International Journal of Computer and Communication Technology

Volume 2 | Issue 2 Article 12

April 2011

An Improvement in Stability of MANET using Flow Admission Control Algorithm

Mohit Ranjan Panda
C. V. Raman College of Engineering, Bhubaneswar, India, mohit1146@gmail.com

Manas Ranjan Mishra
Department of Computer Science, C.V. Raman College of Engineering, Bhubaneswar, Orissa., manascvrce@gmail.com

Recommended Citation
Panda, Mohit Ranjan and Mishra, Manas Ranjan (2011) "An Improvement in Stability of MANET using Flow Admission Control Algorithm," *International Journal of Computer and Communication Technology*: Vol. 2 : Iss. 2 , Article 12.
DOI: 10.47893/IJCCT.2011.1085
Available at: https://www.interscience.in/ijcct/vol2/iss2/12

An Improvement in Stability of MANET using Flow Admission Control Algorithm

Mohit Ranjan Panda[1], Manas Ranjan Mishra[2],
[1,2]Department of Computer Science, C.V. Raman College of Engineering,
Bhubaneswar, Orissa.
mohit1146@gmail.com, manascvrce@gmail.com

Abstract:

An ad hoc network is a collection of wireless mobile hosts forming a temporary network without the aid of any centralized administration or standard support services. However flows transported across mobile ad hoc wireless networks suffer from route breakups caused by nodal mobility. In a network that aims to support critical interactive real-time data transactions, to provide for the uninterrupted execution of a transaction, or for the rapid transport of a high value file, it is essential to identify stable routes across which such transactions are transported. Noting that route failures can induce long re-routing delays that may be highly interruptive for many applications and message/stream transactions, it is beneficial to configure the routing scheme to send a flow across a route whose lifetime is longer, with sufficiently high probability, than the estimated duration of the activity that it is selected to carry. We evaluate the ability of a mobile ad hoc wireless network to distribute flows across routes which is sufficiently stable for successful transmission. As a special case, for certain applications only transactions that are completed without being prematurely interrupted may convey data to their intended users that is of acceptable utility. We describe the mathematical calculation of a network's stable throughput measure, as well as its stable throughput capacity. We proposed the stable throughput and flow admission control routing algorithm (SFAR) to provide for the timely and stable transport of flow transactions across mobile ad hoc wireless network systems.

Keywords:
Ad hoc networks, Mobile wireless networks, QoS routing, Stable Throughput, Link survival time

1. INTRODUCTION:

Mobile ad hoc networks (MANET) are popular architectures because they offer to mobile hosts the opportunity to communicate and share services, in the absence of any infrastructure. MANETs are highly dynamic systems whose topology and hosts membership can change continuously - new nodes are joining, others are leaving, their relative position modifies. During its lifetime, MANET is characterized by the following set of management operations:

• *network setup,* usually executed by one host that also creates the MANET identity;
•*join/leave,* when a host joins or leaves an existing MANET;
• *merge* of two or more MANETs; one larger MANET is created;
• *split* of MANET in several sub MANETs;
• *Termination* when hosts disband and the MANET ceases to exist.

The performance of a mobile ad hoc wireless network is impacted by the dynamic stochastic process characteristics of its underlying links (and the associated noise interferences,

data rates, ranges, communications capacity levels), nodes (e.g., their mobility patterns and resource states), the underlying graph connectivity of the network topology, and the application induced traffic loading processes and their required quality of service (QoS) objectives. Under typical on-demand ad hoc routing algorithms, a source node that wishes to communicate across the network, initiates a route discovery process. Consequently, a route may be discovered for the transport of messages generated as part of the underlying session flow. Nodes located along the selected route keep forwarding entries used to route packets generated by the flow that uses the route. Under a proactive routing scheme, all nodes periodically exchange link state data and thus keep forwarding entries for all network domain destinations. In either case, the stableness of the route is generally not involved as a requirement for its selection. Consequently, route breakups will frequently occur, induced by nodal mobility and/or nodal and link failures as well as by fluctuations in the communications transport quality experienced across the network's communications links. As a result, a flow's route used to transport a file, or a group of packets that are part of an end-to-end user transaction (or burst), may be broken even before the flow's transaction (or corresponding session or call holding time) has expired.

In this paper, we derive new performance metrics that serve to characterize the level of stable transport service provided by the network. We define and discuss in the following such a 'stable throughput' measure. It entails the transport of a flow across a route that can guarantee acceptable communications capacity and stableness level objectives. It can be made based on credits awarded to a flow upon its reception at the intended destination.

We develop and study networking schemes that serve to improve the stable throughput features exhibited by a networked system. We hereby demonstrate such schemes by presenting the stable flow admission and routing algorithm (SFAR). We illustrate this algorithm to yield an operation that enhances the stable throughput behavior of the ad-hoc Networks. We present mathematical models for the analysis and design of such stable network systems, as well as confirm via simulations using NS2, the effectiveness of our on demand routing and flow control schemes in ensuring the system with high stable throughput performance.

2. CONVENTIONAL ROUTING SOLUTIONS

2.1. Description

A natural method for trying to provide routing in an ad hoc network is to simply treat each mobile host as a router and to run a conventional routing protocol between them [3, 9]. In effect, mobile host *B* in Figure 1 acts as a router between the “network” directly reachable by ***A*** and the “network” directly reachable by C. Host **A** transmits its packets for *C* to *B,* which then forwards them on to C. Conventional routing protocols are based on either *distance vector* or *link state* algorithms [l0].

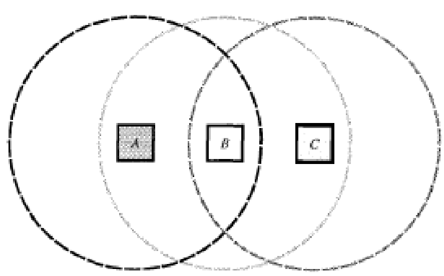

Figure 1 An ad hoc network of three wireless mobile hosts

In *distance vector* routing, each router maintains a table giving the distance from itself to all possible destinations. Each router periodically broadcasts this information to each of its neighbor routers, and uses the values received from its neighbors to compute updated values for its own table. By comparing the distances received for each destination from each of its neighbors, a router can determine which of its neighbors is the correct “next hop” on the shortest path toward each destination. When presented a packet for forwarding to some destination, each router simply forwards the packet to the correct next hop router. By transmitting routing table updates more frequently such as when any information in the table changes, the algorithm converges more quickly to the correct path (for example, when a link comes up or goes down), but the overhead in CPU time and network bandwidth for transmitting routing updates increases. Examples of distance vector routing protocols include the routing protocol used in the DARPA Packet Radio Network [3]; the original routing protocol for the ARPANET [6]; RIP (used in parts of the Internet [l], in Novell’s IPX [13], and in Xerox’s XNS [14]; and RTMP (used in AppleTalk) [14].

In *link state* routing, each router maintains a complete picture of the topology of the entire network. Each router monitors the cost of the link to each of its neighbor routers, and periodically broadcasts an update of this information to all other routers in the network. Given this information of the cost of each link in the network, each router computes the shortest path to each possible destination. When presented a packet for forwarding to some destination, each router forwards the packet to the next hop router based on its current best path to that destination. Link state routing protocols converge much more quickly as conditions in the network change, but generally require more CPU time (to compute the

complete shortest path to each possible destination) and more network bandwidth (to broadcast the routing update from each router to all other routers in the entire network) than distance vector algorithms.

2.2. Problems

Although using either type of conventional routing protocol in an ad hoc network, treating each mobile host as a router, may often work, there are a number of problems with this approach:

- Transmission between two hosts over a wireless network does; not necessarily work equally well in both directions.
- Many ‘‘links’’ between routers seen by the routing algorithm may be redundant.
- Periodically sending routing updates wastes network bandwidth.
- Periodically sending routing updates wastes battery power.
- Finally, conventional routing protocols are not designed for the type of dynamic topology changes that may be present in ad hoc networks.
- MANET Routing Protocols lack load-balancing (Less QOS).
- Dynamic stochastic process characteristic of underlying link (Un-stable).

3. GOAL

The performance of mobile ADHOC network can be improved by implementing SFAR(**stable throughput and flow admission control routing algorithm**) in conventional routing protocols.

4. STABLE THROUGHPUT

We evaluate the ability of a mobile ad hoc wireless network to distribute flows across stable routes by introducing the stable throughput measure as a performance metric .In the following; we illustrate the mathematical calculation of a network's stable throughput measure.

Let X={X_t, t ≥ 0} be the system's session size process. It is a stochastic process over the state space E = {0,1,2,….}, that represents the temporal evolution of the system's total number of supported flows.

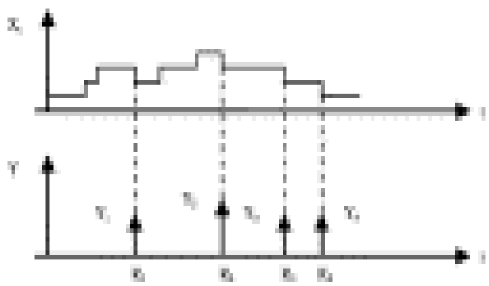

Figure 2. Random processes for total number of sessions in the system and their associated rewards.

The random variable Xt denotes the number of sessions supported by the system at time *t*. The *n*th supported session is admitted into the network system at time A_n and is assumed to be successfully transmitted at time R_n.

For departing session *n*, let random variable Yn denote the payoff (Reward) that the system obtains at time Rn upon the session's departure. The corresponding reward process is denoted as $Y = (Y_n;\ n = 1,2,3 \ldots\ldots\ldots)$, with the rewards assuming values in $R^+ = [0, +\infty)$.

The total reward gained by the system over the period of operation [0,t] is represented by random variable $Z_t = \sum_{n=1}^{N_t^D} Y_n$ where N_t^D denotes the number of session departures occurring over the period (0,t]. The average reward gained by the system per unit time, denoted as η, is an indicator of the operational efficiency of the system:

(1)

$$\eta = \lim_{t\to\infty} \frac{1}{t} E(z_t)$$

For example, when the gained reward process Y=Yn is represented as a sequence of independent identically distributed (reward) random variables, each assuming a finite mean value, and is also statistically independent of the session departure point process (and associated counting process; or more generally assuming that the variable N_t^D to be a stopping time relative to sequence *Y* so that Wald's lemma can be applied, assuming also a finite value for the average number of departures that take place over (0,t], we write:

(2)

$$\lim_{t\to\infty} \frac{1}{t} E\left(\sum_{n=1}^{N_t^D} Y_n\right) = \lim_{t\to\infty} \frac{1}{t} E(N_t^D) E(Y_n)$$

The session departure rate (also identified as the flow throughput, measured in units of flows/s) is defined as: $\lambda_d = \lim_{t\to\infty} \frac{1}{t} E(z_t)$; hence,

(3)

$$\eta = \lambda_d E(Y).$$

The traffic flows occurring across the network, over time and space are divided into flow of K classes $C_1, C_2, \ldots, C_K$. The offered loading traffic of class *k* is expressed (flows/s) $\lambda_0^k = \lambda_0 r(k)$, where $r(k) \in [0,1]$ expresses the relative loading level of the total class *k* flow rate, so that $\sum_K r(K) = 1$; λ_0 [flows/s] represents the total offered flow loading rate level.

Let Yn(k) denote the reward gained by the *n*th departing *k*-flow of K class. N_t^D (K) Representing the departure time of the corresponding flow. The corresponding reward process is denoted as $Y(k)=\{Y_n(k), n\geq 1$.The corresponding counting process is denoted as $N^D(k) = \{N_t^D(k), t \geq 0\}$.

The average long term gained reward rate is defined by

(4)

$$\eta = \lim_{t\to\infty} \frac{1}{t} E\left(\sum_k \sum_{n=1}^{N_t^D(k)} Y_n(k)\right)$$

Assuming reward variables to be finite mean independent and identically with mean value E(Y(k)), and the counting variable $N_t^D(k)$ to be a finite mean stopping time with respect to the sequence{Y(k)}, we conclude by Wald's lemma

(5)

$$\eta = \lim_{t\to\infty} \frac{1}{t} E\left(\sum_k \sum_{n=1}^{N_t^D(k)} Y_n(k)\right) = \sum_k \lambda_d(k) E(Y(k))$$

where

$\lambda_d(k) = \lim_{t\to\infty} E(N_t^D(k))/t$ represents the class k flow departure rate.

The corresponding matrix of session/flow admitted rates is denoted as :

$$\Lambda=\{\lambda_d(k),k=1,2,\ldots,K\}.$$

Let $P_B(k)$be the blocking probability that the system fails. We then calculate rate of admitted class K flows:

(6)

$$\lambda_d(k)=\lambda_o r(k)(1-P_B(k))$$

The gained reward rate is given by with blocking probability is

(7)

$$\eta = \sum_k \lambda_0 r(k)(1-P_B(k))E(Y(k)) = \lambda_d E(Y)$$

And total flow departure rate is equal to

(8)

$$\lambda_d = \sum_k \lambda_d(k) = \lambda_0 \sum_k r(k)(1-P_B(k))$$

4.1. The network's throughput rate

A class-k flow is characterized by parameters: the flow rate F(k)(bits/sec) and of the intended sessions holding time *H(k)* (expressed in units of seconds).

Let *L(k)* be a random variable whose distribution represents the distribution of the survival time of class-k routes. We define *T(k)* to be the random variable that represents the transport time of an admitted k-flow's session across its configured route. It is true that if the route is broken prior to the termination of the session's complete intended transport time, (H(k)).

Thus, we have for each successful transmission

(9)

$$T(k)=\min\{H(k),L(k)\}.$$

4.2. The network's stable throughput rate

As illustrated above, for certain applications, it is essential that the flow's session be carried out to completion to derive full benefit from the underlying interaction or transport of information. In such cases, a disruption, or premature termination, of the communications process in the midst of a session can significantly discount the value of packets delivered to the destination prior to such an early termination event. For example partial transmission of an computer executable program is meaningless.

For this purpose, let $S(k)\in[0,1]$ be a random variable that expresses the fractional success (or success probability) associated with k-session transport events. We set the credit *Y(k)* that a k-session gains upon the termination of its transport to depend upon its level of transport success: $Y(k)=F(k)T(k)S(k)$. In this manner, *Y(k)* designates a portion of the incurred transported information for which a class k session obtains credit upon termination. The sequence of successive class-k success variables forms the process $S(k)=\{S_n(k);n=1,2,\ldots\}$.

We define the efficiency factor associated with such a reward function as the system's stable throughput (denoted as fr and expressed in units of bits/s). That is,

(10)

$$f_s = \eta = \sum_k \lambda_d(k)E(F(k)T(k)S(k))$$

The stable throughput rate (expressed in units of flows/s) is defined as

(11)

$$\lambda_s = \sum_k \lambda_d(k)E(S(k))$$

When $S_n(K)=1$ for all n and k, every session receives full credit for all the data that has been transported. In this case, the stable throughput $f_s(\lambda_s)$ metric measures the system's throughput level $f(\lambda_d)$.

4.3. Throughput capacity and stable throughput capacity

The network system's throughput capacity f^c (measured in units of bits/s), also expressed as λ_d^c in units of flows/s, is defined as the maximum throughput level that a network can sustain, under prescribed operational and traffic loading conditions.

We note that the value assumed by the spatial reuse factor may depend upon the routing scheme R and flow admission algorithm A that are implemented in the network system and upon the underlying loading conditions. We denote the offered traffic flow matrix as Λ_0. Under the implemented flow admission control scheme, the resulting admitted traffic matrix is denoted as Λ. We thus write SRF(Λ),or , to explicitly point out the dependence of the SRF on the corresponding underlying processes.

Thus we have:

(12)

$\lambda_d E(FT\Pi)=R\cdot SRF(\Lambda)$,

where we set (for prescribed loading matrix Λ)

(13)
$$E(FT\Pi) \triangleq E[F(\Lambda)T(\Lambda)\Pi(\Lambda)]$$
$$= \lambda_d^{-1}\sum_k \lambda_d(k)E[F(k,\Lambda)T(k,\Lambda)\Pi(k,\Lambda)]$$

Consequently, for a given routing scheme R (or class of admissible routing policies), and under the fairness conditions imposed on the employed flow admission scheme (such as the conditions noted above and used in subsequent sections to illustrate the performance of the studied networks), the flow throughput rate is upper bounded as

(14)
$$\lambda_d \le \sup_{\Lambda_0 \in \Lambda^0} \frac{R.SRF(\Lambda)}{E[F(\Lambda)T(\Lambda)\Pi(\Lambda)]} \triangleq \lambda_d^C$$

so that λ_d^C is defined as the network's flow throughput capacity. Similarly, using Eq.(10), the stable throughput capacity rate f_s^C is given by

(15)
$$f_s \le \sup_{\Lambda_0 \in \Lambda^0} \frac{R.SRF(\Lambda)E[F(\Lambda)T(\Lambda)S(\Lambda)]}{E[F(\Lambda)T(\Lambda)\Pi(\Lambda)]} \triangleq f_s^C$$

For this calculation, we need to statistically characterize the lifetime of an intact (uninterrupted) network route, for routes of different hop lengths. We carry out such an analysis in the following section.

4.4. Survival time of links and routes

In [7] and [8], we have examined the behavior of link and route lifetimes by focusing on breakups that are induced by nodal mobility. Assuming a random waypoint mobility model (with relatively low values assumed for the times spent by nodes in pausing at the area boundary), we have shown that the distribution of the route survival time due to mobility L_m is well approximated by an exponential distribution. It is thus written as

(16)
$$P(L_m \le t) = 1 - e^{-\kappa t} \quad (t \ge 0)$$

where κ is a constant that is determined by the mobility pattern of the nodes. We have shown the parameter of the underlying link lifetime distribution to be well approximated by setting $\kappa=(v_1+v_2)\mu t/(2r)$, where μ is a parameter determined by the mobility pattern [7] , r denoted the link's communications range, and $v_1$and v_2 represent the speeds of the underlying link's end nodes; so that we have:

(17)
$$P(L_m \le t) = 1 - e^{-\frac{\mu(v_1+v_2)t}{2r}} \quad (t \ge 0)$$

To represent link failure events, we assume the following model. A link breakup can be induced by either one of the following two factors: (1) Nodal mobility that sets the link's end nodes to be at a distance that exceeds the threshold level r, and thus making communication ineffective at the desired bit error rate level. (2) Link outages that occur when the nodes are located within the designated communications range (r). Such outages can be caused by noise and interference processes, as well as the mobile character of the end nodes.

5. STABLE ROUTING AND FLOW ADMISSION CONTROL

5.1. Algorithm description

In mobile ad hoc networks, nodal mobility, as well as link fading events, leads to link breakups. Such failures cause established routes to frequently breakup. When a route is detected to break, the control subsystem will often try to reconstruct the route and/or discover and re-configure a new route, leading often to significant capacity utilization inefficiencies. Re-configuration processes are often not attempted when transporting flows for real-time applications. For the latter applications, it is desirable to implement a flow admission control mechanism that admits flows only if it is possible to discover a network route across which sufficient resources can be guaranteed to ensure a desired stableness target level (so that the route's lifetime is sufficiently long to carry, with high probability, the flow's transaction without early interruption). In addition, the selected route should offer the admitted flow packets with sufficient capacity resources so that prescribed (per application type) end-to-end packet delay (mean and jitter) levels are met. To achieve such an operation, we introduce in the following the distributed on-demand routing and flow admission (SFAR) scheme. This scheme operates in accordance with the following principles (extending the operation of the AODV scheme).

During the route discovery period, route request (RREQ) packets are flooded across the network (or a configured sub network, such as a Mobile Backbone Network, see [11], [10], [12]and[13] attempting to identify a route to the destination and then configure the routing entries in the routing matrices of the intermediate nodes located across the selected route. Under the SFAR scheme, a node receiving such a route request packet will proceed to forward (flood) it to its neighbors (if it has not yet forwarded such a RREQ packet) only if: (a) *stableness control*: It determines that the hence-to-fore route traveled by the RREQ message (including its current forwarding/outgoing link) is sufficiently stable, in relation to the end-to-end stableness level desired for the underlying flow; (b) *capacity/delay control*: The link capacity (or delay, or queue-size) level that it can offer to packets to be generated by this flow, if admitted, is acceptable.

Upon receipt of RREQ packets, the intended destination node waits for a prescribed period of time in collecting several, if any, RREQ packets. It then proceeds to examine the cumulative stableness status indices carried in the header of these packets (indicating the corresponding expected lifetime of the route traveled by each RREQ packet), as well as other indices, such as the hop-length of the traveled route. The destination node then, for example, selects an RREQ packet that provides acceptable stableness level while offering the shortest route among all such received RREQ packets. The selected RREQ packet induces the issue of a route reply (RREP) packet that is transported towards the source node across the selected route, configuring at this phase the forwarding entries at the routers located along the route. Of course, other performance measures can be employed for selecting a RREQ and the associated desired route; e.g., such a measure can be based on a metric that is computed as weighted sum of the involved stableness index, path hop-length and involved nodal delay (or congestion) identifiers. Clearly, this mechanism can be also implemented through an extended DSR scheme. In this way, the identity of the discovered route can be carried in the header of each packet.

Algorithm

SFAR – Processing RREQ packet

1: Waits for a prescribed period of time in collecting several, if any, RREQ packets

2: Examines the received RREQ packets

3: Selects an RREQ packet that provides acceptable stableness level while offering the shortest route among all such received RREQ packets

4: **if** It is the destination node **then**

5: Issue RREP packet

6: **else**

7: Modifies the RREQ packet

8: Forwards the RREQ packet

9: **end if**

In this way, routes can be selected on the following basis: (a) Under a next hop routing approach, each node proceeds to identify the best acceptable route to be used to next forward a received packet (using the cumulative performance index recorded in the header of a received packet). (b) Alternatively, under a source routing operation, the route is selected by the source node in accordance with the stableness and capacity/delay levels prescribed for the underlying flow, if such a route can be found (using the current layout of the weighted graph available to the source node). The identity of the route is indicated in the packet's header. Alternatively, a flow labeling process (as used by MPLS) can be invoked, so that label switching tables are configured by the control process once an acceptable route has been selected by the source node. Under each one of these mechanisms, if no acceptable route is discovered or calculated, the admission of the flow is rejected/blocked by the source node.

In the following, we illustrate the calculation of the stableness indices along a route and the simplified computation process that can be carried out to determine at an intermediate node whether the stableness of the route traveled by a packet is acceptable. For presentation purposes, assume our description to relate to a reactive routing operation under which RREQ packets are flooded for the purpose of discovering an acceptably stable route. During the route discovery phase, consider a RREQ packet received at node-*j* from node-*i*, and assume that node-*j* decides to forward this packet to its neighbors, after carrying out performance comparison with other RREQ packets that it recently received for the purpose of discovering a route for the same underlying flow. The latter received RREQ packet contains in its header a route vulnerability index (RVI), denoted as RVI(i). The latter serves as a cumulative vulnerability (lack of stableness) metric of the path traversed by this RREQ packet at the time received by node-*j*. With respect to such a packet, node-*j* performs the following computation to calculate the new (cumulative) route vulnerability index RVI(j) used to swap the RVI(i) value included in the received message; the new index is placed in the header of the new RREQ packet forwarded by node-*j* to its neighbors:

(18)

$$RVI(j) = RVI(i) + \frac{1}{2}\left(v(i) + v(j)\right) + \frac{r(i,j)}{\mu T_f(i,j)}$$

where v(i),v(j) denote the speeds of node *i* and node *j*, respectively ;r(i,j) denotes the effective communications range (at an acceptable bit-error-rate level) across the (i,j) link; and $\overline{T_f(i,j)}$ represents the mean duration of the time-to-fade along the same link. At source node *s*, we set *RVI*(s)=0.

We assume that the desired flow's route's stableness index (RRI) objective level β (with respect to the identified underlying session holding time *H*) is specified in the RREQ packet or, alternatively, can be determined from information embedded in the RREQ message, such as session or application type (or, for example, a differentiated service code identifier, if employed). Accordingly, it is desired for the selected route to exhibit a lifetime longer than the expected session holding time, so that $P(L \geqslant H) \geqslant \beta$. For this purpose, node-*j* performs the following comparison in determining whether to continue forwarding a RREQ packet that it has received from node-*i*, and tagged as candidate for forwarding (in comparison with other received RREQ packets, as noted above) for establishing a path for a flow of class *k*. The latter packet will be forwarded only if

$$VI(f) = -\frac{r(t_i,f)\ln\beta}{\mu H} \quad (19)$$

For example, assuming a required route's stableness index objective level, β=95% a nodal speed level of 5 m/s, and a session holding time of 3 s, we note by using Eq. (19) that the selected route must not be longer than three hops. However, if the latter holding time level is reduced to 2.4 s, it is noted that a route whose hop length is equal to four hops would also be able to meet the required level of stableness.

An intermediate scheme that does not employ a flow admission control mechanism that serves to block flows if a route that provides an acceptable stableness index is discovered, but that finds for a flow that is admitted based on residual capacity grounds a route that is highly stable, under current network layout conditions is also of interest. We have thus identified such a scheme as stable flow routing (SFR) scheme. Under the on-demand operational version of the latter, link stableness states are not used by nodes in making forwarding decisions for the flooded RREQ packets, though the cumulative stableness levels are still accumulated and recorded in the packet headers. Upon receiving such packets, the destination node selects a route that offers the highest stableness level, or that maximizes a benefit function that combines multiple attributes, such as stableness and path length (or path residual capacity)

5.2. Implementation of the SFAR algorithm

As discussed in previous sections, under the SFAR scheme, the concept is to select a route that will stay intact (with sufficiently high probability) for a certain period of time. This time duration represents the time it takes to transport across the network the data associated with a transaction that is executed by an underlying time critical application. The application may be real-time streaming based or of non-real-time character.

In evaluating the required holding time of a route that is selected for a transaction of a flow or burst, we note the following:

1. Clearly, for circuit switching networks, connection durations can be readily employed.

2. For packet switching networks, we observe two cases:

2a. For real-time applications, the required holding time is dictated by the application layer based duration of the burst/transactions for which we wish to insure that the network will likely employ a route whose lifetime duration is longer than the latter duration with high probability.

2b. In turn, for a critical file that is transported across a packet switching network and produced by non-real-time application, one proceeds to estimate the effective throughput rate that can be granted for the end-to-end (elastic, often feedback based) transaction of its packets.

For implementing the SFAR scheme when a datagram packet switching routing protocol is employed, the flow's source node can act to block the access of (i.e., discard) packets.

6. CONCLUSIONS

We propose stable throughput and stable throughput capacity measures to characterize the ability of a mobile ad hoc wireless network to provide highly survivable transport of flows. Such a service is critically required for supporting applications that often require flow transactions to be carried out to completion without interruption to yield maximum benefit. We note that a network system that is designed to yield a high throughput rate does not necessarily provide its users with a high measure of stable service and consequently may be characterized by low stable throughput performance. To provide a stable throughput transport service for flows that involve multi-packet transactions that should not be prematurely disrupted, we present a new scheme that implements a stable flow admission and routing scheme. Our on-demand stable routing algorithm selectively discovers routes that are probabilistically assured to survive for the duration of the underlying sessions or file transfers.

A number of additional adaptive mechanisms can be integrated into the schemes described in this paper to further enhance the stable performance of mobile ad hoc wireless networks. Future investigations will employ such techniques, using the methods developed in this paper for defining and analyzing the underlying stable performance features of the enhanced network system. Furthermore, to improve the stable behavior of a network layout, it is often advantageous to position stable and capable relay nodes, including unmanned ground vehicles (UGVs) and/or unmanned aerial vehicles (UAVs), in locations that serve to reduce the mobility impact of nodes on the stability of selected routes and therefore enhance the stable throughput performance of the network.

REFERENCES

[1]. C.E. Perkins, E.M. Royer, Ad hoc on-demand distance vector routing, in: Proceedings of the 2nd IEEE Workshop on mobile Computing Systems and Applications, WMCSA'99, February 1999, New Orleans, pp. 90–100.

[2]. D.B. Johnson and D.A. Maltz, Dynamic source routing in ad hoc wireless networks, *Mobile Computing*, Kluwer Academic Publishers (1996) pp. 153–181.

[3]. Z. Ye, S.V. Krishnamurthy, S.K. Tripathi, Effects of multipath routing on TCP performance in ad hoc networks, in: Proceedings of Global Telecommunications Conference,

GLOBECOM'04. IEEE, vol. 6, November 2004, Dallas, pp. 4125–4131.

[4]. Y.C. Hu, D.B. Johnson, Exploiting congestion information in network and higher layer protocols in multihop wireless ad hoc networks, in: Proceedings of 24th International Conference on Distributed Computing Systems, March 2004, Tokyo., Japan.

[5]. K. Fall, A delay-tolerant network architecture for challenged internets, in: Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, 2003, Germany, pp. 27–34.

[6]. J. Burgess, B. Gallagher, D. Jensen, B. Levine, Maxprop: routing for vehicle-based disruption-tolerant networks, in: Proceedings of INFOCOM 2006, 25th IEEE International Conference on Computer Communications, April 2006, Barcelona, Spain, pp. 1–11.

[7]. R. Zhang, I. Rubin, Mobility induced stable throughput behavior in mobile ad hoc networks, in: Proceedings of IEEE 60th Vehicular Technology Conference, 2004. VTC2004-Fall, vol. 4, 2004, Los Angeles, pp. 2708–2711.

[8]. R. Zhang, I. Rubin, Stable flow admission and routing for mobile ad hoc networks, in: Proceedings of Military Communications Conference (Milcom), October 2006, Washington DC.

[9]. D.B. Johnson, Routing in ad hoc network of mobile hosts, in: Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications, December 1994, pp. 158–163, Santa Cruz.

[10]. I. Rubin, R. Zhang, H. Ju, Topological performance of mobile backbone based wireless ad hoc network with unmanned vehicles, in: Proceedings of IEEE Wireless Communications and Networking Conference, WCNC'03, vol. 3, 2003, New Orleans, pp. 1498–1503.

[11]. I. Rubin, A. Behzad, H. Ju, R. Zhang, X. Huang, Y. Liu, R. Khalaf, Ad hoc wireless networks with mobile backbones, Mobile Radio Communications (PIMRC), vol. 1, September 2004, Barcelona, Spain, pp. 566–573.

[12]. X. Huang, I. Rubin, H. Ju, A mobile backbone network routing protocol with flow control, in: Proceedings of Military Communications Conference (MILCOM), vol. 2, 2004, pp. 1086–1092.

[13]. H. Ju, I. Rubin, Backbone topology synthesis for multi-radio meshed wireless LANs, in: Proceedings IEEE INFOCOM 2006 Conference, April 2006, Barcelona, Spain.

[14]. J. Hsu, I. Rubin, Performance analysis of multi-rate capable random access MAC protocols in wireless multi-hop networks, in: Proceedings IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC'06), September 2006, Helsinki, Finland.