Predictive congestion control mechanism for MANET

S. Subburam
Research Scholar, Satyabama University, Chennai, India.
Email:subburam@bsauniv.ac.in

P. Sheik Abdul Khader
Department of Computer Applications, BSA University, Chennai, India.
Email:hodca@bsauniv.ac.in

Abstract

In adhoc networks connection failure between source and destination often occurs, due to mobility of nodes. After every failure the connection between source and destination gets disconnected.The problem is while sending data packets from source to destination, there is a possibility of occurring congestion at any node incurring high packet loss and long delay, which cause the performance degradation of a network. This paper presents predictive congestion control routing protocol for wireless Ad-hoc networks called as PCCAODV. Unlike traditional AODV, predictive congestion index of a node as the ratio of current queue occupancy over total queue size at node level. Based on a congestion index, PCCAODV utilizes the upstream nodes and down stream nodes of a congested node and initiates route finding process Bi-directionally to find alternate non congested path between them for transmitting data. Suppose that the process find more non congested multi-paths and decides a finest single path for transmitting data. The protocol is implemented and simulated using Ns-2 simulator. Performance comparisons of the proposed PCCAODV protocol against AODV is presented and shown that the proposed algorithm performs well.

Keywords:*Ad hoc Networks, Congestion, AODV, PCCAOV*

1. Introduction

Ad-hoc are usually defined as an autonomous system of nodes connected by wireless links and communicating in a multi-hop fashions the benefits of ad-hoc networks are many, but the most important one is their ease of deployment without centralized administration or fixed infrastructure, thereby enabling an inexpensive way to achieve the goal of wireless communications [1].One of the fundamental tasks that an ad hoc network should perform is congestion control and its main objective is to limit the delay and buffer overflow caused by network congestion and provide better performance of the network.

In wire line networks, congestion control is implemented at the transport layer and is often designed separately as the transport layer and is often designed separately from functions of other layers. Since wired links have fixed capacities and are independent, this methodology is well justified and has been studied extensively.However, these result do not apply directly to wireless network because the ad hoc network, result in large amount packet loss, high delay, unfair scenarios and low throughputs , because each mobile node has limited transmission capacity and buffer[2].They mostly intercommunicate by multi-hop relay.

In adhoc networks Routing protocols can be divided into proactive, reactive and hybrid protocols are typically table-driven. Example of this type includes DSDV, WRP. Reactive or source-initiated on-demand protocols, in contrary, do not periodically update the routing information. It is propagated to the nodes only when necessary [1].Example of this type include DSR,AODV and ABR. Hybrid protocols make use of both reactive and proactive approaches. Example of this type includes TORA,ZRP [1] [7] [8].We note that the existing routing protocols are not have congestion control techniques and it is not considered, when establishing a new route and it remains the same until mobility or failure results in disconnection. Our motivation is the congestion is a main cause for packet loss in MANETs. A new perspective on this problem might be to realize congestion control in the MAC or network layer. After all, it might make sense to tackle the problem where it emerges. An exceedingly high network load is a problem closely associated with medium access and packet forwarding.[4].

1.1 AODV Routing Protocols

The Ad Hoc On-Demand Vector (AODV) routing protocol is an adaptation of the DSDV protocol for dynamic link conditions [2] [9]. Every node in an Ad-hoc network maintains a routing table, which contains information about the route to a particular destination. Whenever packet is to be sent by a node, it first checks with its routing table to determine whether a route to the destination is already available. If so, it uses that route to send the packets to the destination. If a route is not available or the previously entered route is inactivated, then the node initiates a route discovery process. This protocol performs Route Discovery using control message route request (RREQ) and route reply (RREP) whenever node wishes to send packet to destination. To control network wide broadcasts of RREQs, the source node use an expanding ring search technique. The forward path sets upon intermediate nodes in its routing table with a lifetime association using RREF. When either destination or intermediate node moves, a routs error (RERR) is sent to the affected source nodes. When source node receives the routs error message (REM), it can reinitiate route discovery if the route is still needed. Neighborhood information is obtained from broadcast Hello packet [2]. The RREQ propagation, 2 Reverse route creation, Forward route creation illustrated in figure 1,2,3.

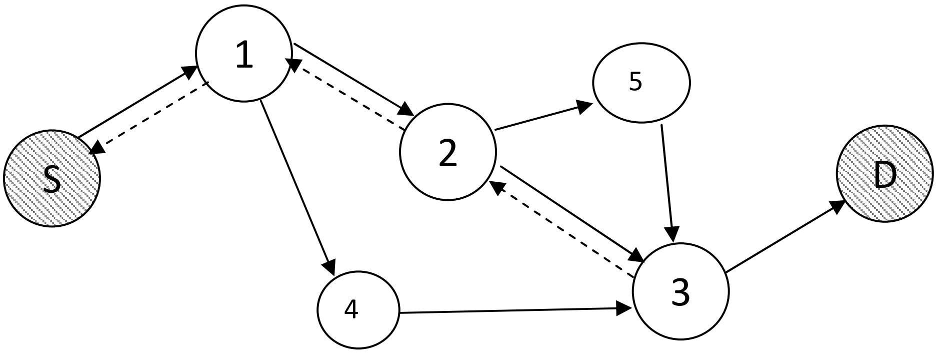

Figure 1 RREQ propagation

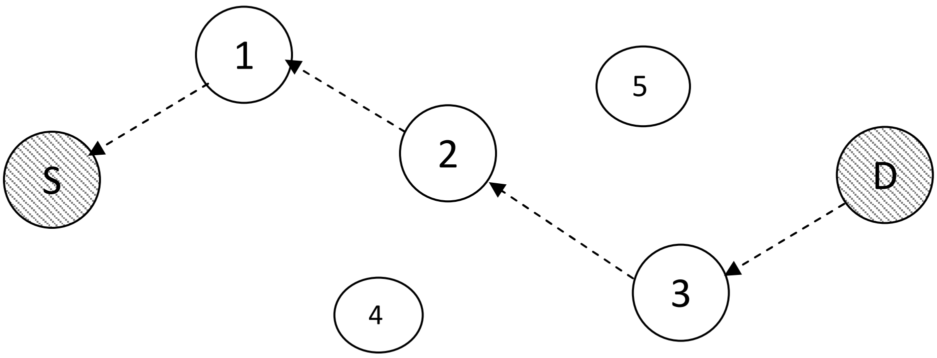

Figure 2 Reverse route creation

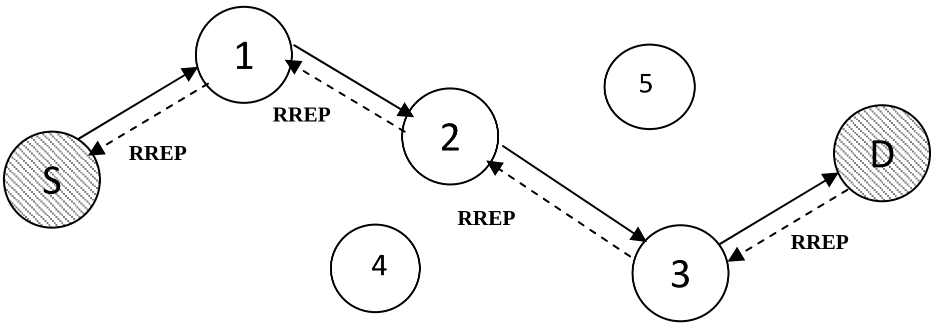

Figure 3 Forward route creation

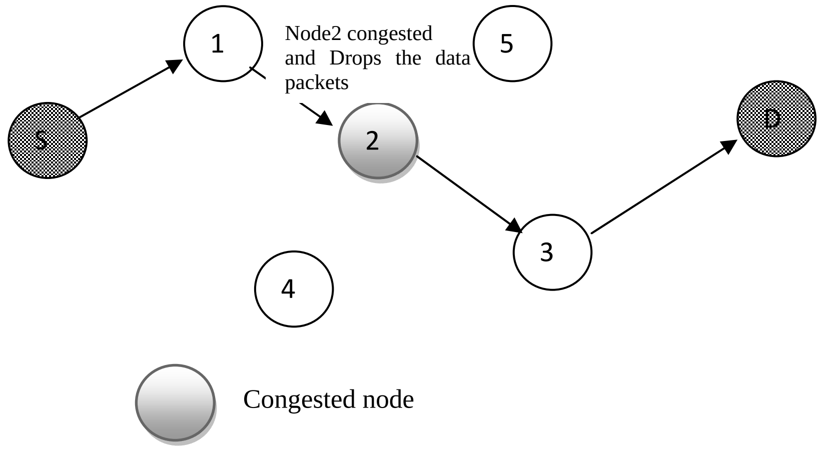

Figure 4 Upstream node 1 and Downstream node 3 Broadcasting RREQ

Since AODV has no congestion control mechanisms. Routing may let a congestion happen which is not handled by congestion control, it may lead to the following problems (i) Long delay (II) Many packet losses (iii) Low throughput. A simplified example is illustrated in Figure 4.A route S->1->2->3--> D, the route is initially found for the sender S to the receiver D, the node 2 has congested because of buffer overflow, it couldn't process all incoming packets so it drops all the packets.

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 of more significance. Unlike well-established networks such as the Internet, in a dynamic network like a MANET, it is expensive, in terms of time and overhead, to recover from congestion. 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.

2. Predictive congestion control routing protocol

In the paper, we used Predictive congestion control congestion technique in AODV routing protocol called as (PCCAODV) is a unicast routing protocol for MANET, tries to prevent congestion from occurring in the first place. Inn PCCAODV, every node participating on a route warns its neighbor nodes before congested based on the current occupancy of node buffer. The buffer consists of three zones.

Table I Three zones of node buffer

Safe Zone	Likely to be congested zone	Congested zone
Minth		Maxth

The PCCAODV consists of the following components.

1. Route discovery
2. Predictive congestion detection
3. Bidirectional path discovery

2.1 Route discovery

In the predictive congestion control routing is like for normal AODV, in the route discovery is dealing with congestion in the network. It broadcasts an RREQ packet toward the destination, the destination responds to the first arrived RREQ and sending back an RREP packet. The RREP will traverse back the path that the RREQ previously followed and adds a new entry in its routing table. This path becomes the primary route between the source and the destination.

Table II : Routing table of a Node

Parameters	**Description**
Src	Source
Dst	Destination
Next_node	Next node on primary path
Next_hop count	Next node hop count
NCongest_Status	Congestion status of next node
Next_noncong_node	Next non congested node
Next_noncong_ Node_hopcount	Next non congested node hop count
Prev_node	Previous node
Pcongest_status	Congestion status of previous node
Prev_noncongested _Node	Predecessor non congested node
Prev_noncong_Node_hopcount	Predecessor non congested node hop count

2.2 Predictive congestion control techniques

Congestion in a network may occur at any interval if the load on the network (the no of packets sent to the network) is greater than the capacity of the network (The no of packets network can handle)

Network Load > Capacity of Network

When the number of packets coming to a node exceeds its carrying capacity, the node becomes congested and starts losing packets. We have to use a congestion metrics at a node to detect eh congestion well in advance otherwise which is harmful when the buffer is near full. We use base design of RED model parameters two preset thresholds ; minimum threshold (minth) and maximum threshold (maxth)

Minth =25% buffer_size (1)

Maxth = 3*Minth (2)

To detect the congestion well in advance compute the average queue size

$Avgque = (1- w_q) * Avgque + Inst\_Que * w_q$ (3)

Where w_q , the queue weight is a constant parameter and Inst_que is a instantaneous queue size

In our Predictive congestion control model, we introduce queue status of actual queue size over average queue size given by Equation 4, which reflects the burstiness of the incoming traffic. Based on the queue_status , the mobile node can get useful information about the incoming traffic. If the queue _status value is large, the incoming traffic becomes bursty traffic. The continuous growth of the queue_status indicates that the incoming bursty traffic is beyond the mobile node’s buffer capacity and buffer overflow is imminent.

Queue_status = Inst_Que - Avg_que (4)

Where Inst_que is actual queue size and Avgque is average queue size.

Is Queue_status is low, the incoming traffic is less bursty. The transient congestion caused by small or short-lived bursty traffic should be accommodated since it does not cause buffer overflow.

A primary path of a node predicts its congestion status and periodically broadcasts a congestion status packet (CSP) with TTL=1. The CSP packet contains the node’s congestion status and a set of tuples (source S Destination D, P_node predecessor node, S_node successor node, S_Hop count, P_hop count)

Pseudo code 1 : Early detection congestion control

```
    Initialization
    Avgque=0
    Inst_que=0
    Min_th=0.25*buff_size
    Max_th=0.75*buff_size
    Warn_line=Buff_size/2
    For each arriving packet in the queue
    Inst_que++
    Calculate average queue size
    If the queue is not empty
Avgque= (1- wq) Avegque+ wq*Inst_que
    If (Inst_que > Avgque)
   Queue_status = Inst_que – Avgque
   Else
   Queue_status  = 0
   If (Inst_que > warn_line)
   If ( Queue_status  > min_th and Inst_que < max_th)
```

In this pseudo code 1 of early detection of congestion control algorithm, the calculation of average queue length involves the previous average queue length and the instantaneous queue length modified by a weight parameter, wq. Since wq is a constant parameter, a short-term increase in queue size resulting from bursty traffic or transient congestion need not result in a significant increase in the average queue size. As a result, the average queue (Avgquenew) changes much slower than the instantaneous queue (Inst_que).

Congestion in a network signifies that a node at any interval became congested and started to lose packets. Several metrics are available to monitor the congestion status at node level. For instance, it could be based on the average queue length and the percentage of packets discarded for lack of buffer space. Every second, a node checks the occupancy of its link layer queue using the dynamic congestion estimation technique so as to detect congestion well in advance. The Energitic Congestion (EC) estimation technique is a queue management algorithm that makes use of a direct measurement of the congestion status.

The EC algorithm uses three parameters, namely Minth, Maxth and w_q to standardize its performance. Minth and Maxth are queue thresholds to present the current status of the queue, and w_q is the queue weight parameter to compute the average queue size from the instantaneous queue length. The performance of EC depends on these thresholds. If the thresholds are small, then link utilization will be very low. If the thresholds are set too high, then congestion might occur even before the nodes are notified. To overcome this problem, we propose an effective threshold selection strategy.

Let us use expressions (1) and (2) to set the minimum and maximum threshold values for the queue length, which are dependent on the preferred average queue size. We initially set Maxth =2*Minth, because the EC would function most effectively when Maxth–Minth is larger than the typical increase in the calculated average queue size in one round-trip time, and a useful rule (from RED queue) is to set Maxth to at least twice of Minth. We then changed Maxth energitically based on the traffic condition. For this reason, we chose to fix the minimum threshold of 35%.

Minth =35% Queue_size (1)

Maxth =2 * Minth (2)

The objective of average queue length is to incorporate all the traffic fluctuations, and it follows the long-term changes of Inst_que, reflecting persistent congestion in the networks. Expression (3) is used to find the average queue length.

$$\text{Avgque}=(1-w_q) * \text{Avgque} + \text{Ins\_que} * w_q \quad (3)$$

The weight parameter, wq, regulates network congestion and acts as a time constant of the low-pass filter. The average queue length is desired to track recurrent network congestion that happens over a long period and, at the same time, filter short time congestion. This condition imposes a setback on the selection of wq.If wq is too small, the average queue length could not grasp the long-range congestion, which might result in ineffective congestion detection. If wq is too large, the average queue length follows the instantaneous queue length, which also degrades the performance of the congestion estimation technique. Therefore, the value of wq should be related to the flow of traffic in the queue. The proposed EC algorithm would concentrate on assigning wq values energitically according to the traffic flow. Initially, wq is set to 0.002. We used expression (4) to set wq values energitically, where N is the number of active flows and P is the packet rate (number of packets per second).

wqnew = wqold * N * P (4)

If the average queue length is less than Minth and instantaneous queue is less than warn_line (warn_line = Queue_size/ 2), then the node is in zone I (safe zone). If the average queue length is greater than Minth and less than Maxth, then the node is likely to be in congestion and an alternative path discovery mechanism is initiated. In the mean time, if the instantaneous queue size is greater than Maxth due to heavy incoming traffic, the status of alternative path discovery becomes false.

Queue utilization=(Maxth + Minth)/2 (5)

In this situation, our algorithm introduces the Queue_utilization parameter, which will help to change the Maxth values energitically until the alternative path discovery becomes true. We used expression (5) to get Queue_utilization value (Minth = 35% Queue_size; Maxth = 70% Queue_size; and Queue_utilization = 87.5% Queue_size), which consists of three ranges. It varies from 85% to 90% queue size with 2.5% difference. Finally, if the average queue length is greater than Maxth, then node's congestion status becomes Zone-III (congested zone). The Pseudo code II of energitic predictive congestion estimation control is shown below.

Pseudo code II : Energitic Predictive congestion estimation control

```
//initialization
Avgnew = 0; Avgold = 0; Inst_Que = 0 ;
Minth = 0.35 * queue_size
Maxth = 2*Minth Queue_util[] = {0.85,0.875,0.9}
Wq = 0.002;
Warn_line = queue_size / 2
 //For each arriving packet in the queue
Inst_Que++
//Calculate average queue size
 If the queue is not empty then
Avgnew = (1-wq) Avgold+ Inst_Que * wq
If (Avgnew < min_th && Inst_que < Warn_line) then
 Begin
Queue_status = ‘‘Safe’’;
Else if (Avgnew > min_th && Avgnew < max_th) then
 Begin
Queue_status = ’’Likely to be congested’’;

// Initiate Alternate Route Discovery Process
 If (inst_que > max_th && alter_path = FALSE) then
 max_th = Queue_util[i++]*buff_size;
Else
 Queue_status = ‘‘Congested’’;
 Avgold = Avg;
 Wq = Wq*N*P
 End
 End For each departing packet in the queue
Inst_que - -
Call procedure create CSP
```

Variable parameters: N: number of active flows; P: packet rate (packets/s); Avgold: previous average queue; Avgnew: new average queue; Inst_que: instantaneous queue.
Fixed parameters: Wq: queue weight; Queue_util: maximum queue utilization; Minth: minimum threshold value; Maxth: maximum threshold value.

```
Pseudo code III : CSP Creation
  Procedure  for create CSP of at X
      Variable : Dst, Next_node, NCongest_status,
      Next_noncong_Node,
      Next_noncong_Nodehopcount, Prev_node;
      Congest_status,Prev_noncongested_Node,
      Prev_noncong_Node_hopcount
      Output:packet p=(Cong,set[D,P_non_node, P_hon-
opcount,
                    S_non_node,S_non_HopCount])
     Cong = current congestion status of X
     For (each destination Dst in routing table of X)
     If (NCongst _status is Zone I)
     D=Dst
     S_non_node= Next_node
     S_non_Hop count= Next_hopcount
     Else
     D=Dst
     S_non_node = Next_noncong_Node
     S_non_Hop count = Next_noncong_Node_hopcount
     If( Pcongest _status is Zone I)
     P_non_node + Prev_node
     P_non_Hop count = Next_hop count -1
     Else
     D=Dst
     P_non_node = Prev_noncongested_Node
     S_non_Hop count= PREv_noncong_Node_hopcount
     insert( [D,S_non_node, S_non_ Hop count, P_non _node, P_non_hop count] to
packet p
```

3 Bi-directional path discovery

When the predecessor node and successor node receive a CSP packet from its primary path node of X regarding to destination D, predecessor node and successor node will be aware of the congestion status of X and know that the Non-congested nodes in the primary path and their hop count. This information is breakthrough to find bidirectional alternate path. The primary table of predecessor and successor node updated accordingly. The pseudo code for receiving CSP below:

```
Pseudo Code IV : Receive CSP at non-congested node
Procedure  recv CSP at non-congested node
Input : packet p = [D,P_non_node, P_non_hop
Count, S_non_node,S_non_Hop count,] From X
For (each destination D in p)
If (route exists in the routing table for destination S_non
-node or P_non_node) then
        Choose non congested route as alternate route
Else if (no route exists) then
Inititate the route discovery process bi directionally
P_non_node source and S_non_node by broadcasting the
BiRREQ packet in the network
Wait until timeout value  ‘ t’ for BiRREP
If(BiRREP at Source P_non_node)
If (Source P_non_node has multiple routes)
    Choose non congested shortest path as an alternative path
```

Update it's routing table

Else

Choose Single non congested path as an alternate path

Update it's routing table

Repeat the above procedure from Step I

The Working principle of the protocol, follows :

1. It initiates a route discovery process. The Prev_noncongest _Node starts to discover a bypass route toward Next_noncong_node of X known from the CSP packet and Next_Noncong_Node starts to discover a bypass route toward Prev_noncongested _Node of X known from the CSP packet. For this purpose, Prev_noncongested_Node broadcasts a BiRREQ (Bi directional route request) packet destined for Next_noncong_Node and Next_noncongested_Node broadcasts a BiRREQ packet destined for Prev_noncong_Node.
2. To reduce the network traffic when any primary path nodes receive BiRREQ packet and a BiRREQ is arrived at any congested node it is dropped automatically.
3. The immediate non-congested node may receive BiRREQ packet checks if it has an entry to the Next_noncong_node or Prev_Noncongested_Node in its routing table. If the intermediate node has a route entry, it will send a route reply BiRREP packet back to the originator nodes with congestion details.
4. If multiple entries to the same destination are found, it will select the ebst route with the metric Congestion status and if it doesn't have route entry the it will rebroadcast the BiRREQ packet in to the network.
5. The destination node is receiving a BiRREQ packet and reply BiRREP packets to source which ensuring that the paths are non congested path.
6. Before sending a BiRREP, the destination node will construct a Route Reply packet by appending the additional field of congestion status. The destination node will send route replies until the time out period has expired.
7. Each intermediate node along forward path to the source will update the congestion status field of RREP packet by calculated buffer capacity and update the value.
8. The destination node on receiving more than one Route Request packet from distinct intermediate nod e and reply to only those BiRREQ packets and ensuring that the paths are node-disjoint paths.
9. The source node Prev_noncongested_Node receives more than one route reply, contructs the routing table runs the algorithm for route selection also the source node updates in its route table for each destination.

10.(i) The bidirectional alternate path discovery process will reduce the time to find on congested alternate path.

(ii) The two disjoint alternate path , the shortest path will be a alternate path

(iii) the alternate path is disjoint with the primary route, except that they join at the end nodes Prev_noncongested_node and Next_noncong_Node

(iv) suppose no alternate oath si found between Prev_noncongested_node and Next_noncong_Node, it is working with primary path.

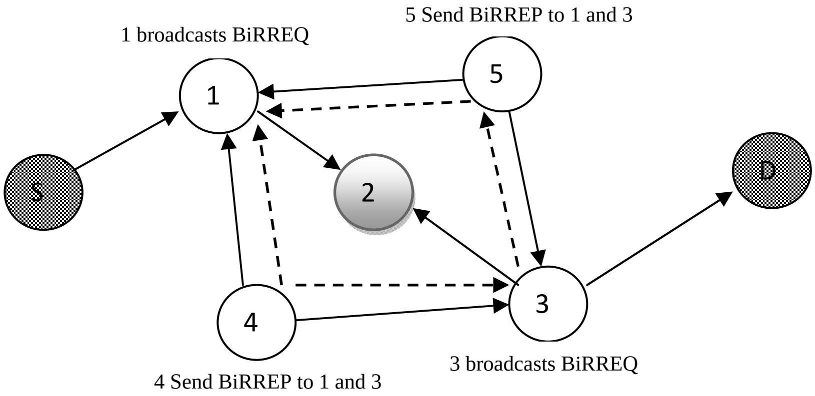

Figure 5 Discover non congested path by Bidirectional

A simplified example is illustrated in Fig 5 the non-congested predecessor node(1) and successor node(3) of congested node(2) receive CSP packet, they broadcast BiRREQ packet to find alternate path between 3 to 1 and 1 to 3 bi directionally. The intermediate nodes (between 1 and 3) 5 and 4 receive BiRREQ packet and sent replay packet BiRREQ to 1 and 3 respectively. If the predecessor source node 1 gets to disjoint multi-path between 1 to 3 (1 to 3 via 5, 1 to 3 via 4) in the two disjoint alternate path the hop count is same but the buffer occupancy of intermediate node 5 is 40% and 4 is 25% so that based on the buffer occupancy level the node 1 will choose (1-> 4->3 instead of 1-> 5->->6). in future this alternate path becomes primary path for S - > D

4.Performance study

We implemented PCCAODV using the Network Simulator Ns-2 version 2.33 [5] with the CMU Monarch wireless extensions [6] . We compared PCCAODV's performance to AODV routing protocols in MANET. We present our observation below.

4.1. Movement and Traffic Model

The random waypoint model [5] is used to model mobility of nodes. In this, minimum speed kept is at 0 m/sec and maximum speed at 20 m/sec. Pause time was varied from 0 sec to 900 sec for determining mobility effects and kept at 400 sec for determining effect of traffic loads. The number of mobile nodes in the simulation was 50. Other parameters are given in Table 1. The number of sources are 20, i.e. the number of sessions is 20 for determining mobility effects and varied from 5 to 30 for determining effects of traffic loads. Traffic sources with 512 bytes data packets at the rate 8 packets/sec and 20 packets/sec are CBR.

4.2. Performance Metrics

In order to investigate the performance of these protocols, we used the following performance metrics:

- Packet Delivery Ratio (PDR): It is the ratio between the packets received at the destination and the packets generated by the sources.
- Normalized Routing Overhead: It is defined as the percentage of control packets with respect to the received data packets. Each hop of any control packets is computed as a new control packet
- End-to-End Delay: It is the delay in transmitting data packets through wireless links plus the delay in the network interface queues due to network congestion.

4.3. Results and Analysis

4.3.1 Effect of Node Mobility

A 50 node model with 20 active sessions, each with 8 packets/second arrival rate with variable pause time from 0 second to 900 seconds were considered for simulation. It is observed that mobility has a great impact on the performance of PCCAODV, AODV . Performance is always better with high mobility in case of PCCAODV than AODV.

a)Average End-to-End Delay

With high mobility PCCAODV has lesser end-to-end delay compared to AODV. This is because of high interference and frequent link breaks. As mobility decreases average end-to- end delays of PCCAODV starts outperforming as compared to AODV. After 700 msec pause time, average end-to-end delay remains almost constant. This is because destination nodes will go for alternate paths if the congestion occurs during transmission. The average end-to-end delay with variable pause time of the mobile nodes for these three protocols is shown in Fig. 6.

b) Normalized Routing Overhead

Routing overhead of PCCAODV is similar to AODV. This happens because PCCAODV uses alternate route strategy in advance of congestion for route maintenance and delay information is piggybacked to data packets. The normalized routing overhead with variable pause time of the mobile nodes for these three protocols is shown in Fig. 7.

c) Packet Delivery Ratio

With high mobility PCCAODV has similar packet delivery ratio compared to AODV. This is because PCCAODV estimates delay dynamically. With high mobility, interference of neighbor nodes will be high. As mobility decreases, PCCAODV starts outperforming AODV because of its better route maintenance policy. Up to 700 msec pause time packet delivery ratio is gradually increased, after 700msec pause time it becomes almost constant. At low mobility, link breaks are very less and routes are balanced. The packet delivery ratio with variable pause time of the mobile nodes for these three protocols is shown in Fig. 8.

4.3.2 Effect of Traffic Load

Different traffic rates were simulated by using different number of sessions. A 50 mobile nodes model using random waypoint mobility with 400 seconds pause time and 8 packets/seconds arrival rate is considered for simulation.

a) Average End-to-End Delay

Average End-to-End delay of PCCAODV is almost similar to AODV at low traffic. As traffic increases, interference with neighbor nodes is increased resulting in frequent route changes. This causes the increase of average end-to-end delay at high traffic rate. Even though PCCAODV estimates node delay dynamically, average end-to-end delay is almost similar to AODV. The average end-to- end delay with variable number of sessions for these two protocols is shown in Fig. 9

b) Normalized Routing Overhead

Normalized routing over head of PCCAODV is slightly greater than AODV as it uses local link repairs and congestion at low traffic load. At high traffic, as the load on the system is more, nodes will become congested. The normalized routing overhead with variable number of sessions for these two protocols is shown in Fig. 10.

c) Packet Delivery Ratio

Packet delivery of PCCAODV is better than AODV as traffic load increases. This shows the efficiency of the PCCAODV. For PCCAODV, average end-to-end delay and normalized routing overhead is better at high mobility. Also the packet delivery ratio is significantly better at high traffic rate. PCCAODV tries to maintain the routes with the required low delays. This makes the routes less loaded and resulting less packet drops as congestion decreases. Therefore packet delivery ratio increases. With little expense of routing overhead, PCCAODV is able to deliver the packets at good percentage. The packet delivery ratio with variable number of sessions for these three protocols is shown in Fig. 11.

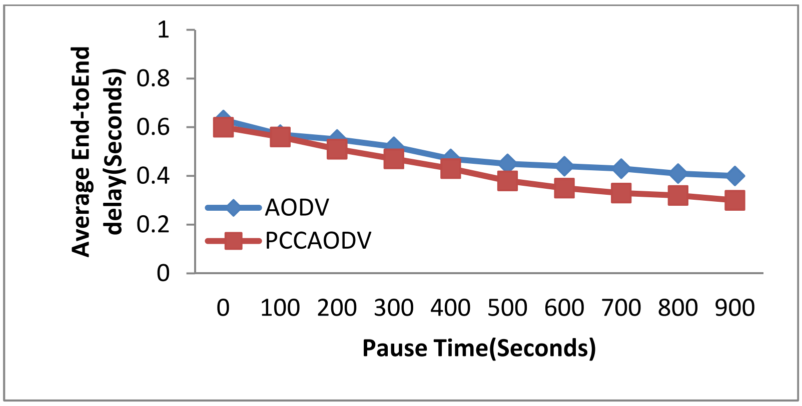

Figure 6 Average end to end delay for 20 sessions at 8 packets/sec

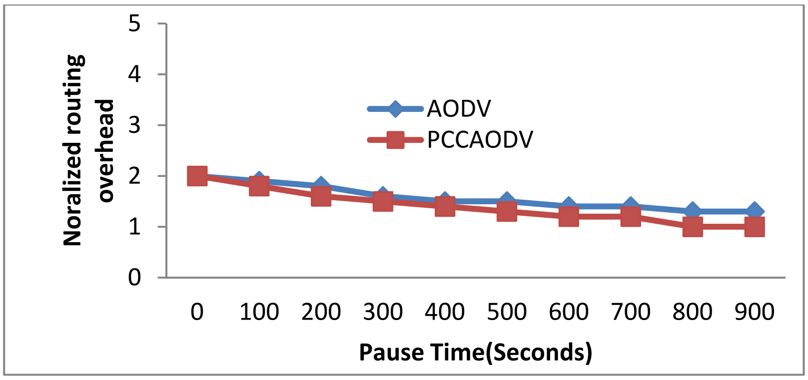

Figure 7 Normalized Routing Overhead for20 sessions at 8 packets/sec

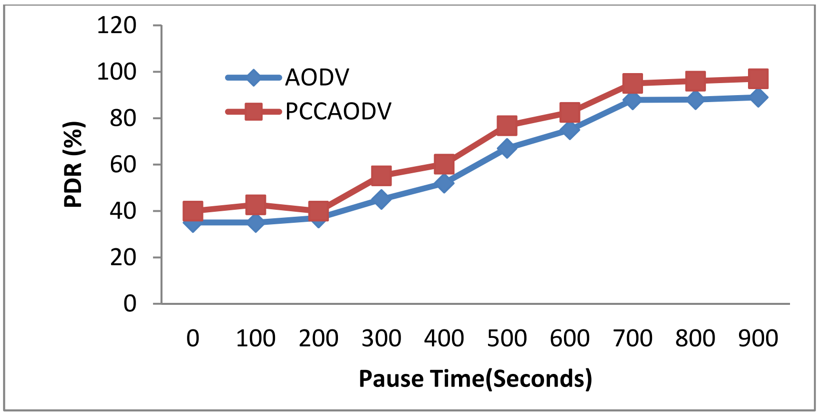

Figure 8 Packet Delivery Ratio for 20 sessions at 8 packets/sec

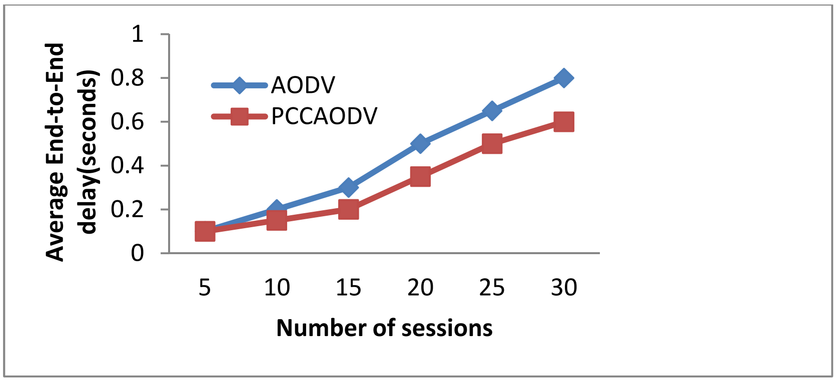

Figure 9 Average End-to-Ed delay at 400sec pause time and 8packets/sec

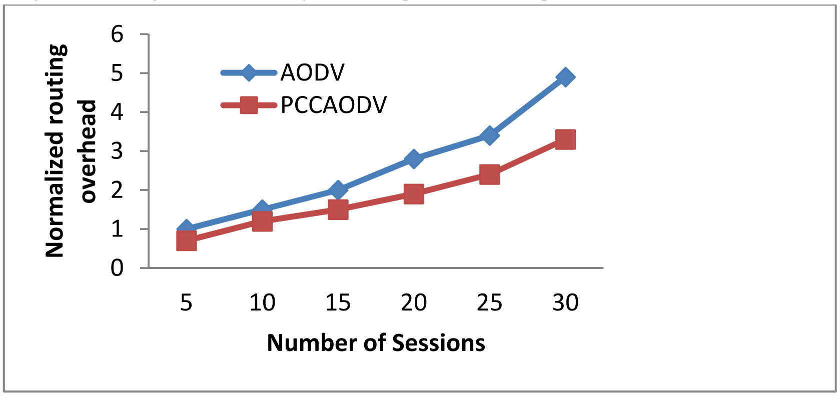

Figure 10 Normalized routing overhead at 400sec pause time and 8packets/sec

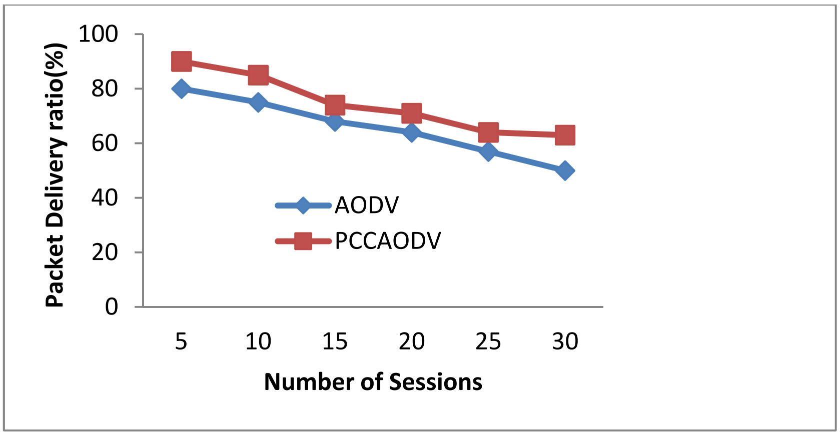

Figure 11 Packet Delivery ratio at 400sec pause time and 8packets/sec

5.Conclusion

We have proposed PCCAODV, a Predictive congestion control routing protocol in MANETs. PCCAODV has lost fewer packets than AODV that are not having congestion control mechanism. This is because PCCAODV tries to predict congestion from occurring in the first place, rather than dealing with it reactively. A key in PCCAODV design is the bi directional alternate path discover concept. This concept tries to find out non congested alternate path and tries to avoid congestion in the network. Our ns – 2- based simulation has confirmed that the advantages of PCCAODV had demonstrated a significantly improvement of end to end delay and packet delivery ratio over AODV.

References

[1] C.Siva Ram Murthy and B.S.Manoj,” Ad hoc Wireless Networks – Architectures and Protocols”, Pearson Education, 2007.

[2] C.E.Perkins, E.M. Belding – Royer, and I. Chakeres, “Ad Hoc On Demand Distance Vector (AODV)routing,” IETF Internet draft, Oct.2003.

[3] Yingun Yu; Giannakis, G.B.”Cross – Layer Congestion and Contention Control for Wireless Ad Hoc Networks” Wireless Communications,
IEEE Transactions on Volume 7, Issue 1, Jan.2008

[4] Christian Lochert Bj” orn s Scheuermann Martin Mauve, A Survey on Congestion Control for Mobile Ad-Hoc Network Wiley Wireless Communication and Mobile Computing 7 (5). Pp.655-676, June 2007.

[5] NS-2 Network Simulator, http://www.isi.edu/nenam/ns/,2008.

[6] CMU. “The CMU Monarch Project: Wireless and Mobility Extensions to ns,”http:www.monarch.cs.smu.edu/cmuns. Html.oct.1999.

[7] J.Broch, D.Johnson, and D. Johnson, and Maltz, “The Dynamic Sources Routing Protocol for Mobile Ad Hoc Networks,” IETF Inernet
draft, Oct.1999.

[8] D.Johnson and D. Maltz, Ad Hoc Networking. Addison – Wesley,2001.

[9] Ur Rahman Khan, K.Reddy, A.V.”An Efficient DSDV Routing Protocol for Wireless Mobile Ad Hoc Networks and its Performance Comparision_”Cpmputer Modelinmg and Simulation.2008.EMS’08.Second UKSIM European Symposium on Oublication Date: 8-10 Sept.2008.