

QoS-aware fair rate allocation in wireless mesh networks [☆]

Bo Wang ^{*}, Matt Mutka

Department of Computer Science and Engineering, Michigan State University, East Lansing, MI 48823, USA

Available online 26 January 2008

Abstract

In wireless mesh networks, quality-of-service (QoS) support and fair rate allocation are usually considered separately. Fair rate allocation frameworks proposed in previous work focus on elastic flows that do not have explicit service quality requirements. Such frameworks may not be applied directly within mesh networks containing real-time flows that have explicit rate and delay requirements. QoS support frameworks for wireless networks guarantee service quality of real-time flows by reserving bandwidth for them. However, the major drawbacks of existing approaches are the inefficiency and the lack of fairness consideration in bandwidth allocation, which results in poor performance of elastic flows. In this paper, we propose QUOTA (quality-of-service aware fair rate allocation), a framework that combines QoS support and fair rate allocation. QUOTA provides higher priority to real-time flows than elastic flows by reserving the necessary bandwidth for the former and fairly allocating the left-over bandwidth to the latter. The performance of QUOTA is evaluated through simulations and is compared with other QoS support and rate allocation frameworks.

© 2008 Elsevier B.V. All rights reserved.

Keywords: QoS; Fairness; Rate allocation; Wireless mesh networks

1. Introduction

Wireless mesh networking (WMN) is an emerging technology that uses wireless multi-hop networking to provide a cost-efficient way for community or enterprise users to have broadband Internet access and share network resources.

Fig. 1 shows a typical wireless mesh network. The network contains mesh routers and mesh clients [11]. Several wireless mesh routers connected through wireless links form the multi-hop backbone of the network. Mesh routers have limited mobility and some of them have wired Internet connections, which become Internet gateways. Mesh clients connect to mesh routers to access the Internet and share network resources among themselves. In order to provide stable and high-speed Internet access for mesh clients, inter-router and client-router communications usually

use different radio technologies, which effectively eliminate interference between them.

One major challenge for the wide deployment of wireless mesh networks is to provide QoS support and fair rate allocation. It is well-known that using TCP and CSMA/CA MAC protocols (e.g., IEEE 802.11 [3]) in multi-hop wireless networks results in severe unfairness [24]. Users that are farther away from the Internet gateway receive less bandwidth and are sometimes starved. In addition, QoS support for real-time applications, such as video and voice over the mesh, is still an open problem.

Flows within wireless mesh networks may be classified as real-time and elastic [33]. Real-time flows include video and voice applications that have requirements on the transmission rate and are very sensitive to latency and jitter. For example, voice over the mesh requires that the one-way latency is less than 200 milliseconds (ms) and the jitter is less than 50 ms [27]. Elastic flows are from applications that can adjust their transmission rate gradually. Such applications include file transfer, email and remote terminal [33].

In this paper, we focus on the problem of providing QoS support for real-time flows, while allocating bandwidth to elastic flows fairly. Previous work [10,14,16,21,23,28,34–38]

[☆] This research is supported in part by the US National Science Foundation under Grants No. CNS-0721441 and CNS-0524163.

^{*} Corresponding author. Tel.: +1 517 355 8152.

E-mail addresses: wangbo1@cse.msu.edu (B. Wang), mutka@cse.msu.edu (M. Mutka).

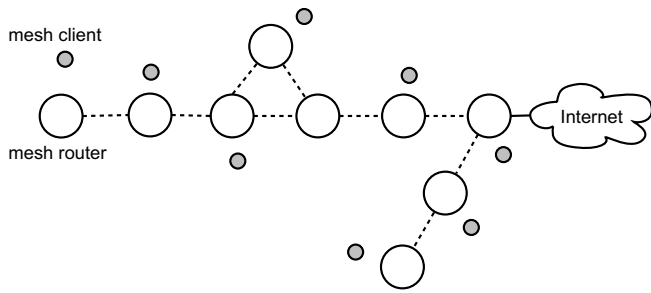


Fig. 1. A mesh network scenario.

normally consider these two problems separately by focusing either on admission control of real-time flows or rate allocation of elastic flows. These approaches cannot be applied directly in a wireless mesh network that contains both real-time and elastic flows. The major contribution of our paper is the proposed QUOTA framework providing both QoS support and fair, efficient rate allocation for wireless mesh networks.

The paper is organized as follows: In Section 2 we review the related work and highlight our contributions. Section 3 gives a detailed introduction to the QUOTA framework. Section 4 introduces the implementation details of our framework. We provide a simulation evaluation of the QUOTA framework in Section 5. Conclusions and future work are given in Section 6.

2. Related work

Providing QoS support for real-time flows in wireless networks is an active research area. SWAN [10] is a service differentiation framework for wireless ad hoc networks. The SWAN framework provides service guarantee for real-time flows by controlling the rate of elastic flows. The rate controller is deployed between the IP and the MAC layer. The flow rate control algorithm used in SWAN is additive increase, multiplicative decrease (AIMD). When the MAC layer detects packet delay that exceeds a threshold, which is determined by the delay requirement of the real-time flows, the rate control algorithm is triggered and a new transmission rate is determined. The rate adjustment is achieved through a leaky bucket traffic shaper. One drawback of SWAN is the lack of fair rate allocation for elastic flows. We will compare the performance of SWAN and QUOTA through simulation results given in Section 5 and highlight our contributions.

Another type of service differentiation framework focuses on modifying the IEEE 802.11 MAC protocol [35,37,38]. In fact, IEEE 802.11e [3] has been proposed to provide a set of QoS enhancements to the IEEE 802.11 standard. Different levels of services are achieved by using different MAC protocol parameters, such as the contention window size. However, fair rate allocation for elastic flows is not considered in these approaches. In addition, while the QUOTA framework will benefit if the underlying MAC protocol is

improved, it is important and practical to find solutions when a traditional IEEE 802.11 MAC protocol is used.

A contention graph and utility maximization based framework has been used for fair rate allocation in several research papers. Some of them [21,23] focus on rate allocation in single hop wireless networks. Others [16,28,34] pay more attention to multi-hop wireless networks. Our framework is different from the work mentioned above in three very important aspects. First, we consider real-time flow QoS support in addition to rate allocation for elastic flows. Higher priority is given to real-time flows by reserving bandwidth for them while utility maximization based rate allocation is used as a tool to allocate the left-over bandwidth among elastic flows fairly. Second, rate allocation is performed at the Internet gateway instead of relying on distributed mesh routers. To support the QoS of real-time flows in mesh networks, a centralized method is more suitable than a distributed one because it provides more accurate bandwidth reservation for real-time flows and faster, more stable rate allocation for elastic flows. This is vital for real-time flows since they are very sensitive to network conditions and normally last for a short period of time. In fact, the IEEE 802.16 (WiMAX) mesh network [4] provides a centralized mechanism for rate allocation and link scheduling. More detailed discussion on the centralized rate allocation will be given in Section 4 when we introduce the implementation details of the QUOTA framework. Third, we employ an adaptive method for accurately estimating the capacity of the network, which is the basis for efficient bandwidth reservation and rate allocation.

3. Theoretical framework

In this paper, we focus on the wireless mesh backbone formed by several mesh routers. We consider the scenario where all the mesh routers are deployed by a single organization, for example, an Internet service provider (ISP). In this case, all the mesh routers are configured and controlled by the organization. For the ease of presentation, we consider a scenario shown in Fig. 2, which is a branch of the wireless mesh network shown in Fig. 1. Different branches can be configured to use orthogonal channels to eliminate interference between them. This scenario is similar to the one that was studied in [24]. In this paper, we assume that wireless links are bidirectional. In addition we do not

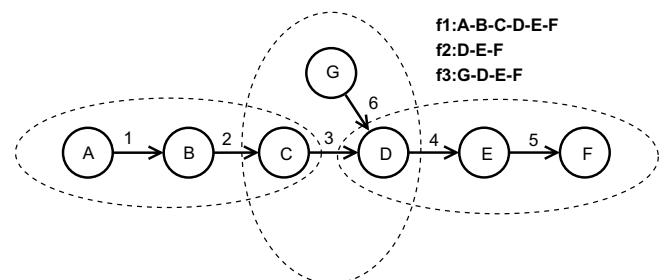


Fig. 2. A branch of the mesh network.

consider adaptive link capacity, so we assume the capacities of all links are equal. Note that similar link capacity assumptions are made in [21,34]. Our model can be easily modified to take different link capacities into consideration, such as in [16].

In the mesh branch shown in Fig. 2, node F is the Internet gateway. We denote the set of *network flows* (end-to-end application flows) as F . Also we denote the set of *link flows* (flows between directly connected nodes) as L . Every network flow $f \in F$ consists of one or more link flows. Every link flow $l \in L$ carries at least one network flow. The legend of Fig. 2 shows three network flows in the mesh branch.

3.1. Link contention graph and rate allocation feasibility

In an IEEE 802.11 MAC based wireless network, two link flows contend for channel access if the source or destination of one link flow is within the interference range of the source or destination of another one [21,34]. We can define a *link contention graph* $G(V, E)$ based on the contention relationship between different link flows. The vertex set V contains all the link flows in the network. An edge in set E indicates that two vertices (two link flows) contend with each other. Fig. 3 shows the contention graph of the mesh branch shown in Fig. 2.

The link contention graph captures the interference among different link flows. An important concept associated with the link contention graph is the *maximal clique*. A complete subgraph of a graph is called a clique. A maximal clique is defined as the clique that is not contained in any other clique [12]. We denote the set of maximal cliques of a contention graph as C . The maximal cliques can be obtained from the link contention graph using the Bierston algorithm [12]. In the link contention graph given in Fig. 3, we have three maximal cliques, $C_1 = \{1, 2, 3\}$, $C_2 = \{2, 3, 4, 6\}$ and $C_3 = \{3, 4, 5, 6\}$. In fact, a maximal clique represents a contention region in which all link flows interfere with each other. The dotted ellipses shown in Fig. 2 illustrate contention regions corresponding to the above maximal cliques. A maximal clique can also be considered as a limited resource shared and contended by different link flows within it. For any maximal clique, at any time, only a single link flow can be active. The above constraint is denoted as the *clique constraint*. In the rest of this paper, we will simply use clique to refer to maximal clique.

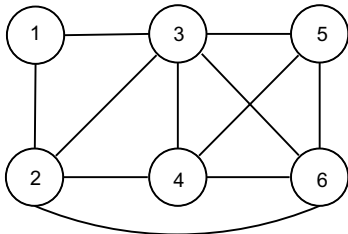


Fig. 3. Link contention graph of the mesh network scenario shown in Fig. 2.

In order to formally state the clique constraint, let $\mathbf{x} = (x_f, f \in F)$ denote the *rate allocation vector* of network flows; let $\mathbf{y} = (y_l, l \in L)$ denote the rate allocation vector of link flows derived from \mathbf{x} . Also let b_l denote the link capacity for link flow l when it is active in isolation. Rate vector \mathbf{x} and \mathbf{y} are said to be *scheduling feasible* if they can be scheduled under the clique constraint. The scheduling protocols considered here are medium access control (MAC) layer protocols. In this paper, we use the IEEE 802.11 MAC as the underlying scheduling protocol.

We denote the necessary and sufficient condition that guarantees scheduling feasibility as the *scheduling feasibility constraint*. In order to derive this constraint, let us first define a matrix $\mathbf{R} = \{R_{lf}\}$:

$$R_{lf} = \begin{cases} 1, & \text{if link flow } l \text{ carries network flow } f \\ 0, & \text{otherwise} \end{cases}$$

Also define matrix $\mathbf{Q} = \{Q_{nl}\}$:

$$Q_{nl} = \begin{cases} 1, & \text{if clique } n \text{ contains link flow } l \\ 0, & \text{otherwise} \end{cases}$$

As an example, for the mesh scenario shown in Fig. 2 and the associated link contention graph shown in Fig. 3, we have

$$\mathbf{R} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

and

$$\mathbf{Q} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

It has been shown that [16,21], for a perfect contention graph [18], the scheduling feasibility constraint can be written as

$$\sum_{l: Q_{nl}=1} \frac{y_l}{b_l} \leq 1, \quad \forall n \in C$$

where

$$y_l = \sum_{f: R_{lf}=1} x_f, \quad \forall l \in L$$

In fact, the constraint states that the sum of the normalized rate of all link flows in the clique, cannot exceed the *normalized clique capacity*, which is 1 [21].

However, it is difficult and expensive to check if the link contention graph is perfect. It has been shown that [21], for a general graph, the scheduling feasibility constraint can be written as

$$\sum_{l:Q_{nl}=1} \frac{y_l}{b_l} \leq \frac{2}{3}, \quad \forall n \in C$$

by reducing the normalized clique capacity to 2/3.

Based on our link capacity assumption given at the beginning of this section, we denote the capacity of all wireless links to be b . So we can write the scheduling feasibility constraint in another form

$$\sum_{l:Q_{nl}=1} y_l \leq \frac{2}{3}b, \quad \forall n \in C \tag{1}$$

where the right-hand side of the inequality is denoted as the *clique capacity*.

Scheduling feasibility does not guarantee that the rate vector is *throughput feasible*, which means the throughput of a flow equals to the allocated rate. This is due to the inefficiency of the underlying scheduling protocols. For example, due to the random nature of the IEEE 802.11 MAC, some idle time is wasted during the back off period. We introduce the *effective clique capacity* τ_n for every clique $n \in C$ such that the constraint

$$\sum_{l:Q_{nl}=1} y_l \leq \tau_n, \quad \forall n \in C$$

guarantees the throughput feasibility of the rate vector. We denote the above constraint as the *throughput feasibility constraint*. To simplify the representation of the throughput feasibility constraint, let us define the *clique-flow matrix* $\mathbf{P} = \mathbf{QR}(\mathbf{P}_{nf} = Q_{nl}R_{lf})$. In fact P_{nf} represents the number of link flows in clique n that carries network flow f . As an example, for the mesh scenario shown in Fig. 2 and the associated link contention graph shown in Fig. 3, we have

$$\mathbf{P} = \begin{bmatrix} 3 & 0 & 0 \\ 3 & 1 & 2 \\ 3 & 2 & 3 \end{bmatrix}$$

Denote $\boldsymbol{\tau}$ as the vector of effective clique capacities. Then the throughput feasibility constraint can be written as

$$\mathbf{P}\mathbf{x} < \boldsymbol{\tau} \tag{2}$$

The effective capacity of a clique depends on several factors, such as the underlying scheduling protocol, the number of competing link flows in this clique and the location of link flows [23]. It is difficult to determine the effective capacity of a clique in advance. In our framework, the effective capacity of a clique has the maximum value of $(2/3)b$ and is adaptively estimated according to the network conditions.

3.2. Admission control

Assume that a network planner has a global view of the contention graph and the traffic pattern, and the network flow set F is divided into real-time flow set F^r and elastic flow set F^e . We give a higher priority to real-time flows over elastic flows by performing admission control and band-

width reservation first and allocate the left-over bandwidth to elastic flows afterwards. We sequentially process all real-time flows. The order of processing will be discussed in Section 4.2. For a real-time flow $f \in F^r$, it is admitted if

$$P_{nf}r_f < \tau'_n, \quad \forall n \in C$$

Otherwise, it is rejected. P_{nf} denotes the number of link flows in clique n that carries network flow f ; r_f is the rate requested by real-time flow f . So $P_{nf}r_f$ represents the bandwidth used by flow f in clique n . τ'_n is the available bandwidth of clique n , which is set to the effective capacity of the clique at the beginning of the admission control process. If real-time flow f is admitted, which means its requested rate can be supported by all cliques, the rate allocation x_f is equal to r_f ; otherwise, $x_f = 0$. In addition, the available bandwidth of every clique n is updated as $\tau'_n = \tau'_n - P_{nf}x_f$. The bandwidth reservation of real-time flows will be discussed in the following subsection.

The above admission control strategy gives absolute priority to real-time flows. However, other admission control strategies can be adopted according to the policy of the mesh network operator. For example, before admission control, a certain proportion of the available bandwidth of every clique can be reserved for elastic flows to prevent their starvation.

3.3. Utility maximization based rate allocation

After the admission control is completed for real-time flows, we may allocate bandwidth to elastic flows. In order to guarantee fairness among different elastic flows, we use a well-developed utility maximization framework. This framework was originally proposed for Internet congestion control [26,29] and recently has been used for fair rate allocation in wireless networks [16,21,23,34].

We associate a utility function $U_f(x_f)$ to every elastic flow $f \in F^e$ [33]. Further, we assume that this function is increasing, strictly concave and twice continuously differentiable [16,21,34]. It has been shown that [26,32], by maximizing $\sum_{f \in F^e} U_f(x_f)$, a certain targeted fairness is achieved among elastic flows. Furthermore, if we choose the utility function $U_f(x_f) = \ln(x_f)$ for every elastic flow, proportional fairness is achieved [26,32]. More formally, the objective can be written as

$$\mathbf{P} : \max_{x_f \geq 0} \sum_{f \in F^e} U_f(x_f), \quad \text{subject to } \mathbf{P}\mathbf{x} \leq \boldsymbol{\tau} \tag{3}$$

The above problem is referred to as the system primal problem. It is a typical convex optimization problem [13], which can be solved by using the Lagrange duality [26,29]. The dual problem is defined as

$$\mathbf{D} : \min_{\lambda \geq 0} D(\boldsymbol{\lambda}) \tag{4}$$

where

$$\begin{aligned}
D(\lambda) &= \max_{x_f \geq 0} \left(\sum_{f \in F^e} U_f(x_f) - \lambda^T (\mathbf{P}\mathbf{x} - \boldsymbol{\tau}) \right) \\
&= \sum_{f \in F^e} \max_{x_f \geq 0} \left(U_f(x_f) - x_f \sum_{n \in C} \lambda_n P_{nf} \right) \\
&\quad + \sum_{n \in C} \lambda_n \tau_n - \sum_{f \in F^r} x_f \sum_{n \in C} \lambda_n P_{nf}
\end{aligned}$$

Let us define

$$\mu_f = \sum_{n \in C} \lambda_n P_{nf}$$

In fact, the Lagrange multiplier λ_n can be interpreted as the shadow price [26] of clique n , which is the cost of a unit flow accessing the channel in clique n [34]. Also $x_f \mu_f$ may be interpreted as the total price charged for flow f for transmitting at rate x_f .

To solve the dual problem, we use an iterative algorithm with the help of the gradient projection method [34]. The algorithm involves all network flows and maximal cliques and is given an initial rate allocation vector \mathbf{x} . For real-time flow f , x_f is determined by the admission control process described in the previous subsection and does not change during the iterative process. For elastic flows, initial rate can be randomly chosen.

At every iteration, for every clique n , it receives the rate information of all flows f where $P_{nf} \neq 0$ and updates the shadow price λ_n using

$$\lambda_n(t+1) = \left[\lambda_n(t) - \gamma \frac{\partial D(\lambda(t))}{\partial \lambda_n} \right]^+$$

where γ is the step size and t is the iteration number. Since

$$\frac{\partial D(\lambda)}{\partial \lambda_n} = \tau_n - \sum_{f \in F^e} x_f P_{nf} - \sum_{f \in F^r} x_f P_{nf}$$

we have

$$\lambda_n(t+1) = \left[\lambda_n(t) - \gamma \left(\tau_n - \sum_{f \in F^e} x_f(t) P_{nf} - \sum_{f \in F^r} x_f P_{nf} \right) \right]^+ \quad (5)$$

The above equation reflects the law of supply and demand [21,34]. When the rate demand exceeds (less than) the effective capacity of clique n , the cost of a unit flow accessing the channel in clique n increases (decreases). The above equation also shows the fact that every clique n reserves necessary bandwidth, $\sum_{f \in F^r} x_f P_{nf}$, for real-time flows. Note that the rates of real-time flows do not change during the iteration, so the bandwidth reservation will not be affected by the rate allocation of the elastic flows. After the price update, the clique shadow price is reported to the source of all elastic network flows f where $P_{nf} \neq 0$.

For every elastic flow f , the source of this flow receives shadow price information from all maximal cliques n where $P_{nf} \neq 0$. Then a new transmission rate of the flow is obtained by solving the following problem

$$\max_{x_f \geq 0} (U_f(x_f) - x_f \mu_f) \quad (6)$$

which can be interpreted as maximizing the net utility (utility minus cost). Since function $U_f(x_f)$ is strictly concave, the unique solution of the above problem exists and can be expressed as

$$x_f^* = U_f^{-1}(\mu_f)$$

Since $U_f(x_f) = \ln(x_f)$, the maximizer $x_f^* = \frac{1}{\mu_f}$. The new rate information of flow f is reported to all maximal cliques n where $P_{nf} \neq 0$ and will be used in the next iteration.

It has been shown that [16,34], by choosing the appropriate step size γ , starting from any initial rate vector \mathbf{x} , the above iterative algorithm will converge to the optimal solution $(\mathbf{x}^*, \boldsymbol{\lambda}^*)$, and the solution is primal-dual optimal, which means \mathbf{x}^* is also the optimal rate vector for the primal problem.

The above discussion introduces the general idea of our framework. However, in order to use this framework in a wireless mesh network, there are many practical issues that need to be considered. First, the construction of the contention graph and its maximal cliques is a challenging problem. Second, it is a difficult task to perform clique price update since the clique is a virtual concept. Third, the communication between the sources of network flows and the cliques should be reliable enough to guarantee the correct execution of the algorithm. Last, in order to guarantee the throughput feasibility and the QoS of real-time flows, we need an efficient on-line algorithm to estimate the effective clique capacities according to the network conditions. In the next section, we introduce the implementation of the QUOTA framework by solving the above-mentioned practical issues.

4. Framework implementation and practical issues

The implementation of the framework executes in a round-by-round fashion. Every round has a fixed length duration consisting of two phases: distributed clique construction and centralized admission control and rate allocation. The detailed description of these two phases and the associated control messages are explained in the following subsections.

It is reasonable to assume that all branches share a common control channel that is orthogonal to the channels used for data transmission. The control channel can be used for mesh backbone management. It will also be used by QUOTA for control message exchange. The multi-interface and multi-channel mesh network architecture has been proposed and investigated in both academia and industry [5,8,6,7,9,24] and becomes essential for the wide deployment of the wireless mesh networks. The use of a dedicated control channel can improve the reliability of the framework and leave more bandwidth for data transmission. When the control channel for the mesh backbone is not available, QUOTA control messages have to compete with

data packet transmissions, which results in potential control message loss and less bandwidth for data transmission. In this case, a high priority queue can be used for control messages when packet transmissions are scheduled. We will compare the performance of QUOTA with and without a dedicated control channel in Section 5.

4.1. Distributed clique construction

The framework presented in Section 3 requires the construction of a global link contention graph and its maximal cliques. The Bierstone algorithm used for the maximal clique problem has exponential time complexity [25]. In our implementation, we use a distributed maximal clique construction method similar to [20,21,34] through which we

obtain the maximal cliques without the construction of a global link contention graph. Since every mesh router has limited number of neighbors, even with exponential time complexity, to obtain the maximum clique set of the local link contention graph is fast.

Every link flow in the network has an associated local link contention graph that contains itself and all other link flows that interfere with it. Local link contention graphs are subgraphs of the global link contention graph and they are usually overlapping. We let individual mesh routers construct the local link contention graphs for all link flows that originate from it. The mesh router then applies the Bierstone algorithm [12] on the contention graphs to obtain a set of maximal cliques. It has been shown that, the collection of maximal cliques of different local link contention graphs gives us the complete set of maximal cliques of the global link contention graph [34].

In order to construct the local link contention graph of a given link flow, the mesh router needs to know the information about all other link flows that are in the interference range of the sender or the destination of this flow. In our implementation, the interference range is set to be identical to the transmission range so the mesh router can construct the local link contention graph via control message exchange. We use two types of messages: *beacon messages* and *link messages*. Similar types of messages are used in [20,21]. A beacon message is associated with a link flow and contains the source and destination identifiers of this link flow. Similarly a link message is associated with a link flow and contains the information

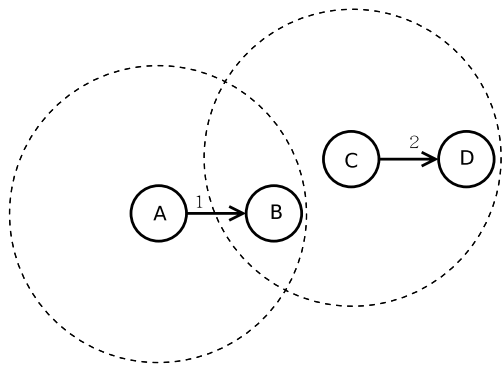


Fig. 4. Local link contention graph construction example.

	node A (link flow 1)	node B (link flow 2)	node C (link flow 3)	node D (link flow 4)	node E (link flow 5)	node G (link flow 6)
local contention graph						
maximal cliques						

Fig. 5. Distributed clique construction example. The associated network topology is shown in Fig. 2; the associated global link contention graph is shown in Fig. 3.

of itself and all other flows that interfere with it. Every mesh router periodically broadcasts beacon messages and link messages that are associated with all the link flows originate from it or are destined to it. Link messages are constructed based on the information obtained from beacon messages so they are sent less frequently than beacon messages. In order to see why we need two types of messages and how these messages are used, let us consider a simple example shown in Fig. 4. A similar example is given in [20]. Dotted circles in the figure shows the transmission range of node A and node C. Assume node A needs to construct the local link contention graph of link flow 1. Link flow 2 should be included in the contention graph since node B is in the transmission range of node C. By receiving beacon messages from node C, node B knows that flow 2 interferes with flow 1. However, node A cannot receive beacon messages from node C since it is out of the transmission range of node C. By receiving link messages from node B, node A then knows the information about link flow 2 and includes it in the contention graph of link flow 1.

At the end of the clique construction phase, each mesh router sends *clique messages* to the gateway node. Each clique message is associated with a link flow that originates from the mesh router and contains three parts: the source and destination of this flow; all the maximal cliques to which this link flow belongs; network flows that this link flow carries. If the mesh router is the source of a real-time network flow, the rate and delay requirements of this flow are also included. Note that elastic flows do not have specific rate and delay requirements. By receiving all these clique messages, the gateway node is able to construct the set of maximal cliques (denoted as C) of the global link contention graph, and the set of network flows (denoted as F). In addition, based on the information of network flows contained in the clique messages, it can construct the clique-flow matrix P .

Fig. 5 shows an example of the clique construction process. The corresponding network topology and traffic pattern are shown in Fig. 2. The associated global link contention graph is shown in Fig. 3. In this scenario, each mesh router only has one local link flow that originates from it. The row “local contention graph” of the table shows the local contention graphs constructed at different mesh routers. The row “maximal cliques” shows the maximal clique decomposition at different mesh routers.

4.2. Centralized admission control and rate allocation

Based on the information collected during the clique construction phase, the gateway node performs admission control and rate allocation using the algorithm described in Section 3. In fact, the gateway node can be considered as a simulator that executes the above-mentioned algorithm. The input to the simulator consists of the maximal clique set C , network flow set F , the clique-flow matrix P and the QoS requirements of real-time flows. The output

of the simulator is the converged rate allocation vector. For a real-time flow, if it is rejected, the rate assignment will be zero; otherwise, the rate assignment is the rate requested by the source. In order to reduce the probability that admitted real-time flows get rejected in later rounds, during the admission control process described in Section 3, we give higher priority to real-time flows admitted in previous rounds by processing their rate requests before newly arrived real-time flows.

When the rate allocation vector is available, the gateway node sends *rate messages* to the source of each network flow. Rate messages contain the assigned rate of the associated network flow. By receiving the rate message, the source of a network flow knows the rate to use for the next round.

As explained in Section 1, the motivation of relying on the gateway node instead of several mesh routers to perform the admission control and rate allocation is for better QoS support of real-time flows. A distributed implementation [21,34] would require several rounds to perform the iterative process described in Section 3.3 and to find the correct effective clique capacities described in the following subsection. Each round of the distributed implementation corresponds to one iteration of the rate allocation algorithm and it involves information exchange about flow rate and clique price among wireless routers. The long process of finding the optimal rate allocation and determining correct effective clique capacities, which can last for tens of seconds [34], has a detrimental effect on the QoS of real-time flows. In contrast, executing the admission control and rate allocation algorithm virtually at the gateway node yields an optimal rate allocation in a single round and determines the correct effective clique capacities in a few rounds. We will show the benefits of using a centralized rate allocation through simulation results in Section 5.

The utility maximization based rate allocation problem is a convex optimization problem. It has been pointed out that using a centralized algorithm, the optimal solution can be obtained in worst-case polynomial-time complexity [22,13]. Therefore, our centralized implementation for the rate allocation phase is scalable.

4.3. Adaptive effective clique capacity

As we mentioned in Section 3, the effective clique capacity cannot be determined in advance. In our implementation, we use a measurement-based method to dynamically estimate the effective capacities of maximal cliques. When there are no real-time flows in the network, we use a conservative value $0.6b$ as the clique capacity. This value is slightly smaller than $(2/3)b$ that guarantees scheduling feasibility as shown in Eq. (1).

When there are real-time flows in the network, the receiver of any real-time flow measures the average delay of this flow during the clique construction phase and this information is attached to the clique messages sent to the gateway. The effective clique capacity adjustment is performed

before admission control and rate allocation. If the maximal clique set and the traffic pattern are the same as the last round, the initial effective capacity of every maximal clique is set to the value determined in the last round. Then for every real-time flow f , the gateway node checks that if the average measured delay exceeds the delay requirement. If yes, for every clique n , the effective clique capacity τ_n will be reduced by $P_{nf}a$, where a is a small constant. This is equivalent to reserving extra bandwidth $P_{nf}a$ for the real-time flow f in clique n as shown in Eq. (5). If the maximal clique set or the traffic pattern changes in a new round, the effective capacity of every maximal clique will be set to $0.6b$ and no further capacity adjustment will be performed in this round.

The major purpose of dynamically adjusting the effective clique capacities is to meet the rate and delay requirements of the real-time flows by guaranteeing throughput feasibility. By reserving extra bandwidth for real-time flows, we expect to reduce the delay and at the same time meeting its rate requirement. However, when there are only elastic

flows in the network, we use a pre-determined conservative value as the effective clique capacity, which may not guarantee throughput feasibility.

5. Performance evaluation

In this section, we present simulation results for the QUOTA framework in mesh networks consisting of real-time and elastic flows. Also we compare the performance of QUOTA with SWAN. We choose SWAN since its QoS support for real-time flows is based on the rate control of the elastic flows, which is very similar to QUOTA. The evaluation metrics include the delay and loss rate of real-time flows, the aggregate throughput of the network and the fairness index of elastic flows. Given a rate vector \mathbf{x} of size n , the fairness index [17] is defined as

$$F(\mathbf{x}) = \frac{(\sum x_i)^2}{n(\sum x_i^2)}$$

5.1. Simulation setup

We use NS2 [2] (version 2.29) for our simulation. For the QUOTA framework, the duration of each round is set to be 4 s. In the distributed clique construction phase, 2 s are used for beacon message and link message exchange. Each mesh router periodically sends beacon messages and link messages. The inter-message intervals of beacon and link messages are 0.2 and 0.9 s, respectively. Another one second is used for sending clique messages to the Internet gateway. After admission control and rate allocation, the gateway node sends rate messages to the

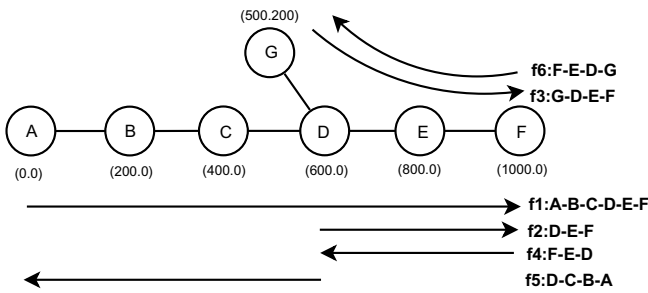


Fig. 6. A single-branch scenario.

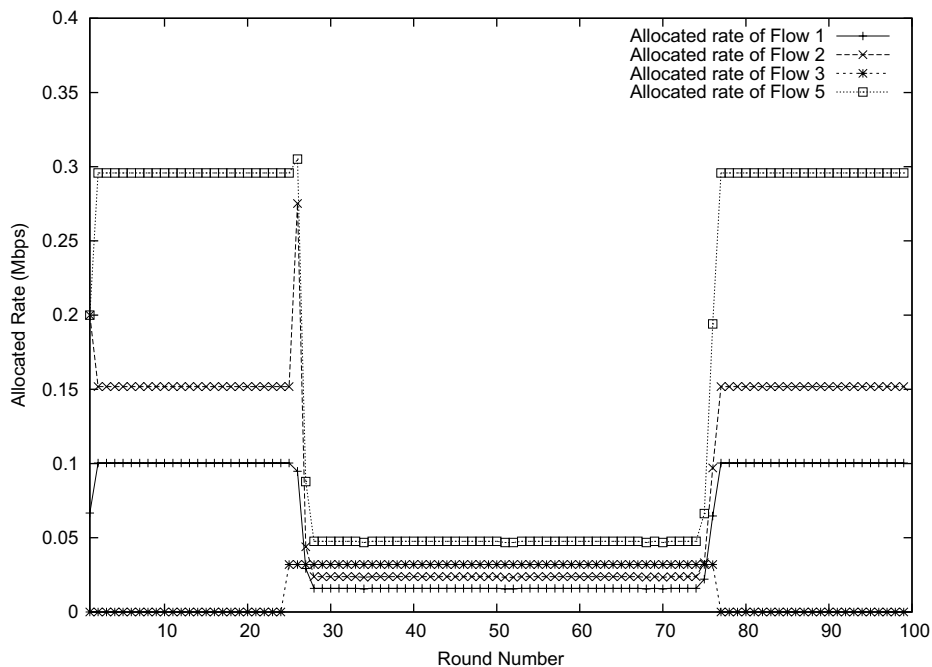


Fig. 7. Allocated rate of different flows at different rounds under QUOTA.

source of each network flow. If the source fails to get the rate message during a round, it will continue using the rate assigned at the last round. Some part of the QUOTA implementation in NS2 is based on the code used in [21]. The implementation code of the SWAN framework is obtained from the authors' website.

5.2. Simulation results of a single-branch scenario

The simulated single-branch scenario is shown in Fig. 6. All nodes use two 2 Mbps 802.11 radios tuned to orthogonal channels used by data transmission and control message exchange. The transmission range and interference range are set to be 250 m. The coordinates of nodes are given in the figure. In this scenario, we have six network flows. The routes for these flows are manually set and are shown in the figure. Flows f_1, f_2, f_4 and f_5 are elastic flows with constant bit rate (CBR) traffic source. The packet size is set to be 1000 bytes. The sending rate of the traffic source is adjusted at the application layer under QUOTA framework while SWAN adjusts the source rate using a traffic shaper between the IP and MAC layer. Note that both QUOTA and SWAN can work with a TCP-based traffic source. However, in this paper we consider TCP as a form of rate allocation for elastic flows [26,34] and use it as a baseline for comparison. Elastic flows last for the whole simulation period, which is 400 s. Real-time flows f_3 and f_6 are from voice applications simulated as CBR traffic sources with 32 kbps rate requirement and 50 ms maximal tolerable one-way delay. The packet size is set to be 80 bytes. Real-time flows start at 100 s and end at 300 s.

Fig. 7 shows the rate allocation at different rounds under the QUOTA framework. The rate allocations of flow f_4 and

flow f_6 are not shown in the figure since they are the same as flow f_2 and flow f_3 , respectively. During [0, 100] seconds, there are no real-time flows in the network and the available bandwidth is allocated to elastic flows. At 100 s, two real-time flows enter the network and the rate allocation is quickly adjusted to meet the QoS requirements of them. When real-time flows leave the network at 300 s, the available bandwidth for elastic flows increases and the new rate allocation is the same as that in the period [0, 100]. Fig. 8 shows the throughput of flows at different rounds under the QUOTA framework. The throughput of flow f_4 and flow f_6 are not shown in the figure since they are similar to flow f_2 and flow f_3 , respectively. We observe that before real-time flow enters the network, the rate allocation is not throughput feasible. Starting at 100 s, the effective clique capacities are adjusted according to the delay requirements of real-time flows. After a few rounds, the rate requirement of real-time flows is satisfied and the loss rates of real-time flows are under 1% during the period [100, 300]. The rate allocation of elastic flows also becomes throughput feasible during this period. Fig. 9 shows the average delay of real-time flows in logarithmic scale at different rounds under the QUOTA framework. When real-time flows enters the network, the delay requirement is not satisfied. After a few rounds of clique capacity adjustment, the delays of real-time flows are under the maximal tolerable value and are stably maintained during the whole session.

We also simulate SWAN and TCP in the single-branch scenario. For SWAN, we tuned the maximum sending rate of elastic flows and the delay threshold mentioned in the Section 2 to make sure that real-time flows are admitted and their delay and rate requirements are successfully met. When TCP is used for elastic flows, both rate and

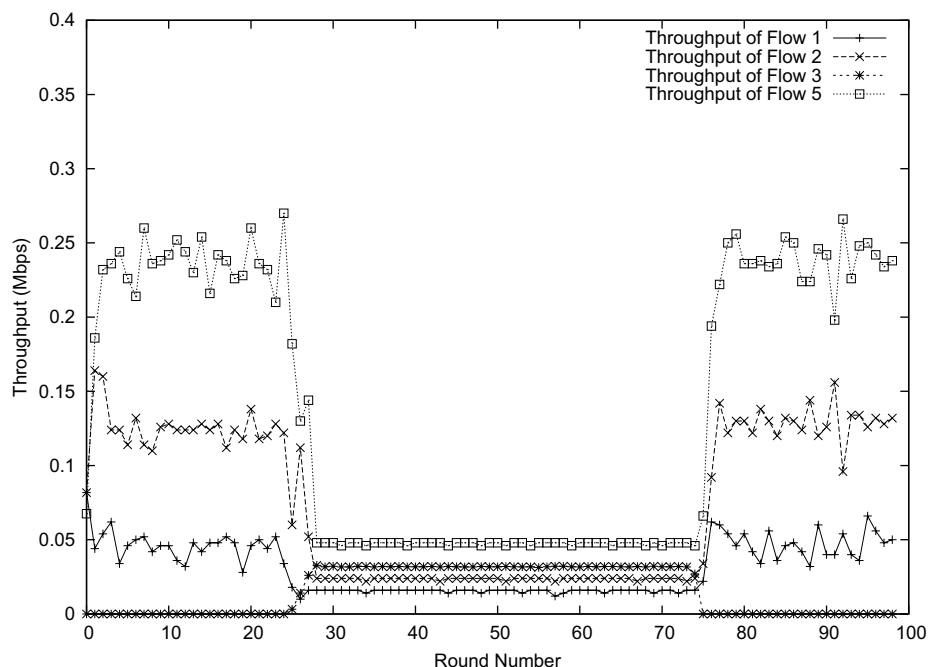


Fig. 8. Throughput of different flows at different rounds under QUOTA.

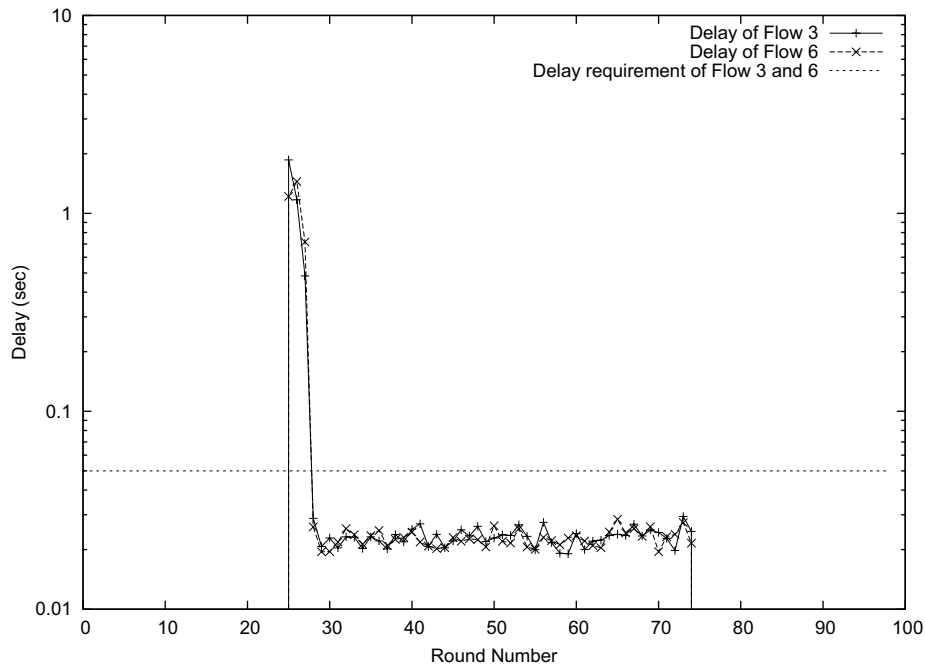


Fig. 9. Average delay of two real-time flows at different rounds under QUOTA.

Table 1
Performance comparison of the single-branch scenario (CBR real-time traffic)

	TCP	SWAN	QUOTA
Aggregate throughput (kbps)	406	297	385
Average delay of real-time flows (ms)	431	49	46
Fairness index	0.62	0.46	0.81

delay requirements of real-time flows are violated, which shows the necessity of using admission control and bandwidth reservation to support the service requirements of real-time flows. Table 1 shows the aggregate throughput of all flows, the average delay of real-time flows and the fairness index of elastic flows under QUOTA, SWAN and TCP. The aggregate throughput under QUOTA is higher than SWAN and slightly smaller than TCP. In addition, the fairness index of elastic flows under QUOTA is greatly improved over SWAN and TCP. In previous work on fair rate allocation for elastic flows, utility maximization based framework usually achieves higher aggregate throughput and better fairness than TCP [23]. However, QUOTA supports QoS of real-time flows by reducing the available bandwidth for elastic flows, which may result in a smaller aggregate throughput than TCP as seen in this scenario. QUOTA outperforms SWAN in two aspects: First, to support the same level of QoS for real-time flows, QUOTA maintains more bandwidth for elastic flows than SWAN. Second, QUOTA allocate bandwidth more fairly than SWAN.

Here are some reasons for the performance gap between SWAN and QUOTA. First, SWAN performs rate control of elastic flows on a per-node basis by using a traffic shaper

between the IP and MAC layer; QUOTA directly adjusts the sending rate of elastic flows at the application layer, which provides more accurate per-flow rate adjustment. Second, the delay threshold value used in SWAN, which triggers the rate control of elastic flows, only reflects the contention level of a single maximal clique (contention region). For networks with multi-hop flows spanning several maximal cliques, it is difficult to choose an appropriate delay threshold according to the delay requirement of the real-time flow. By using the link contention graph and maximal clique decomposition, QUOTA's bandwidth reservation for real-time flow is more flexible and efficient. The rate allocation of elastic flows is fairer with the help of the utility maximization framework.

The voice traffic can also be modeled as an ON/OFF source with exponential distributed ON and OFF period [19]. We perform simulation using traffic sources with exponential ON and OFF period for real-time flows f_3 and f_6 . The average length of ON and OFF period is 300 ms. Packets are generated during ON periods at a constant bit rate of 32 kbps. Other simulation parameters are the same as the previous scenario with CBR real-time traffic. Since the length of the ON or OFF period is much smaller than the interval of a round in QUOTA, we use the rate at the ON period as the rate requirements in QUOTA. This approach is conservative in the sense that it reserves bandwidth for real-time flows even they are in the OFF period. However, it is efficient in maintaining the QoS of real-time flows.

Table 2 shows the aggregate throughput of all flows, the average delay of real-time flows and the fairness index of elastic flows under QUOTA, SWAN and TCP. Compared with simulation results of the previous scenario with CBR

Table 2
Performance comparison of the single-branch scenario (VBR real-time traffic)

	TCP	SWAN	QUOTA
Aggregate throughput (kbps)	449	304	355
Average delay of real-time flows (ms)	401	46	36
Fairness index	0.6	0.47	0.8

real-time traffic, we observe that SWAN and TCP take advantage of the additional bandwidth given by the OFF period of the real-time traffic while QUOTA does not. However, the performance benefits of QUOTA are still obvious.

5.3. Simulation results of multi-branch scenarios

In this subsection, we present the simulation results of multi-branch wireless mesh network scenarios. We simulate a flat area of 600 m by 600 m with 40 randomly positioned stationary wireless nodes. In addition, a gateway node is placed in the center of the network. The network is divided into four branches and each branch contains one maximal clique. Different branches use orthogonal channels with 2 Mbps capacity for data transmission while sharing a single control channel for QUOTA control messages. Routes between nodes are determined offline using shortest-path routing algorithms. The simulation results presented are the average values of 5 randomly generated network topologies.

In these scenarios we simulate network flows that originate from or are destined to the gateway, and we vary the number of network flows in a branch from 3 to 7 in each topology. There is one real-time flow in each branch.

Real-time flows are from video applications simulated as CBR traffic sources with rate requirement 200 kbps and 20 ms maximal tolerable delay. The packet size is set to be 512 bytes. Elastic flows are generated using CBR traffic sources with packet size of 1000 bytes. Each simulation lasts for 400 s.

Fig. 10 shows the aggregate throughput of all flows with different number of flows in each branch. The figure shows that QUOTA always achieves higher aggregate throughput than SWAN. When the number of flows in a branch is large, TCP achieves higher aggregate throughput than QUOTA and SWAN. Fig. 11 shows the average delay of real-time flows in logarithmic scale. We observe that both the QUOTA and SWAN satisfy the delay requirements of real-time flows while TCP cannot provide the service guarantee. The average fairness index of branches are shown in Fig. 12. We observe that QUOTA has a better fairness index than both SWAN and TCP, especially when the number of flows in each branch is large. The above simulation results indicate that, while maintaining the same level of delay for real-time flows, QUOTA allocates bandwidth to elastic flows more efficiently and more fairly than SWAN.

Finally, we evaluate the overhead of the QUOTA framework by relaxing the assumption of a dedicated control channel. In a new set of simulations, QUOTA control messages are sent through the respective data transmission channels of different branches. The simulation results are presented as “QUOTA without control channel” in the Figs. 10–12. We observe that without the dedicated control channel, QUOTA still successfully maintains the QoS of real-time flows and achieves high fairness index. Since control message transmissions

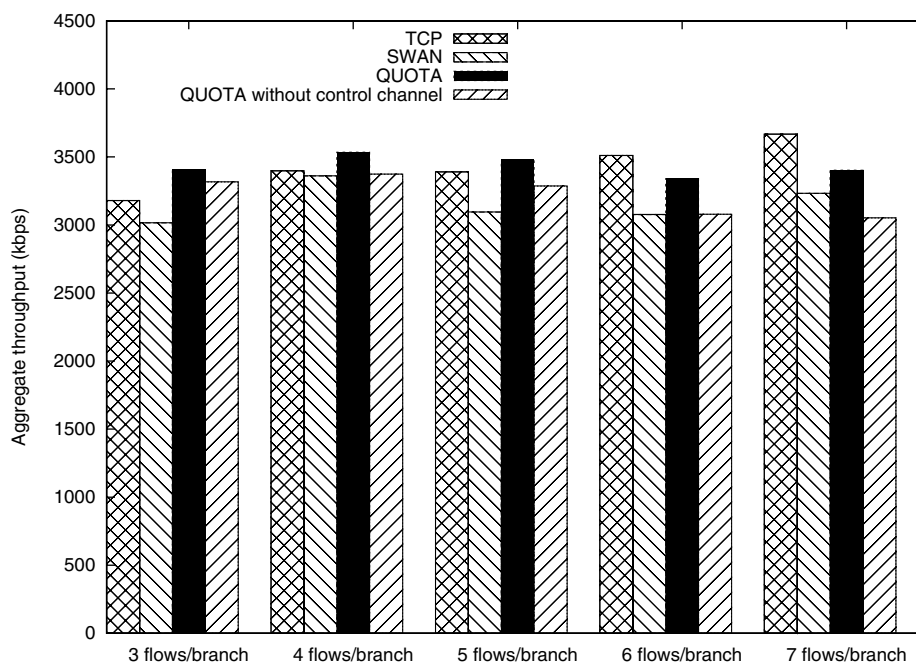


Fig. 10. Aggregate throughput of network flows.

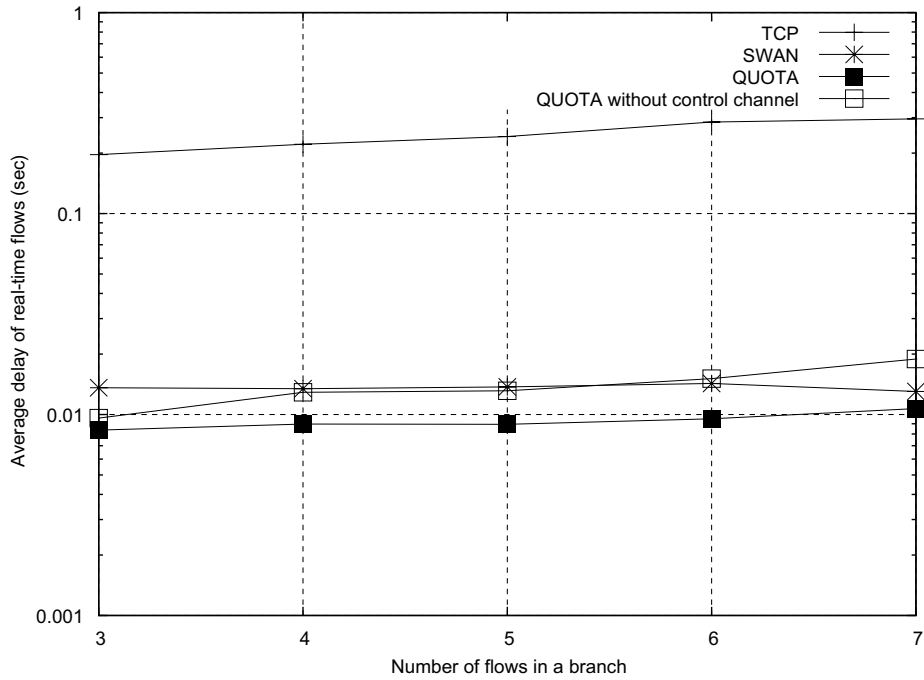


Fig. 11. Average delay of real-time flows.

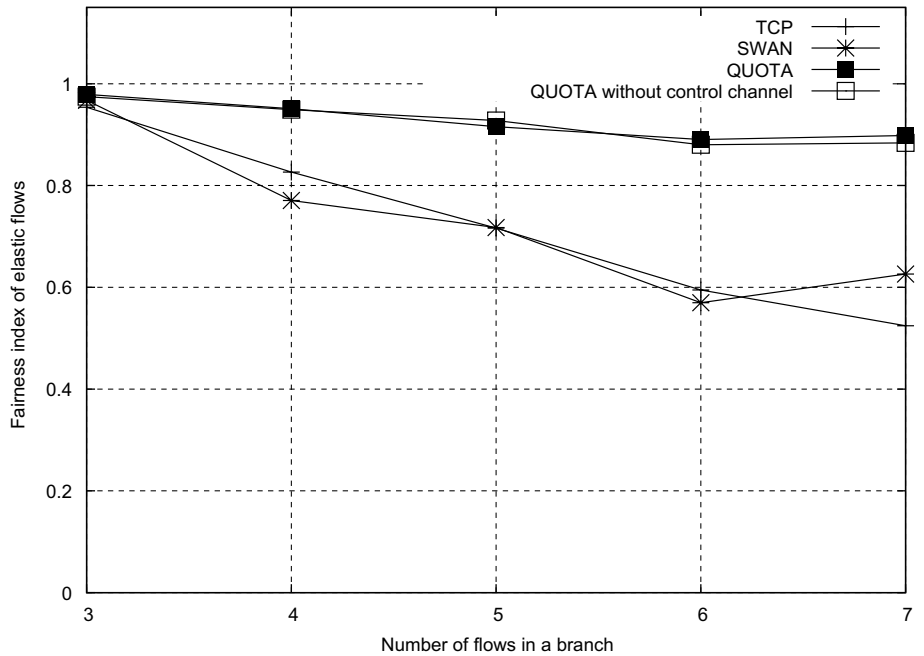


Fig. 12. Average fairness index of branches.

compete with data packet transmissions, the aggregate throughput of QUOTA without dedicated control channel is lower than QUOTA with dedicated control channel. The normalized message overhead of QUOTA is shown in Fig. 13. The normalized overhead is defined as the ratio between the control messages transmitted in bytes and data packets received in bytes. The message overhead increases with the increase of the number of flows in a branch. For a mesh branch consisting of large

number of flows, using a dedicated control channel would be more appropriate. However, typical mesh network deployment limits the number of mesh routers that connect to a gateway. For example, in Google’s Muni WiFi [30,1] mesh network deployment, small clusters of 6–8 access points are connected to a gateway using the same frequency channel. Most of the nodes are one hop away from the gateway and typical network flows are from or to the gateway. QUOTA framework operates on

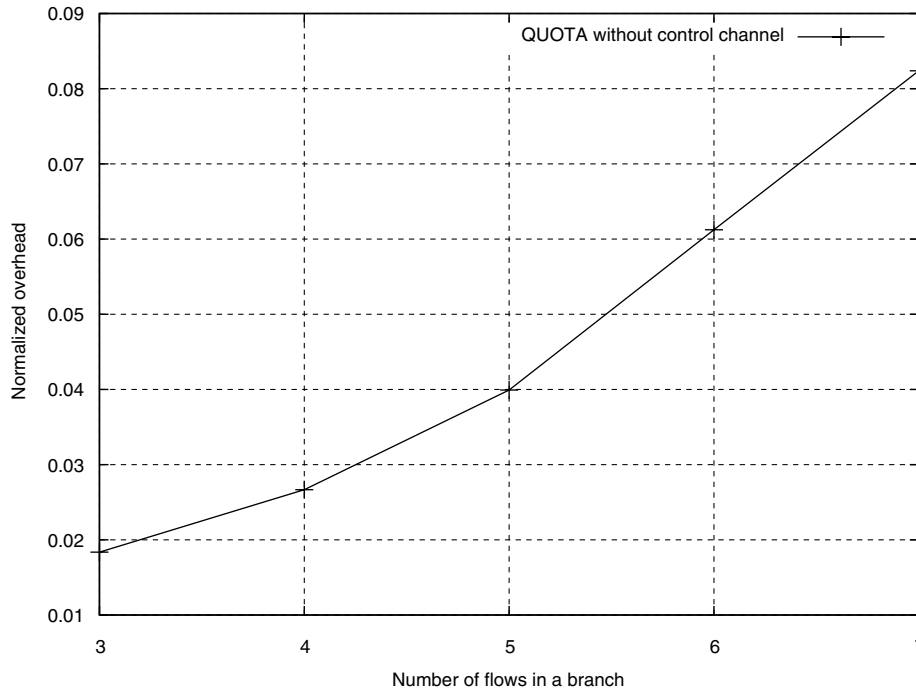


Fig. 13. QUOTA control message overhead.

aggregated flows between the mesh routers and the gateway, whose number is limited by the number of mesh routers in each branch.

6. Conclusions and future work

In this paper, we proposed a framework named QUOTA for QoS support and fair rate allocation in wireless mesh networks. Our framework uses link contention graph and utility maximization framework to perform admission control and rate allocation. Simulation results show that our framework successfully guarantees the QoS of real-time flows and fairly, efficiently allocates bandwidth for elastic flows in different wireless mesh network scenarios. We also compared the performance of QUOTA with the SWAN framework and explained the advantages of our framework. Both QUOTA and SWAN provide QoS support for real-time flows by controlling the rate of elastic flows. However, QUOTA outperforms SWAN in terms of the efficiency and fairness of rate allocation for elastic flows. We observed large performance gap between QUOTA and SWAN in various mesh network scenarios.

The further reduction of message overhead of QUOTA is our future work. One promising direction is to combine QUOTA with a routing protocol, such as AODV [31]. This approach has been discussed in [34].

Our framework will benefit from a more efficient underlying scheduling protocol. An improved scheduling protocol will help to increase the effective capacity of maximal cliques. We would like to investigate the joint design of QoS-aware rate allocation framework and scheduling protocol [15,16] in the future.

References

- [1] Google WiFi Mountain View. Available from: <<https://wifi.google.com/support/>>.
- [2] Available from: <<http://www.isi.edu/nsnam/ns/index.html>>.
- [3] IEEE 802.11 Standard. Available from: <<http://standards.ieee.org/getieee802/802.11.html>>.
- [4] IEEE 802.16 Standard. Available from: <<http://standards.ieee.org/getieee802/802.16.html>>.
- [5] Kiyon Autonomic Networks. Available from: <<http://www.kiyon.com>>.
- [6] Meshdynamics. Available from: <<http://www.meshdynamics.com/>>.
- [7] Microsoft Mesh Networks. Available from: <<http://research.microsoft.com/mesh>>.
- [8] Tropos Networks. Available from: <<http://www.troposnetworks.com/>>.
- [9] UCSB MeshNet. Available from: <<http://moment.cs.ucsb.edu/mesh-net/>>.
- [10] G.S. Ahn, A.T. Campbell, A. Veres, L.H. Sun, SWAN: service differentiation in stateless wireless ad hoc networks, in: Proceedings of the IEEE INFOCOM'2002, New York, June 2002.
- [11] I.F. Akyildiz, X. Wang, W. Wang, Wireless mesh networks: a survey, *Computer Networks Journal* (2005).
- [12] J.G. Augustson, J. Minker, An analysis of some graph theoretical cluster techniques, *Journal of the Association of Computing Machinery* 17 (4) (1970) 571–586.
- [13] S. Boyd, L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.
- [14] I.D. Chakeres, E.M. Belding-Royer, Pac: perceptive admission control for mobile wireless networks, in: Proceedings of the First International Conference on Quality of Service in Heterogeneous Wired/Wireless Networks (QShine), Dallas, TX, October 2004.
- [15] L. Chen, S.H. Low, M. Chiang, J.C. Doyle, Cross-layer congestion control, routing and scheduling design in ad hoc wireless networks, in: IEEE INFOCOM'06, 2006.
- [16] L. Chen, S.H. Low, J.C. Doyle, Joint congestion control and media access control design for wireless ad hoc networks, in: IEEE INFOCOM'05, 2005.

- [17] D. Chiu, R. Jain, Analysis of the increase and decrease algorithms for congestion avoidance in computer networks, *Computer Networks* (1989).
- [18] Reinhard Diestel, *Graph Theory*, Springer, 2005.
- [19] M.C. Domingo, D. Remondo, Analysis of VBR VoIP traffic for ad hoc connectivity with a fixed IP network, in: *VTC 2004-Fall*, 2004.
- [20] Z.Y. Fang, B. Bensaou, Design and implementation of a MAC scheme for wireless ad-hoc networks based on a cooperative game framework, in: *Proceedings of the IEEE ICC*, June 2004, pp. 4034–4038.
- [21] Z.Y. Fang, B. Bensaou, Fair bandwidth sharing algorithms based on game theory frameworks for wireless ad-hoc networks, in: *Proceedings of the IEEE INFOCOM'04*, 2004.
- [22] M. Fazel, M. Chiang, Network utility maximization with nonconcave utilities using sum-of-squares method, in: *Conference on Decision and Control*, 2005.
- [23] V. Gambiroza, E. Knightly, Congestion control in CSMA-based networks with inconsistent channel state, in: *Proceedings of ACM International Wireless Internet Conference (WICON)*, Boston, MA, August 2006.
- [24] V. Gambiroza, B. Sadeghi, E. Knightly, End-to-end performance and fairness in multihop wireless backhaul networks, in: *Proceedings of Mobicom*, 2004.
- [25] R. Gupta, J. Walrand, Approximating maximal cliques in ad hoc networks, in: *IEEE Personal, Indoor, and Mobile Radio Communications Conference (PIMRC)*, Barcelona, Spain, September 2004.
- [26] F.P. Kelly, A.K. Maulloo, D.K.H. Tan, Rate control in communication networks: shadow prices, proportional fairness and stability, *Journal of the Operational Research Society* 49 (1998) 237–252.
- [27] Kiyon, Inc., High quality voip over kiyon multi-service 802.11 wireless, White Paper, 2005.
- [28] B. Li, End-to-end fair bandwidth allocation in multi-hop wireless ad hoc networks, in: *ICDCS 2005*, 2005, pp. 471–480.
- [29] S.H. Low, D.E. Lapsley, Optimization flow control – I: basic algorithm and convergence, *IEEE/ACM Transactions on Networking* 7 (1999) 861–874.
- [30] O. Malik, Google Launches WiFi Network in Mountain View, GigaOM. Available from: <<http://gigaom.com/2006/08/15/google-launches-wifi-network-in-mountain-view/>>.
- [31] C.E. Perkins, E.M. Royer, Ad hoc on-demand distance vector routing, in: *Proceedings of the Second IEEE Workshop on Mobile Computing Systems and Applications*, New Orleans, LA, February 1999, pp. 90–100.
- [32] B. Radunovic, J.Y.L. Boudec, Rate performance objectives of multi-hop wireless networks, in: *Proceedings of the INFOCOM*, 2004.
- [33] S. Shenker, Fundamental design issues for the future internet, *IEEE JSAC* 13 (7) (1995) 176–1188.
- [34] Y. Xue, B. Li, K. Nahrstedt, Optimal resource allocation in wireless ad hoc networks: a price-based approach, *IEEE Transactions on Mobile Computing* (2005).
- [35] Y. Yang, R. Kravets, Distributed QoS guarantees for realtime traffic in ad hoc networks, in: *IEEE International Conference on Sensor and Ad Hoc Communications and Networks (SECON)*, 2004.
- [36] Y. Yang, R. Kravets, Contention-aware admission control for ad hoc networks, *IEEE Transactions on Mobile Computing* 4/4 (2005) 363–377.
- [37] Y. Yang, R. Kravets, Achieving delay guarantees in ad hoc networks through dynamic contention window adaptation, in: *IEEE INFOCOM*, 2006.
- [38] Y. Yang, R. Kravets, Throughput guarantees for multi-priority traffic in ad hoc networks, *Elsevier Ad Hoc Networks Journal* (2006).