Cross-Layer optimization to improve TCP performance in MANETs

Mahesh Bakshi, 05MCMB16
Advisors: Ms. Anupama Potluri & Dr. Atul Negi

-

Abstract

In MANETs, packet losses occur due to less reliable wireless medium of communication as well as mobility of the nodes that leads to loss of routes. TCP performs poorly in terms of the throughput achieved as it interprets these packet losses as congestion. We hope to improve TCP performance through cross layer optimization to handle these packet losses separately.

-

Overview

- Introduction
- Related work
  - Definition of cross layer design
  - Functionality of TCP flavors
  - Classification of proposals
  - TCP-F, ELFN proposals
- Our approach
- Implementation and simulation results
- Concluding remarks

Background/Introduction

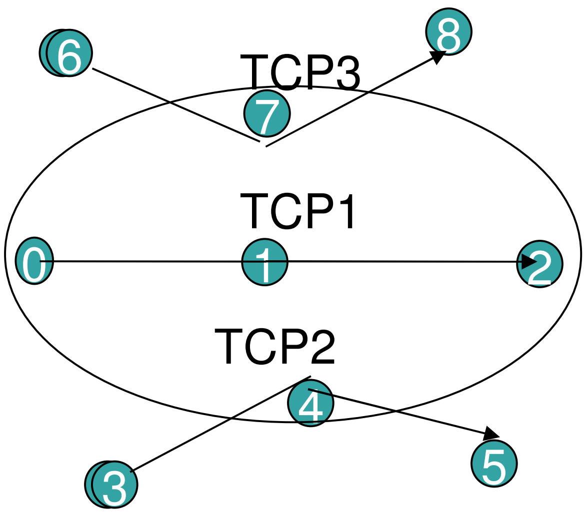
Problems of TCP performance in MANETs

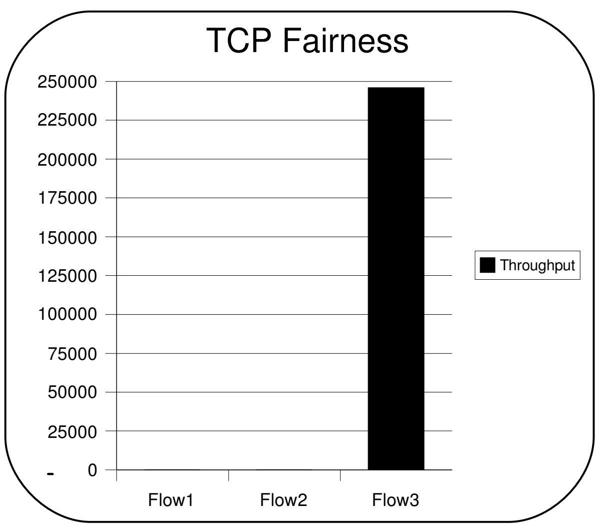
➔TCP is unable to distinguish between losses due to route failures and losses due to network congestion.

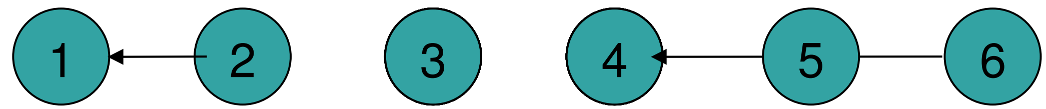
➔TCP unfairness.

-

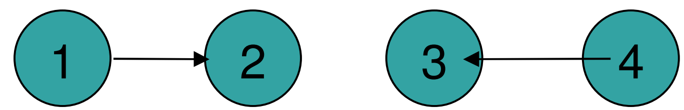
TCP Unfairness problem



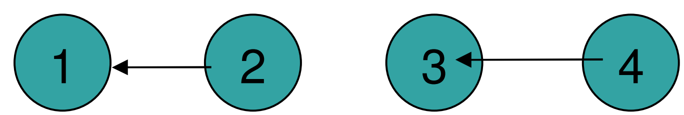




One-Hop Unfairness



Scenario 1



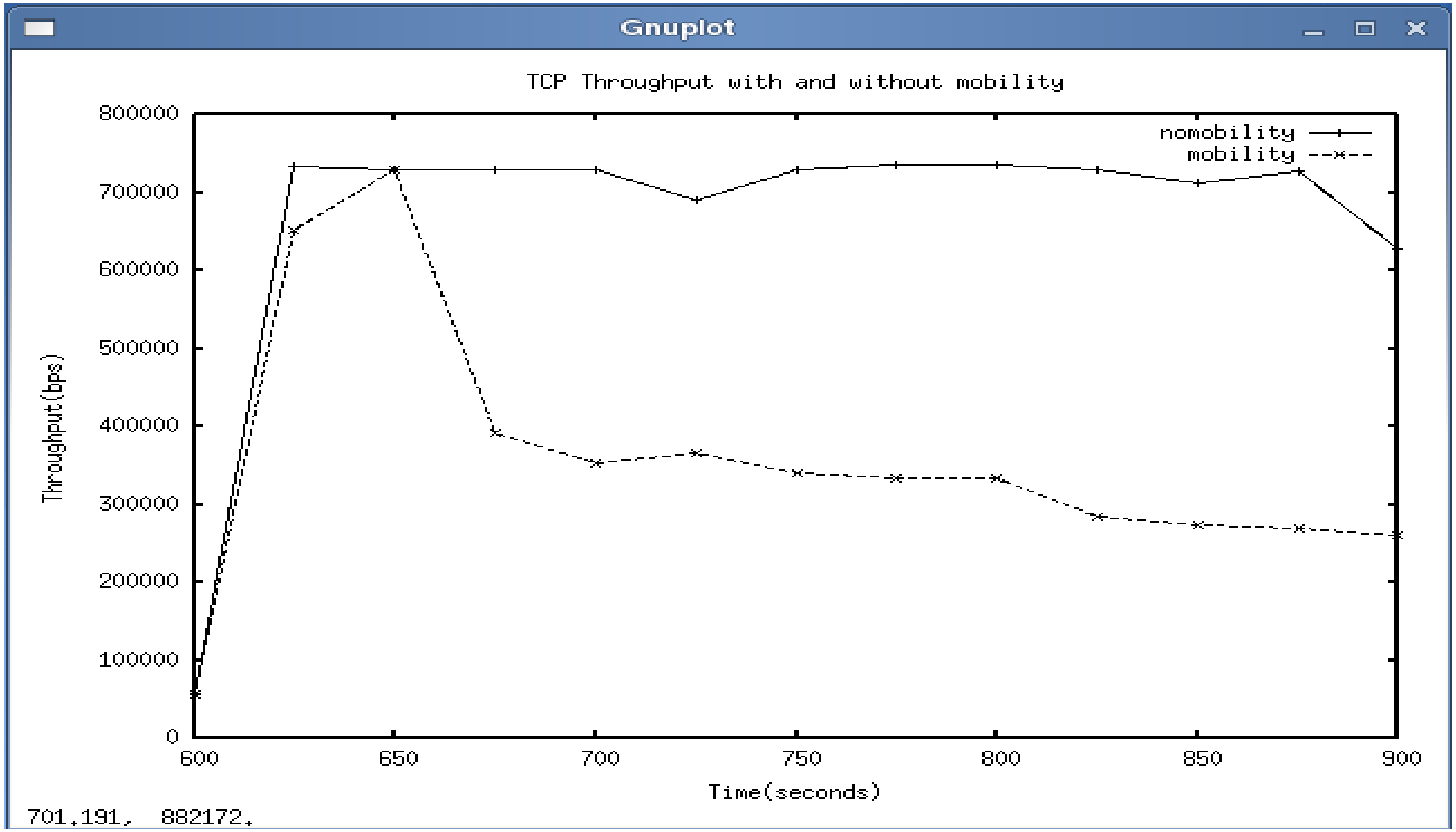
scenario 2

Statement of problem

- *The main purpose of our approach is to distinguish link failures occurring due to mobility of the nodes from congestion. We aim to achieve this using cross layer information with out adding traffic into network.*

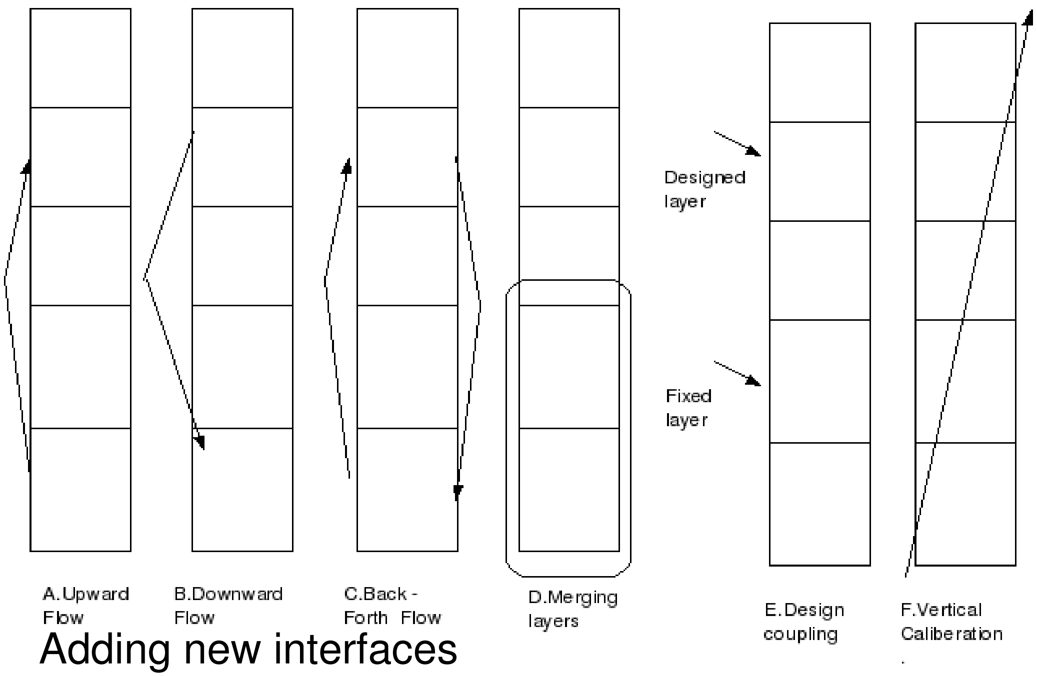
-

Motivation



TCP Throughput with and without Mobility (50n-20m/s-953L-2334R-no congestion)

Cross-layer design proposals(general)



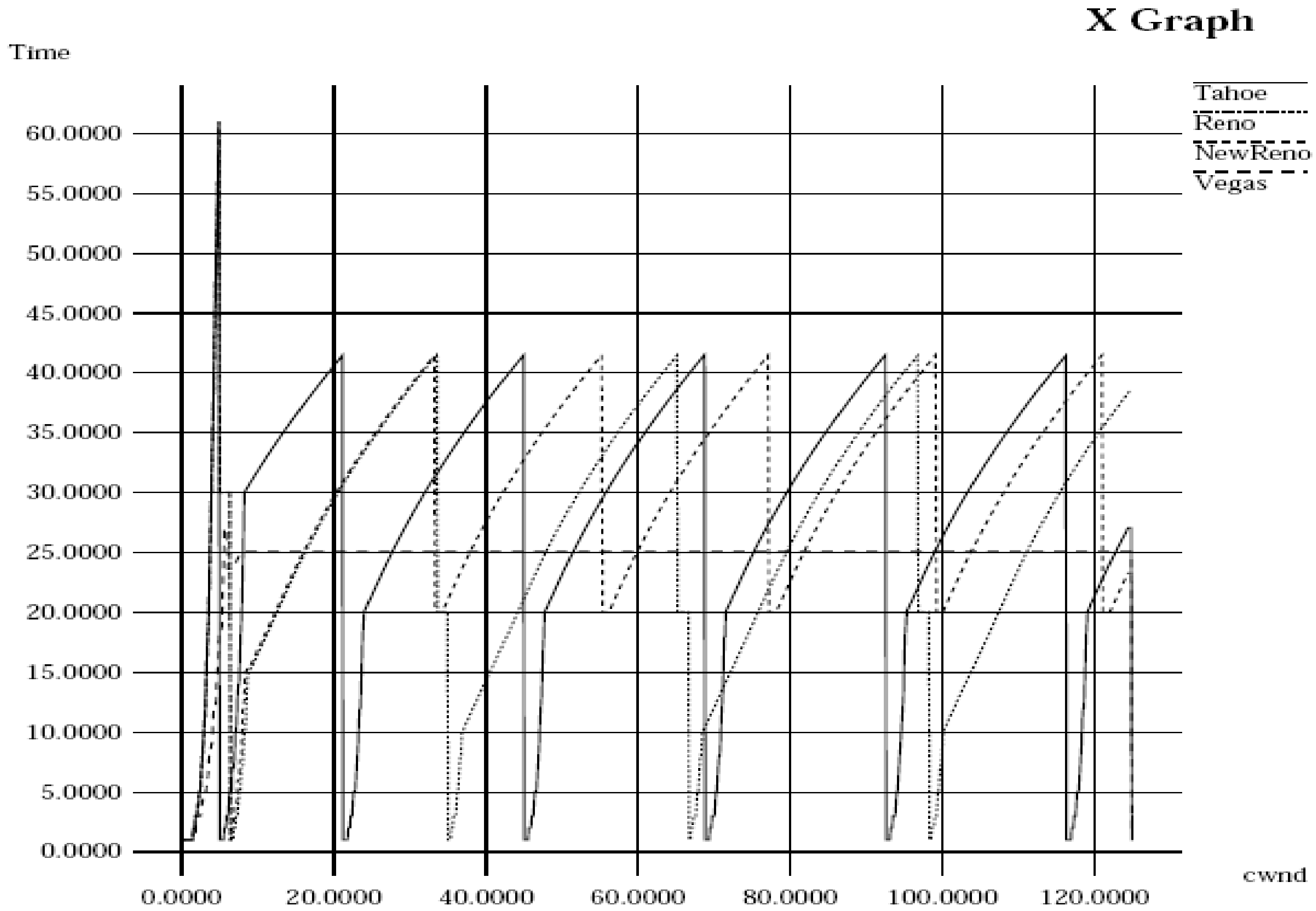
Adding new interfaces

Snapshot of cross-layer proposals [13]

- **A Definition of Cross-Layer Design[13]**: Protocol design that violates the layered communication architecture of the protocol stack is cross-layer design with reference to that particular layered architechture.

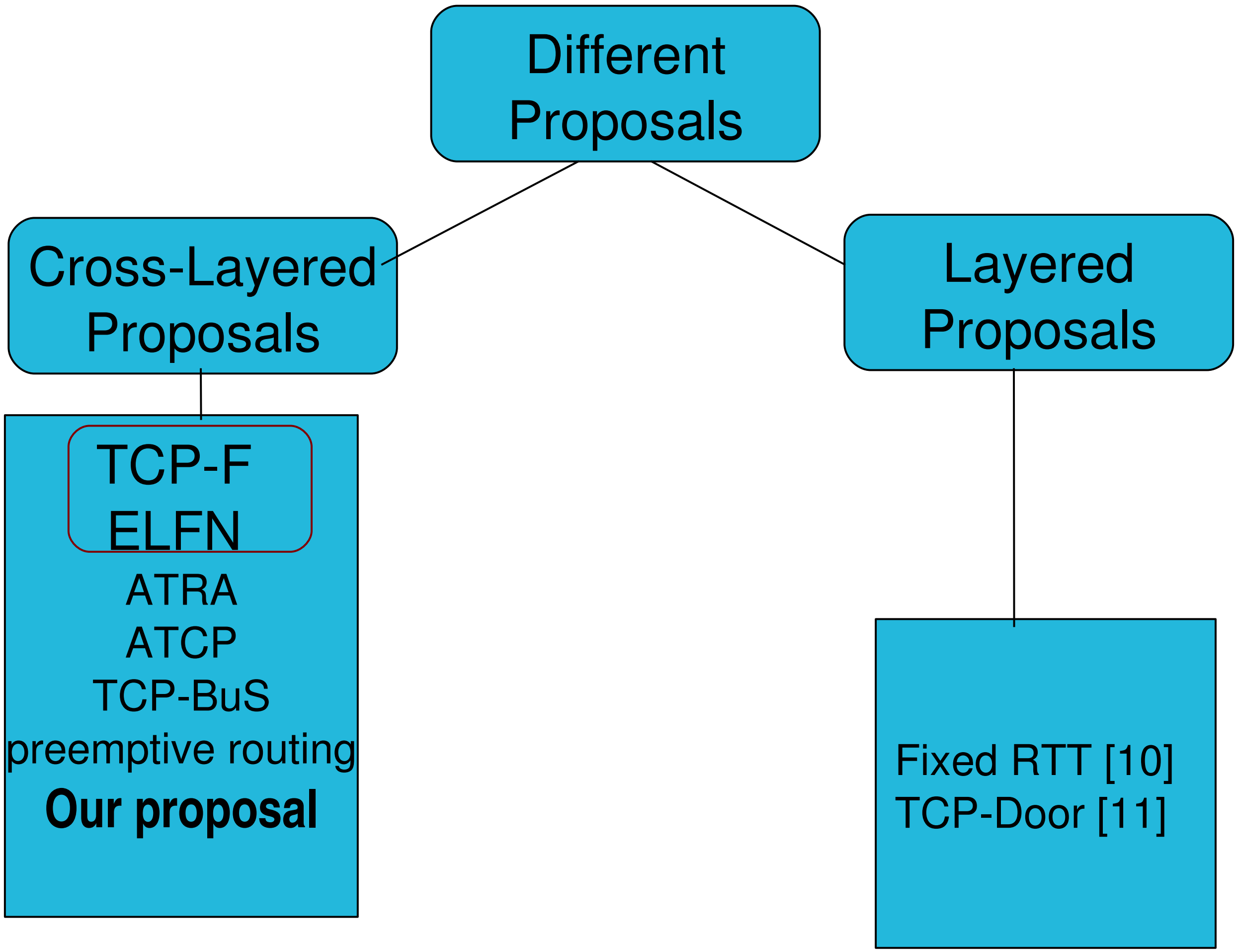
[13] V. srivastava and M. Motani, “Cross-layer design: A survey and the road ahead,” in IEEE Communications magazine, pp. 112–119, Dec 2005.

Explanation of TCP Flavors



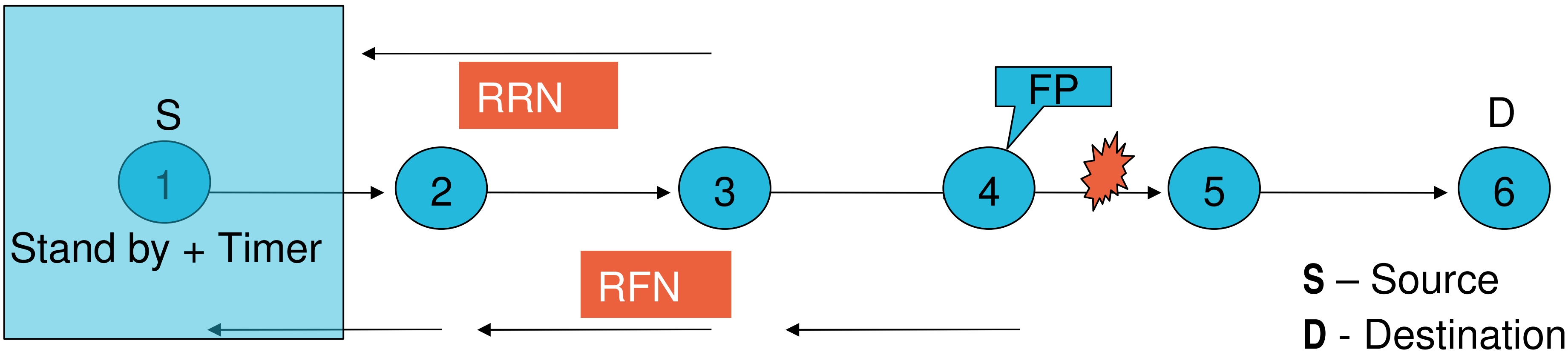
Simulation results of *cwnd* values for one FTP flow for different flavors of TCP

Classification of proposals to distinguish between losses due to link failure and congestion



-

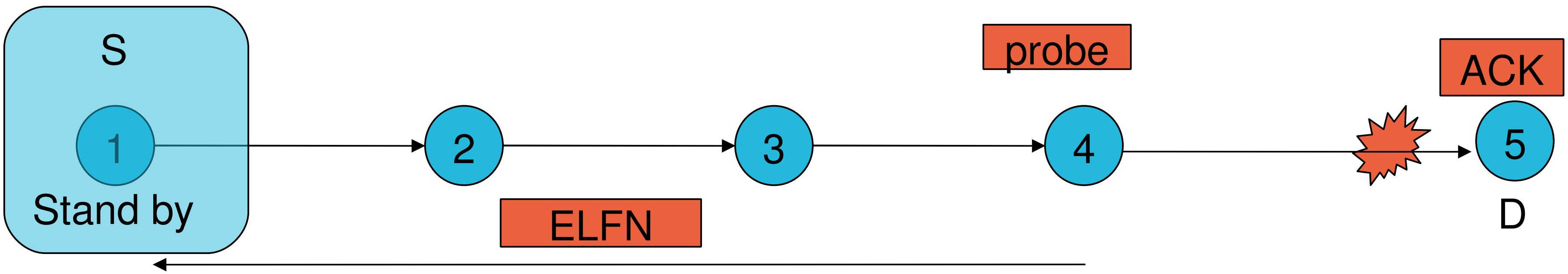
Explanation of TCP Feedback[2]



- Assumes routing protocol has a mechanism to transfer feedback.
- Assumes **worst case RRD** for handling congestion.
- Intermediate nodes are responsible for **RRN** (through routing updates)
- Emulation is done with network as a **black box** and **failure rate** & **RRD** as input parameters.

[2]. K.Chandran, et al., "A feedback-based scheme for improving TCP performance in ad hoc wireless networks,"*Personal Communications, IEEE*, Feb 2001, vol 8, Issue 1, pp.34-39.

Explanation of TCP-ELFN [3]

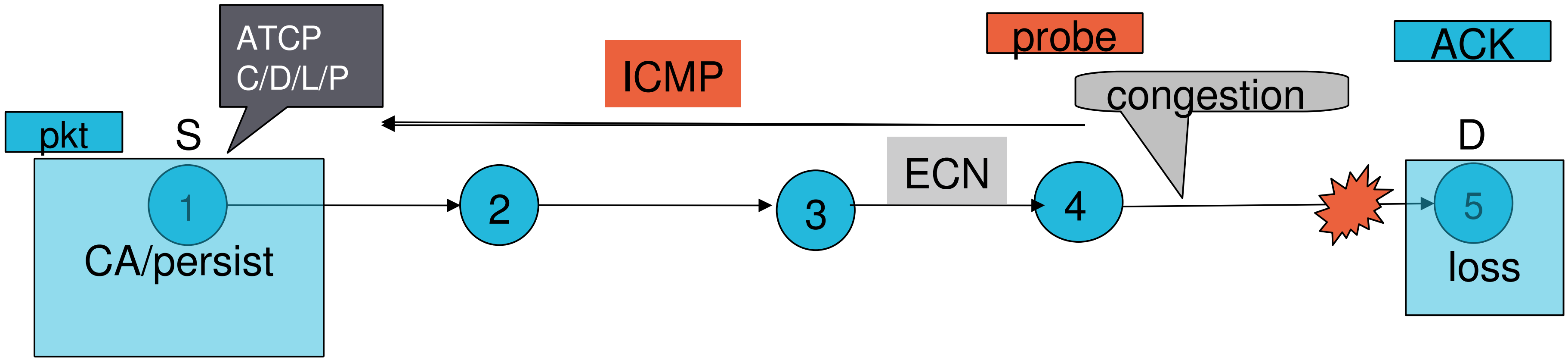


S – Source
D - Destination

- ELFN performance is worse than basic TCP in heavy load scenarios with mobility due to **aggressive probing.**

[3] G.Holland and N.Vaidya,"Analysis of TCP performance in mobile ad hoc networks,"*ACM Wireless networks*, Mar. 2002, Vol. 8, No. 2, pp.275-88.

Explanation of ATCP [7]



Event	ATCP-State	TCP-State
Congestion	Congested(C)	CA
Link failure	Normal(D)	Persist
high BER	Loss(L)	Persist
Partition	Disconnect(P)	Persist

S – Source
D - Destination

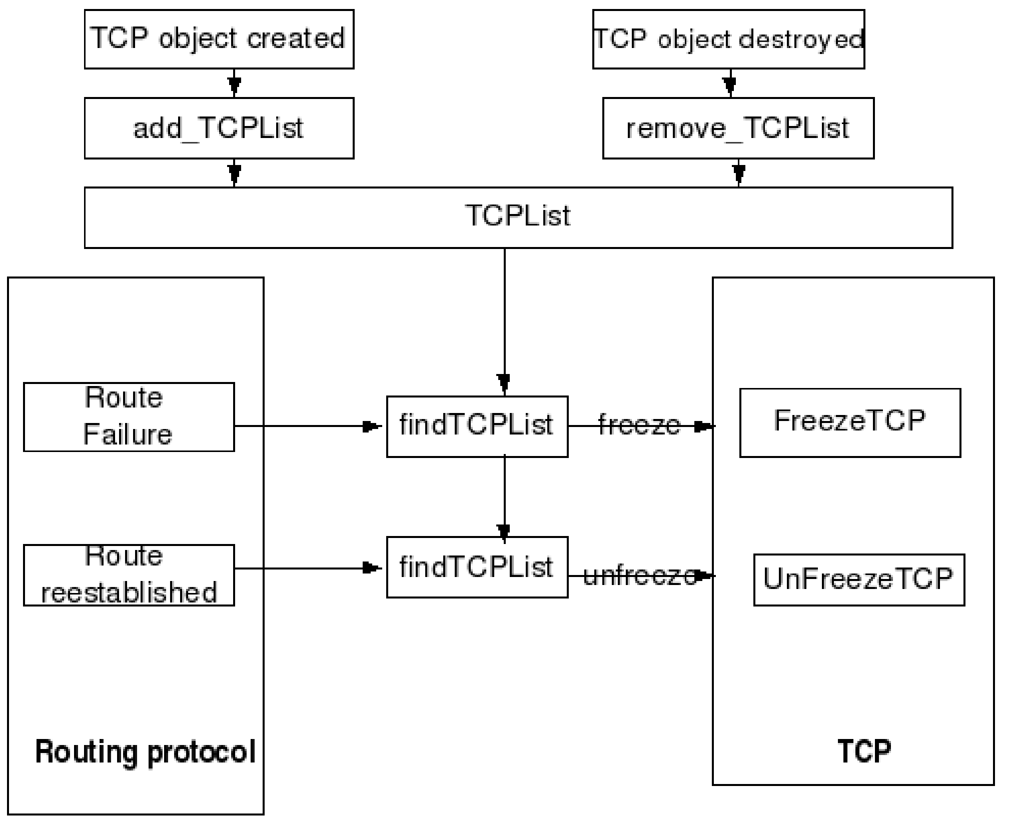
[7] . J. Liu and S. Singh, “ATCP: TCP for Mobile Ad Hoc Networks,”*IEEE JSAC*, July 2001, vol. 19, no. 7, pp. 1300-1315.

Our Approach

- Our approach is to **avoid extra signaling** messages to signal TCP that the loss is due to **route failure** and not congestion. We modify DSR so that RERR messages are used to send a **cross-layer trigger** to TCP, which **freezes** its transmission on all connections with that destination. Later, when a new route is discovered we send another cross-layer trigger to **resume transmission**.

-

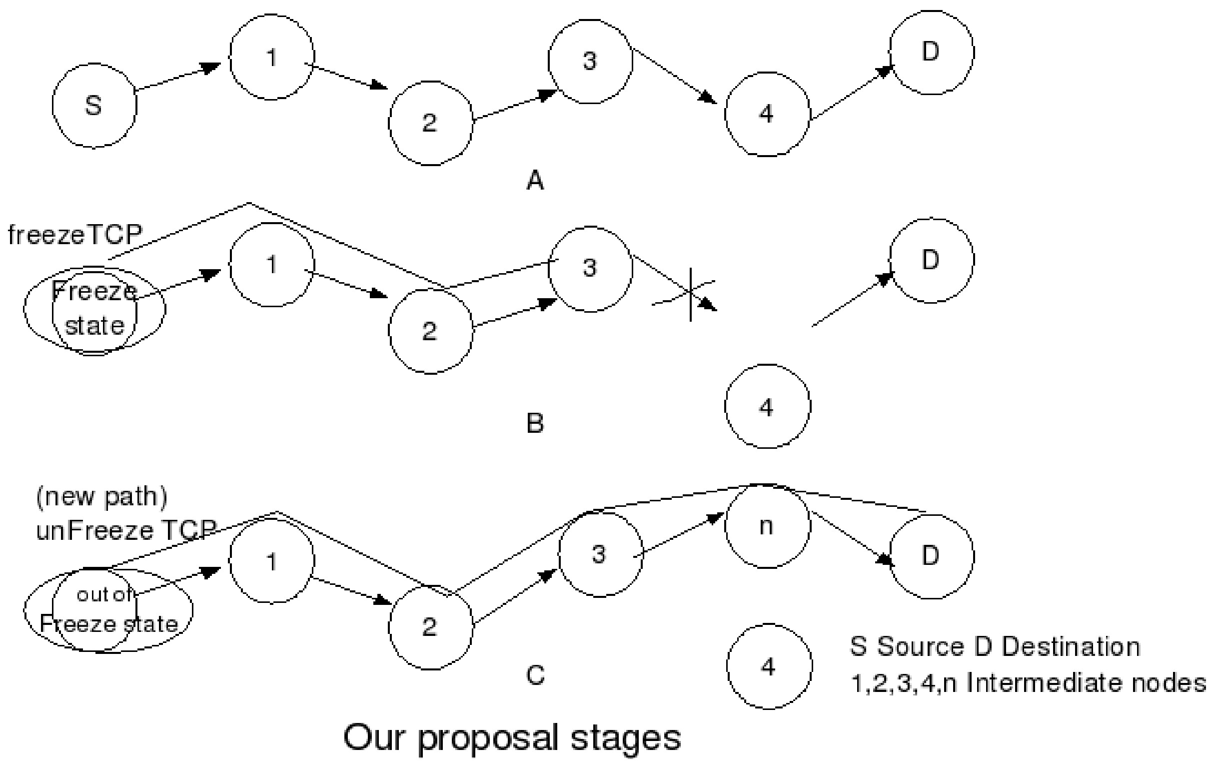
Block diagram of our approach



-

Block diagram of our approach

Explanation of our proposal



Stages in our new cross-layer proposal

-

State transition diagram of our proposal

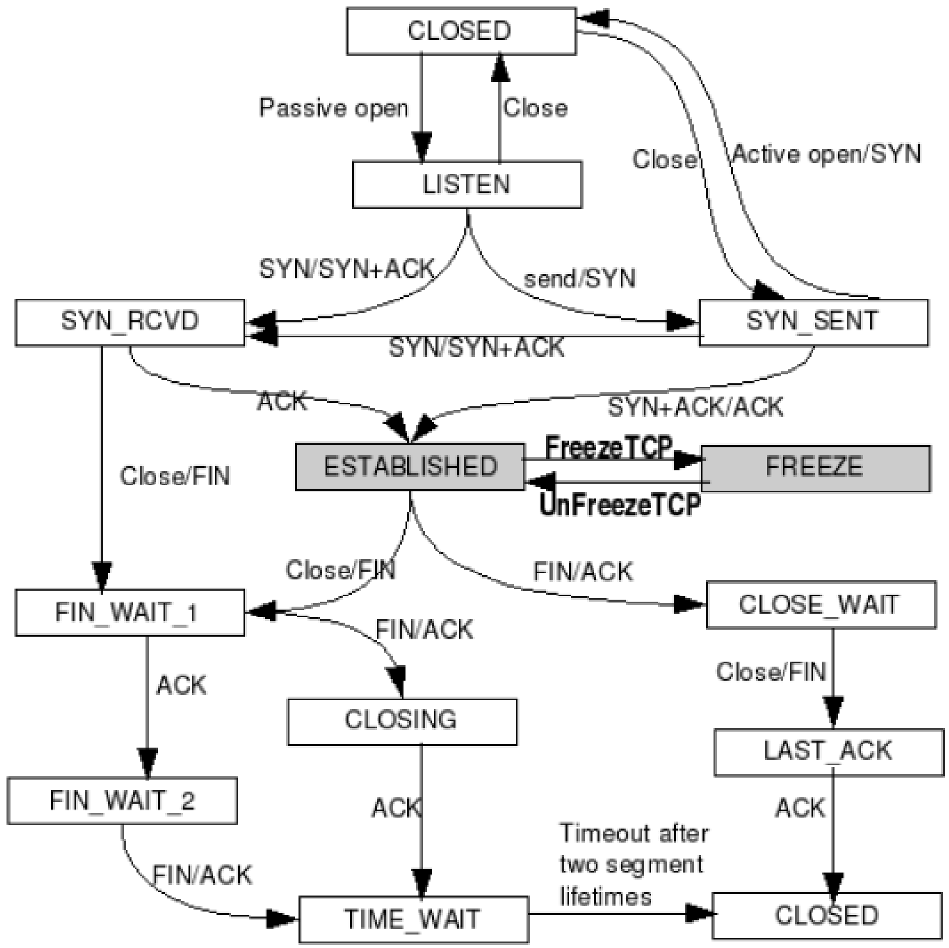


Figure 3.3: State-machine of new proposal

-

Functionality of TCP and DSR in *ns-2*

- When ever we start FTP then it calls *send(nbytes)* with argument -1, ***send(nbytes)*** is API between application and transport protocol.
- Here argument **-1** corresponds to an **infinite send**; that is, the TCP agent will act as if its send buffer is continually filled by the application.
- Which later invokes ***send_much****, which attempts to send as many packets as the* ***current sent window allows*** *and sets retransmission timer. Generally the sending TCP never actually sends data (it only sets the packet size)*
- *When ever we want to reduce congestion window then* ***slowdown(how)*** *function is called. The argument how tells how to change cwnd and ssthresh.*

-

Functionality of TCP and DSR in *ns-2*

- When a packet with source route first arrives at **recv(Packet, Handler)**, then packets destination address is checked against node’s net id and the broadcast address. If it matches with either of these addresses then it is send to handlePacketReceipt(p) for further processing. If packet is of type route reply then it invokes **acceptRouteReply(p)** function. If a packet is of type route error then it invokes **processBrokenRouteError(p)**.

-

Functions added or modified in *ns-2*

- TcpAgent::freezeTcp()
- TcpAgent::unFreezeTcp()
- TcpAgent::set_timers()
- TcpAgent::cancel_timers()
- TcpAgent::TcpAgent()
- TcpAgent::slowdown(int how)
- TcpAgent::recv(Packet*, Handler*)

- DSRAgent::processBrokenRouteError(SRPacket& p)
- DSRAgent::acceptRouteReply(SRPacket &p)
- DSRAgent::replyFromRouteCache(SRPacket &p)
- DSRAgent::handlePktWithoutSR(SRPacket& p, bool retry)

Parameters used in simulation

Topology	**1500*300**
No. of nodes	**25, 50**
Pause time	**0, 20, 50 secs**
Max. speed	**1 m/s**
No. of FTP flows	**10, 14, 20**
Simulation time	**900 secs**
TCP flavors	**Tahoe, Reno, New Reno**
Packet size	**512**

Mobility scenarios used in simulations

S.No.	Topology	Nodes	Pause time	max. speed	FTP Flows	link changes	route changes	destination unreach-able
s1	*1500*300*	25	0	1	10	236	1090	24
s2	*1500*300*	25	20	1	10	219	1826	0
s3	*1500*300*	25	0	1	14	236	1090	24
s4	*1500*300*	25	20	1	14	219	1826	0
s5	*1500*300*	25	50	1	14	195	1713	0
s6	*1500*300*	50	0	1	10	798	5279	0
s7	*1500*300*	50	20	1	10	990	4863	0
s8	*1500*300*	50	20	1	20	990	4863	0
s9	*1500*300*	50	50	1	10	788	4863	0

-

Throughput comparison for TCP-Tahoe

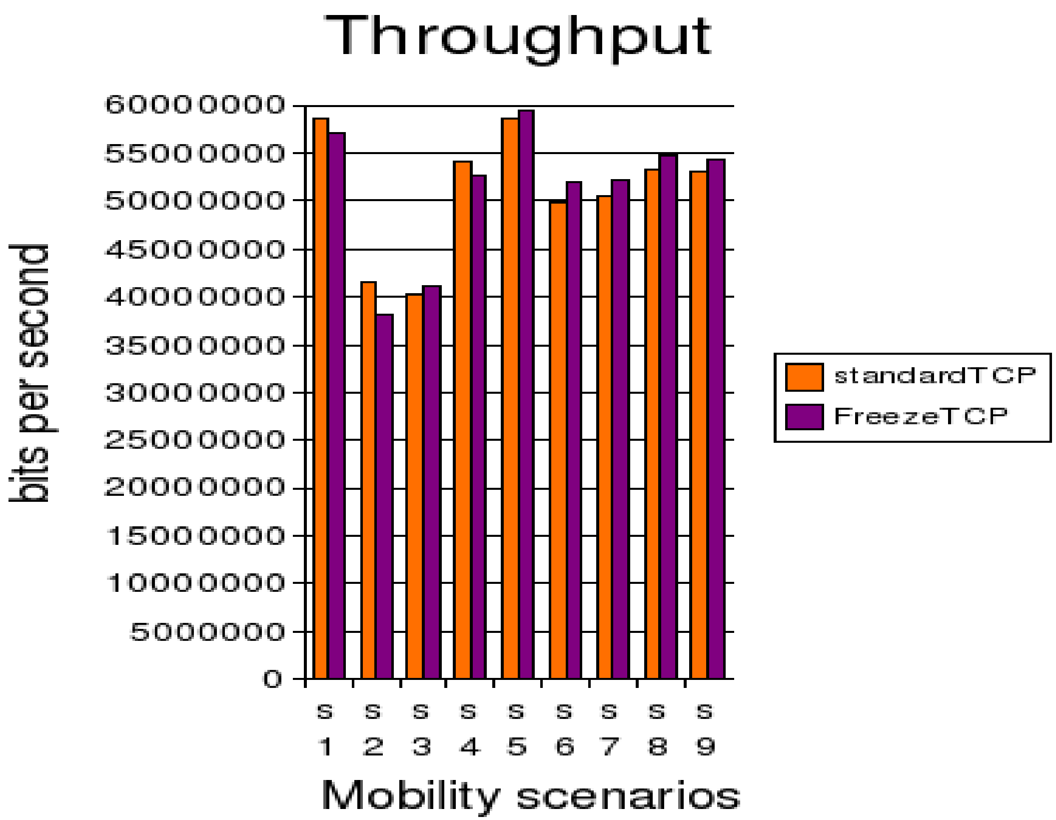
Throughput for TCP-Tahoe

Nodes(n). pause(p). flows(f)

- s1 n25-p0-f10
- s2 n25-p20-f10
- s3 n25-p0-f14
- s4 n25-p20-f14
- s5 n25-p50-f14
- s6 n50-p0-f10
- s7 n50-p20-f10
- s8 n50-p20-f20
- s9 n50-p50-f20

Topology size = 1500*1500,

Max. speed =1m/s



-

Throughput comparison for TCP-NewReno

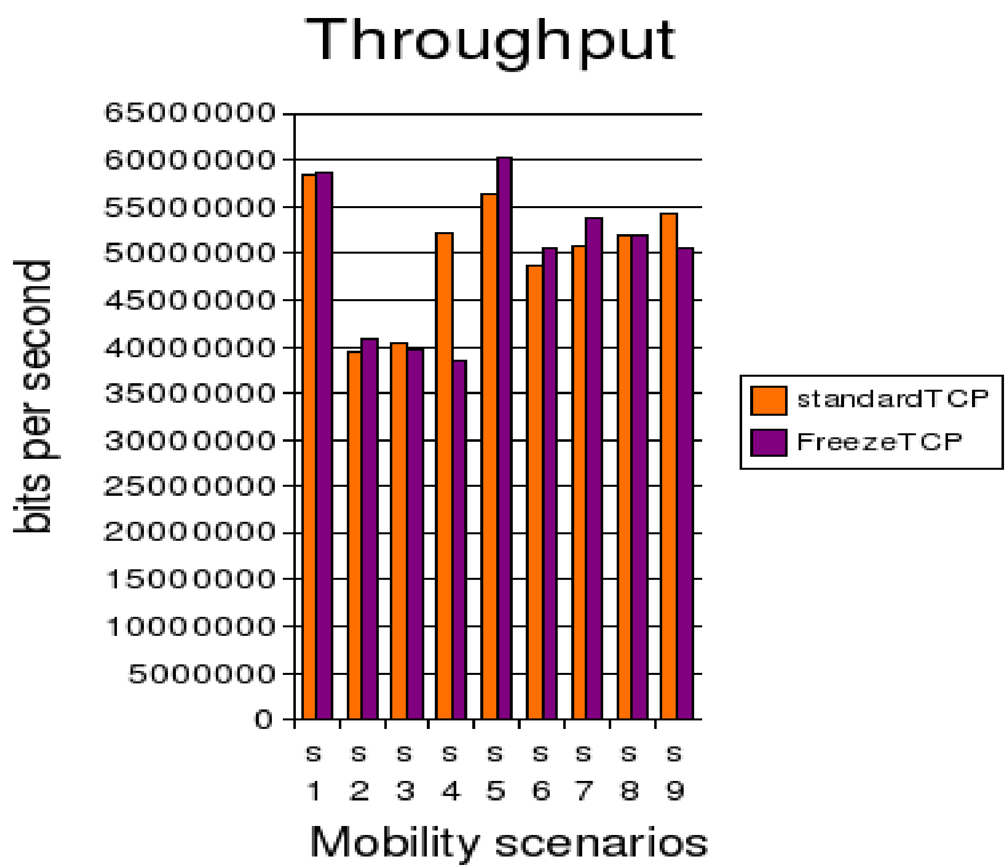
Throughput for TCP-New Reno

Nodes(n). pause(p). flows(f)

- s1 n25-p0-f10
- s2 n25-p20-f10
- s3 n25-p0-f14
- s4 n25-p20-f14
- s5 n25-p50-f14
- s6 n50-p0-f10
- s7 n50-p20-f10
- s8 n50-p20-f20
- s9 n50-p50-f20

Topology size = 1500*1500,

Max. speed =1m/s



Throughput comparison for TCP-Reno

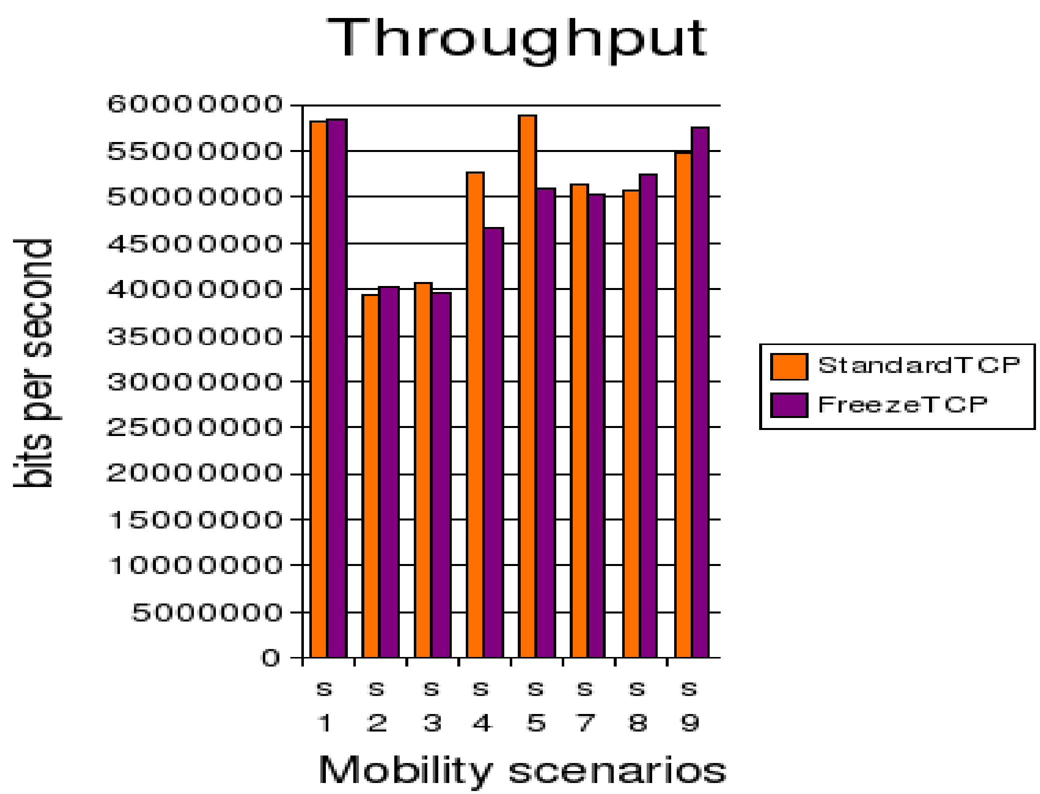
Throughput for TCP-Reno

Nodes(n). pause(p). flows(f)

- s1 n25-p0-f10
- s2 n25-p20-f10
- s3 n25-p0-f14
- s4 n25-p20-f14
- s5 n25-p50-f14
- s6 n50-p0-f10
- s7 n50-p20-f10
- s8 n50-p20-f20
- s9 n50-p50-f20

Topology size = 1500*1500,

Max. speed =1m/s



-

Concluding remarks

- We proposed a solution which appears to be a lighter loading solution as compared to other approaches since it does not add new probe packets into the network.
- We made simulations using standard protocols and by using our proposal for various mobility patterns and captured *cwnd* values for each flow.
- We repeated simulations for three TCP flavors namely Tahoe, Reno and NewReno.
- In most of the cases our proposal gives better result than that of standard proposal.

Paper submitted and simulation study

- Submitted a paper titled "*Comparison of goodput using different flavors of TCP for different transmission rates*" to M.V. Chauhan students paper contest, IEEE Indian council, August, 2006.
- Made simulation study on Throughput comparison with and with-out mobility for various TCP flavors using different mobility patterns and FTP flows.

References

[1] E. A. Ahmad Al Hanbali and P. Nain, “A survey of tcp over ad hoc networks,” in IEEE communications surveys, third quarter, vol. 7, pp. 22 – 36, 2005.

[2] S. V. Kaetik chandran, sudharshan Raghunathan and R. Prakash, “A feedback-based scheme for improving tcp performance in ad hoc wireless networks,” in IEEE Personal Communications, vol. 8 of 1, pp. 34 – 39, Feb 2001.

[3] G.Holland and N.Vaidya, “Analysis of tcp performance in mobile ad hoc networks,” in ACM Wireless networks, vol. 8, pp. 275 – 288, Mar 2002.

[4] V. Anantharaman and R. Sivakumar, “Tcp performance over mobile ad hoc networks: A quantitative study,” in J. wireless commun. and mobile computing, vol. 4, pp. 203 – 222, Mar 2004.

[5] K. X. et al., “Tcp unfairness in ad hoc wireless networks and a neighborhood red solution,” in Wireless Networks, pp. 383 – 399, Mar 2005.

References (contd.)

[6] D. A. M. David B. Johnson and J. Broch, “Dsr the dynamic source routing protocol for multihop wireless ad hoc networks,” in Ad Hoc Networking, Chapter 5, no. 139 - 172, 2001.

[7] J. Liu and S. Singh, “Atcp: Tcp for mobile ad hoc networks,” in IEEE JSAC, vol. 19, pp. 1300 – 1315, July 2001.

[8] S. F. M. Mathis, J. Mahdavi and A. Romanow, TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms. Request for Comments: 2001, 1999.

[9] M. Allman, V. Paxson, and W. Stevens, TCP Congestion Control. Request for Comments: 2581, 1999.

[10] S. Floyd, T. Henderson, and A. Gurtov, The NewReno Modification to TCP’s Fast Recovery. Request for Comments: 3782, 1999.

[11] S. F. M. Mathis, J. Mahdavi and A. Romanow, TCP Selective Acknowledgment Options. Request for Comments: 2018, 1996.

References (contd.)

[12] J. Singh and B. Soh, “Tcp new vegas: Improving the performance of tcp vegas over high latency links,” IEEE International Symposium on Network Computing and Applications, Sep 2005.

[13] V. srivastava and M. Motani, “Cross-layer design: A survey and the road ahead,” in IEEE Communications magazine, pp. 112–119, Dec 2005.

[14] S. Corson and J. Macker, Mobile Ad hoc Networking (MANET):Routing Protocol Performance Issues and Evaluation Considerations. Network Working Group, Request for Comments: 2501, Category: Informational, Jan 1999.

[15] G. Holland and N. Vaidya, “Analysis of tcp performance in mobile ad hoc networks part ii: Simulation detils and results,” tech. rep., Texas A and M University, Feb 1999.

[16] J. Monks, P. Sinha, and V. Bharghavan, “Limitations of tcp-elfn for ad hoc networks,” 2000. J. P. Monks, P. Sinha and V. Bharghavan, Limitations of TCP-ELFN for Ad Hoc Networks, MOMUC 2000.

Thank you

-