A Feedback Control Scheme for Resource Allocation in Wireless Multi-hop Ad Hoc Networks

Wenbo He
wenbohe@cs.uiuc.edu

Xue Liu
xueliu@cs.uiuc.edu

Klara Nahrstedt
klara@cs.uiuc.edu

Computer Science Department
University of Illinois at Urbana-Champaign

Abstract

We present a new feedback control scheme for resource allocation in wireless ad hoc environment. The congestion measure on a wireless link is symbolized as a virtual price, which depends on the recent history of wireless node's queue size. The aggregate price along a route is the feedback to control the transmission rate at the source. In this way, the transmission rate adjustment will dynamically achieve efficient resource allocation, and make the queue size on wireless links stabilize around desired equilibrium. Simulation results show that the proposed scheme can not only help allocate bandwidth fairly, but also achieve the goal of optimal sharing the channel capacity among multi-hop flows.

1. Introduction

1.1 Framework for Resource Allocation in the Internet

TCP has been a tremendous success during the last 20 years. Different versions of TCP for congestion control are designed and deployed on the Internet [1] [2] [3]. Currently, the most widely deployed congestion control protocol on the Internet is TCP-Reno, which is based on packets loss feedback. However, in wireless networks, packet loss is caused not only by congestion, but also by channel conditions such as noise and fading, so congestion control scheme based on packet loss is not efficient in wireless environment [4]. Recently researchers began modeling the congestion by other measures. These measures (typically a scalar variable), are explicitly fed back to the flow source as the degree of congestion, then the source adjusts the transmission rate according to the congestion measure.

Let $R_{|S|\times|L|}$ be the routing matrix of the network with S being the set of all sources and L being the set of all links. $R_{i,j} = 1$ if source $i's$ route passes through link j, $R_{i,j} = 0$ otherwise. Let c_l denote the capacity of link l. Kelly et al [5] [6] first formulated congestion control as a resource allocation problem and give analytical solutions. They use utility function $U_s(r_s)$ to describe the degree of satisfaction for sender s as an increasing concave function of its transmission rate r_s. The cost of sender s to maintain the transmission rate r_s depends on the load on each router on the path to its destination. The net gain for a user s is expressed by its gained value of utility function minus its cost incurred on its route. Hence, the total network gain is:

$$\sum_{s\in S} U_s(r_s) - \sum_{l\in L} \int_0^{\sum_s r_s} R_{s,l}\, f_l(y)dy \qquad (1)$$

where $s \in S$, $l \in L$. Selections of $U_s(r_s)$ reflect the fairness consideration. $f_l(y)$ is non-decreasing continuous function, which can be viewed as the unit price of a certain rate y. We call y aggregate rate on link $l \in L$, which is the total rate of flows passing through a link l. So $y = \sum_{s:} r_s \quad \forall\, l \in L, \quad R_{s,l} = 1$. The resource allocation objective is to maximize objective function (1) with the link capacity constraint:

$$\begin{aligned} &\sum\nolimits_{s:} r_s \le c_l \quad \forall\, l \in L, \quad R_{s,l} = 1 \\ &\quad and \qquad r_s \ge 0 \qquad \forall\, s \in S \end{aligned} \qquad (2)$$

In the following, we call the optimization problem of maximization of objective function (1) with constraints (2) the Resource Optimization (RO) problem. Intuitively, when the derivative of equation (1) over r_s for a certain $s \in S$ is positive, we can increase the transmission rate r_s to increase the value of (1). So the primal algorithm to solve the RO problem is by adjusting the transmission rate of flows as follows:

$$\dot{r}_s = k_s(r_s)[U'_s(r_s) - P_s(t)] \qquad (3)$$

where $k_s(r_s)$ can be any positive non-decreasing function and $P_s(t) = \sum_{l\in\Re} f_l(y(t))$ is route price on route $\Re$. Low et al. [7] solves the RO problem using dual algorithm, which is derived from Karush-Kuhn-Tucker conditions. It is in fact adjusting the price on routers as sending rate is given by $r_s = U^{-1}(P_s)$:

$$\overset{\bullet}{p_l} = h_l(p_l)(y_l - c_l)^+_{p_l} \tag{4}$$

where p_l and c_l are link price and capacity on l, $(y_l - c_l)^+_{p_l} = \begin{cases} y_l - c_l, & p_l > 0 \\ \max(y_l - c_l,\ 0), & p_l = 0 \end{cases}$ and $h_l(p_l) > 0$ is a non-decreasing continuous function. It has been proven that both primal and dual algorithms are asymptotically stable and converge to optimal solution of the RO problem.

RO problem and its solutions establish the relationship between resource allocation and congestion control [8] [9]. At sender side, various versions of TCP congestion control protocols can be explained in this framework. For example, when dropping probability is small, TCP Reno can be approximated by selecting utility function as $U_s(r_s) = -\frac{1}{r_s T_s}$, where r_s is the rate of source s and T_s is its Round Trip Time (RTT). For TCP Vegas, the utility function can be expressed by $U_s(r_s) = \alpha T_{ps} \log r_s$, where T_{ps} is the round-trip propagation delay of source s. TCP Vegas implies weighted proportional fairness in resource allocation. At the router side, active queue management (AQM) protocols which drop or mark the packets when congestion occurs can also be explained in this framework. Here, link price can be considered as the probability that the link marks a packet, thus the route price $P_s = 1 - \prod_{l:l \in \Re} (1 - p_l)$ is equal to the dropping probability on the route.

1.2 Related Work in Wireless Networks

For the optimization problem, the constraint is that the total traffic on a certain link should be less than or equal to the link capacity c_l, which is a known constant in wired networks. However, in wireless ad hoc networks, we cannot precisely predict the available one-hop bandwidth due to the shared medium nature. In wireless networks, contention in the spatial space is unavoidable. By current MAC protocols for collision control (IEEE 802.11x), once the channel is idle, a node which has data to transmit has to wait for a randomly chosen duration (backoff interval) before attempting to retransmit. The backoff interval is represented by [0, *cw*], where *cw* is the contention window. When the channel becomes idle, backoff interval decreases until it reaches zero or the channel becomes busy again. If the channel becomes busy before the backoff interval reaches zero, the backoff interval stops decreasing till the channel becomes idle again. When backoff interval reaches zero, the node transmits a RTS signal. The time which each wireless node spends on counting down backoff intervals contributes a part of MAC layer overhead. Considering the contention overhead, the available bandwidth for a single wireless link can not be predicted precisely. The example in Figure 1 shows that flows f_1, f_2 and f_3 contend with each other. Wireless links l_{13}, l_{23}, l_{34} and l_{45} compete for the total capacity of ***C***. In this case, even though the wireless link has capacity of ***C***, the available bandwidth on the involved links (l_{13}, l_{23}, l_{34} and l_{45}) is much less than ***C***. The unpredictability of the available wireless link capacity is the major obstacle to apply the primal and dual methods for optimal resource allocation used in wired networks directly in the wireless networks.

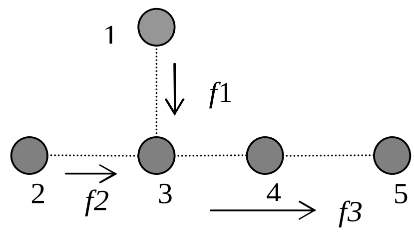

Figure 1: Contention flows in a wireless network

Xue et al [10] aimed to solve the problem by using clique-associated price for wireless networks rather than link-associated price. A clique is the group of neighboring nodes, and the nodes within a group share the wireless link capacity ***C*** at any time. However the clique graph is location and traffic dependent, and the overhead to build the clique graph is expensive. This paper seeks an inexpensive way to do the resource allocation in wireless ad hoc environment.

Other researchers [11] [12] [13] [14] have also discussed resource allocation in wireless networks using price based mechanisms. In [11], a node must forward a certain amount of packets for the benefit of others to get enough credits to maximize its own benefit. [12] [13] assumes selfishness of individual mobile host and apply the non-cooperative game theory to achieve optimal resource allocation. [14] implements a resource allocation scheme based on generalized Vickrey auction with reserve pricing. The assumption of "selfishness" in the above papers requires each mobile host to work with a secure payment and accounting system, and to keep per-flow information to guarantee the pricing mechanisms work, which will bring large overhead. Actually ad hoc wireless networks usually cover a small area and it is more reasonable to assume that mobile nodes are willing to forward packets for others if no congestion occurs.

In this paper, we propose a light-weight resource allocation scheme in multi-hop wireless ad hoc networks. The proposed algorithm will also achieve fairness among flows. We use the virtual price to represent the degree of congestion, which is determined according to the queue length of the outgoing interface of wireless links. In our feedback control scheme, when a packet arrives to the destination, it first obtains the accumulative price on the route, and then an ACK will be sent back to the source with the accumulative price information. While receiving the ACK and retrieving the accumulative price, the source will update the transmission rate of the flow accordingly. If a queue is piled up in a node, that means wireless link is busy (sending or waiting to send), as a result the price of forwarding packets will rise. The price increase gives senders incentives to slow down their transmission. Since all the flows which pass through the node will sense the price change and take the actions to adjust their sending rate accordingly, the feedback control scheme is fair among flows. The price shift and transmission rate adjustment interact with each other to make the resources utilization stay around a desired level. The feedback control scheme also stabilizes queue size around a predefined value, so that as long as the overshoot of the queue length stays within the capacity of the queue, no packets will be dropped. Another major difference of our scheme compared with previous approaches is that our feedback scheme for resource allocation is developed without explicit usage of utility functions.

The rest of the paper is organized as follows. In section 2, we introduce the framework of the feedback scheme for resource allocation in wireless ad hoc environment. In section 3, we present the feedback control scheme, which is a decentralized algorithm for price update and transmission rate adjustment. A major difference of our scheme compared with previous approaches is that our feedback scheme for resource allocation is developed without explicit usage of utility function. However, we derive the corresponding utility functions for the feedback mechanism and give the explanation of the utility function in section 4. In section 5, we prove the convergence of the proposed feedback control scheme. In section 6, we provide simulation results to evaluate the proposed feedback control scheme. Finally, conclusions are made in section 7.

2. Feedback Framework

2.1 Model for Resource Allocation in Wireless Ad Hoc Networks

Consider a wireless ad hoc network with a set of static nodes $N_{set} = \{1, 2, \cdots, N\}$. Consider a data source $s \in N_{set}$ who sends data to destination $d \in N_{set}$ along the route $R_s = \{s,\ i_1,\ i_2, \cdots, i_j,\ d\}$, where $i_j \in N_{set}$ and $j+1$ is the number of hops from the source to destination. Assume there are totally M_i flows passing through node i. Consider a flow $f_k,\ \ k \in 1, \cdots, M_i$, which needs to be relayed at router (node) i, and f_k is sent at the rate $r_k,\ \ k \in 1, \cdots, M_i$. The total traffic relayed by the node i is $\sum_{k=1}^{M_i} r_k$.

In the feedback scheme, each wireless node maintains a scalar variable P_i, the virtual price representing the degree of congestion. When a flow is forwarded by node i, P_i is added to the aggregate route price of the flow. P_i is updated autonomously at node i based on the current and previous value of queue length on the wireless interface. If a flow f has the route $R_s = \{s,\ i_1,\ i_2, \cdots, i_j,\ d\}$, the virtual price on the route is $P_f = \sum_{k=1}^{j} P_{i_k}$. If the total price P_f increases, the congestion degree on the route R_s increases, so the flow should decrease the sending rate. The details on how to update P_i and how to determine the transmission rate of a flow are given in section 3.1 and 3.2.

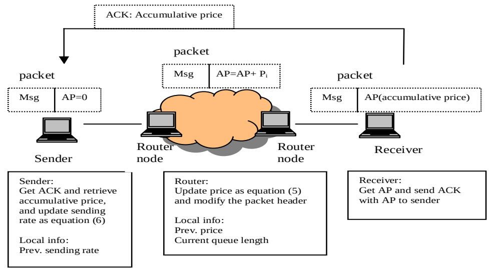

Figure 2: Roles of sender, receiver and router nodes

Figure 2 shows how the proposed feedback mechanism works. Senders are responsible for updating the transmission rate according to the accumulative price on the route. Each router is responsible to update price based on current queue length and its previous price, and to add its current price to the accumulative price field in the packet header. The information listed in the dashed boxes is required for router nodes to update P_i and for source node to determine the transmission rate.

2.2 Interaction Between Virtual Price, Transmission Rate and Queue Length

The virtual price of a wireless link is considered as the indication of the demands of resources (e.g. bandwidth) at the link. We determine the virtual price P_i at router i based on its queue size, since the queue size reflects the difference between network load and network processing capacity (wireless bandwidth).

In a desired feedback scheme, if the queue size on a router is above the desired level, the virtual price will increase; otherwise the virtual price will decrease. Figure 3 shows an example about the desired relationship among queue size, virtual price and transmission rate of a flow, where we want to stabilize the queue size at around 20% of the total buffer capacity (buffer utilization stays around 20%). We can see from the Figure 3 that when utilization is greater than 20% the price will rise; otherwise, the price will decrease. As the price goes up, the rate slows down and vice versa. Each time when the utilization reaches 20%, the

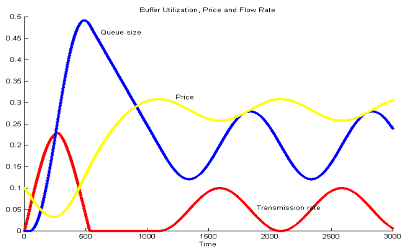

Figure 3: Relationship among buffer utilization, price and transmission rate

price reaches its maximum or minimum. If the utilization goes down through 20%, the price reaches its maximum; otherwise the price reaches its minimum. We will give the details on how to update price and adjust transmission rate in section 3. Since we only care about how the virtual price and the transmission rate adjust with regard to the changes of queue size, the unit of price and rate are omitted.

3. Decentralized Algorithm for Virtual Price and Transmission Rate Update

3.1 Algorithm for Virtual Price Update

In wireless networks, a node forwards a data packet by broadcasting it to its neighbors. Each interface of a wireless node needs to maintain a queue to store the data packets waiting to be forwarded. The maximum queue size is the total store-and-forward buffer size at each mobile host. If total buffer capacity is $\boldsymbol{B}$, we can choose the desired size of occupied buffer $q*$, so that $q* < \boldsymbol{B}$, around which we hope to stabilize the queue size. The choice of $q*$ depends on how much delay the network can tolerate. A smaller $q*$ introduces less queuing delay. Also, packets are not likely to be dropped if the margin $B - q*$ is large. On the other hand, we do not want to have $q*$ too small in order to get higher throughput.

We use the following equation to update the virtual price according to queue size at the router nodes.

$$P_i(t+1) = (P_i(t) - \lambda_i(q_i^* - q_i(t)) + \zeta_i(q_i(t) - q_i(t-1)))^+ \quad (5)$$

where $P_i(t)$ is the virtual price that the node i charges from other users to forward message at the time t. And q_i is the occupied buffer size (queue length) at node i. λ_i and ζ_i are positive numbers which reflects the update step size. And function $[F(t)]^+ = \max(F(t), 0)$. Consider the term $\lambda_i(q_i^* - q_i(t)$, if $q_i^* > q_i$, i.e. current queue size is less than the desired level, then the price goes down. Once the source notices the price decreased, it will increase the sending rate. Otherwise, when queue size exceeds the desired level q*, the price goes up, and the sender will decrease the sending rate. The term $\zeta_i(q_i(t) - q_i(t-1))$ is a prediction term. It helps to make the real queue size approach the desired value q* faster. If the current queue size is changing, the price at router side should be adjusted according to how much the queue size changes. This term accelerates the queue size to reach $q*$. Note that equation (5) is maintained at relay hosts.

3.2 Algorithm for Transmission Rate Update

The sending rate of a flow should be adjusted to the accumulative price on the route from the sender to the destination, which is the sum of the prices charged by routers on the route. Similar to the market rule, the demand of a product goes down when the price increases, if the accumulative price increases, the sender of the flow will slow down the sending rate, and vice versa. In this way, the flow will achieve the proper rate at routers on the route to the destination for the long run. The transmission rate update rule is as follows:

$$r_f(t+1) = [r_f(t) - \delta_f(\sum_{i \in R} P_i(t+1) - \sum_{i \in R} P_i(t))]^+ \quad (6)$$

where r_f is the transmission rate of flow f, and P_i is the price charged by the router i on the route. R is the set of routers which are involved in relaying flow f. δ_f is a positive number with the unit of "rate unit/price". Equation (6) is only maintained at sender side. When the accumulated price goes up, the resources at the routers are being dissipated more than expected, and congestion is more likely to occur, so the sending rate should go down to alleviate this situation.

Consider the queue size at router i, assuming its service rate c_i, which is the number of buffer cells being consumed (transmitted) at the unit time. Note c_i is not necessarily constant. The $r(t)$ can be considered as the number of buffer cells taken by arriving flows at the unit time. We have:

$$q_i(t+1) = [q_i(t) + (\sum_{s \in S_i} r_s(t) - c_i)]^+ \quad (7)$$

where S_i is the collection of the senders of flows, which are relayed by router i. Equation (7) is the dynamic model of the queue length at router i, and no control strategy is applied. However, (7) sets up a connection between virtual price and sending rate by queue size.

Figure 4 shows how a node relays packets for others. The pricing module is responsible to maintain an up-to-date price and attach the accumulative price to the headers of packers. The accumulative price is the sum of the virtual price P_i at the current router i and the accumulative price (AP) in the packet header.

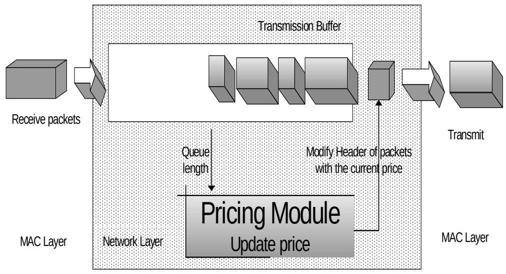

Figure 4: Router architecture

The calculation at either router side or at sender side to update the price is a constant time operation. To implement the algorithm, a field in packet header is needed to store the accumulative price for a flow. We can see that both operation overhead and space overhead of the distributed algorithm are small. The algorithm at sender side and the algorithm at router side work autonomously. However, the system as a whole works as a feedback system.

3.3 Stability Analysis

In this subsection we will study the stability of the proposed scheme discussed above. From equation (5) (6), we know that the stability of queue size implies the stability of the virtual price and sending rate, hence the stability of the whole system. First, we prove the stability under the single-flow case. Then we will extend the results to multiple-flow case.

A. Stability of Single Flow

Denote $x_i(t) = q_i(t) - q_i^*(t)$. We use the continuous version of equation (5), (6) and (7) as follows.

$$\frac{d}{d\,t}P_i(t) = \lambda_i x_i(t) + \zeta_i \frac{d}{d\,t}x_i(t) \quad (8)$$

$$\frac{d}{d\,t}r_f(t) = -\delta_f \sum_{j\in R} \frac{d}{d\,t}P_j(t) \quad (9)$$

$$\frac{d}{d\,t}x_i(t) = \sum_{s\in S} r_s(t) - c_i \quad (10)$$

Consider a single flow f in a wireless network, equation (10) can be expressed as:

$$\frac{d}{d\,t}x_i(t) = r_f(t) - c_i \quad (11)$$

Differentiate (11), we get

$$\frac{d^2}{d\,t^2}x_i(t) = \frac{d}{d\,t}r_f(t) \quad (12)$$

Substitute (9) to (12) we get

$$\frac{d^2}{d\,t^2}x_i(t) = -\delta_f \sum_{j\in R} \frac{d}{d\,t}P_j(t) \quad (13)$$

Let $X(t) = \sum_{i\in R} x_i(t)$, substitute (8) to (13), then sum the result over i, we have

$$\frac{d^2}{d\,t^2}X(t) = -\delta_f N(\lambda_i X(t) + \zeta_i \frac{d}{d\,t}X(t)) \quad (14)$$

where N is the number of routers on the route of flow f and δ_f, λ_i, and ζ_i are all positive. It is easy to verify that the two characteristic roots of equation (14) always have negative real parts, hence the system is stable when a single flow is considered. Next, we will use this result to show that in multiple flow case, the system is still stable.

B. Stability of Multiple Flows

Consider any router in the network. Assume that at time t, there are k flows passing through it, and the channel capacity is c, which is shared by neighboring wireless links, such that $\sum c_i = c - overhead$, where c_i is the capacity share that node i obtains. c_i is time-varying and shared by k flows. Assume each of k flows get $c_{i1}, c_{i2}, \cdots, c_{ik}$ capacity respectively at time t, where $\sum_{j=1}^{k} c_{ij} = c_i$. The k flows interweave with each other in the queue of node i. The j-th flow takes the queue size as $q_{ij}(t+1) = [q_{ij}(t) + (r_j(t) - c_{ij})]^+$, $j = 1, \cdots, k$. We have shown in ***A*** that for each $j = 1, \cdots, k$, $q_{ij}(t)$ is stable. The queue length of the router is $q_i(t) = \sum_{j=1}^{k} q_{ij}(t)$. If each q_{ij} is stabilized at q_{ij}^*, the queue $q_i(t)$ at router i can be stabilized around $q_i^* = \sum_{j=1}^{k} q_{ij}^*(t)$. So we know that at any given router, the feedback system is still stable when multiple flows exist.

4. Interpretation of the Feedback Model

By equation (8) (9) and (10), we have

$$\dot{r}_f(t) = -\sum_{i\in R_f} \delta_f \lambda_i x_i - \delta_f \sum_{i\in R_f} \zeta_i (\sum_{s\in S_i} r_s - c_i) \quad (15)$$

where R_f is the set of routers on the route of the flow$f$; S_i is the set of source whose flows go through the router i. By comparing equations (3) and (15), we can get $k_r(r_f) = \delta_f$, route price $P_s = \sum_{i\in R_s} \lambda_i x_i$, and $U_r'(r_f) = -\sum_{i\in R_s} \zeta_i (\sum_{s\in S_i} r_s - c_i)$. So the utility function can be

$$U_r(r_f) = \sum_{i \in R_f} \zeta_i[(c_i - \sum_{s \in S_i} r_s)r_f + \frac{r_f^2}{2}] \quad (16)$$

$U_r(r_f)$ is a concave function, since $U_r''(r_f) < 0$; and $U_r(r_f) \rightarrow -\infty$ as $r_f \rightarrow \infty$. The utility function doesn't only depend on the transmission rate r_f, but also the capacities of routers on the route of sender s. We can also write equation (16) as $U_r(r_f) = \sum_{i \in R_f} \zeta_i(c_i - \sum_{s \in S_i, s \neq f} r_s - \frac{r_f}{2})r_f$. Let's consider the case in which a flow uses a single relay node i, and no other flows pass through node i. In this case the utility function is shown in Figure 5. The utility function gets its maximum when transmission rate reaches the capacity of the relaying node, C_i.

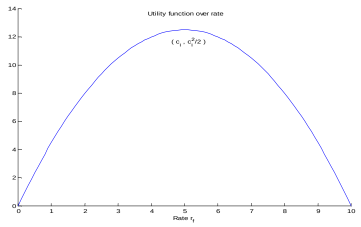

Figure 5: Utility function of the feedback scheme

This means that if the transmission rate is larger than the capacity of the relay node, the sender will suffer from the congestion, even if the capacity is not predictable in wireless networks. Therefore, by feedback mechanism we don't need to know the capacity on the wireless links on the route to maximize equation (1). Hence, the feedback mechanism can tolerate the capacity shift by contention and collision on the wireless links.

Since $(c_i - \sum_{s \in S_i} r_s)$ is the residual capacity on node i, the equation (16) implies the proportional residual capacity fairness.

5. Convergence of the Resource Allocation Problem

Define N to be the set of nodes in the wireless network, R_f be the set of routers on the route of flow f, and S_i be the set of sources which go through the router i. Consider the utility function $U_r(r_f) = \sum_{i \in R_f} \zeta_i(c_i - \sum_{s \in S_i, s \neq f} r_s - \frac{r_f}{2})r_f$, and route price $P_f = \sum_{i \in R_f} \lambda_i x_i = \sum_{i \in R_f} \lambda_i(q_i - q_i*)$, which are derived by the feedback scheme presented in section 3.

Theorem Under the update law of equation (5), (6) and the system model (7), we can achieve the maximum benefit (net gain) of the whole network (total gain minus total cost).

$$\begin{aligned} V(r) &= \sum_{f \in F} \left\{ \sum_{i \in R_f} \zeta_i[(c_i - \sum_{s \in S_i} r_s)r_f + \frac{r_f^2}{2}] \right\} \\ &- \sum_{i \in N} \int_0^{\sum_{f:\, f \text{ relayed by } i} r_f} \lambda_i x_i dy \qquad (17) \end{aligned}$$

where y is the aggregate rate on the wireless channel at router i.

Proof: Consider the Lyaponov function of the system as equation (17). It is trivial to show $V(r)$ is strictly concave, since $\sum_{i \in R_f} \zeta_i[(c_i - \sum_{s \in S_i} r_s)r_f + \frac{r_f^2}{2}]$ is concave and $\lambda_i x_i$ is continuous increasing under equation (7). Further,

$$\begin{aligned} \dot{V} &= \sum_{f \in S} \frac{\partial V}{\partial r_f} \dot{r}_f \\ &= \sum_{f \in S} \left\{ \delta_f \sum_{i \in R_f} \zeta_i(c_i - \sum_{s \in S_i} r_s) - \sum_{i \in R_f} \delta_f \lambda_i x_i \right\}^2 \\ &\geq 0 \qquad (18) \end{aligned}$$

6. Simulation and Results

In this section, we present our simulation results of the proposed scheme. The simulation scenario is shown in Figure 6. It models a multi-hop wireless ad hoc network with 3 flows and 6 nodes i, $i \in [1, 6]$. We take $\lambda_i = \zeta_i = 0.001$, and $\delta_f = 0.01$. Total buffer size is 1000 packets and the pre-designed value of $q*$ is 400 cells. Assume the channel capacity is ***C***=1*Mbps*. We have 3 flows competing with each other for limited channel capacity. Flows $f1$, $f2$ and $f3$ travel 1, 3 and 5 hops, respectively.

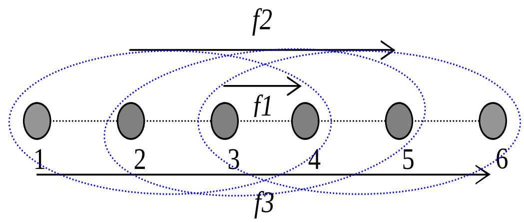

Figure 6: Simulation scenario

In this scenario, all 3 neighboring wireless links share the channel capacity ***C***. In our simulation, we randomly assign wireless bandwidth on each wireless link and make the total bandwidth of all 3 neighboring links bounded by ***C***. In ideal case, the optimal allocation of bandwidth is that each flow can get about 0.14*Mbps* bandwidth, wireless link l_{12}

and l_{56} get 0.14*Mbps* bandwidth, l_{23} and l_{45}get 0.28*Mbps* bandwidth, and l_{34} get *0.42Mbps* bandwidth.

A. Simulation Results

As Figure 7 shows, the average equilibrium transmission rate of all three flows is around 0.12*Mbps*. After t=500 iterations, the rate of $f2$ and $f3$ overlap with each other, and after t=3000 iterations, the rates of $f1$, $f2$ and $f3$ are identical.

Figure 8 and 9 show the queue length and virtual price of node 2, 3 and 4, respectively. Node 3 is the bottleneck in this scenario, since all the flows pass through node 3. The equilibrium queue length of node 3 is stabilized around 400 after 8000 iterations. Node 2 and node 4 are pretty much idle, since they are not bottleneck links.

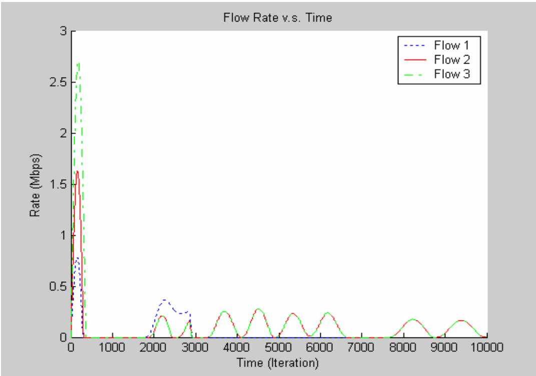

Figure 7: Simulation result (1)— flow rate v.s. time

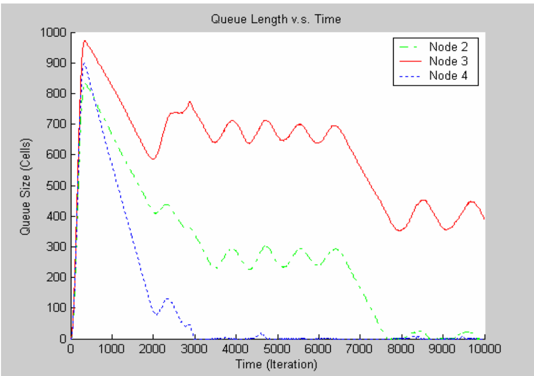

Figure 8: Simulation result (2) — queue length v.s. time

B. Stability

The simulation shows that the buffer occupation (as Figure 8) and price of bottleneck link (node 3 in Figure 9) are stabilized in the feedback model. The non-bottleneck links are almost idle, so the price tends to zero. The route price is dominant by bottleneck link.

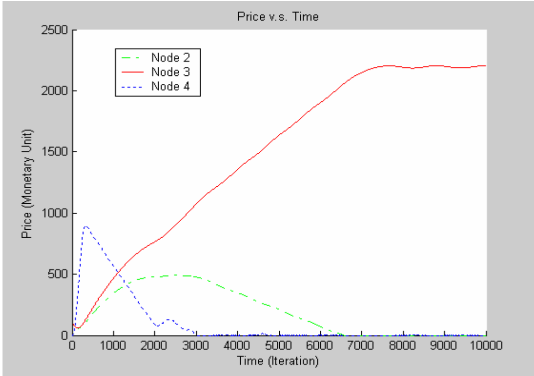

Figure 9: Simulation result (3) — price v.s. time

C. Fairness of the bandwidth allocation

From Figure 7, we see the transmission rates of the three flows are identical after t = 3000 iterations. This means that the bandwidth allocation by the feedback control scheme is pretty fair. All the flows slow down sending data when the route price increases, and speed up sending when the route price goes down.

D. Effectiveness on bandwidth allocation

In ideal case (without contention overhead), each flow can get bandwidth of 0.14Mbps in the simulated scenario. The simulation shows that by our feedback model, each flow can achieve about 0.12Mbps by ignoring the contention overhead. So the presented algorithm is very efficient to achieve the global optimal share of channel capacity among multi-hop flows.

In figure 8, the total buffer capacity is 1000 cells, the feedback scheme eventually stabilize the queue size at around 400 cells, and the peaks of the queue size curve of bottleneck link never exceeds 1000 in this simulation case. This means that we can avoid packets dropping as much as possible in the feedback scheme.

7. Conclusions and Future Work

In this paper, we proposed a feedback control scheme for resource allocation in wireless environment. The feedback control algorithm takes effect by price adjustment and flow rate modification. The distributed algorithm presented is composed of two parts, which are maintained at sender and router respectively. They work together to get the efficient resource allocation. The feedback control algorithm can also stabilize the queue size at their desired level at each wireless node. The overhead associated with the distributed algorithm is very small. Simulation result shows that the proposed feedback control scheme in wireless ad hoc environment is efficient and fair.

The scheme and analysis would be applied to wired networks as well, which is a strength of the paper. However, the scheme is more specific to wireless networks, since in wired network we can have more precise model for resource allocation.

In the future, we will work on simulations with more elaborate scenarios, and compare the feedback scheme with other congestion control schemes.

References

[1] L. S. Brakmo, S. W. O'Malley, and L. L. Peterson, "TCP vegas: New techniques for congestion detection and avoidance," in *SIGCOMM*, 1994, pp. 24–35.

[2] Z. Wang and J. Crowcroft, "Eliminating periodic packet losses in the 4.3-tahoe bsd tcp congestion control algorithm," *SIGCOMM Comput. Commun. Rev.*, vol. 22, no. 2, pp. 9–16, 1992.

[3] D. Borman, R. Braden, and V. Jacobson, "Tcp extensions for high performance (RFC1323)."

[4] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz, "A comparison of mechanisms for improving TCP performance over wireless links," *IEEE/ACM Transactions on Networking*, vol. 5, no. 6, pp. 756–769, 1997.

[5] F. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: shadow prices, proportional fairness and stability," in *Journal of the Operational Research Society*, vol. 49, 1998.

[6] R. Gibbens and F. Kelly, "Resource pricing and the evolution of congestion control," *Automatica*, vol. 35, pp. 1969–1985, 1999.

[7] S. H. Low and D. E. Lapsley, "Optimization flow control — I: basic algorithm and convergence," *IEEE/ACM Transactions on Networking*, vol. 7, no. 6, pp. 861–874, 1999.

[8] S. H. Low, L. Peterson, and L. Wang, "Understanding vegas: A duality model," *Journal of ACM*, vol. 49, no. 2, pp. 207–235, 2002.

[9] R. Srikant, "The mathematics of internet congestion control," 2004.

[10] Y. Xue, B. Li, and K. Nahrstedt, "Price-based resource allocation in wireless ad hoc networks," in *IWQoS*, 2003, pp. 79–96.

[11] Y. Qiu and P. Marbach, "Bandwidth allocation in wireless ad hoc networks: A price-based approach," in *INFOCOM*, 2003.

[12] V. Srinivasan, P. Nuggehalli, C. Chiasserini, and R. Rao, "Cooperation in wireless ad hoc networks," 2003.

[13] M. Felegyhazi, L. Buttyan, and J. Hubaux, "Equilibrium analysis of packet forwarding strategies in wireless ad hoc networks – the static case," 2003.

[14] K. Chen and K. Nahrstedt, "ipass: an incentive compatible auction scheme to enable packet forwarding."