Journal of Zhejiang University SCIENCE A
ISSN 1009-3095 (Print); ISSN 1862-1775 (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Optimal congestion control algorithm for ad hoc networks: Penalty function-based approach

XU Wei-qiang[1,2], WU Tie-jun[†‡1]

([1]*School of Information Science and Engineering, Zhejiang University, Hangzhou 310027, China*)

([2]*College of Informatics and Electronics, Zhejiang Sci-Tech University, Hangzhou 310018, China*)

[†]E-mail: tjwu@zju.edu.cn

Received Feb. 23, 2006; revision accepted Aug. 3, 2006

Abstract: In this paper, based on the inherent characteristic of the contention relation between flows in ad hoc networks, we introduce the notion of the link's interference set, extend the utility maximization problem representing congestion control in wireline networks to ad hoc networks, apply the penalty function approach and the subgradient method to solve this problem, and propose the congestion control algorithm Penalty function-based Optical Congestion Control (POCC) which is implemented in NS2 simulator. Specifically, each link transmits periodically the information on its congestion state to its interference set; the session at each source adjusts the transmission rate based on the optimal tradeoff between the utility value and the congestion level which the interference set of the links that this session goes though suffers from. MATLAB-based simulation results showed that POCC can approach the globally optimal solution. The NS2-based simulation results showed that POCC outperforms default TCP and ATCP to achieve efficient and fair resource allocation in ad hoc networks.

Key words: Ad hoc networks, Congestion control, Subgradient, Penalty function
doi:10.1631/jzus.2006.A2110 **Document code:** A **CLC number:** TP202

INTRODUCTION

Congestion control is a critical issue for ensuring the efficient and fair allocation of network resource among communication flows. In wireline networks, TCP has resolved this issue of concern. However, it was reported that TCP performance in a multihop ad hoc environment degrades severely (Xu and Wu, 2006). Thus, some optimization-based congestion control schemes have been proposed to improve TCP performance in ad hoc networks (Chiang, 2005; Chen *et al*., 2005; Lee *et al*., 2006). Chiang (2005) addressed the jointly optimal congestion control and power control, and Chen *et al*.(2005) and Lee *et al*. (2006) considered joint congestion control and media access control design in ad hoc networks. In this paper, based on the inherent characteristic of the contention relation between flows, we proposed another kind of congestion control scheme for ad hoc networks.

In wireline networks, flows only contend at the router that performs flow scheduling. The capacity of link l represents the constraint on flows which go though link l, which is independent of other flows which go though links other than link l. However, in ad hoc networks, flows also compete for the channel resource of link l if link l is within the interference ranges of links which these flows go though (Xu and Wu, 2006). Such a fundamental difference motivates us to design a more efficient congestion control algorithm for optimal resource allocation in ad hoc networks.

In ad hoc networks, when one link is transmitting data, all links in its interference range remain silent. Thus, congestion control protocols for ad hoc networks must give feedbacks based on the congestion information in the entire interference set, rather than individual links. In this paper, we introduce the

‡ Corresponding author

notion of the link's interference set, extend the basic utility maximization formulation of the congestion control problem for wireline networks in (Kelly *et al*., 1998; Low and Lapsley, 1999) to ad hoc networks. We apply the penalty function method to take into account the capacity constraints and present a new formulation, apply the subgradient method to obtain this optimal solution, and propose a distributed algorithm Penalty function-based Optimal Congestion Control (POCC). POCC can be implemented in NS2 simulator through parallel fashion at each source/link. Specifically, each link transmits periodically the information on the congestion state to its interference set; each session at each source adjusts the transmission rate based on the optimal tradeoff between the utility value and the congestion level which the interference set of the links that this session goes though suffers from. Thus, it can cope with absence of the central coordination in ad hoc networks. Meanwhile, computation only based on local information can effectively save on the network resource. In particular, the link does not need to maintain any state information for each session, leading to lower computation and communication overhead. Thus, it is scalable to a broad range of network scenarios. The MATLAB-based simulation results showed that POCC can rapidly approach the globally optimal solution. The NS2-based simulation results showed that POCC can achieve efficient and fair resource allocation.

PROBLEM STATEMENT

In this section, we first introduce the notation to model the system under study, and then represent the congestion control problem as a nonlinear optimization problem. Finally, we specify a number of inherent characteristics of ad hoc networks imposing important restrictions to efficiently solve this optimization problem.

Notation and assumption

Consider ad hoc networks represented by a directed graph $G(N^*,L^*)$, where N^* is a set $N^*=\{1, 2, \ldots, N\}$ of nodes and L^* is a set $L^*=\{1, 2, \ldots, L\}$ of logical links. Let C_l be the capacity of link $l\in L^*$. The network is shared by a set of the end-to-end multihop flows $S^*=\{1, 2, \ldots, S\}$. Each flow $s\in S^*$ originated from source s goes through a set of wireless links $L^*(s)\subset L^*$. A single-hop data transmission in the flow s along a particular wireless link is referred to as a subflow of s. For each link, let $S^*(l)=\{s\in S^*|l\in L^*(s)\}$ be the set of flows/sources that goes through link l. Source s is characterized by a utility function $U_s(x_s)$, which satisfies two assumptions:

(1) $U_s(x_s)$ is increasing, strictly concave, and twice continuously differentiable at $x_s^{\min}\le x_s\le x_s^{\max}$, where x_s is the transmission rate required by source s, and $x_s^{\min}\ge 0$ and $x_s^{\max}<\infty$ are the minimum and maximum transmission rate required by source s, respectively;

(2) The curvatures of $U_s(x_s)$ are bounded away from zero.

Link's interference set

In ad hoc networks, subflow going through the link l suffers from two kind of contentions: (1) subflows originated from the source node of the link l contend for the link l with each other; (2) subflows going through the other links in whose interference range the link l is located in, compete for the link l. Thus, among a set of mutually contending subflows, only one of them may transmit at any given time.

In this paper, we model the contention relation between subflows as the link's interference set interfering with the link l, including the link l itself, denoted by IS_l. This set indicates which groups of subflows interfere with the subflows which go through the link l. Because links included in the interference set IS_l share the same channel resource C_l of the link l, only one of the subflows going through link k ($k\in IS_l$) may transmit at any given time. Thus, the aggregated rate of all subflows in such a set may not exceed the channel capacity, i.e.

$$\sum_{k\in IS_l}\sum_{s\in S^*(k)} x_s \le C_l. \tag{1}$$

The accurate interference set of the link can be constructed based on the SIR model as proposed in (Gupta and Kumar, 2000), which may be very difficult in reality. In practice, two links interfere with each other when either the sender or the receiver of one is within interference range of the sender or the receiver of the other (Xue *et al*., 2006). Thus, when the interference range is K times that of the transmis-

sion range, the interference set of the link includes all links that contain the K+1 neighbor set NS^{K+1} of the sender and receiver of that link, i.e.

$$IS_l=\{k|(T_k\in NS^{K+1}(T_l))\cup(T_k\in NS^{K+1}(R_l))\cup (R_k\in NS^{K+1}(T_l))\cup(R_k\in NS^{K+1}(R_l))\}, \quad (2)$$

where T_k and R_k are the sender and the receiver of link k, respectively, and

$$NS^{K+1}(N)=\begin{cases}\{N'|(N,N')\in L^*\}, & K=0;\\ NS^1(NS^K(N))\cup NS^K(N), & K>0.\end{cases} \quad (3)$$

Problem formulation

Lastly, TCP congestion control algorithm has been interpreted as a distributed primal-dual optimization algorithm over the Internet to maximize aggregate utility, and to achieve efficient and fair allocation of network resource (Kelly *et al*., 1998; Low and Lapsley, 1999). In this paper, we extend the basic utility maximization formulation to ad hoc networks with new constraints. Here, we now formally present the congestion control problem in ad hoc networks as the nonlinear optimization problem P:

$$\begin{aligned}\text{P}:\ &\max_{\boldsymbol{x}}\sum\nolimits_s U_s(x_s)\\ \text{s.t.}\ &\sum\nolimits_{k\in IS_l}\sum\nolimits_{s\in S^*(k)}x_s\le C_l,\forall l,\end{aligned} \quad (4)$$

where $\boldsymbol{x}\in\boldsymbol{X}=\{x_s\,|\,x_s^{\min}\le x_s\le x_s^{\max},\ s\in S^*\}$.

The objective function maximizes the aggregated utility of all multihop flows. The constraint of the inequality is the resource constraint from the shared wireless channel as discussed above. The efficient and fair allocation of network resource among all flows may be achieved by choosing the appropriate utility functions, and maximizing the aggregated utility.

The objective function is differentiable and strictly concave, and the feasible region in inequality is convex and compact. Therefore, based on nonlinear optimization theory, there exists a unique maximizer for the optimization problem P.

Restrictions on algorithm design

However, some inherent characteristics of ad hoc networks impose important restrictions on the algorithm design to effectively solve the problem P. We outline these restrictions below:

(1) Absence of central coordination: In principle, ad hoc networks are independent of any established infrastructure or centralized administration. Each node operates in a distributed peer-to-peer mode, acting as an independent router and an end host. In such an infrastructure-less network, there does not exist any central coordination and control. Also, ad hoc networks are often geographically distributed systems. Thus, the centralized solutions are not practicable. The designed algorithm has to be implemented in a distributed manner in the absence of a central coordination, through the cooperation of the nodes.

(2) Network resource constraints: In ad hoc networks, each node has only limited power supply. More seriously, as each node is acting as both an end host and a router at the same time, additional energy is required to support network operations. Moreover, due to the limited bandwidth and other resources in ad hoc networks, the designed algorithm has to require only very low overhead of communication and computation for the congestion control.

(3) A large-scale scenario: In ad hoc networks, many applications involve a large-scale network with tens of thousands of nodes, such as wireless sensor networks. Thus, the designed algorithm has to be scalable to a large-scale scenario. Scalability of the algorithm implies the efficient support of large numbers of links, nodes, simultaneous sessions/flows, etc.

(4) Dynamical network topology: In ad hoc networks, the network topology may change frequently when nodes arbitrarily move, resulting in link changes between nodes and route changes between the source and the destination. Moreover, the set of sources and destinations also changes from time to time, as some nodes start to transmit the data, others quit the transmission.

PENALTY FUNCTION-BASED APPROACH: POCC

In this section, we apply the penalty function method and the subgradient method to solve the

problem Eq.(4). Specifically, the penalty function method transforms Eq.(4) into a new formulation, and the subgradient method obtains the optimal solution of this transformed problem. Correspondingly, we propose a distributed algorithm POCC which is robust to those challenges that ad hoc networks pose.

Penalty function-based problem

Let $f(\boldsymbol{x})=\sum_s U_s(x_s)$, and $g_l(\boldsymbol{x})=\sum_{k\in IS_l}\sum_{s\in S^*(k)} x_s - C_l$. It follows from Theorem 9.2.2 of (Mokhtar *et al*., 1979) that,

$$\inf\{f(\boldsymbol{x}):\boldsymbol{x}\in\boldsymbol{X},\ g_l(\boldsymbol{x})\le 0,\ \forall l\} = \lim_{\sigma\to\infty}\left(\inf\left\{f(\boldsymbol{x})+\sigma\sum_l P_l(\boldsymbol{x}):\boldsymbol{x}\in\boldsymbol{X}\right\}\right), \quad (5)$$

where σ, the penalty scaling factor, is a large number, and $P_l(\boldsymbol{x})$, $\forall l$ is a suitable penalty function. From this result, it is clear that we can get arbitrarily close to the optimal objective value of the primal problem by computing for a sufficiently large σ.

We choose the following penalty function $P_l(\boldsymbol{x})$:

$$P_l(\boldsymbol{x}) = \max\left\{0, \sum_{k\in IS_l}\sum_{s\in S^*(k)} x_s - C_l\right\}. \quad (6)$$

If $\sum_{k\in IS_l}\sum_{s\in S^*(k)} x_s - C_l \le 0$, then $P_l(\boldsymbol{x})=0$, and no penalty is incurred. On the other hand, if $\sum_{k\in IS_l}\sum_{s\in S^*(k)} x_s - C_l > 0$, then $P_l(\boldsymbol{x})>0$, and the penalty term $\sigma P_l(\boldsymbol{x})$ is realized. Therefore, $P_l(\boldsymbol{x})$ is a suitable penalty function, as it can incur a positive penalty for infeasible points and no penalty for feasible points. We can interpret $\sigma P_l(\boldsymbol{x})$ as the penalty associated with the violation of the capacity constraint of link l.

We apply the penalty function method to transform the problem Eq.(4) into a new formulation as following:

$$\max_{\boldsymbol{x}} \sum_s U_s(x_s) - \sigma\sum_l \max\left(0, \sum_{k\in IS_l}\sum_{s\in S^*(k)} x_s - C_l\right). \quad (7)$$

We will obtain a solution to the problem Eq.(4) by solving the unconstrained problem Eq.(7).

The subgradient method

We now present the subgradient method to solve Eq.(7) iteratively. Let

$$F(\boldsymbol{x}) = \sum_s U_s(x_s) - \sigma\sum_l \max\left(0, \sum_{k\in IS_l}\sum_{s\in S^*(k)} x_s - C_l\right),$$

the vector $\boldsymbol{g}^F(\boldsymbol{x}_0)$ is a subgradient of the function $F(\boldsymbol{x})$ at the point $\boldsymbol{x}_0$, where $\boldsymbol{g}^F(\boldsymbol{x}_0)=\{\boldsymbol{g}_s^F(\boldsymbol{x}_0), s\in S^*\}$, $x_s(n)$ denote the values of x_s at the n iterative step, and $\alpha(n)$ be the step size at the nth iteration, then

$$\boldsymbol{g}_s^F(\boldsymbol{x}(n)) = U_s'(x_s(n)) - \sigma\sum_l\left(CI_l(n)\sum_{k\in IS_l} I_{s\in S^*(k)}\right) = U_s'(x_s(n)) - \sigma\sum_{m\in L^*(s)}\left(\sum_{l\in IS_m} CI_l(n)\right), \quad (8)$$

where I_z is the indication function: $I_z=1$ when z is true, otherwise, $I_z=0$. $CI_l(n)$ is the link congestion indicator for link l at the nth iteration,

$$CI_l(n) = \begin{cases} 0, & \sum_{k\in IS_l}\sum_{s\in S^*(k)} x_s(n) - C_l \le 0, \\ 1, & \text{otherwise.} \end{cases} \quad (9)$$

Thus x_s is updated as

$$x_s(n+1) = \left[x_s(n) + \alpha(n)\boldsymbol{g}_s^F(\boldsymbol{x}(n))\right]_{x_s^{\min}}^{x_s^{\max}} = \left[x_s(n) + \alpha(n)\left(U_s'(x_s(n)) - \sigma\sum_{m\in L^*(s)}\left(\sum_{l\in IS_m} CI_l(n)\right)\right)\right]_{x_s^{\min}}^{x_s^{\max}}, \quad (10)$$

where $[z]_a^b = \min\{\max\{z,a\},b\}$.

Remarks (1) Note that the subgradient algorithm converges to the optimal solution, with step size satisfying the following conditions (Shor, 1985):

$$\alpha(n)>0,\ \lim_{n\to\infty}\alpha(n)=0 \text{ and } \sum_{n=1}^{\infty}\alpha(n)=\infty. \quad (11)$$

(2) If we interpret CI_m as the "net" congestion indicator for link m, and $\sum_{l\in IS_m\setminus m} CI_l$ as the "additional" congestion indicator for link m, then $\sum_{l\in IS_m} CI_l$ is the total congestion indicator for link m. This kind of congestion information is consistent with this characteristic of ad hoc networks where transmissions x_s on one link $m\in L^*(s)$ contribute the congestion over all links in its interference set $l\in IS_m$. Therefore, congestion control protocols for ad hoc

networks must give feedbacks based on the congestion information in the entire interference set, rather than the individual link.

(3) Note that the distributive nature of Eq.(10) is attractive: Each source s can solve Eq.(10) separately without the need to coordinate with other sources. Source s acquires $\sum_{m\in L(s)}\left(\sum_{l\in IS_m} CI_l\right)$ by summing up the link congestion indicator over the interference set of all links which f_s goes through. Thus, the session at each source can independently adjust the transmission rate so that the optimal tradeoff can be achieved between the derivative of the utility function and the total congestion level in the interference set of the links that the session goes through.

(4) In order to implement Eq.(9), the capacity of link l is required to be known in advance, which is very difficult in reality. In practice, the queue size at the sender of the link can break down this difficulty, because $\sum_{k\in IS_l}\sum_{s\in S^*(k)} x_s - C_l$ can be estimated by counting the aggregate change of the queue size at the sender of link l over a certain time window, i.e.,

$$\int_{t=T_n}^{T_{n+1}}\left(\sum_{k\in IS_l}\sum_{s\in S^*(k)} x_s(t) - C_l(t)\right)\mathrm{d}t = \sum_{k\in IS_l}\left(q_l(T_{n+1}) - q_l(T_n)\right)^+, \quad (12)$$

where $q_l(T_n)$ is the instantaneous queue size in the sender of link l at T_n, the time instant at the nth iteration of link l.

Thus, we can determine the link congestion indicator $CI_l(n)$ through monitoring the queue size at the sender of link l,

$$CI_l(n) = \begin{cases} 0, & q_l(T_n) \le \eta Buffer_l; \\ 1, & \text{otherwise}, \end{cases} \quad (13)$$

where $Buffer_l$ is the threshold of the queue buffer at the sender of link l, and η is chosen as 0.2 in our simulation.

When the average queue size is greater than the threshold, $CI_l(n)$ is set to 1 immediately, which ensures that the average queue size does not significantly exceed $\eta Buffer_l$. Note that $\sum_{k\in IS_l}\sum_{s\in S^*(k)} x_s$ is also not required to calculate $CI_l(n)$ in Eq.(13).

To overcome the occasional bursts of packets in the queue, the instantaneous queue size replaces the average queue size, which is calculated through an exponential weighted moving average. Thus,

$$CI_l(n) = \begin{cases} 0, & q_l^{\text{ave}}(T_n) \le \eta Buffer_l; \\ 1, & \text{otherwise}, \end{cases} \quad (14)$$

where

$$q_l^{\text{ave}}(T_{n+1}) = wq_l^{\text{ave}}(T_n) + (1-w)q_l(T_{n+1}). \quad (15)$$

We use a value of 0.35 for w in our simulation.

From the discussion above, at link l, no per-flow information needs to be stored or maintained.

Distributed implementation

We summarize the distributed algorithm as follows:

At the nth iteration:

(1) Link m's (in fact, its sender) algorithm:

Step 1.1: Receives the information on the queue size from the link l, $\forall l \in IS_m$ to compute $\sum_{l\in IS_m}\left(q_l(n+1) - q_l(n)\right)^+$;

Step 1.2: Computes the link congestion indicator CI_m according to Eq.(14);

Step 1.3: Communicates new information on CI_m to its inference set IS_m;

Step 1.4: Informs the session f_s about $\sum_{l\in IS_m} CI_l$.

(2) Source s' algorithm:

Step 2.1: Sums up the total congestion information of all links in its path $\sum_{m\in L^*(s)}\left(\sum_{l\in IS_m} CI_l(n)\right)$;

Step 2.2: Chooses a new transmission rate for the next period $x_s(n+1)$ according to Eq.(10);

Step 2.3: Transmits data at new rate to links in its path.

Remarks (1) In the proposed algorithm, one kind of coordinating information between sources and links/nodes needs to be exchanged i.e. the aggregated congestion indicator over the links' interference set. Note that one byte can represent 255 congested links in the interference set of links on a path. For most real networks, allocating just one byte for the congestion indicator field should be sufficient. Thus, the overhead of the network congestion indicator is quite small.

(2) To implement Steps 1.1 and 1.3, each link needs to communicate periodically its information about CI_m to its inference set IS_m. This can be

achieved by the sender and the receiver of link l, respectively, piggybacking this information onto HELLO packets in the AODV routing protocol and sending at a certain time interval periodically to their K+1 neighbor set. The procedure to describe in detail how to send the congestion indicator to the K+1 neighbor set is tedious. Considering the space of this paper, we omit this procedure.

(3) The proposed algorithm can adapt naturally to changes in network conditions. Although we assume that the links' capacity and interference set, the set of sources and their utility functions, and the paths that the sources use, are unchanged, this algorithm can extend directly to the case when these quantities are time varying. The simulation results showed that it can track the moving optimal solution, if the changing network conditions are slow relative to its convergence.

From the discussion above, we can know that the proposed algorithm can be implemented in a parallel fashion at each receiver/link, copes with the absence of the central coordination in ad hoc networks, has lower overhead of computation and communication, and is scalable to a broad range of network scenarios. Meanwhile, due to computation only based on local information, it can effectively save on the network resource. Thus, it is suitable for online implementation, and breaks down those challenges that ad hoc networks pose.

SIMULATION INVESTIGATION

We study the proposed algorithm POCC in two different simulation environments: the MATLAB environment and the NS2 environment.

MATLAB environment

The network scenario is randomly generated by drawing node positions from a uniform distribution on the unit square [0,1]×[0,1], and by allowing two nodes to communicate if their distance is smaller than the threshold 0.25. Fig.1 shows a static network scenario randomly generated at a time. The network has 30 nodes and several links, and is shared by four end-to-end multihop flows/sessions f_s (s=1, 2, 3, 4). The source, the destination, and the path for each session are labelled in Fig.1. The aggregated utility function is chosen as $\sum_{s=1}^{4}\log x_s$, which enforces proportional fairness (Kelly *et al*., 1998). The minimum and maximum rates are 0 and 500, respectively. The capacity of links 5, 6, 9, 10, 14, 15, 17 and 18 is 2000. The other links' capacity is 1000.

We investigated the convergence of the proposed algorithm POCC, and demonstrate the simulation results (Table 1 and Fig.2).

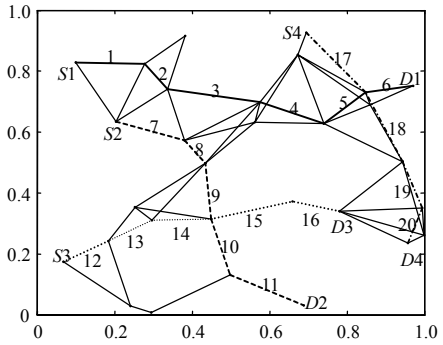

Fig.1 Ad hoc network scenario

Table 1 Globally optimum solutions and solutions given by POCC

Variables	Globally optimum solutions	Solutions solved by POCC	Relative error (%)
x_1	111.1111	111.86	0.675
x_2	129.6314	129.10	0.409
x_3	194.4419	196.47	1.043
x_4	333.3333	332.52	0.244

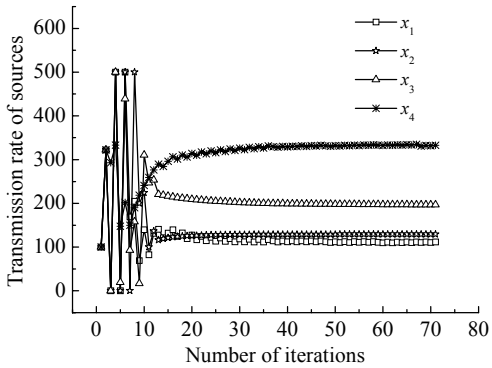

Fig.2 End-to-end transmission rate of sources

In Table 1, we compare the globally optimal rate of sources solved by the optimization toolbox in MATLAB, and the rate of sources given by POCC after 70 iterations. The results showed that the algo-

rithm could approach the globally optimum after 70 iterations. The plot in Fig.2 illustrated that they begin to approach global optimum after 40 iterations.

Remarks (1) Larger stepsize can result in higher convergence speed, but may cause larger fluctuation. The choice of stepsize is a trade-off between the convergence speed and the magnitude of fluctuation. Therefore, the stepsize must be chosen appropriately. A large stepsize is initially chosen to ensure fast convergence, and the stepsize is reduced step by step along the iteration procedure. In our simulation, the inverse of the number of iterations is chosen as the stepsize.

(2) From the algorithm, we know that, in each iteration, each source requires information on the prices of the links' interference set on the path. Therefore each iteration requires 1 RTT for end-to-end information exchange. As RTT is typically in the order of milliseconds to tens of milliseconds (Wang and Kar, 2005), we can infer that the overall convergence time of POCC should range from:

Convergence time of POCC
= 40 iterations×1 RTTs in each iteration
×milliseconds to tens of milliseconds
= 40 milliseconds to tens of milliseconds

Apparently, at this period, it is reasonable in most cases to consider the network situation as static in the practical ad hoc networks.

NS2 environment

We have implemented POCC in NS2 simulator and conducted extensive packet-level simulations. We demonstrated that POCC achieved efficient and fair resource allocation in ad hoc networks. In NS2 environment, the two-ray ground reflection model is used as the radio propagation model, IEEE 802.11 DCF as the MAC protocol, and AODV as the routing protocol. The transmission range and the interference range are 250 m and 550 m, respectively. The data transmission rate is 2 Mbps. The packets generated are of size 1000 bytes in all the simulations. All the simulations were run for 200 s. Every flow in the network exists in the entire simulation run, and each data point on the graph is averaged over 15 simulation runs.

1. Grid topology

POCC is evaluated and compared against default TCP (Reno version), and ATCP (Liu and Singh, 2001) for the grid topology shown in Fig.3. The metrics employed to measure the performance of POCC are the aggregate throughput and fairness. The aggregate throughput is measured in bps and reflects the number of packets successfully received at the destination.

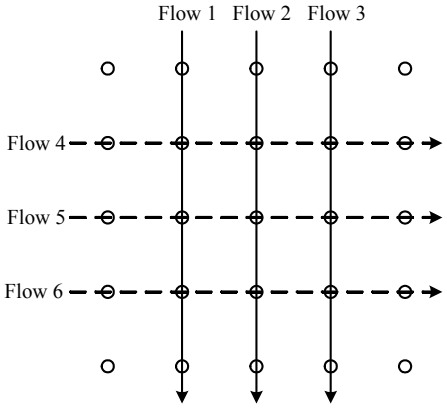

Fig.3 Grid topology

The aggregate throughput of each flow using POCC, TCP and ATCP is shown in Fig.4 showing that POCC preserved well fairness for all flows. Note that flow 2 and flow 5 received respectively 13.762% and 11.0536% of the overall throughput in the grid topology. In this case, TCP and ATCP failed to fairly allocate resource to these flows. The flows 1, 3, 4 and 6 almost occupied all channel resource and achieve high throughput, while the flows 2 and 5 were eventually starved, only received 3.353% and 5.032% of the overall throughput in TCP, and 6.873% and 5.121% in ATCP. It is evident from the results that

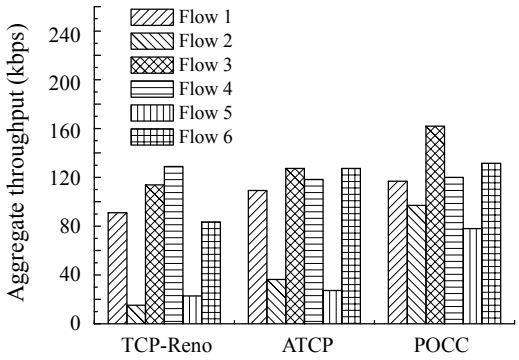

Fig.4 Aggregate throughput in grid topology

POCC can increase the degree of fairness in the network, since a link periodically sends its congestion indicator to its inference set, when it experiences congestion.

Also, we observe in Fig.4 that, for the grid topology, the throughput achieved by POCC can provide improvement of around 55% over TCP, and around 30% over ATCP. POCC can utilize the underlying network bandwidth more efficiently than TCP and ATCP, because POCC uses rate based congestion control mechanisms and explicit congestion indication.

2. Random topology

The setdest tool in NS2 is used to generate the random topologies for the simulations. The mobility model used for topology generation is the random waypoint model. All the simulations were performed for a 1500×1500 grid consisting of 50 nodes, distributed randomly over 2D grid. The source-destination pairs were randomly chosen from the set of 50 nodes in the network. Speeds of 0, 1, 5, 10, 15 and 20 m/s are considered in our simulations. Load on the network is 15 flows. FTP was the application that is used for all the flows in the network.

The performance of POCC is evaluated and compared against TCP and ATCP for the network scenarios outlined above. The metric employed to measure the performance is the overall throughput shown in Fig.5. We observe in Fig.5 that for the random topology, the aggregate throughput achieved by POCC can provide improvement of around 15%~55% over default TCP and ATCP.

These results showed that the performance of POCC is better than that of default TCP and ATCP.

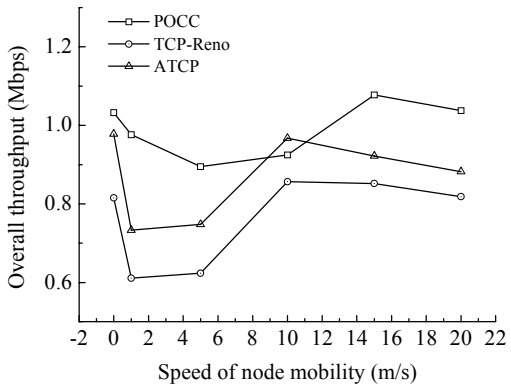

Fig.5 Overall throughput versus mobility

CONCLUSION

Due to the characteristics of ad hoc networks, congestion control must give feedback based on the congestion information in the link's interference set, rather than individual link. We introduce the notion of the link's interference set, extend the utility maximization problem of congestion control in wireline networks to ad hoc networks, apply the penalty function method and the subgradient method to solve this problem, and propose the distributed algorithm POCC. We have implemented POCC in NS2 simulator and conducted extensive packet-level simulations. We demonstrated that POCC achieved efficient and fair resource allocation in ad hoc networks.

References

Chen, L., Low, S., Doyle, J., 2005. Joint Congestion Control and Media Access Control Design for Ad Hoc Wireless Networks. Proc. Infocom 2005, Miami, USA.

Chiang, M., 2005. Balancing transport and physical layers in wireless multihop networks: jointly optimal congestion control and power control. *IEEE Journal on Selected Areas in Communications*, **23**(1):104-116. [doi:10.1109/JSAC.2004.837347]

Gupta, P., Kumar, P.R., 2000. The capacity of wireless network. *IEEE Trans. on Information Theory*, **46**(2):388-404. [doi:10.1109/18.825799]

Kelly, F.P., Maulloo, A.K., Tan, D.K.H., 1998. Rate control in communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research Society*, **49**(3):237-252. [doi:10.1038/sj.jors.2600523]

Lee, J.W., Chiang, M., Calderbank, R.A., 2006. Jointly optimal congestion and contention control in wireless ad hoc networks. *IEEE Communications Letters*, **10**(3):216-218. [doi:10.1109/LCOMM.2006.1603389]

Liu, J., Singh, S., 2001. ATCP: TCP for mobile ad hoc networks. *IEEE Journal on Selected Areas in Communications*, **19**(7):1300-1315. [doi:10.1109/49.932698]

Low, S.H., Lapsley, D.E., 1999. Optimal flow control I: basic algorithm and convergence. *IEEE/ACM Trans. on Networking*, **7**(6):861-874. [doi:10.1109/90.811451]

Mokhtar, S., Bazaraa, C., Shetty, M., 1979. Nonlinear Programming: Theory and Algorithms. Wiley, New York, p.338.

Shor, N.Z., 1985. Minimization Methods for Non-differentiable Functions. Springer-Verlag.

Wang, X., Kar, K., 2005. Cross-layer Rate Control for End-to-end Proportional Fairness in Wireless Networks with Random Access. Proc. ACM MobiHoc'05, Chicago, IL, USA, p.157-168.

Xu, W., Wu, T., 2006. TCP issues in mobile ad hoc networks: challenges and solutions. *Journal of Computer Science and Technology*, **21**(1):72-81. [doi:10.1007/s11390-006-0072-2]

Xue, Y., Li, B., Nahrstedt, K., 2006. Optimal resource allocation in wireless ad hoc networks: A price-based approach. *IEEE Trans. on Mobile Computing*, **5**(4):347-364. [doi:10.1109/TMC.2006.1599404]