Available online at www.sciencedirect.com
ScienceDirect
Ad Hoc Networks 5 (2007) 429–443
Ad Hoc Networks
www.elsevier.com/locate/adhoc

Sufficient rate constraints for QoS flows in ad-hoc networks ☆

Rajarshi Gupta *, John Musacchio, Jean Walrand

Department of Electrical Engineering and Computer Science, University of California, Berkeley, CA 94720, United States

Received 30 December 2004; received in revised form 26 January 2006; accepted 28 January 2006
Available online 21 February 2006

Abstract

The capacity of an arbitrary ad-hoc network is difficult to estimate due to interference between the links. We use a conflict graph that models this interference relationship to determine if a set of flow rates can be accommodated. Using the cliques (complete subgraphs) of the conflict graph, we derive constraints that are *sufficient* for a set of flow rates to be feasible, yet are guaranteed to be within a constant bound of the optimal. We also compute an alternate set of sufficient constraints that can be easily derived from the rows of the matrix representation of the conflict graph. These two sets of constraints are particularly useful because their construction and verification may be distributed across the nodes of a network. We also extend the ad-hoc network model to incorporate variations in the interference range, and obstructions in the network.

Keywords: Ad-hoc networks; Graph theory; QoS

1. Introduction

Ad-hoc networks are becoming commonplace in today's world as wireless enabled devices proliferate. This leads to newer applications based on these networks, and hence the need to support quality of service. For a broader discussion and motivating examples, see Chapter 1 in [1].

A vital problem faced by researchers in this field is to determine the capacity of an arbitrary ad-hoc network. This is difficult because neighboring links using the same channel interfere, and the interference relationships between all of the links in a network can be quite complex.

Several researchers interested in the capacity of ad-hoc networks have modelled the ad-hoc network using randomized models, and evaluated asymptotic bounds on the capacity. Other work has addressed the question of whether a given flow vector is feasible on a particular ad-hoc network, where "feasible" means that a global scheduler with access to all the information in the network could find a link scheduling policy that would achieve the desired rates. In this work, we are also interested in methods for determining whether a flow vector is feasible, but we are particularly interested in methods that are suitable for distributed control in an ad-hoc network.

☆ This work was supported by the Defense Advanced Research Project Agency under Grant N66001-00-C-8062.

* Corresponding author. Tel.: +1 858 658 2080.
E-mail addresses: guptar@eecs.berkeley.edu (R. Gupta), musacchj@eecs.berkeley.edu (J. Musacchio), wlr@eecs.berkeley.edu (J. Walrand).

1570-8705/$ - see front matter
doi:10.1016/j.adhoc.2006.01.003

As in [2,3], we make use of a "conflict graph" that models the interference relationships between the different links of a network. Every link in the connectivity graph $G = (V, E)$ is represented by a node in the conflict graph $CG = (V^C, E^C)$. Two nodes in CG are connected by an edge if the nodes correspond to links in G that interfere. In Fig. 1, we show an example of a connectivity graph in which the interference between links is marked using dotted lines. The corresponding conflict graph is shown on the right. The authors of [2,3] show that a set of necessary and sufficient conditions to whether a set of flows is feasible is found by a computationally expensive process of finding all of the independent sets of the conflict graph, and then writing constraints in terms of the independent sets. (We review the details of the "independent set" method, as we call it, in Section 3.2.)

Because the independent set constraints are computationally expensive and require global information, they are not suitable to be used in a distributed scheme. We therefore look to find a different set of conditions that can be computed in a distributed way and that are at least sufficient, though perhaps not necessary, for a flow vector to be feasible.

One such set of constraints we refer to are the "row" constraints, because they are derived by using the rows of the matrix representation of the conflict graph. While they are sufficient conditions that are relatively easy to compute, they are much more restrictive than is necessary in many cases.

This motivates us to develop a different set of sufficient conditions using cliques (complete subgraphs) of the conflict graph. While previous work [3] has used cliques to find *necessary* conditions for a set of flow rates to be feasible, we find *sufficient*

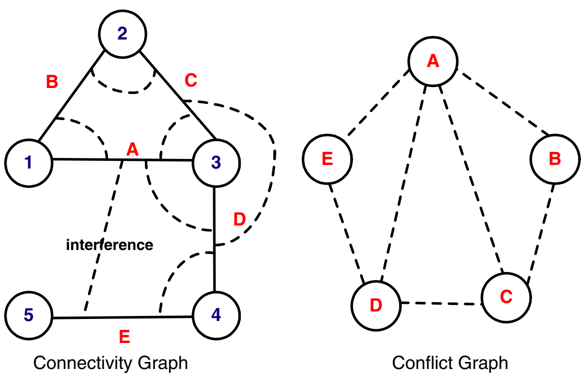

Fig. 1. Example of a connectivity graph with its conflict graph.

conditions. We refer to our sufficient clique constraints as "scaled" clique constraints because they are constructed by modifying the necessary clique constraints by a constant scaling factor. Unlike the row constraints, the scaled clique constraints are within a constant bounded factor of being necessary; a factor that is independent of the structure of the network, or the flows on it.

One potential drawback of a scheme based on cliques is that the number of cliques in a graph can grow exponentially with the number of nodes. However, we discuss a technique for identifying approximate cliques that can be implemented distributedly, and that grows only polynomially with the number of nodes. We discuss how the sufficiency of the clique constraints is maintained when using this approximation.

At first, we model the nodes having a constant interference range, and utilize the resulting unit disk structure (defined in Section 5.4) of the graph. However, to consider more realistic ad-hoc network scenarios, we augment the network model to allow for variances in the interference range, and evaluate its effect on the scaling factor. We also incorporate obstructions in the network, and extend the proof of sufficiency to include this.

Finally, we attempt to validate our theoretical constraint-based approach by simulating various ad-hoc networks, and comparing the achieved rates in the simulation to the theoretical limits predicted by our model. The results are encouraging at first sight, as the networks appear to satisfy our constraint-based approach. However, a deeper study reveals that in some cases, large inefficiencies in the underlying MAC protocol may be the primary cause for the limited throughput achieved; our constraints are satisfied as a corollary of this effect. We conclude that the constraint-based framework presented here becomes universally practicable only when used in conjunction with an efficient MAC protocol.

This paper does not try to present a new architecture to achieve QoS in ad-hoc networks. What we propose is a theoretical framework that allows us to answer questions about the feasibility of a specific set of flows on an arbitrary ad-hoc network. However, this framework may indeed be used as tools to develop new and improved protocols for QoS routing in ad-hoc networks, as we discuss in Section 2.1.

The rest of the paper is organized as follows. In Section 2, we describe the related work in the field, while Section 3 summarizes the network model that we base our results upon. Sections 4 and 5 are used to present the row and clique constraints, and prove sufficiency and bounds on these. In Section 6, we compare these two sets of constraints. The application of these constraints in distributed algorithms are described in Section 7. Simulation results are presented in Section 8, before we conclude the paper.

2. Related work

Many researchers have looked at modelling the capacity of ad-hoc networks. In [5], Gupta and Kumar study how the capacity of an ad-hoc network scales asymptotically with the number of nodes n in the network. The authors study the problem under two different models of interference: the *protocol model* and the *physical model*. In the protocol model, a receiver can successfully receive a sender's transmission if the sender is geographically closer, by some margin, than any other node that is actively transmitting. In contrast, the physical model models the transmissions of other nodes as noise, and assumes a receiver successfully receives a sender's transmission if the signal to interference (SIR) power ratio is above a threshold. The authors show that under both the protocol and physical models, the maximum capacity is $\Theta(1/\sqrt{n})$ per node, assuming optimal node placement in a disk of unit area. They also show that under random node placement, and under the protocol interference model, the capacity is $\Theta(1/\sqrt{n \log n})$. In [6], the authors extend this result to show that randomly scattered nodes can in fact achieve the same $\Theta(1/\sqrt{n})$ per-node transmission rate of arbitrarily located nodes. Other researchers have also extended the work of [5] by considering the effects of node mobility [7], and throughput-delay trade-offs [8].

Li et al. analyze the capacity of specific network topologies, and run packet level simulations of both the specific topologies and of random graphs [9]. With a packet level simulator, the authors show that the maximum throughput achieved in the simulation, where the link schedule is determined by a distributed MAC protocol (802.11), is somewhat less than that predicted by the analytical capacity model which assumes an ideal schedule. For example, for a network consisting of a chain of nodes relaying a single flow, analysis suggests that a rate of up to $\frac{1}{4}$ the channel capacity should be feasible, but with nodes using the 802.11 MAC, a rate of only $\frac{1}{7}$ is achieved.

Luo et al. [2] as well as Jain et al. [3] study the problem of finding rate constraints on a set of flows in an ad-hoc network, modelling what would be possible if there were a global scheduler. Both works model the interference between links in an ad-hoc network using conflict graphs and find capacity constraints by finding the independent sets of the conflict graph. The concepts of conflict graph and independent sets are discussed in more detail in the next section. In [10], Kodialam and Nandagopal model the routing and scheduling of flows in an ad-hoc network as a graph edge coloring problem, and find necessary and sufficient conditions for the achievability of a rate vector. The caveat is that this model only considers conflicts between links incident at the same node, and does not take into account interference due to all other neighboring links.

2.1. Extensions

The theoretical framework presented in this paper has already proved to be vital in our study of practical algorithms for quality of service in ad-hoc networks. Our understanding of cliques, and the crucial role they play in ad-hoc networks, has been central to the development of a suite of algorithms for the routing and MAC layers.

We have utilized the clique-based framework to propose realistic admission control and QoS routing mechanisms for ad-hoc networks. In [11], we proposed a distributed ad-hoc shortest widest path (ASWP) routing algorithm, which transforms the well-known shortest widest path paradigm to the ad-hoc domain by taking interference into account. We further proposed IQRouting [12] – a source-based heuristic mechanism that is able to select QoS routed paths in a dynamic manner, using only localized state information. Each of these algorithms relies on the computation of available bandwidth (Section 7.2), and utilizes cliques as the central unit of QoS.

Even after proving the existence of a feasible schedule, a more important practical problem is to find a mechanism that can achieve such a schedule. While a distributed solution may be too difficult to achieve, it is fair to hope for approximate mechanisms which are able to approach the feasible network throughput to within a bounded factor of the optimal. We suggested a novel backoff mechanism in [13] that takes a step in this direction, achieving a fair distribution of resources between wireless nodes sharing the medium.

3. Modelling the ad-hoc network

3.1. Determining conflict graph

We consider a wireless ad-hoc network with N stations. Each station is equipped with a radio having communications range ρ, and a potentially larger interference radius ω. Our model of interference is similar to that of the protocol model introduced in [5]. A transmission from station i to station j is successful if both of the following conditions are satisfied:

$$d_{ij} < \rho \quad \text{and} \quad d_{kj} > \omega \tag{1}$$

for every other station k that is simultaneously transmitting. Here d_{ij} denotes the distance between i and j.

The connectivity graph G is a directed graph whose vertices correspond to wireless stations and the edges correspond to the wireless links. There is a directed link from vertex i to vertex j iff $d_{ij} < \rho$. The nodes of the conflict graph represent links in the connectivity graph. A pair of nodes, l_{ij} and l_{kl} in the conflict graph are connected by an edge if they cannot have simultaneous transmissions according to the protocol interference model. This is similar to the model of the conflict graph as presented in [2,3].

To avoid confusion in the rest of the paper, we adopt the convention of using the prefix 'CG' (e.g. CG-node, CG-edge) when referring to the conflict graph. Additionally, a wireless device participating in the ad-hoc protocol is sometimes referred to as an ad-hoc station.

We envision the following technique for computing the conflict graph. We assume each station knows its position (e.g. using GPS) and disseminates its position information to other stations in the local neighborhood. Each station then geometrically computes which stations are within an interference radius; we call such stations interfering neighbors.

However, the transmission and interference footprints are not perfect circles in reality, due to factors such as obstacles, multi-path fading, etc. A better scheme would use measurements, as well as any available position information from GPS, to determine which stations interfere. In Sections 5.5 and 5.6, we attempt to augment our model to incorporate more realistic interference patterns.

In cases where the network's MAC protocol uses RTS/CTS (request to send/clear to send) or acknowledgements, such as 802.11 [14], one might choose to use stricter rules for identifying conflicting links than the conditions presented in (1). For a successful transmission to occur, these stricter conditions require that $d_{kj} > \omega$ and $d_{ki} > \omega$, for every other station k that is simultaneously transmitting *or receiving*. This is because a receiving station will be sending acknowledgements or will have sent a CTS message that silences other stations in the vicinity.

An alternate approach to constructing a conflict graph, is to compare the geometric centers of each link, i.e. the mid-point of a line segment connecting receiver and transmitter. This is in contrast to comparing distances between the stations themselves [15]. A sufficient condition for a pair of links, l_{ij} and l_{kl} not to conflict would be $|c_{ij} - c_{kl}| > (\omega + \rho)$, where c_{ij} is the position of the geometric center of link l_{ij}. Note that this is sufficient whether one uses the conditions (1) or the stricter RTS/CTS rules, for a link transmission to be successful. One may then construct a conflict graph by assuming that pairs of links that do not satisfy these link-center conditions conflict. However, this approximation would lead to some pairs of links being modelled as conflicting, even though they do not conflict in reality, and thus would result in a conservative view of network capacity.

It is useful to note that the discussions about row and clique constraints presented in this paper are *not* dependent on the way the CG is calculated. A more accurate CG representation is still amenable to the same analysis as presented here.

3.2. Independent set constraints

One can find a set of necessary and sufficient conditions for a proposed set of flow rates to be feasible, by looking at the independent sets in CG [2,3]. An independent set in the CG is a set of CG-nodes that have no edges between them. The idea is to identify all of the maximal independent sets of CG, calling them $\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_z$. Then, the independent set constraints say that a set of flow rates $\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_n$ is feasible *iff* there exists non-negative weights $\lambda_1, \lambda_2, \ldots, \lambda_z$ such that $\sum_{i=1}^{z} \lambda_i \leqslant 1$ and

$$\mathcal{F}_j \leqslant C \sum_{i:j \in \mathcal{I}_i} \lambda_i \quad \forall j \in \{1, \ldots, n\} \tag{2}$$

where C is the capacity of the channel.

Unfortunately, computing all of the independent sets in a conflict graph is exponential in the number of CG-nodes [3], so using this method in a large CG is not practical. Furthermore, the method requires global information about the entire graph. For these reasons, we look for methods that require less computation, and that can be done with local information at each CG-node.

4. Row constraints

We describe here one set of sufficient conditions for a set of flow rates to be feasible. We represent a set of flow rates as the vector $\mathcal{F}$ of size $n \times 1$, where n is the number of links in the network and $\mathcal{F}_i$ is simply the flow rate assigned to link i. We also make use of a conflict graph incidence matrix $\mathcal{M}$ defined as follows:

$$\mathcal{M}_{jk} = \begin{cases} 1 & \text{if links } j \text{ and } k \text{ are connected by an edge,} \\ 0 & \text{otherwise.} \end{cases}$$

Theorem 1. *A set of flow rate assignments $\mathcal{F}$ has a feasible schedule if*

$$\mathcal{M}\mathcal{F} \leqslant \mathcal{C}. \tag{3}$$

where $\mathcal{C}$ is a $n \times 1$ vector, with all entries equal to the channel capacity $\mathcal{C}$. We refer to expression (3) as the "row constraints" because each row of the conflict graph matrix $\mathcal{M}$ is used to construct a linear inequality on the values of $\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_n$.

Proof. We assume that the entries of the flow rate vector $\mathcal{F}$ are rational multiples of each other. Let T be the smallest integer such that flow rates $\mathcal{F}_i$ are integer multiples of $\frac{C}{T}$ for all $i \in \{1,\ldots,n\}$, and let $\mathcal{K}$ be the vector of integers such that $\mathcal{F}_i = \mathcal{K}_i \times \frac{C}{T}$. We will construct a periodic schedule with period T.

We begin by transforming the conflict graph CG by replacing a CG-node i by a clique consisting of $\mathcal{K}_i$ nodes. Each of the new nodes in the replacement clique needs to be connected to every neighbor of the replaced CG-node.

Let this transformed graph be called $\mathrm{CG_f}$. Observe that a coloring of $\mathrm{CG_f}$ implies a schedule for CG. This is because each node in the clique replacing i has a unique color. Further, if CG-nodes i and j were adjacent in CG, all the components of their replacement cliques are adjacent in $\mathrm{CG_f}$, and hence have unique colors. Consequently, by letting each color correspond to a slot, we can ensure that CG-nodes i and j ate scheduled for appropriate durations, which are disjoint.

Next we observe that $\mathcal{M}\mathcal{F} \leqslant \mathcal{C}$ implies $\mathcal{M}\mathcal{K} < \mathcal{T}$, where $\mathcal{T}$ is a $n \times 1$ vector, with all entries equal to T. This in turn implies that each $\mathrm{CG_f}$-node has strictly less than T neighbors. We may now color the graph with the greedy coloring algorithm that follows. Label the $\mathrm{CG_f}$-nodes with indices $\{1,\ldots,N\}$ and begin coloring the nodes in increasing order of index, by assigning each $\mathrm{CG_f}$-node a color with index in $\{1,\ldots,T\}$. For $\mathrm{CG_f}$-node i, we assign it the lowest indexed color not already assigned to a neighbor. We can always find such a color with index in $\{1,\ldots,T\}$, simply because $\mathrm{CG_f}$-node i has less than T neighbors. Thus we have a valid coloring for $\mathrm{CG_f}$, which implies the existence of a feasible schedule for the flow vector $\mathcal{F}$. □

The row constraints may be evaluated in a localized and distributed manner by evaluating

$$\mathcal{M}^i\mathcal{F}^i \leqslant \mathcal{C}^i, \quad \forall i, \tag{4}$$

where $\mathcal{F}^i$ and $\mathcal{C}^i$ are the flow vector and capacity vector representing only the neighbors of link i. Since all the non-zero elements of the ith row of $\mathcal{M}$ lies in the interference neighborhood of CG-node i, we only need to consider those relevant entries of $\mathcal{F}$ and $\mathcal{C}$. Each CG-node only needs local information from all its neighbors to check the validity of these constraints.

While the row constraints are sufficient, they are not necessary constraints. Indeed, in some examples they can be much more restrictive than is necessary. Fig. 2 shows a conflict graph that illustrates such an example. The row constraints imply that

$$\mathcal{F}_\mathrm{A} + \mathcal{F}_\mathrm{B} + \mathcal{F}_\mathrm{C} + \mathcal{F}_\mathrm{D} + \mathcal{F}_X \leqslant C. \tag{5}$$

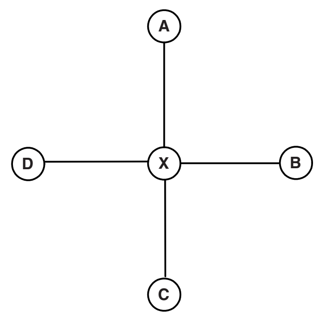

Fig. 2. Row constraints – star graph.

However, expression (5) is much stronger than what is necessary to be feasible. For example, one could achieve the rates $\mathcal{F}_A = \mathcal{F}_B = \mathcal{F}_C = \mathcal{F}_D = C$ if CG-node X is switched off. Yet, if we used expression (5) as our guide (with $\mathcal{F}_X = 0$), we would have to set all the rates to $C/4$. Thus, in this case, obeying the row constraints would lead us to rates that are only 1/4 of actually feasible rates. In theory, the row constrained solution could be arbitrarily far from optimal, as in the case if the star had many rays instead of 4.

Because the row constraints could possibly be overly conservative, we are motivated to develop a different set of constraints using cliques. We describe the method in the next section.

5. Clique constraints

5.1. Cliques: definition and background

We begin with a few definitions well-known in graph theory. Consider a bi-directional graph with nodes and edges. An *induced subgraph* is a subset of the nodes together with any edges whose endpoints are both in this subset. An induced subgraph that is a complete graph is called a *clique*. A *maximal clique* of a graph is a clique such that it is not contained in any other clique. In the conflict graph in Fig. 1, *ABC, ACD* and *ADE* are all maximal cliques.

A clique in a conflict graph is closely related to the capacity of ad-hoc networks. CG-nodes that form a clique are all connected to each other – consequently only one CG-node in a clique may be active at once.

There are two main reasons to utilize cliques in place of independent sets. First, cliques in a CG are inherently local structures and therefore amenable to localized algorithms. Second, as we describe in Section 7.1, we can approximate the maximal cliques around a link in a computationally simple and distributed fashion.

5.2. Specifying clique constraints

Assume that each CG-node (i.e., link in the connectivity graph) is aware of all the maximal cliques that it belongs to. This information may be described by an incidence matrix $\mathcal{Q}^i$, which is of order $q \times n$. Here, q is the number of maximal cliques that this link i belongs to, and n is the total number of links. In this matrix,

$$\mathcal{Q}^i_{jk} = \begin{cases} 1, & \text{if links } i \text{ and } k \in \text{clique } j, \\ 0, & \text{if links } i \text{ or } k \notin \text{clique } j. \end{cases}$$

Note that this matrix $\mathcal{Q}^i$ does not include information about the entire network – it covers only the interference neighborhood of link i. The union of the clique matrices across all the links gives the global clique matrix $\mathcal{Q}$.

Since a network must satisfy the capacity constraints for all cliques, we can write the 'clique constraints' in a matrix form. As in Section 4, we denote the flow vector $\mathcal{F}$ and the capacity vector $\mathcal{C}$. Hence we have,

$$\mathcal{Q}\mathcal{F} \leqslant \mathcal{C}. \tag{6}$$

Consider the conflict graph as shown in Fig. 1. Let the allocated flow on each CG-node be denoted by $\mathcal{F}_A, \mathcal{F}_B$ etc. Then, the clique constraints $\mathcal{Q}\mathcal{F} \leqslant \mathcal{C}$ are:

$$\mathcal{F}_A + \mathcal{F}_B + \mathcal{F}_C \leqslant C,$$
$$\mathcal{F}_A + \mathcal{F}_C + \mathcal{F}_D \leqslant C,$$
$$\mathcal{F}_A + \mathcal{F}_D + \mathcal{F}_E \leqslant C.$$

5.3. Insufficiency of clique constraints

The clique constraints provide a *necessary* condition for a realizable schedule to exist, since there cannot be a feasible schedule over links that form a violated clique constraint. One might hope that these constraints would also be sufficient conditions for a realizable schedule. Unfortunately, that is only true for a special sub-class of graphs called *Perfect Graphs* [16]. Perfect graphs are those whose chromatic number (least number of colors required to color the graph, such that every adjacent node has a separate color) and clique number (size of the largest clique) are equal for all induced subgraphs.

As noted in [3], the simplest example of an imperfect graph where the clique constraints are insufficient is the conflict graph shaped like a pentagon, shown in Fig. 3. Although the clique constraints suggest a valid flow allocation of $0.5C$ on each link, in reality only $0.4C$ on each link is achievable since at most two out of the five CG-nodes may be active simultaneously.

5.4. Sufficiency using scaled clique constraints

In Section 4, we proved the sufficiency of row constraints for a schedule to exist. In this section, we demonstrate another *sufficient* condition – based on clique constraints, scaled by 0.46.

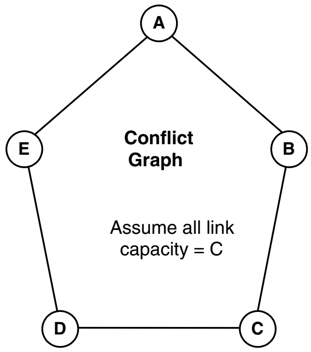

Fig. 3. A pentagon conflict graph.

When modelling a link by its mid-point (Section 3.1), the resulting CG has an unit disk graph structure. A graph is said to be a *unit disk graph* (UDG) [17] when there is an edge between two nodes if and only if their Euclidean distance is at most 1 (or a constant value ω). We then use properties of UDG to prove the following theorem.

Theorem 2. *When the conflict graph is modelled as a UDG, a set of flow rate assignments $\mathcal{F}$ has a feasible schedule if*

$$\mathcal{QF} \leqslant \mathcal{C} \times 0.46. \tag{7}$$

Proof. Following in the lines of the proof of Theorem 1, we impose an integer flow rate vector $\mathcal{K}$ by assuming a T-periodic slotted time schedule, where $\mathcal{F}_i = \mathcal{K}_i \times \frac{C}{T}$. Next, we transform the conflict graph CG by replacing a CG-node i by a clique consisting of $\mathcal{K}_i$ nodes. Let this transformed graph be called $\mathrm{CG_f}$. As we observed in the proof of Theorem 1, a coloring of $\mathrm{CG_f}$ implies a schedule for CG.

Denote $\chi(\mathrm{CG_f})$ as the chromatic number of $\mathrm{CG_f}$, and let $\kappa(\mathrm{CG_f})$ be the clique number of the graph. Then it is well known that $\chi(\mathrm{CG_f}) \geqslant \kappa(\mathrm{CG_f})$, since we at least need to use a different color for every member of the largest clique.

We would have a *feasible* schedule for CG if we could color $\mathrm{CG_f}$ using at most T colors (i.e., $\chi(\mathrm{CG_f}) \leqslant T$). This would ensure that all CG-nodes in a clique are scheduled for disjoint slots, yet the number of available slots in the periodic schedule is not exceeded.

In [4], the authors define the imperfection ratio imp(G) of a transformed weighted graph (e.g. $\mathrm{CG_f}$) as the supremum of the ratio between its chromatic number and its clique number. They further show bounds on imp(G) if the graph belongs to the class of UDG. For a UDG,

$$\mathrm{imp(CG)} = \sup_f \frac{\chi(CG_\mathrm{f})}{\kappa(CG_\mathrm{f})} \leqslant 2.155 \approx \frac{1}{0.46}. \tag{8}$$

We sketch the proof of this result in Section 5.6.

To complete our proof, we observe that $\mathcal{QF} \leqslant \mathcal{C} \times 0.46$ implies that $\mathcal{QK} \leqslant 0.46\mathcal{T}$. This in turn implies that $\kappa(\mathrm{CG_f}) \leqslant 0.46T$, since the clique number $\kappa(\mathrm{CG_f})$ is simply the largest element of $\mathcal{QK}$. Now applying expression (8), we have $\chi(CG_\mathrm{f}) \leqslant \frac{1}{0.46}\kappa(CG_\mathrm{f}) \leqslant T$. Thus we have a sufficient condition for the existence of a feasible schedule. □

Consider the scaled clique constraints of expression (7): In addition to being sufficient, they are within a bounded factor of the necessary conditions of expression (6). Consequently, we are assured that flow vectors satisfying the scaled clique constraints are no further than a factor of 0.46 from an optimally feasible flow vector.

It is also worth noting that the clique constraints may be evaluated in a distributed fashion by checking

$$\mathcal{Q}^i\mathcal{F}^i \leqslant \mathcal{C}^i \times 0.46, \quad \forall i, \tag{9}$$

where $\mathcal{F}^i$ and $\mathcal{C}^i$ are the flow vector and capacity vector as known to link i. The only non-zero elements of the clique matrix $\mathcal{Q}^i$ lie in the interference neighborhood of CG-node i, and so the only affected elements of $\mathcal{F}^i$ and $\mathcal{C}^i$ are the corresponding ones.

5.5. Obstructions in ad-hoc network

Typically, interference regions in a real ad-hoc network are not shaped like perfect disks, due to the presence of obstructions. In such situations, the underlying CG may not be a UDG. Thus, condition (7) may not be sufficient to guarantee the existence of a feasible schedule, as the proof of Theorem 2 depends on the UDG property.

One solution to this problem is to construct what we call a "virtual" conflict graph, where links that lie within an interference range of each other are always modelled as conflicting, even if in reality an obstacle prevents the links from interfering each other. Note that the virtual-CG is a UDG by construction, even if the underlying CG is not. Using the virtual-*GG* we may state and prove the following.

Theorem 3. *A set of flow rate assignments $\mathcal{F}$ has a feasible schedule if*

$$\mathcal{Q}_\mathrm{V}\mathcal{F} \leqslant \mathcal{C} \times 0.46, \tag{10}$$

where Q_V is the clique incidence matrix of the virtual-CG, as defined above. This is valid even if the "true" CG is not UDG.

Proof. Note that the virtual-CG has the same set of CG-nodes (corresponding to links in the connectivity graph) as the true-CG, while the set of CG-edges of the virtual-CG are a superset of the edges of the true-CG. We may define integer flow rates $\mathcal{K}_i$ where $\mathcal{K}_i = \mathcal{F}_i \times \frac{C}{T}$, a virtual-CG$_f$, and a true-CG$_f$ in the same way that we did in the proof of Theorem 1. By Theorem 2, condition (10) implies that there exists a coloring of the virtual-CG$_f$ with at most T colors. However, a valid coloring of the virtual-CG$_f$ is a valid coloring of the true-CG$_f$ as well, because each node in the true-CG$_f$ has a subset of the neighbors it has in the virtual-CG$_f$. Thus, there exists a valid coloring of the true-CG$_f$, which in turn implies the existence of a feasible schedule to achieve the flow rate vector $\mathcal{F}$. □

5.6. Variance in interference range

Often, the interference region may not be shaped like a perfect disk, but be uneven near the edges, due to fading effects. Such an interference region may be modelled as being bounded between two disks. Two CG-nodes cannot interfere if their distance >1, and two CG-nodes will always interfere if their distance $\leqslant x \leqslant 1$. But if two CG-nodes are separated by a distance between x and 1, they may or may not interfere. We would like to take this variance into account while employing our constraint-based approach.

In order to present the proof of our extension, it is useful to sketch the proof of the original imperfection theorem, as given in [4].

Theorem 4 [4]. *For a UDG,*

$$\mathrm{imp}(G) \leqslant \frac{\frac{\sqrt{3}}{2}+1}{\frac{\sqrt{3}}{2}} \approx 2.155. \qquad (11)$$

Proof. The proof uses the 'Stripe Lemma' from [17] which implies that in a UDG, stripes of width $\sqrt{3}/2$ are perfect. The technique used in the proof is to cover the UDG with a large number of randomly positioned stripe-graphs. A stripe-graph consists of stripes of width $\sqrt{3}/2$ separated by a distance of 1. Since each stripe is perfect, and their separation is greater than 1, the entire stripe-graph is perfect. Then, the probability p that any node is covered by a particular stripe-graph is given by $p = \frac{\frac{\sqrt{3}}{2}}{\frac{\sqrt{3}}{2}+1}$. Extending a probability argument, the authors show that the imperfection of the graph is bounded above, by the reciprocal, i.e., $\mathrm{imp}(G) \leqslant \frac{1}{p}$. □

We define the 'connectivity band' of a node in a stripe in the CG. In Fig. 4, it follows from geometry that a node A will necessarily be connected to all other nodes in the stripe which lie within a *connectivity band* of height 1/2 in either direction (shaded region in the figure). So if two nodes are not connected to each other, their Y coordinates must be separated by at least 1/2.

The following theorem now follows from Theorem 4.

Theorem 5. *When the interference range ω in a CG varies between $\frac{1}{2} \leqslant x \leqslant w \leqslant 1$, the imperfection ratio is bounded by,*

$$\mathrm{imp}(CG) \leqslant \frac{1+z}{z}, \quad \text{where } z = \sqrt{x^2 - \left(\frac{1}{2}\right)^2}. \qquad (12)$$

Proof. In [17], the authors show that the Stripe Lemma holds if for all nodes located within the stripe, two nodes on either side of a third node, but not adjacent to the third node (e.g. nodes B and C in Fig. 4) are never connected themselves. This is a corollary of having a connectivity band of width 1/2 on either side of a node – B and C on either side of A are necessarily separated by a distance greater than 1.

When the interference range varies between x and 1, a modification of the Stripe Lemma using

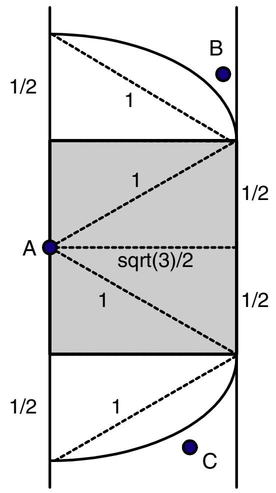

Fig. 4. Connectivity band of a node in a stripe.

Table 1
Imperfection ratio bound as a function of interference unevenness

x	z	imp	1/imp
1	0.866	2.155	0.46
0.9	0.75	2.33	0.43
0.8	0.62	2.60	0.38
0.7	0.49	3.04	0.33
0.6	0.33	4	0.25

triangle geometry ensures that the stripe is always perfect provided $z^2 + \left(\frac{1}{2}\right)^2 \leqslant x^2$, where z is the width of the stripe. Consider Fig. 4 again: We need to ensure that the band of height 1/2 on either direction of A is within its range of connectivity, i.e. the diagonal of the connectivity band $\leqslant x$. At the limit, we get $z = \sqrt{x^2 - (1/2)^2}$. Note that since $x \leqslant 1$, we have $z \leqslant \sqrt{3}/2$.

Following the proof technique of Theorem 4 [4], the imperfection ratio in this case is bounded by $\mathrm{imp}(CG) \leqslant \frac{1+z}{z}$. □

Hence, existence of a feasible schedule is guaranteed provided $\mathcal{QF} \leqslant \mathcal{C} \times \frac{z}{1+z}$.

Table 1 tabulates the values of the bound on imperfection ratio as a function of the unevenness in the interference region. As seen, the imperfection can grow as the interference range varies more.

This extension can also account for the approximation error introduced by representing a link by its mid-point (Section 3.1). To account for this, we can model the interference range as lying between two discs of radius ω and $\omega + \rho$.

Once again it is important to realize that the clique constraints applied to a virtual conflict graph akin to the one described in Section 5.5 will also imply a feasible schedule. In this case, the virtual CG corresponds to the *maximum* interference range (i.e., modelling the variance in the interference range by its upper bound). A feasible schedule in the virtual CG ensures a feasible schedule in the true CG as well. Depending on the value of the imperfection bound as given in Table 1, we may choose to use the clique constraints on the virtual $CG(\mathcal{Q}_{\mathrm{V}}\mathcal{F} \leqslant \mathcal{C} \times 0.46)$ or the constraints on the true $CG(\mathcal{QF} \leqslant \mathcal{C} \times \frac{z}{1+z})$.

6. Row vs. clique constraints

We have presented two sets of constraints, *both* of which may be evaluated in a distributed fashion. In this section, we discuss the efficacy of using one set of constraints versus the other.

6.1. Relating row and clique constraints

We have seen that the row constraints are sufficient, but may be arbitrarily far from being necessary, as we observed in Section 4. On the other hand, the scaled clique constraints are not only sufficient, but also within a bounded factor of the necessary constraints (Section 5.4).

We present a schematic of the relationship between the various sets of constraints discussed earlier. Each of the constraint sets defines a polytope in $\mathbb{R}^n_+$, describing the feasible range of values for the flow vector $\mathcal{F}$. We show a two dimensional representation of these regions in Fig. 5.

The independent set (IS) polytope corresponds to the necessary and sufficient condition for a feasible schedule to exist, and is therefore the benchmark. The outer clique polytope is derived from necessary conditions and so contains the IS polytope. Scaling this by a factor of 0.46 gives us the scaled clique polytope. This corresponds to sufficient conditions for feasibility, and is hence entirely contained within the IS polytope. The row constraints also describe sufficient conditions, and so the row polytope lies within the IS polytope as well. The row and the scaled clique constrained polytopes certainly overlap (e.g. at the point when the flow vector is 0), but do not contain one another.

Indeed, there are notable cases (typically simple networks, and/or few flows) where the row constraints are less pessimistic than scaled clique constraints. An example is the pentagon CG (Fig. 3), where row constraints allow a flow rate of $C/3 = 0.33C$ is possible on each link, while scaled clique constraints only allow a flow rate of $0.46 \times$

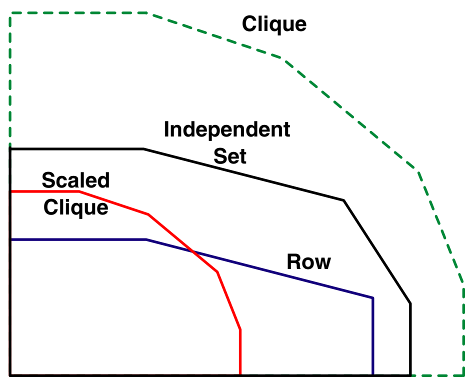

Fig. 5. Polytopes describing feasible regions for the flow rate vector, as defined by the row, clique and independent set constraints.

$C/2 = 0.23C$. However, as the number of flows increases, we are likely to get situations where a CG-node has many neighbors, which do not all form a single clique. Such topologies make the row constraints too restrictive, as is often the case in realistic ad-hoc networks.

6.2. Using both row and clique constraints

Because the row constraints may be less restrictive in some circumstances than the clique constraints, we would like to give each station in the network the flexibility to choose to use either the row constraints or the scaled clique constraints, depending on which are less restrictive for the situation. However, we need to be sure that in a network where some stations use clique constraints, and others use row constraints, the union of the constraints being checked across the network constitute sufficient conditions for a flow rate vector to be feasible. We show that this is indeed the case in the theorem that follows.

Theorem 6. *Suppose the CG-nodes are partitioned into two disjoint sets* **A** *and* **B**. *Let* $\mathcal{Q}^{\mathbf{A}}$ *denote a reduced clique matrix, containing only the rows of* $\mathcal{Q}$ *that correspond to cliques containing one or more CG-nodes in* **A**. *Similarly let* $\mathcal{M}^{\mathbf{B}}$ *denote a reduced conflict graph incidence matrix, containing only the rows of* $\mathcal{M}$ *that describe the neighbors of CG-nodes in* **B**.

The flow rate vector $\mathcal{F}$ *is feasible if*

$$\mathcal{Q}^{\mathbf{A}}\mathcal{F} \leqslant 0.46 \times C,$$

$$\mathcal{M}^{\mathbf{B}}\mathcal{F} \leqslant C.$$

Proof. As in the proof of Theorem 1 we define integer flow rates $\mathcal{K}_i$ where $\mathcal{K}_i = \mathcal{F}_i \times \frac{C}{T}$, and we define the graph $\mathrm{CG_f}$ by replacing each node i of CG with a clique of $\mathcal{K}_i$ "children" nodes. Recall that a coloring of graph $\mathrm{CG_f}$ using at most T colors implies the existence of a feasible schedule. Let $CG_{\mathrm{f}}^{\mathbf{A}}$ denote the subgraph of CG_{f} restricted to CG_f-nodes whose parents are in **A**. Theorem 2 implies the existence of a coloring for $CG_{\mathrm{f}}^{\mathbf{A}}$. We color the nodes of $CG_{\mathrm{f}}^{\mathbf{A}}$ using such a coloring, and now we seek to color the remaining nodes of $\mathrm{CG_f}$.

We observe that $\mathcal{M}^{\mathbf{B}}\mathcal{F} \leqslant C$ implies $\mathcal{M}^{\mathbf{B}}\mathcal{K} \leqslant T$ which in turn implies that each node in $CG_{\mathrm{f}}^{\mathbf{B}}$ has less than T neighbors. It is worth emphasizing that each node in $CG_{\mathrm{f}}^{\mathbf{B}}$ has less than T neighbors in total, including neighbors in $CG_{\mathrm{f}}^{\mathbf{B}}$ and neighbors in $CG_{\mathrm{f}}^{\mathbf{A}}$. We may now color the remaining nodes of $\mathrm{CG_f}$ by using the same greedy coloring algorithm as in Theorem 1. Because the remaining nodes each have less than T neighbors, we can always find a color with index in $\{1, \ldots, T\}$. Thus we have found a coloring for all of $\mathrm{CG_f}$, and therefore there exists a feasible schedule to accommodate flow rate vector $\mathcal{F}$. □

7. Distributed algorithms

7.1. Computing cliques

For our purposes, we would like to compute maximal cliques in a computationally simple, distributed and localized manner. General algorithms to generate cliques (e.g. [18–20]) are centralized in nature. Also, these are exponential algorithms since the number of maximal cliques in a graph (even in a UDG) is exponential. So we aim for a polynomial approximation algorithm.

The essence of the approximation is to use slightly 'super-maximal' cliques. When the number of cliques grows large, several nearby cliques will have a significant intersection, i.e., their membership will differ only at a few nodes. In this case, the approximation algorithm generates the *union* of these as the super-maximal clique. The exact set of maximal cliques generated depends upon the location of the nodes.

By using approximate cliques, the constraints are only strengthened further as multiple individual constraints are replaced by their union. Thus, the sufficiency of the approximated clique constraints implies the sufficiency of the actual set of clique constraints.

The heuristic algorithm presented by authors of this paper in [21] distributedly approximates all maximal cliques that a CG-node belongs to. The algorithm in fact works on any UDG, a more general class of graphs than conflict graphs alone. It makes use of certain key geographic structures of these graphs. For each edge, we limit the set of vertices that may form cliques with this as the longest edge. We then consider several characteristic shapes determined by that edge, and prove that all cliques involving this edge are included in one of the sets of vertices contained in these shapes. The evaluation of the vertices located within these shapes may be done in polynomial time, enabling us to limit the running time of the algorithm.

The algorithm works in $O(m\Delta^2)$ time and generates $O(m\Delta)$ cliques, where m is the number of edges in the graph and Δ is its maximum degree. We also provide a modified version of the algorithm which improves the performance in many cases, albeit without affecting the worst case running time.

7.2. Capacity estimation

We can use the clique constraints to estimate the capacity of an ad-hoc network, in a localized and distributed way. Assume that each CG-node is aware of all its interference neighbors, and their allocated flows. Using this localized information, each CG-node can estimate its available capacity Γ^i as

$$\Gamma^i = \min\{(\mathcal{C}^i \times 0.46) - \mathcal{Q}^i\mathcal{F}^i\}. \tag{13}$$

Γ^i is the available bandwidth on link i, taking into account flows allocated on i, as well as interference from neighboring links. We consider all maximal cliques that i belongs to, and take the worst case available capacity over all the cliques. This ensures that the available capacity satisfies the conditions for sufficiency, as described earlier in the paper.

The value Γ^i is a vital commodity for algorithms involving quality or rate guarantees in a network. In a wired network, the width or available bandwidth of a path is determined by the remaining bandwidth on the bottleneck link in the path. In the case of an ad-hoc network, it is the bandwidth available on the bottleneck *clique* that offers a parallel metric of comparison. We therefore expect our methods of computing Γ^i to be utilized frequently, in distributed algorithms for admission control and QoS routing.

7.3. Admission control

A distributed admission control scheme may now be overlaid on the capacity estimation framework. We assume as above that the CG-nodes in the conflict graph are aware of all their interference neighbors, and their allocated flows.

Using the approximation algorithm described in [21], each CG-node (in reality, the node at one end of this link) keeps track of the maximal cliques it is part of, and the available bandwidth on them. The admission control algorithm is effected when a new flow request {*src*, *dest*, *path*, *bw*} is received by the network. The new flow description is sent out along the path of the flow. Every station that receives the flow request updates its flow vector $\mathcal{F}^i$ with the new flow parameters. And then it recomputes $\Gamma^i = \min\{(\mathcal{C}^i \times 0.46) - \mathcal{Q}^i\mathcal{F}^i\}$ for all its links. If $\Gamma^i < 0$ for any link i, an 'admission denied' message is generated.

We assume that the route for the flow is known ahead of time. This allows us to accept or reject flow requests based on the ad-hoc bandwidth available in the network. We can further incorporate routing techniques that make use of interference knowledge, as in [12].

8. Simulation results

We present simulation results to test the various ideas discussed earlier in the paper. We perform simulations using OPNET [22], which implements detailed packet level simulation models of channels, interference, as well as the 802.11 MAC and ad-hoc routing protocols.

8.1. Feasible vs. actual schedule

In Sections 4 and 5, we have shown that a feasible schedule exists when the row constraints or the scaled clique constraints are satisfied. However, it is important to note that the mere existence of a feasible schedule does not imply our being able to find it, let alone impose it on all the ad-hoc stations. The task of finding a distributed scheduling mechanism that achieves the feasible schedule is a well-known open question, and is beyond the scope of this paper.

Instead, we compare our theoretical models with a practical MAC protocol – the default 802.11b. We make no changes to the existing 802.11b, and simply use it to check against the capacity limits predicted by our model.

8.2. Row constraints

First, we evaluate the row constraints presented in Section 4. The conflict graph evaluated here is shaped like a star, as shown earlier in Fig. 2. All the links A, B, C, and D interfere with link X, but none of these interfere with each other. Assuming the effective capacity of the channel to be 5 Mbps, satisfying the row constraints (at CG-node X) requires

$$\mathcal{F}_A + \mathcal{F}_B + \mathcal{F}_C + \mathcal{F}_D + \mathcal{F}_X \leqslant 5.$$

In fact, *each* of the link A, B, C, and D should be able to achieve close to the 5 Mbps capacity.

The results of this simulation are presented in Table 2. All rates are in Mbps, and the notation

Table 2
Simulation results – star conflict graph

Comment	Generated	Received
One outer link	5	5
Two outer link	2×5	2×4.5
Three outer link	3×5	3×4.5
Four outer link	4×5	4×4.5
Two outer + center	$2\times5+5$	$2\times4.5+0.25$
Four outer + center	$4\times5+5$	$4\times4.5+0.01$
Limit outer links	$4\times2.5+2$	$4\times2.5+0.85$
Limit outer links	$4\times0.5+4$	$4\times0.5+3.5$

$3\times p+q$ in the table implies that three of the outer links are all generating/receiving p Mbps, while link X at the center gets q Mbps.

As seen from the first set of rows of the table, all four of the outer links can indeed achieve 4.5 Mbps each – so these rates exceed the row constrained rates by a large margin. This corroborates the fact that the row constraints may be overly pessimistic, as discussed in Section 4.

The conflict graph shown in Fig. 2 is perfect, and so the unscaled clique constraints of expression (6) should be sufficient [3]. The clique constraints for this graph look like $\mathcal{F}_A+\mathcal{F}_X\leqslant 5, \mathcal{F}_B+\mathcal{F}_X\leqslant 5$, etc. Indeed, as seen from the received rates in Table 2, the clique constraints are always satisfied.

We also observe the unfair nature of the sharing of links. The second set of rows show that link X is starved when multiple of its neighbors are transmitting simultaneously. Only by rate limiting the neighboring traffic (third set of rows) can link X hope to achieve its share.

8.3. Clique constraints and scaling

In order to observe the insufficiency of the unscaled clique constraints, we need to simulate an imperfect CG – the pentagon in Fig. 3. The unscaled clique constraints on this conflict graph suggests that a rate of 2.5 Mbps should be achievable on each link (assuming a 5 Mbps channel capacity); although analysis confirms that each link is limited to 2 Mbps at most (Section 5.3).

The simulation results presented in Table 3 support the analysis quite well. When the traffic on each link is shaped to the predicted limit of 2 Mbps, all links are able to achieve 2 Mbps each. Increasing the offered rates further, to 2.5 Mbps on each link, only makes matters worse – the links get saturated and the throughput is reduced to only 1.8 Mbps per link. The row constraints would only allow $5/3=1.67$ Mbps per link, and so they are again too restrictive.

Table 3
Simulation results – pentagon conflict graph

Comment	Generated	Received
Unsaturated	5×1	5×1
Unsaturated	5×1.5	5×1.5
Reaching saturation	5×2	5×2
Saturated	5×2.5	5×1.8
Over-saturated	5×5	5×1.8

8.4. Randomly generated network

We generate a random ad-hoc network consisting of 50 ad-hoc stations in a 2.5 km × 2.5 km area. The locations of the stations are fed into OPNET. Transmission range is set to 500 m, and the interference range is 1 km. These numbers roughly correspond to a battalion of tanks in a battlefield, with powerful radios. We pick five pairs of stations at random and set up video flows between them (running for 5 min), and alter these rates in order to change the load on the network.

We use the DSR routing protocol [23] to determine the routes, and measure the amount of traffic received at each of the receiving stations in the OPNET simulation. In parallel, we feed the routes generated into the theoretical model implemented in MATLAB [24]. Using Eq. (13), we calculate the minimum spare capacity Γ^i on each link i, to compare against the actual rates received by the flows. The minimum spare capacity Υ in the network is calculated by taking the minimum of Γ^i over all the links.

We run simulations involving three, four and five flows. In many of the simulations, the calculated spare capacity on the network is negative, which predicts that some of the flows may be losing traffic. Each simulation run yields a single point on the plot, with the rates and the routes of all the flows together determining the Γ^i values for each link, and thereby Υ for the network. A summary of the simulation results is shown in Fig. 6. For an individual simulation run, we assign the same video rate to each of the three, four or five flows. On the X-axis, we plot the rate of video traffic sent out on each of these flows; while the Y-axis plots the average rate received over all the flows.

By looking at the routes, and interpolating between the various video rates, we can also determine the exact transmitted rate at which $\Upsilon=0$, as predicted by our model. We determine this limit

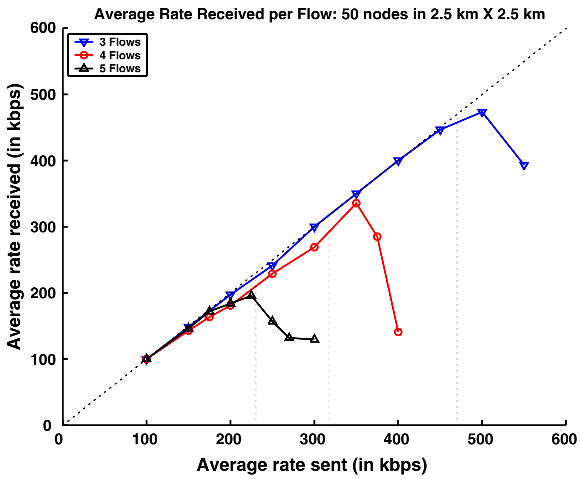

Fig. 6. Comparing received rates with theoretical model.

for each of the three, four and five flow cases, and plot these using the dotted vertical lines. As seen from the figure, the flows appear to receive almost all their traffic until the predicted limit. In each case, the flows experience a sharp loss of quality soon after the theoretical limit is crossed.

8.5. MAC inefficiencies and graph imperfection

Our simulation results indicate that the achieved throughput in an ad-hoc network is often limited to a point close to the scaled clique limits. However, the bound of $\frac{1}{0.46} = 2.155$ on the imperfection ratio may be significantly larger than the typical case imperfection ratio. In fact, [4] states a conjecture that the imperfection ratio is in fact bounded above by $\frac{3}{2}$.

Recall that the imperfection ratio is the ratio between the chromatic number $\chi(\mathrm{CG_f})$ and the largest clique size $\kappa(\mathrm{CG_f})$, maximized across *all* possible flow vectors. Consequently, for the random networks generated in our simulations, it is not possible to analytically determine that imperfection ratio. At best, we can find $\frac{\chi(\mathrm{CG_f})}{\kappa(\mathrm{CG_f})}$ for *particular* flow vectors.

On the other hand, there are several inefficiencies in the distributed nature of the MAC protocol (e.g. [9,25]). When graph imperfection is significantly less than 2.155, our observation that the throughput falls off soon after the offered load exceeds the scaled clique constraints, must be because MAC inefficiency in addition to any graph imperfection is limiting the achievable rates. In fact, given an ideal scheduler, the unscaled clique constraints ought to be necessary and sufficient in Fig. 6, since we find that $\frac{\chi(\mathrm{CG_f})}{\kappa(\mathrm{CG_f})} = 1$ in this specific case. Thus, in this example, the limits on the throughput is primarily due to the inefficiency of the MAC. We are not violating the clique constraints here – they are merely being superseded by the inefficiencies in the MAC protocol.

The scaling factor of 0.46 is *required*, to account for the *worst* case imperfection in a CG, since we cannot predict the flow vectors ahead of time. In simple networks and/or few flows (e.g. Section 8.3), the effect of the graph imperfection is easily visible. In some other situations however, the effects of MAC inefficiency, rather than graph imperfection, might be the dominating cause for the gap between a flow vector satisfying the unscaled clique constraints and one that is actually achievable with a distributed MAC. Thus, the constraint-based framework presented here becomes universally practicable only when used in conjunction with an efficient MAC protocol.

9. Conclusions

This paper presents a theoretical model to predict the capacity of an *arbitrary* ad-hoc network, with a *given* set of desired flows. We model the ad-hoc network, and the underlying interference between links, as a conflict graph. We then propose two sets of constraints – the row constraints and the scaled clique constraints – to determine if a flow vector is feasible on this network. Our main contribution is to prove that each of the above constraints are sufficient for the existence a feasible schedule. We also discuss the tightness of these constraints under ad-hoc network conditions.

Our second contribution is to expand the model of the network to incorporate variations in the interference range, and consider obstructions (like buildings or hills) in the network. We extend the above proofs of sufficiency to incorporate these changes.

An important motivation to utilize the row and clique constraints is that they are localized in nature and amenable to distributed approximation algorithms. Our third contribution is to propose distributed algorithms for capacity estimation that utilize these constraints. The estimated capacity may then be utilized by other algorithms to implement admission control and QoS routing schemes in ad-hoc networks.

While this paper does not claim to propose an implementable protocol for distributed ad-hoc

QoS routing, it provides the underlying theoretical frame work that makes it possible to develop such schemes.

Acknowledgement

Special thanks to Dr. Stephanie Gerke for her advice and comments. We are also grateful to Bill Hodge for his help in running many of the laborious OPNET simulations.

References

[1] R. Gupta, Quality of Service in Ad-Hoc Networks, PhD Thesis, University of California, Berkeley, May 2005.
[2] H. Luo, S. Lu, V. Bhargavan, A new model for packet scheduling in multihop wireless networks, in: Proceedings ACM Mobicom, 2000, pp. 76–86.
[3] K. Jain, J. Padhye, V.N. Padmanabhan, L. Qiu, Impact of interference on multi-hop wireless network performance, in: Proceedings ACM Mobicom 2003, San Diego, CA, September 2003.
[4] S. Gerke, C. McDiarmid, Graph imperfection I, Journal of Combinatorial Theory, Series B 83 (2001) 58–78.
[5] P. Gupta, P.R. Kumar, The capacity of wireless networks, IEEE Transactions on Information Theory 34 (5) (2000) 910–917.
[6] M. Franceschetti, O. Dousse, D. Tse, P. Thiran, On the throughput capacity of random wireless networks, in review.
[7] M. Grossglauser, D. Tse, Mobility increases the capacity of ad-hoc wireless networks, IEEE/ACM Transactions on Networking 10 (4) (2002) 477–486.
[8] A. El Gamal, J. Mammen, B. Prabhakar, D. Shah, Throughput-delay trade-off in wireless networks, in: Proceedings IEEE INFOCOM, Hong Kong, March 2004.
[9] J. Li, C. Blake, D.S.J. De Couto, H.I. Lee, R. Morris, Capacity of ad hoc wireless networks, in: Proceedings ACM Mobicom 2001, Rome, Italy, July 2001.
[10] M. Kodialam, T. Nandagopal, Characterizing the achievable rates in multihop wireless networks, in: Proceedings ACM Mobicom 2003, San Diego, CA, September 2003.
[11] Z. Jia, R. Gupta, J. Walrand, P. Varaiya, Bandwidth guaranteed routing for ad-hoc networks with interference consideration, in: Proceedings ISCC 2005, Cartagena, Spain, June 2005.
[12] R. Gupta, Z. Jia, T. Tung, J. Walrand, Interference-aware QoS routing (IQRouting) for ad-hoc networks, in: Proceedings Globecom 2005, St. Louis, Missouri, November 2005.
[13] R. Gupta, J. Walrand, Achieving fairness in a distributed ad-hoc MAC, in: E.H. Abed (Ed.), Advances in Control, Communication Networks, and Transportation Systems, Systems and Control: Foundations and Applications Series, Springer-Birkhauser, Boston, 2005 (Book Chapter).
[14] M. Gast, 802.11 Wireless Networks: The Definitive Guide, O'reilly and Associates, 2002.
[15] R. Negi, A. Rajeswaran, Physical layer effect on MAC performance in ad-hoc wireless networks, in: Proceedings Communications, Internet and Information Technology CIIT, 2003.
[16] L. Lovasz, A characterization of perfect graphs, Journal of Combinatorial Theory, Series B 13 (1972) 95–98.
[17] A. Graf, M. Stumpf, G. Weisenfels, On coloring unit disk graphs, Algorithmica 20 (1998) 277–293.
[18] F. Harary, I.C. Ross, A procedure for clique detection using the group matrix, Sociometry 20 (1957) 205–215.
[19] J.G. Augustson, J. Minker, An analysis of some graph theoretical cluster techniques, Journal of the ACM (JACM) 17 (4) (1970) 571–588.
[20] C. Bron, J. Kerbosch, Finding all cliques in an undirected graph, Communications of the ACM 16 (1973) 575–577.
[21] R. Gupta, J. Walrand, O. Goldschmidt, Maximal cliques in unit disk graphs: polynomial approximation, in: Proceedings INOC 2005, Lisbon, Portugal, March 2005.
[22] OPNET Modeller, OPNET Technologies Inc. Available from: <http://www.opnet.com>.
[23] D.B. Johnson, D.A. Maltz, Dynamic source routing in ad-hoc wireless networks, in: T. Imielinski, H. Korth (Eds.), Mobile Computing, Kluwer, 1996, pp. 153–181.
[24] Matlab Simulation Environment, The Mathworks Inc. Available from: <http://www.mathworks.com>.
[25] G. Bianchi, Performance analysis of the IEEE 802.11 distributed coordination function, IEEE Journal on Selected Areas in Communications 18 (3) (2000).

Rajarshi Gupta is currently a Senior Engineer with Qualcomm Inc, having completed his Ph.D. in EECS at UC Berkeley. Prior to this, he completed his MS also at Berkeley, and his BS at the University of Maryland. From 1999 to 2003, he worked with Extreme Networks as a Senior Designer, where he has been the author of 10 patents.

He is interested in algorithms to ensure quality in networks – both wired and ad-hoc. This includes: analysis of network capacity; switching and scheduling mechanisms for efficient utilization of resources; and, routing algorithms to guarantee quality of service.

John Musacchio received the BS degree in electrical engineering from the Ohio State University in 1996 and the MS degree in electrical engineering from the University of California, Berkeley in 1998. From 2000 to 2003, he was with the Silicon Valley start-up Terablaze, where he helped architect a high-speed switch fabric chip-set. Since 2003 he has been at the University of California, Berkeley where he expects to complete his Ph.D. in electrical engineering by January 2005. In January 2005 he will join the Department of Information Systems and Technology Management of the University of California, Santa Cruz as an Acting Assistant Professor. His current research interests include stochastic modelling and control of queuing networks, applications of game theory in network pricing, and ad-hoc networks.

Jean Walrand received the Ph.D. degree from the Department of Electrical Engineering and Computer Sciences of the University of California at Berkeley where he is now Professor. His research interests include decision theory, stochastic processes, and communication networks. He is the author of An Introduction to Queueing Networks (Prentice Hall, 1988) and of Communication Networks: A First Course (2nd ed. McGraw-Hill,1998) and co-author of High-Performance Communication Networks (2nd ed, Morgan Kaufman, 2000). He is a Fellow of the Belgian American Education Foundation and of the IEEE and a recipient of the Lanchester Prize and of the Stephen O. Rice Prize.