Routing with Congestion Awareness and Adaptivity in Mobile Ad hoc Networks

Duc A. Tran and Harish Raghavendra
Multimedia and Collaborative Networking Group
Department of Computer Science
University of Dayton
Dayton, OH 45469
Email: duc.tran@notes.udayton.edu

***Abstract*— Mobility, channel error, and congestion are the main causes for packet loss in mobile ad hoc networks. Reducing packet loss typically involves congestion control operating on top of a mobility and failure adaptive routing protocol at the network layer. In the current designs, routing is not congestion-adaptive. Routing may let a congestion happen, which is detected by congestion control, but dealing with congestion in this reactive manner results in longer delay and unnecessary packet loss and requires significant overhead if a new route is needed. This problem becomes more visible especially in large-scale transmission of heavy traffic such as multimedia data, where congestion is more probable and the negative impact of packet loss on the service quality is more of significance. We argue that routing should not only be aware of but also be adaptive to network congestion. In this paper, we propose a routing protocol with such properties.**

I. Introduction

Routing in mobile ad hoc networks (MANETs) is an important problem in need of a solution that not only works well with a small network, but also sustains efficiency and scalability as the network gets expanded and the application data gets transmitted in larger volume.

Different dimensions can be used to categorize routing algorithms in MANETs: *proactive* routing versus *on-demand* routing, or *single-path* routing versus *multi-path* routing. In proactive protocols [2], [11], routes between every two nodes are established in advance even though no transmission is in demand. This is realized by a node periodically updating its neighbors with the routing information it has known thus far, hoping that every node eventually has a consistent and up-to-date global routing information for the entire network. This approach is not suitable for large networks because many unused routes still need to be maintained and the periodic updating may incur overwhelming processing and communication overhead. The on-demand approach (e.g., [1], [4], [7], [12]) is more efficient in that a route is discovered only when needed for a transmission and released when the transmission no longer takes place. However, when a link is disconnected due to failure or node mobility, which occurs often in MANETs, the delay and overhead due to new route establishment may be significant. To address this problem, multiple paths to the destination may be used as in multi-path routing protocols (e.g., [6], [8], [9], [13], [14]). An alternate path can be found quickly in case the existing is broken. The tradeoff, as compared to single-path routing, is the multiplied overhead due to concurrent maintenance of such paths. Furthermore, the use of multiple paths does not balance routing load better than single-pathing unless we use a very large number of paths (which is costly and therefore infeasible) [15].

There is another dimension for categorizing routing protocols: *congestion-adaptive* routing versus *congestion-unadaptive* routing. The existing routing protocols belong to the second group. In this paper, we propose a new routing protocol that belongs to the first group. We name the proposed protocol CRP (*C*ongestion-adaptive *R*outing *P*rotocol).

Our motivation is that congestion is a dominant cause for packet loss in MANETs. Typically, reducing packet loss involves congestion control running on top of a mobility and failure adaptive routing protocol at the network layer. Routing may let a congestion happen, which is later detected and handled by congestion control. Congestion unawareness in routing in MANETs may lead to the following problems:

- Long delay: It takes time for a congestion to be detected by the congestion control mechanism. In severe congestion situations, it may be better to use a new route. The problem with an on-demand routing protocol is the delay it takes to search for the new route.
- High overhead: In case a new route is needed, it takes processing and communication effort to discover it. If multi-path routing is used, though an alternate route is readily found, it takes effort to maintain multiple paths.
- Many packet losses: Many packets may have already been lost by the time a congestion is detected. A typical congestion control solution will try to reduce the traffic load, either by decreasing the sending rate at the sender or dropping packets at the intermediate nodes or doing both. The consequence is a high packet loss rate or a small throughput at the receiver.

The above problems become more visible in large-scale transmission of traffic intensive data such as multimedia data, where congestion is more probable and the negative impact of packet loss on the service quality is more of significance. Unlike well-established networks such as the Internet, in such

a dynamic network like a MANET, it is expensive, in terms of time and overhead, to recover from a congestion. Our proposed CRP protocol tries to prevent congestion from occurring in the first place and be adaptive should a congestion occur. Our ns-2 [10] simulation results confirm that CRP significantly improves the packet loss rate and end-to-end delay while enjoying small protocol overhead and high energy efficiency as compared to AODV and DSR.

The remainder of this paper is organized as follows. Next, we present the protocol details of CRP. We provide the results of our performance study in Section III. The paper is concluded in Section IV with pointers to our future work.

II. Congestion Adaptive Routing

CRP is a congestion-adaptive unicast routing protocol for MANETs. Every node appearing on a route warns its previous node when prone to be congested. The previous node uses a "bypass" route for bypassing the potential congestion area to the first non-congested node on the primary route. Traffic is split probabilistically over these two routes, primary and bypass, thus effectively lessening the chance of congestion occurrence. CRP is on-demand and consists of the following components: (1) Congestion monitoring, (2) Primary route discovery, (3) Bypass discovery, (4) Traffic splitting and congestion adaptivity, (5) Multi-path minimization, and (6) Failure recovery. We present the details of these constituent components in what follows.

A. Congestion Monitoring

When the number of packets coming to a node exceeds its carrying capacity, the node becomes congested and starts losing packets. A variety of metrics can be used for a node to monitor congestion status. Chief among these are the percentage of all packets discarded for lack of buffer space, the average queue length, the number of packets timed out and retransmitted, the average packet delay, and the standard deviation of packet delay. In all cases, rising numbers indicate growing congestion. While any of these methods can work with CRP in practice, we adopt the following simple method as an example in the paper. A node periodically checks the occupancy of its link-layer buffer. The congestion status is determined based on the ratio r between the the number of packets currently buffered to the buffer size. A node is said to be "green" (i.e., far from congested), "yellow" (i.e., likely congested), or "red" (very likely or already congested) if $r \leq$ 1/2, $r \in$ (1/2, 3/4], or $r \in$ (3/4, 1], respectively. As later discussed, a bypass is a path from a node to its *next green node*. The next green node is the first "green" node at least two hops away downstream on the primary route.

B. Primary Route Discovery

The sender discovers the route to the receiver in a simple way. It broadcasts a REQ packet toward the receiver. The receiver responds to the first copy of REQ by sending back a REP packet. The REP will traverse back the path that the REQ previously followed. This path becomes the *primary route* between the sender and the receiver. Nodes along this route are called *primary nodes*.

TABLE I
Primary Routing Table

Attribute	Description
dst	destination
hop	next node on the primary route
hop_status	congestion status of the next primary hop
prob	probability to forward a packet to the primary link
bypass_dst	destination of the bypass route
bypass_hop	next node on the bypass route
bypass_status	congestion status of the bypass route
green_hop	next node on the primary route that is green
green_metric	distance to green_hop in hops

Each node has two routing tables: primary table (denoted as $prTab$) and bypass table (denoted as $brTab$). $prTab$ is used to direct packets on the primary route while $brTab$ directs packets on bypass routes. Thus, $brTab = \emptyset$ for a node that does not appear on a bypass route of any connection. We will revisit $brTab$ later in subsection II-C. For $prTab$, the attributes are described in Table I. An entry in $prTab$ is unique to a destination node. We denote by $prTab[N, R]$ the entry for destination R in the routing table of node N, and $prTab[N, R].attr$ the value for attribute $attr$.

To reduce traffic due to primary route discovery and better deal with congestion in the network, we employ two strategies. First, a REQ is dropped if arriving at a node with a "red" congestion status. Second, if a REQ for receiver R arrives at a node N, this node may already have an entry for R as a result of a previous connection establishment. In this case, N just needs to forward REQ to $prTab[N, R].hop$ – the next primary node of N towards destination R. Broadcasting is avoided, and therefore we do not send too much traffic over the network.

On receipt of a REP initiated by destination R, a node N adds a new entry for R, or replaces the old entry with the new one, into $prTab$. The entry for R is removed if no data packet destined for R arrives or $prTab[N, R].hop$ is not heard of after a certain timeout period.

C. Bypass Discovery

A node N periodically broadcasts a UDT (update) packet with TTL = 1. The UDT packet contains N's congestion status and a set of tuples [*destination* R, *next green node* G, *distance to green node* m], each for a destination appearing in $prTab[N]$. This packet is created by Procedure *createUDT* shown in Figure 1. The purpose is that when a node N receives a UDT packet from its next primary node N_{next} regarding destination R, N will be aware of the congestion status of N_{next} and learn that the next green node is G which is m hops away on the primary route. This information is crucial in case a bypass is needed. The primary table is updated accordingly as shown in Procedure *recvUDT* in Figure 1. An exception applies to the node immediately before destination R. For this node, $prTab[., R].green\_hop = R$, $prTab[N, R].hop\_status =$

TABLE II

BYPASS ROUTING TABLE

Attribute	Description
dst	final destination
bypass_src	start node of bypass route
hop	next node on bypass route
status	congestion status of bypass route

"green", and $prTab[., R].green\_metric$ = 1. For every other node, $prTab[., R].green\_hop$ is set to -2 initially, meaning that this information is not yet available.

Suppose that a node N receives a UDT packet from its next primary node N_{next} (regarding a destination R). If N_{next} is yellow or red, a congestion is likely ahead if data packets continue to be forwarded on link $N \rightarrow N_{next}$. Since CRP tries to avoid congestion from occurring in the first place, N starts to discover a bypass route toward node G - the next green node of N known from the UDT packet. The bypass search is similar to primary route search except that: (1) the bypass request packet's TLL is set to $2 \times m$ (m is the distance from N to G on the primary route), and (2) the bypass request is dropped if arriving at a node (neither N nor G) already present on the primary route. Thus it is not costly to find a bypass and the bypass is disjoint with the primary route, except that they join at the end nodes N and G. It is possible that no bypass is found due to the way the bypass request approaches G. In this case, we continue using the primary route. However, [3] finds that the chance for a "short-cut" to exist from a node to another on a route is significant.

Aforementioned in subsection II-B, a node maintains $brTab$ - a bypass routing table to direct packets along bypass paths. This table, whose attributes are described in Table II. Each entry in $brTab$ is identified by a tuple {destination *dst*, bypass source *bypass_src*}. Similar to the primary routing table $prTab$, an entry of $brTab$ is removed if, after a certain timeout period, no data packet corresponding to this entry arrives or the next bypass node is not heard of. Bypass routing tables are only used by bypass nodes.

D. Traffic Splitting and Congestion Adaptability

Now that the bypass at a node has been found, data packets coming to this node are not necessarily spread over the bypass and the primary link. Indeed, as long as the next primary node is not red, no packet is forwarded on the bypass. This is because the primary route is still far from congested and we do not want to impose any unnecessary burden on the bypass nodes. We find bypass proactively as we can use it immediately if the next primary node becomes red (indicating severe congestion).

Let us consider a node N on the primary route from sender S to receiver R and assume that a bypass from N to the bypass destination $prTab[N, R].bypass\_dst$ is currently maintained but unused. When the next primary node of N (i.e., node $prTab[N, R].hop$) first becomes red, incoming packets will follow primary link $N \rightarrow prTab[N, R].hop$ with a probability

---

Procedure createUDT at node N
  output:packet p = (s, set of [R, G, m])
  s = current congestion status of N
  for (each destination R in $prTab[N]$)
    N_{next}: next primary node
    if(N_{next} is green OR N_{next} is R)
      G = N_{next}
      m = 2
    else
      G = $prTab[N, R].green\_hop$
      m = $prTab[N, R].green\_metric$ + 1
    add [R, G, m] to packet p

Procedure recvUDT at node N
  input: packet p = (s, set of [R, G, m])
  from: node N'
  for (each destination R in p)
    if(N' is the next primary node of N regarding R)
      $prTab[N, R].hop\_status$ = s
      $prTab[N, R].green\_hop$ = G
      $prTab[N, R].green\_metric$ = m

Fig. 1. Algorithms to determine the next green node

$p = prTab[N, R].prob$ = 0.5 and follow bypass link $N \rightarrow prTab[N, R].bypass\_hop$ with an equal chance ($1 - p = 0.5$). Hence, this traffic splitting effectively reduces the congestion status at the next primary node.

To adapt with congestion due to network dynamics, the probability p is modified periodically based on congestion status of the next primary node and the bypass route. The congestion status of a bypass is the accumulative status of every bypass nodes. For instance, in the bottom diagram of Figure 2, the status of the bypass route $A \rightarrow X \rightarrow Y \rightarrow C$ is red. The basic idea is that we should increase the amount of traffic on the primary link if the primary link leads to a less congested node and reduce otherwise. The probability adjustment policy is described in Table III.

We now work on an example demonstrated by Figure 2, where the bypass from A is $A \rightarrow X \rightarrow Y \rightarrow C$, from B is $B \rightarrow Y \rightarrow Z \rightarrow E$, and from D is $D \rightarrow W \rightarrow F$. We start from the top diagram and suppose that node B is aware of C becoming red. B adjusts the probability to forward on the primary link $B \rightarrow C$ to $0.5 - 0.5/2 = 0.25$ because the bypass congestion status is green. Similarly, if node D learns that its bypass becomes yellow, it will change the probability to forward on link $D \rightarrow E$ to $0.4 + 0.6/3 = 0.6$. The new probability values are shown in the middle diagram. Now, suppose that node Y becomes red and nodes C and W remain green, the new probability values on link $A \rightarrow B$, $B \rightarrow C$, and $D \rightarrow E$, shown on the bottom diagram, are changed to

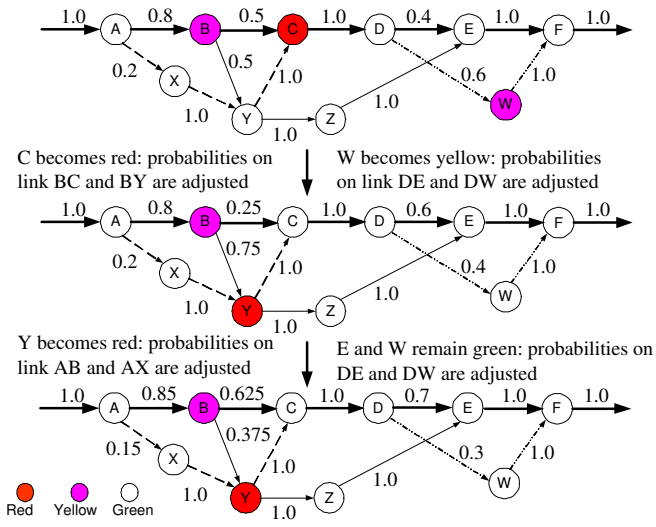


Fig. 2. Examples of splitting probability being adjusted adaptively to congestion

$0.8 + 0.2/4 = 0.85$ (because we assume node B is currently yellow), $0.25 + 0.75/2 = 0.625$ (because we assume node C is currently green), and $0.6 + 0.4/2 = 0.7$, respectively.

Although a bypass and the primary route cannot include more than two common nodes, we allow different bypass paths to share nodes. For instance, bypass node Y in Figure 2 belongs to two bypass paths. The rationale is that enabling bypass paths to share nodes increases the chance to discover a bypass. A bypass node may become too congested if it has to carry large loads of bypass traffic. This case is handled by our congestion adaptation explained above: this node will be eventually removed. We will discuss this kind of problem in the following subsection.

E. Multi-path Minimization

To keep the protocol overhead small, CRP tries to minimize the use of multiple paths. If $prTab[N, R].prob$ approaches 1.0 (e.g., within a pre-defined threshold), this means the next primary node is far from congested or the bypass route is very congested. In this case, N removes the bypass. If $prTab[N, R].prob$ approaches zero, this means that the next primary node is very congested. In this case, the primary link is disconnected and the bypass becomes primary. In either case, all the bypass nodes are informed of the decision and their routing tables are modified accordingly.

To further reduce the use of multi-pathing and keep the protocol simple, CRP does not allow a node to use more than one bypass. Therefore, the bypass route discovery is only initiated by a node if no bypass currently exists at this node. The protocol overhead for using bypass is also reduced because of short bypass lengths. A bypass connects to the first non-congested node after the congestion spot, which should be just a few hops downstream.

F. Failure Recovery

A desirable routing protocol should gracefully and quickly resume connectivity after a link breakage. CRP is able to do so by taking advantage of the bypass routes currently available. There are three main cases of failures and we address them below. For ease of presentation, we consider only one connection with sender S and receiver R. Figure 3 demonstrates these cases, where the bypass from B is $B \rightarrow Y \rightarrow D$ and from A is $A \rightarrow X \rightarrow Y \rightarrow Z \rightarrow E$.

1) A primary link fails (Figure 3(a)): When a primary link fails, say link $D \rightarrow E$, its initial node sends a DISC packet upstream towards the sender along the primary route. DISC records the nodes it visits. DISC stops at nodes that have a bypass. If a node receives the DISC (node B) and finds that its bypass destination (node D) is included in DISC, the bypass cannot be used because it leads a node before the failed link. In this case, DISC is further forwarded upstream. Eventually, DISC will stop at a node (node A) with a bypass whose destination (node E) is after the failed link, or DISC will reach the sender S. In the latter case, S finds a new primary route to R as in subsection II-B. If the former case occurs, the bypass will be used as the primary route.

2) A bypass link or bypass node fails (Figure 3(b)): This case is handled simply. The bypass node (node Y) that detects this failure sends a BPS_DISC packet upstream along the bypass until it reaches a primary node (node A). The bypass will be removed. Note that in Figure 3(b), the bypass $B \rightarrow Y \rightarrow D$ is not removed because the BPS_DSC only visits nodes that correspond to {destination R, bypass source A}.

3) A primary node fails (Figure 3(c)): This case is considered as a combination of the two cases above. If a primary node detects this failure, e.g., node D, it sends a DISC upstream along the primary route. If a bypass node detects this failure (node Z), it sends a BPS_DISC upstream along the bypass until reaching a primary node. When a primary node with a bypass (node A) receives DISC, it waits a period to see if a BPS_DISC is coming. If BPS_DISC comes, the bypass is removed and DISC will be forwarded upstream along the primary route. From then onwards, it is handled similar to the first case above. If BPS_DISC does not arrive within the waiting period, the bypass will be used as the primary route. However, it is possible that BPS_DISC comes late. In this case, it will be ignored, but the bypass that we just converted into primary actually remains broken at the failed node. This failure will be detected and fixed shortly because another DISC packet will be sent back (from node Z), as in Case 1.

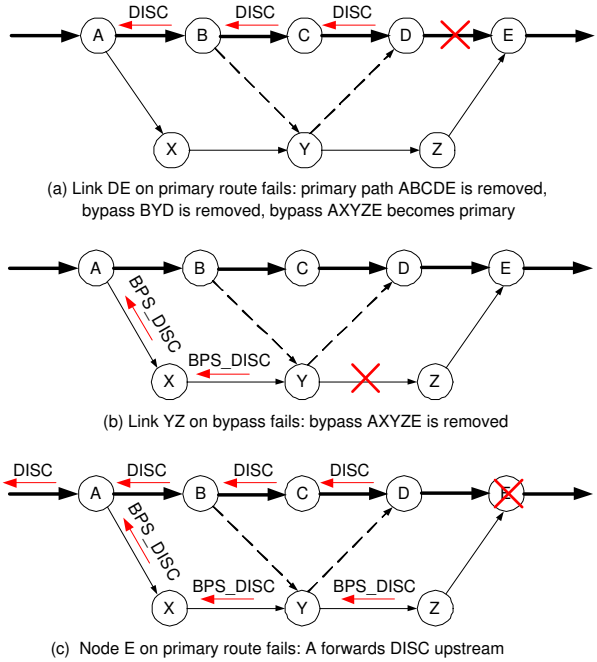
III. Performance Study

We implemented CRP using the Network Simulator Ns-2 version 2.27 [10] with the CMU Monarch wireless extensions. We compared CRP to DSR and AODV, two of the most popular MANET routing protocols. We present our observations in this section.

TABLE III

SPLITTING PROBABILITY ADJUSTMENT FOR CONGESTION ADAPTION

Congestion	bypass status = **green**	bypass status = **yellow**	bypass status = **red**
next primary node is **green**	$p := p + (1-p)/4$	$p := p + (1-p)/3$	$p := p + (1-p)/2$
next primary node is **yellow**	p unchanged	p unchanged	$p := p + (1-p)/4$
next primary node is **red**	$p := p - (1-p)/2$	$p := p - (1-p)/4$	find another bypass



(a) Link DE on primary route fails: primary path ABCDE is removed, bypass BYD is removed, bypass AXYZE becomes primary

(b) Link YZ on bypass fails: bypass AXYZE is removed

(c) Node E on primary route fails: A forwards DISC upstream

Fig. 3. Failure recovery: Bypass from B is $B \rightarrow Y \rightarrow D$ and from A is $A \rightarrow X \rightarrow Y \rightarrow Z \rightarrow E$

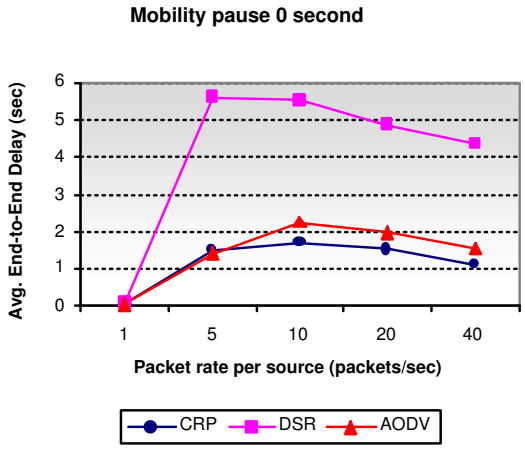


Fig. 4. Average end-to-end delay

A. Simulation Configuration

The network consisted of 50 nodes in a 1500m $\times$ 300m rectangular field. The radio model used was Lucent's WaveLAN radio interface whose nominal bit rate is 2Mbps and radio range 250m. The MAC layer was based on IEEE 802.11 DCF (distributed coordination function). The interface queue at the MAC layer could hold 50 packets before they were sent out to the physical link. The routing buffer at the network layer could store up to 64 data packets.

We used the random waypoint mobility model [5]. [7] suggested that by setting the maximum node speed to 4m/s we could cover most mobility effects by just varying the pause period. Therefore, we used this maximum speed and considered a highly mobile network by setting the pause period to zero. Two simple ways can be used to illustrate different traffic loads: (1) fix the packet rate and vary the number of connections, or (2) fix the number of connections and vary the packet rate. We employed the latter approach. Each simulation run lasted 300 seconds, during which 20 connections were generated and remained open until the simulation ended. For each connection, the source generated 512-byte data packets at a constant bit rate (CBR). This rate was varied among 1, 5, 10, 20, or 40 packets/s.

B. Performance Metrics

We considered the following important metrics for the evaluation: (1) Data Packet Delivery Ratio, (2) End-to-End Delay, (3) Normalized Routing Overhead (the ratio of the amount in bytes of control packets transmitted to the amount in bytes of data received), and (4) Normalized Power Consumption (the ratio of the amount in bytes of both control and data packets transmitted to the amount in bytes of data received). A desirable routing protocol should offer a small packet delivery ratio, small end-to-end delay, small routing overhead, and low power consumption.

C. Simulation Results

The results were collected as average values over 10 runs of each simulation setting. In what follows, the numeric results are demonstrated in figures. The percentage improvement of CRP over DSR and AODV is summarized in Table IV.

For end-to-end delay, we computed the average, worse-case, and standard deviation values, which are shown in Figure 4, Figure 5, and Figure 6, respectively. In all of these measures, CRP outperformed DSR and AODV, especially when the network traffic was heavily loaded. If we used DSR, every data packet carried the entire route information, thus making the network severely congested. Consequently, DSR suffered the worst delay.

AODV would result in a less congested network because neither data nor control packets need to include the entire

TABLE IV

IMPROVEMENT OF CRP OVER DSR AND AODV: POSITIVE VALUES MEAN IMPROVEMENT. E.G., 39.28% MEANS 39.28% BETTER

Rate	over DSR	over AODV	over DSR	over AODV	over DSR	over AODV	over DSR	over AODV
	Avg. delay	Avg. delay	*Delivery ratio*	*Delivery ratio*	Overhead	Overhead	*Energy*	*Energy*
1	28.57%	0%	*-2.8%*	*-1.00%*	-15.3%	16.66%	*-6.7%*	*-2.20%*
5	73.39%	-5.67%	*12.24%*	*-10.61%*	-77.77%	13.51%	*37.88%*	*-5.79%*
10	69.98%	26.22%	*42.04%*	*-2.8%*	-31.57%	26.47%	*48.54%*	*1.65%*
20	68.93%	22.16%	*78.51%*	*11.51%*	0%	39.28%	*57.08%*	*14.34%*
40	75.17%	28.48%	*118.37%*	*23.53%*	50%	50%	*62.08%*	*22.26%*

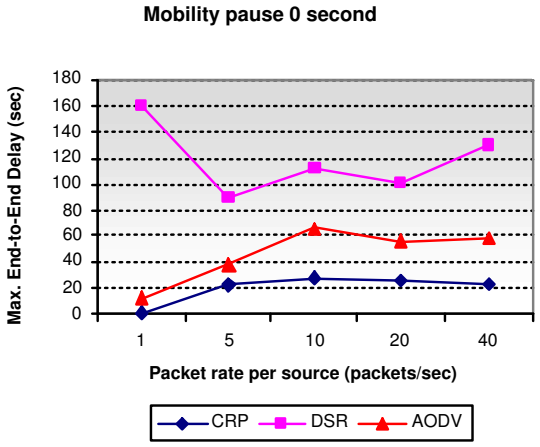


Fig. 5. Worse-case end-to-end delay

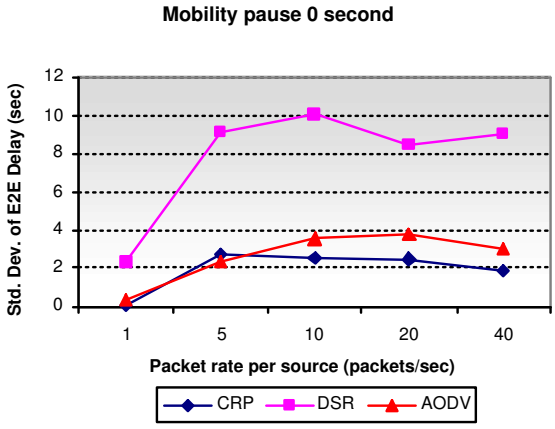


Fig. 6. Uniformity of end-to-end delay

route information. However, when the traffic load was high, AODV could not handle congestion. In contrast, CRP distributed traffic over the primary routes and bypass routes. Congestion was better resolved and therefore CRP offered the shortest end-to-end delay, on average and in the worst-case scenario. For instance, when the packet rate was higher than 10 packets/s, CRP improved over AODV by 57.28% and 26.22% in the worst case and on the average delay, respectively. The improvement over DSR was even more significant (75.32% and 69.98%, respectively). An interesting observation was that the delay variation in CRP was less than that of DSR and AODV (Figure 6), making CRP more suitable for multimedia applications.

In regard of data packet delivery ratio (Figure 7), both AODV and CRP performed better than DSR. Packets lost due to congestion in DSR were more than in the other protocols. When the packet rate was small (1 or 5 packets/s), AODV delivered a few more data packets than CRP. This was because network load was not yet heavy. But if more data were transmitted from the source, CRP delivered more successfully. Indeed, CRP successfully delivered 11.51% and 23.53% more data than AODV when the packet rate was 20 packets/s and 40 packets/s, respectively. The reason, again, was the ability of CRP to adapt to network congestion.

It is important to notice that not only CRP delivered more

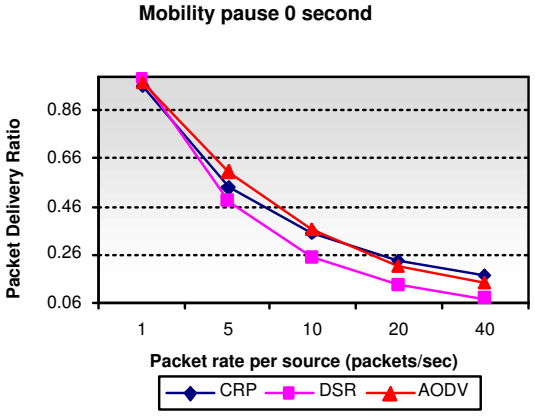


Fig. 7. Data packet delivery ratio

data than AODV did in heavily loaded networks, but CRP incurred less routing overhead (Figure 8). Especially, when the packet rate was 40 packets/s, while CRP delivered 23.53% more data than AODV did, the former incurred a normalized control overhead only half of AODV's. It is because, upon link breakage, while AODV tried to establish a new route to the destination by broadcasting a route request, CRP tried to make use of an available bypass. Therefore, route requests were sent

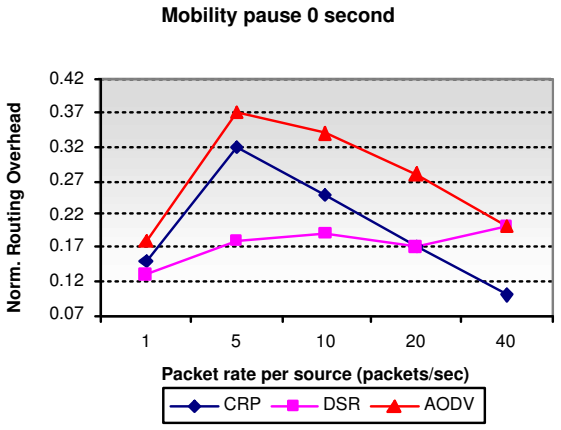


Fig. 8. Protocol Overhead

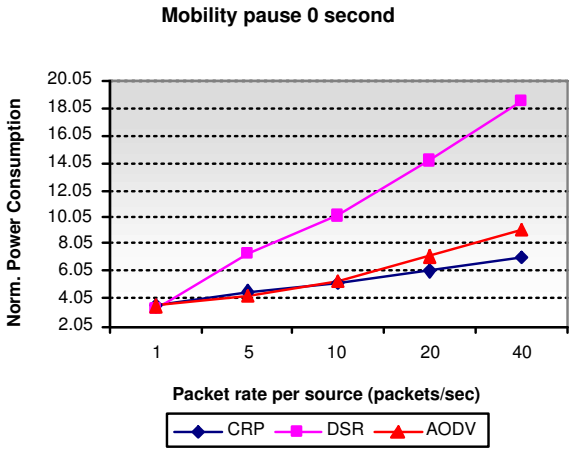


Fig. 9. Energy Efficiency

less often in CRP. One could argue that it costs some overhead to maintain bypass paths in CRP. However, this overhead is kept small because of the way we minimize the use of multiple paths as discussed in Section II-E.

Energy efficiency is crucial to any MANET. AODV and CRP were both competitive and more efficient than DSR. The gap between the first two became more noticeable in densely loaded networks, when DSR consumed even greater power (Figure 9). As mentioned earlier, CRP carried less control traffic. Therefore, the fact that CRP's energy efficiency was higher than AODV's implies that CRP forwarded less data traffic. Interestingly, the amount of data received in CRP was higher. This convinced us that being adaptive to congestion helps increase both the effectiveness and efficiency of routing.

IV. Conclusions

The uniqueness of CRP is its adaptability to congestion. CRP enjoys fewer packet losses than routing protocols that are not adaptive to congestion. This is because CRP tries to prevent congestion from occurring in the first place, rather than dealing with it reactively. A key in CRP design is the bypass concept. A bypass is a sub-path connecting a node and the next non-congested node. If a node is aware of a potential congestion ahead, it finds a bypass that will be used in case the congestion actually occurs or is about to. Part of the incoming traffic will be sent on the bypass, making the traffic coming to the potentially congested node less. The congestion may be avoided as a result. Because a bypass is removed when the congestion is totally resolved, CRP does not incur heavy overhead due to maintaining bypass paths. The bypass maintenance cost is further reduced because a bypass is typically short and a primary node can only create at most one bypass. A short end-to-end delay is also provided by CRP. Indeed, since CRP makes the network less congested, the queueing delay is less. Furthermore, since recovery of a link breakage is realized gracefully and quickly by making use of the existing bypass paths, the delay due to new-route establishment is also low.

Our future work will be focused on optimization techniques for CRP and how different congestion predication and control mechanisms cooperate with CRP to better reduce congestion in MANETs.

References

[1] S. Corson and V. Park. Temporally-ordered routing algorithm (tora) version 1 functional specification. Mobile ad hoc network (MANET) working group, IETF, October 1999.

[2] J. J. Garcia-Luna-Aceves and M. Spohn. Source-tree routing in wireless networks. In *IEEE International Conference on Network Protocols (ICNP)*, 1999.

[3] C. Gui and P. Mohapatra. SHORT: Self-healing and optimizing routing techniques for mobile ad hoc networks. In *ACM Mobihoc*, Annapolis, Maryland, 2003.

[4] J.Broch, D. Johnson, and D. Maltz. The dynamic source routing protocol for mobile ad hoc networks. IETF Internet draft, October 1999. Work in progress.

[5] D. Johnson and D. Maltz. *Ad hoc Networking*. Addision-Wesley, 2001. Dynamic Source Routing in Ad Hoc Wireless Networks.

[6] S.-J. Lee and M. Gerla. Split multipath routing with maximally disjoint paths in ad hoc networks. In *IEEE International Conference on Communication*, pages 3201–3205, Helsinki, Finland, June 2001.

[7] Y. Lu, W. Wang, Y. Zhong, and B. Bhargava. Study of distance vector routing protocols for mobile ad hoc networks. In *Proceedings of IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 187–194, Texas, March 2003.

[8] M. Marina and S. Das. On-demand multipath distance vector routing in ad hoc networks. In *IEEE International Conference on Network Protocols (ICNP)*, pages 14–23, 2001.

[9] A. Nasipuri, R. Castaneda, and S. R. Das. Performance of multipath routing for on-demand protocols in mobile ad hoc networks. *ACM/Baltzer Mobile Networks and Applications Journal (MONET)*, 6:339–349, 2001.

[10] NS-2. Network simulater. http://www.isi.edu/nsnam/ns/.

[11] C. E. Perkins. Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers. In *ACM SIGCOMM*, pages 234–344, 1994.

[12] C. E. Perkins, E. M. Belding-Royer, and I. Chakeres. Ad hoc on demand distance vector (aodv) routing. IETF Internet draft, October 2003. Work in progress.

[13] A. Valera, W. Seah, and S. Rao. Cooperative packet caching and shortest multipath routing in mobile ad hoc networks. In *IEEE Infocom*, San Francisco, CA, April 2003.

[14] S. Vutukury and J. Garcia-Luna-Aceves. Mdva: a distance-vector multipath routing protocol. In *IEEE Infocom*, pages 557–564, 2001.

[15] G. Yashar and A. Keshavarzian. Load balancing in ad hoc networks: Single-path routing vs. multi-path routing. In *Proceedings of the IEEE INFOCOM'04*, Hong Kong, March 2004.