[11] A. I Khuri and S. Mukhopadhyay. Response surface methodology. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(2):128–149, 2010.

[12] H. Tan, M. Chan, W. Xiao, P. Kong, and C. Tham. Information quality aware routing in event-driven sensor networks. In *IEEE INFOCOM, 2010*, pages 1–9, 2010.

[13] Z. M. Charbiwala, S. Zahedi Y. Kim, Y.H. Cho, and M. B. Srivastava. Toward Quality of Information Aware Rate Control for Sensor Networks. In *Fourth International Workshop on Feedback Control Implemenation and Design in Computing Systems and Networks*, April 2009.

[14] E.N. Ciftcioglu, A. Yener, and M.J. Neely. Maximizing quality of information from multiple sensor devices: The exploration vs exploitation tradeoff. *Selected Topics in Signal Processing, IEEE Journal of*, 7(5):883–894, Oct 2013.

[15] J. Edwards, A. Bahjat, Y. Jiang, T. Cook, and T.F. La Porta. Quality of information-aware mobile applications. *Pervasive and Mobile Computing*, 11:216–228, 2014.

[16] E. N. Ciftcioglu, A. Michaloliakos, A. Yener, K. Psounis, T. F. La Porta, and R. Govindan. Operational information content sum capacity: From theory to practice. *Computer Networks*, 75:1–17, 2014.

[17] E. N. Ciftcioglu, A. Michaloliakos, K. Psounis, T. F. La Porta, and A. Yener. Power minimization with Quality-of-Information Outages. In *Proc. IEEE WCNC*, Istanbul, Turkey, April 2014.

[18] M.Y.S. Uddin, H. Wang, F. Saremi, G.-J. Qi, T. Abdelzaher, and T. Huang. Photonet: a similarity-aware picture delivery service for situation awareness. In *Real-Time Systems Symposium (RTSS), 2011 IEEE 32nd*, pages 317–326. IEEE, 2011.

[19] Y. Jiang, X. Xu, P. Terlecky, T. Abdelzaher, A. Bar-Noy, and R. Govindan. Mediascope: selective on-demand media retrieval from mobile devices. In *Proceedings of the 12th international conference on Information processing in sensor networks*, pages 289–300. ACM, 2013.

[20] Arash Asadi, Qing Wang, and Vincenzo Mancuso. A survey on device-to-device communication in cellular networks. *IEEE Communications Surveys & Tutorials*, 16(4):1801–1819, 2014.

[21] Richard A Comroe and Daniel J Costello. Arq schemes for data transmission in mobile radio systems. *IEEE transactions on vehicular technology*, 33(3):88–97, 1984.

[22] A. Bar-Noy, G. Cirincione, R. Govindan, S. Krishnamurthy, T.F. LaPorta, P. Mohapatra, M. Neely, and A. Yener. Quality-of-information aware networking for tactical military networks. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2011 IEEE International Conference on*, pages 2–7. IEEE, 2011.

[23] Karen D Frazer. *NSFNET: A Partnership for High-speed Networking: Final Report, 1987-1995*. Merit Network, 1996.

[24] N. Gupta, P.R. Kumar, et al. A performance analysis of the 802.11 wireless lan medium access control. *Communications in Information & Systems*, 3(4):479–304, 2003.

[25] A. Jindal and K. Psounis. On the efficiency of csma-ca scheduling in wireless multihop networks. *Networking, IEEE/ACM Transactions on*, 21(5):1392–1406, 2013.

[26] R. Ramanathan, Ciftcioglu E., A. Samanta, Urgaonkar R., and T.L. Porta. An approximate model for analyzing real-world wireless network scalability. Technical report, Raytheon BBN Technologies, 2015. http://www.ir.bbn.com/~ramanath/pdf/symptotics-techreport.pdf.

[27] S.A. Chatzichristofis and Y.S. Boutalis. Cedd: color and edge directivity descriptor: a compact descriptor for image indexing and retrieval. In *Computer Vision Systems*, pages 312–322. Springer, 2008.

[28] T.T. Tanimoto. An elementary mathematical theory of classification and prediction. *International Business Machines Corporation*, 1958.

[29] G. Barrenechea, B. Beferull-Lozano, and M. Vetterli. Lattice sensor networks: Capacity limits, optimal routing and robustness to failures. In *Information Processing in Sensor Networks, 2004. IPSN 2004. Third International Symposium on*, pages 186–195. IEEE, 2004.

[30] M. Mitzenmacher and E. Upfal. *Probability and computing: Randomized algorithms and probabilistic analysis*. Cambridge University Press, 2005.