CS 5229
ADVANCED COMPUTER NETWORKS

Dr. Chan Mun Choon
School of Computing
National University of Singapore

TCP Congestion Control

TCP uses four different mechanisms:

1. Congestion Avoidance : behaviour with mid congestion
2. Slow Start: behaviour after serious congestion
3. Fast Retransmit
4. Fast Recovery

⌘ Since TCP's congestion control is window-based, all these mechanisms affect the size of the congestion window

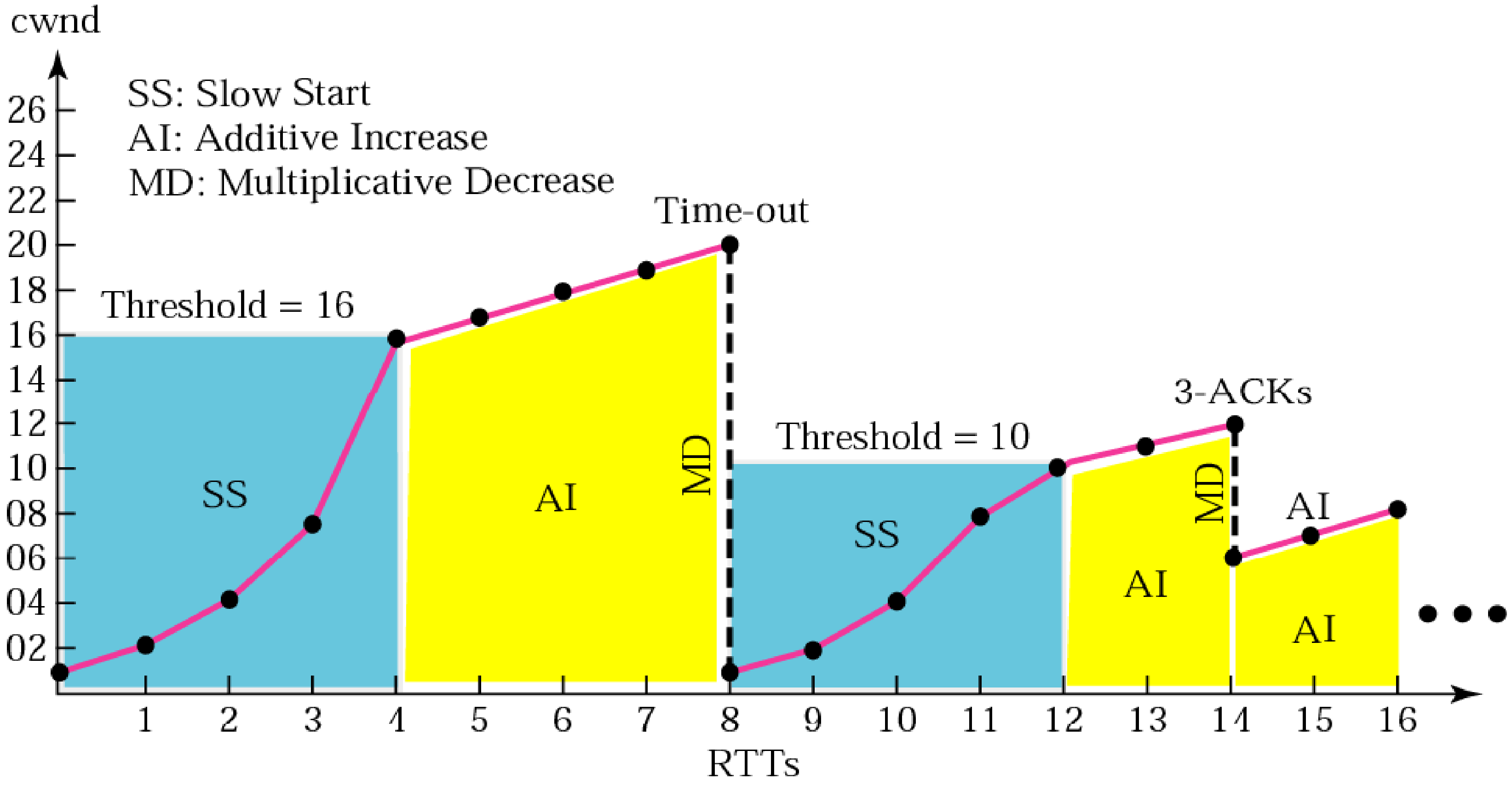
J Padhye, V Firoiu, D Towsley, J Kurose, "Modeling TCP Throughput: A Simple Model and its Empirical Validation," SIGCOMM 1998.

Overview

- Study effect of TCP's loss detection mechanism on throughput
  - Triple duplicate
  - Timeout
- Model can predict throughput over a significantly wider range of loss
- Takes into account
  - Dependency of congestion avoidance on ACK
  - How loss is inferred
  - Limited receiver window size
  - RTT

Assumptions

- A flow with unlimited amount of data to send
- Based on TCP Reno



Model/Assumption

- Window increase by 1/w each time an ACK is received
- Decrease depends on mode of detection
- Model in “rounds”
  - Starts with back-to-back transmission of W packets
  - No packet send until ACK is received
  - Time to send W packet is less than RTT
  - b packets acknowledged per ACK
  - Slope is 1/b per RTT

Loss Modeling

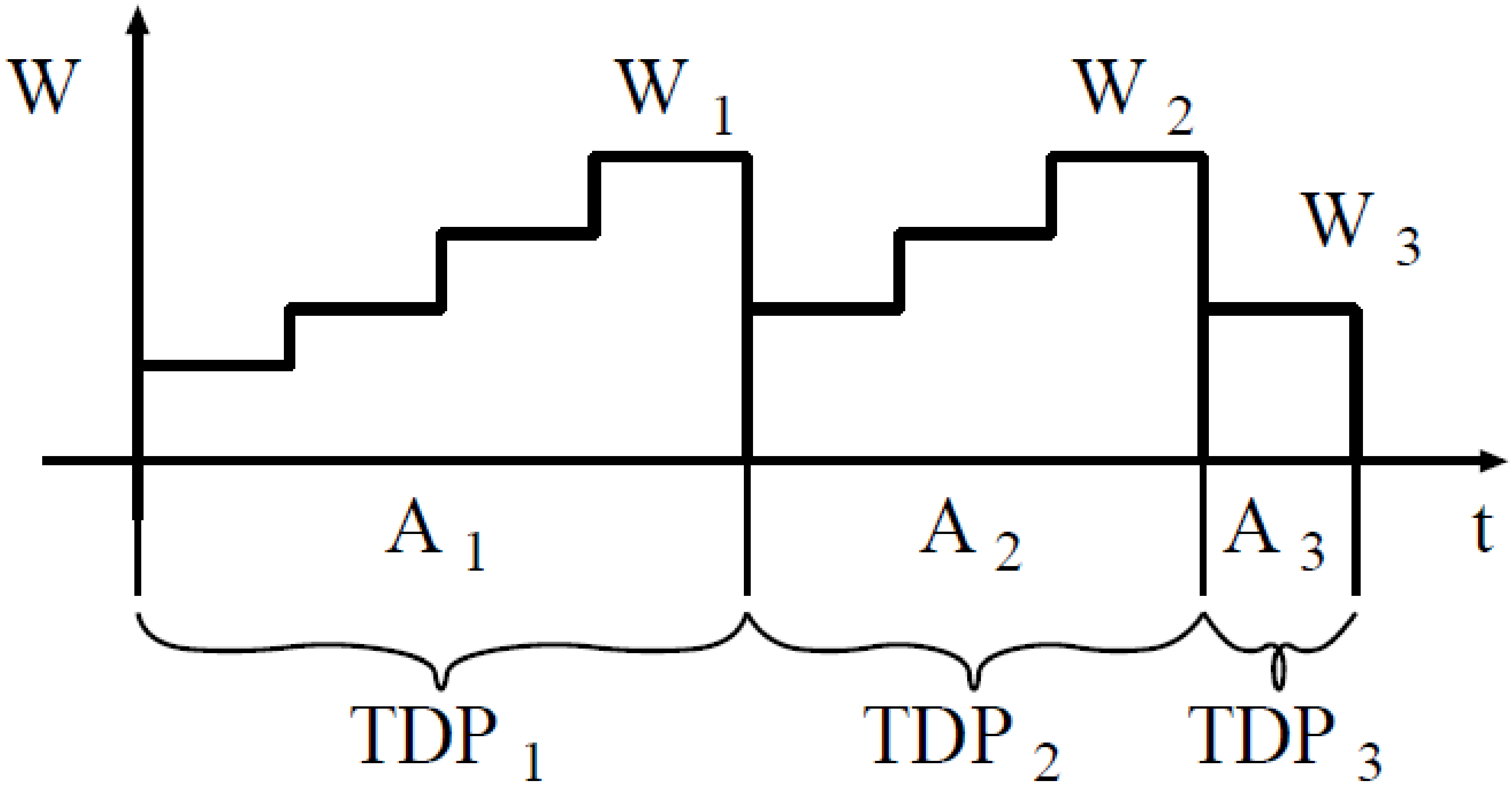
- Loss independent between rounds
  - If a packet is lost, all remaining packets in the round are lost

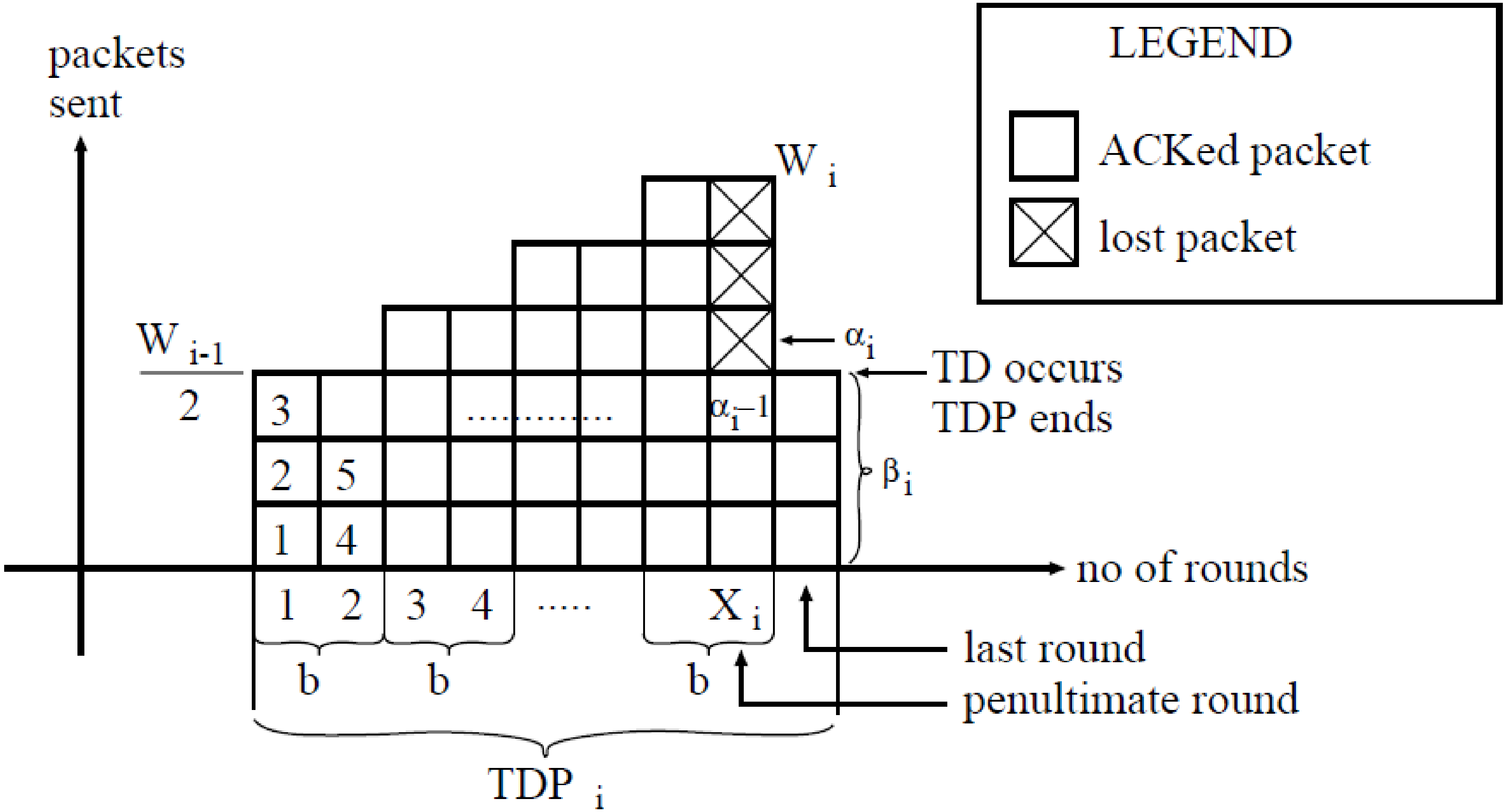
1. Losses are only detected by triple duplicate ACK (TD)
2. Losses are TD and Timeout (TO)
3. Window limited

- Does not model fast recovery
- Throughput is measured in packets per second

Only TD

- p: probability that a packet is lost
  - First packet in round to be lost
- B: throughput (not goodput)
- Y: number of packets send in a period
- A: duration of one TD period (TDP)


W
W 1
W 2
W 3
A 1
A 2
A 3
t
TDP 1
TDP 2
TDP 3



$$B = \frac{E[Y]}{E[A]}$$

$$E[Y] = E[\alpha] + E[W] - 1$$

$$E[A] = (E[X] + 1)E[r]$$

$$E[Y] = \boxed{E[\alpha]} + E[W] - 1$$

<u>Loss is iid</u>

$$P[\alpha = k] = (1-p)^{k-1}p, \quad k = 1, 2, \ldots$$

$$E[\alpha] = \sum_{k=1}^{\infty}(1-p)^{k-1}pk = \frac{1}{p}$$

$$E[Y] = E[\alpha] + E[W] - 1$$

$$\boxed{E[Y]} = \frac{1-p}{p} + \boxed{E[W]} \qquad (5)$$

Another way to derive E[Y]

$$W_i = \frac{W_{i-1}}{2} + \frac{X_i}{b}, \quad i = 1, 2, \ldots$$

$$\begin{aligned} Y_i &= \sum_{k=0}^{X_i/b-1} \left(\frac{W_{i-1}}{2} + k\right)b + \beta_i \\ &= \frac{X_i W_{i-1}}{2} + \frac{X_i}{2}\left(\frac{X_i}{b} - 1\right) + \beta_i \\ &= \frac{X_i}{2}\left(\frac{W_{i-1}}{2} + W_i - 1\right) + \beta_i \quad \text{using (7)} \end{aligned}$$

Assume iid

$$E[W] = \frac{2}{b}\boxed{E[X]} \qquad \boxed{E[\beta]} = E[W]/2.$$

$$E[Y] = \frac{E[X]}{2}\left(\frac{E[W]}{2} + E[W] - 1\right) + E[\beta]$$

The following can now be derived:

$$E[W] = \sqrt{\frac{8}{3bp}} + o(1/\sqrt{p})$$

Recall that $$E[W] = \frac{2}{b}E[X]$$

$$E[A] = (E[X]+1)E[r]$$

$$\begin{aligned} E[X] &= \frac{2+b}{6} + \sqrt{\frac{2b(1-p)}{3p} + \left(\frac{2+b}{6}\right)^2} \\ E[A] &= RTT\left(\frac{2+b}{6} + \sqrt{\frac{2b(1-p)}{3p} + \left(\frac{2+b}{6}\right)^2} + 1\right) \end{aligned}$$

Observe that,

$$E[X] = \sqrt{\frac{2b}{3p}} + o(1/\sqrt{p})$$

Recall that

$$B = \frac{E[Y]}{E[A]}$$

Put the results together:

$$B(p) = \frac{\frac{1-p}{p} + E[W]}{E[A]}$$

$$= \frac{\frac{1-p}{p} + \frac{2+b}{3b} + \sqrt{\frac{8(1-p)}{3bp} + \left(\frac{2+b}{3b}\right)^2}}{RTT\left(\frac{2+b}{6} + \sqrt{\frac{2b(1-p)}{3p} + (\frac{2+b}{6})^2} + 1\right)}$$

Which can be expressed as:

$$B(p) = \frac{1}{RTT}\sqrt{\frac{3}{2bp}} + o(1/\sqrt{p})$$

A "Simpler" Derivation

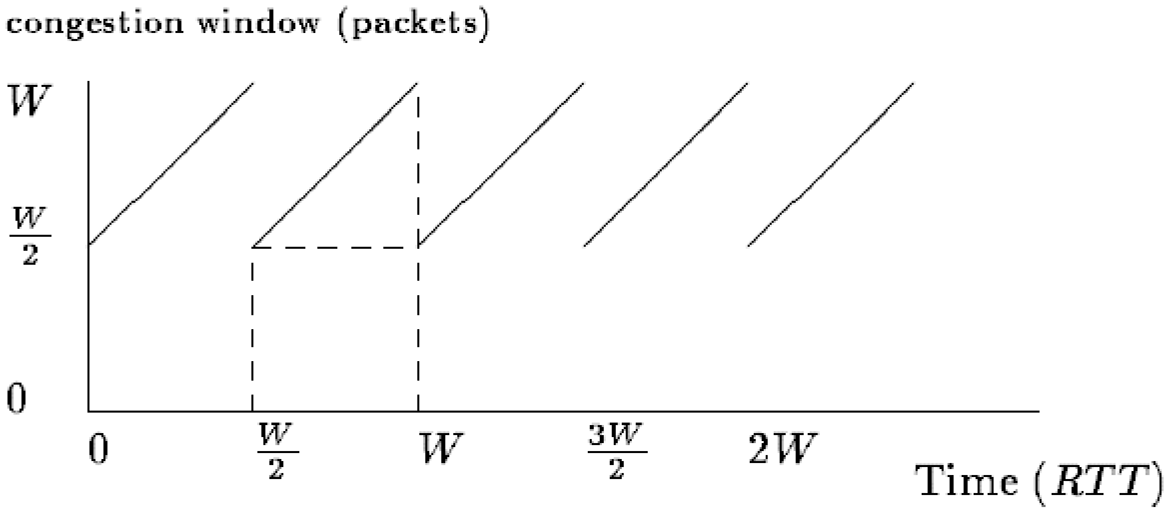
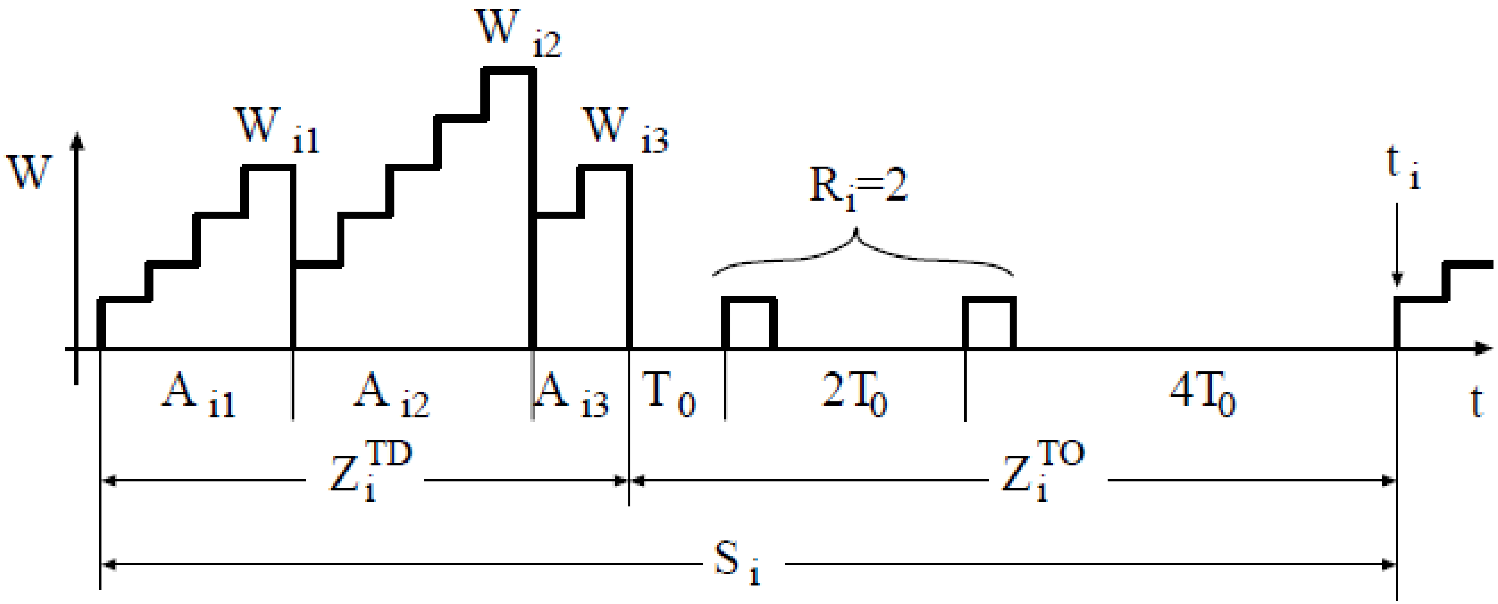


Figure 1: TCP window evolution under periodic loss
Each cycle delivers $(\frac{W}{2})^2 + \frac{1}{2}(\frac{W}{2})^2 = 1/p$ packets and takes $W/2$ round trip times.

M. Mathis, J. Semske, J. Mahdavi, and T. Ott. The macroscopic behavior of the TCP congestion avoidance algorithm. *Computer Communication Review, 27(3), July 1997.*

Case with Timeout

- Main difference: timeout occurs if when packets (or ACK) are lost, less than 3 duplicate ACKs are received
- In the model, consider the cases when 3 or less (duplicate) ACKs are received

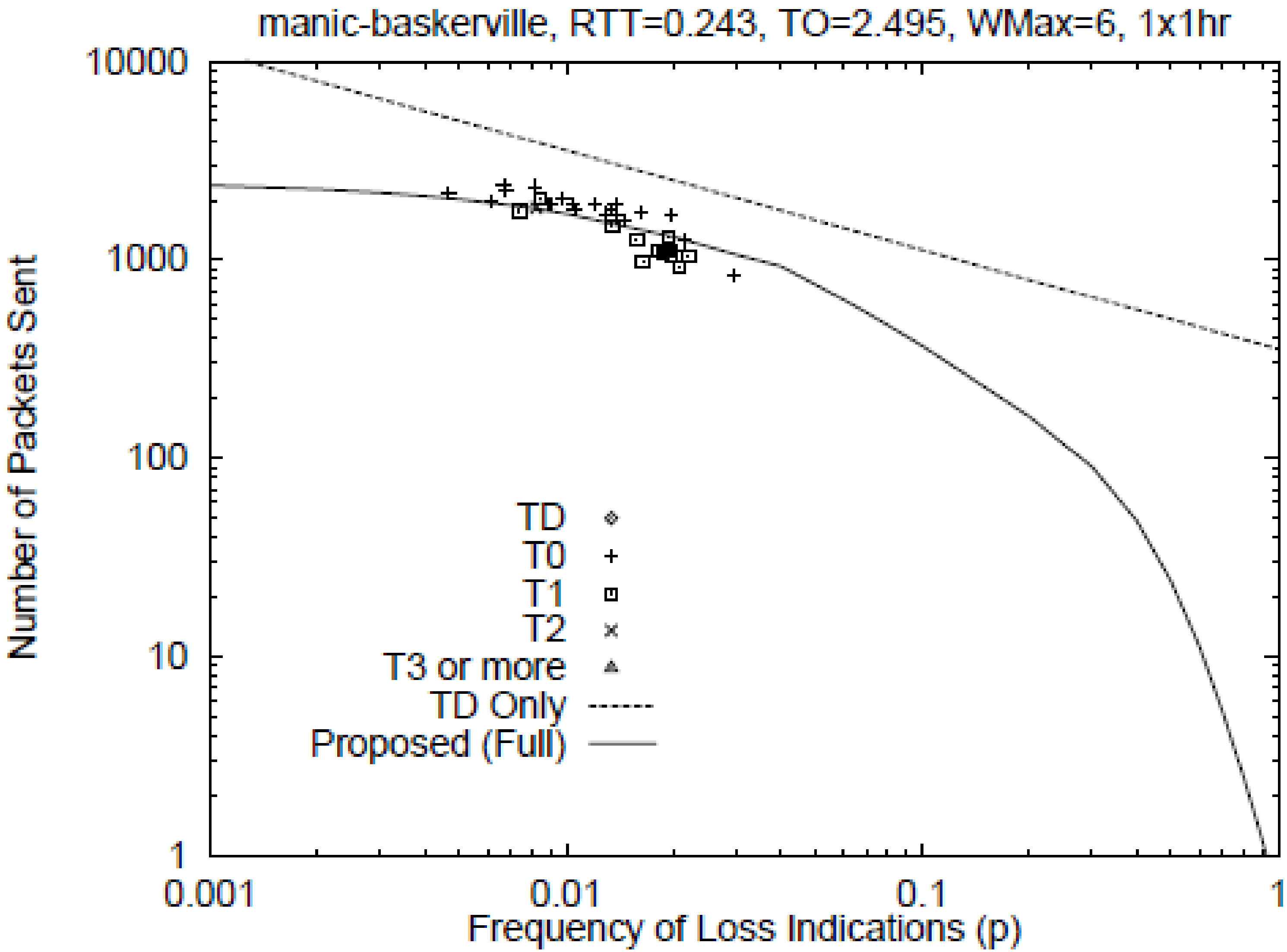

W
W i1
W i2
W i3
R i=2
t i
A i1
A i2
A i3
T 0
2T0
4T0
t
Z i TD
Z i TO
S i

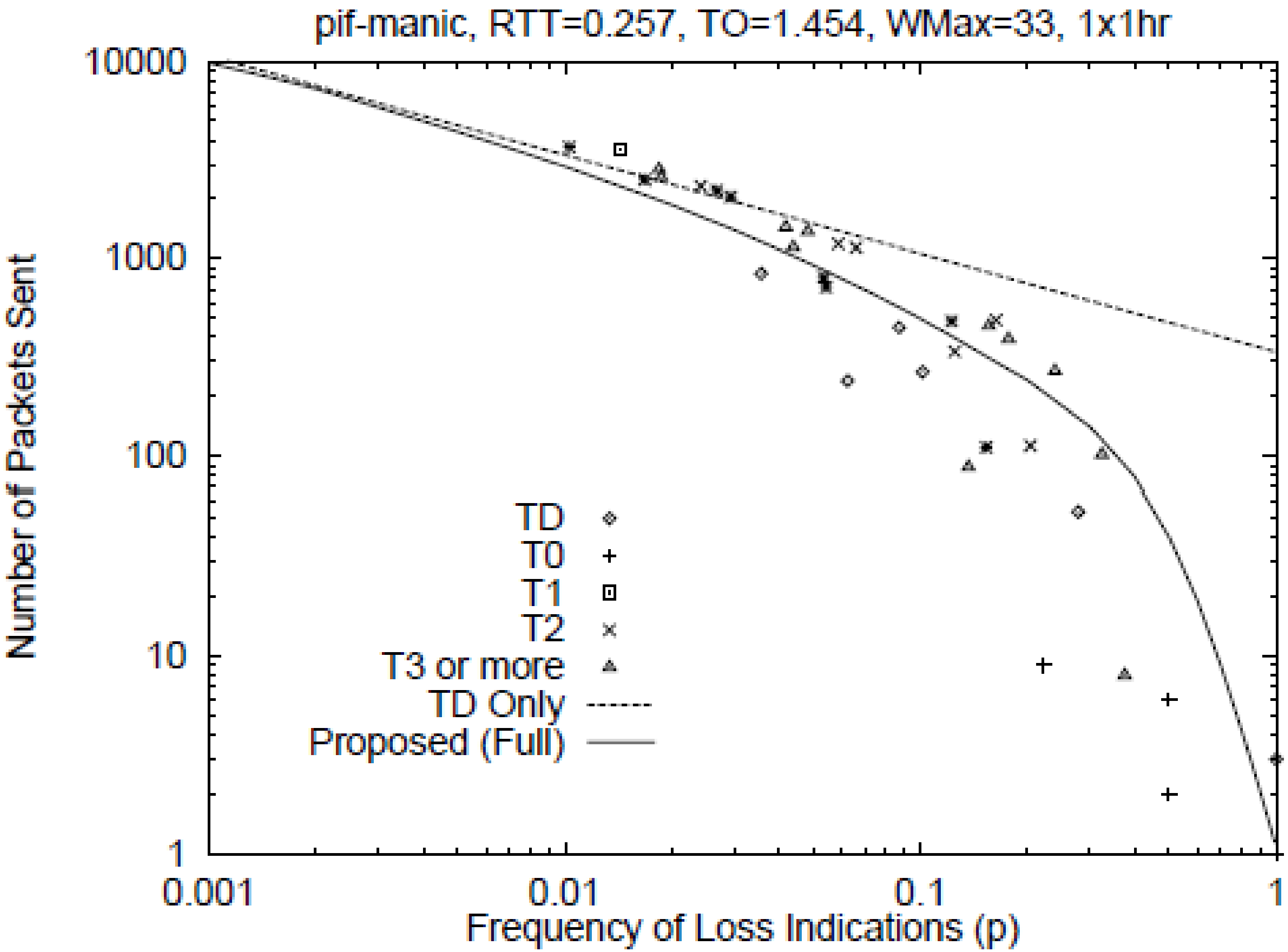
With TD and TO

$$B(p) \approx \frac{1}{\underbrace{RTT\sqrt{\frac{2bp}{3}}}_{\text{small p}} + \underbrace{T_0 \min\left(1, 3\sqrt{\frac{3bp}{8}}\right) p(1+32p^2)}_{\text{larger p}}}$$

Full Model (TD, TO, window limited)

$$B(p) \approx \min\left(\frac{W_{max}}{RTT}, \frac{1}{RTT\sqrt{\frac{2bp}{3}} + T_0 \min\left(1, 3\sqrt{\frac{3bp}{8}}\right) p(1+32p^2)}\right)$$


manic-baskerville, RTT=0.243, TO=2.495, WMax=6, 1x1hr
Number of Packets Sent
10000
1000
100
10
1
0.001
0.01
0.1
1
Frequency of Loss Indications (p)
TD
T0
T1
T2
T3 or more
TD Only
Proposed (Full)


pif-manic, RTT=0.257, TO=1.454, WMax=33, 1x1hr
Number of Packets Sent
10000
1000
100
10
1
0.001
0.01
0.1
1
Frequency of Loss Indications (p)
TD
T0
T1
T2
T3 or more
TD Only
Proposed (Full)

Sender	Receiver	Packets Sent	Loss Indic.	TD	T_0	T_1	T_2	T_3	T_4	T_5 or more	RTT	Time Out
manic	alps	54402	722	19	611	67	15	6	2	2	0.207	2.505
manic	baskerville	58120	735	306	411	17	1	0	0	0	0.243	2.495
manic	ganef	58924	743	272	444	22	4	1	0	0	0.226	2.405
manic	mafalda	56283	494	2	474	17	1	0	0	0	0.233	2.146
manic	maria	68752	649	1	604	35	8	1	0	0	0.180	2.416
manic	spiff	117992	784	47	702	34	1	0	0	0	0.211	2.274
manic	sutton	81123	1638	988	597	41	7	3	1	1	0.204	2.459
manic	tove	7938	264	1	190	37	18	8	3	7	0.275	3.597
void	alps	37137	838	7	588	164	56	17	4	2	0.162	0.489
void	baskerville	32042	853	339	430	67	12	5	0	0	0.482	1.094
void	ganef	60770	1112	414	582	79	20	9	4	2	0.254	0.637
void	maria	93005	1651	33	1344	197	54	15	5	3	0.152	0.417
void	spiff	65536	671	72	539	56	4	0	0	0	0.415	0.749
void	sutton	78246	1928	840	863	152	45	18	9	1	0.211	0.601
void	tove	8265	856	5	444	209	100	51	27	12	0.272	1.356
babel	alps	13460	1466	0	1068	247	87	33	18	8	0.194	1.359
babel	baskerville	62237	1753	197	1467	76	10	3	0	0	0.253	0.429
babel	ganef	86675	2125	398	1686	38	2	1	0	0	0.201	0.306

Motivation for Highspeed TCP

- Consider transmitting over a link with 1Gbps and RTT 200ms
- Using default maximum window size of 64KB, what is the maximum throughput?
  - Too slow, solution: use window scaling
- However, problem still exists
  - Starting from window of say 64KB, how long does it take TCP (Reno), based on AIMD to fully utilize the 1Gbps without loss (MTU=1500B)?

Options

- MIMD?
- Loss Based
  - Cubic TCP (Linux-based OS)
- Rate Based
  - Compound TCP (Window Vista and later)

- KTJ Song, Q Zhang, M Sridharan, “Compound TCP: A Scalable and TCP-Friendly Congestion Control for High-speed Networks,” PFLDNet 2006.
- Rhee, I., and Xu, L. “CUBIC: A new TCP-friendly high-speed TCP variant,” PFLDNet 2005.

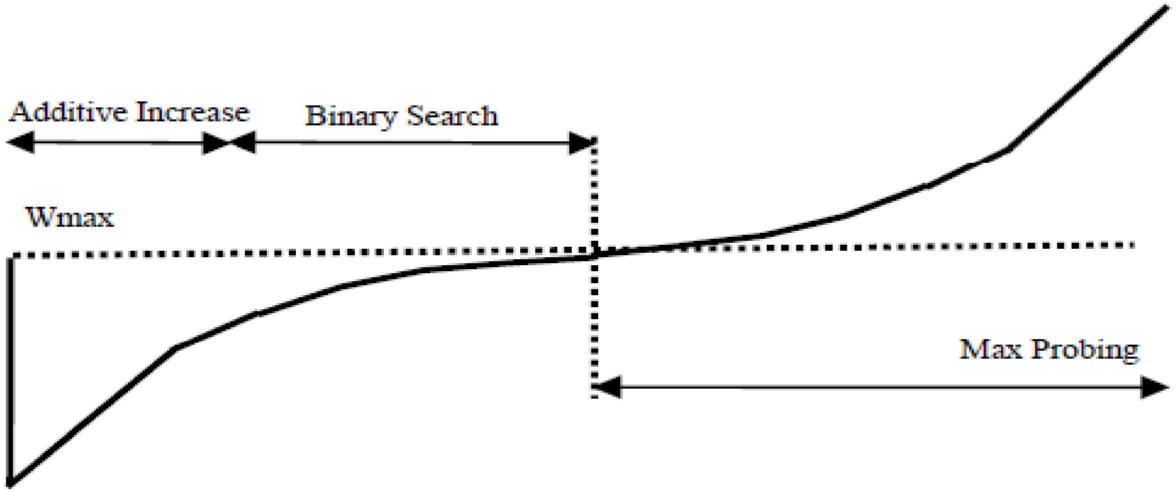


Fig. 1: The Window Growth Function of BIC

- After loss, BIC computes Wmin
- Performs binary search between Wmax and Wmin for the “right” window size
    - but window increase is capped by Smax (linear increase)
    - otherwise, do binary search (logarithmic increase)
- However, increment can be too aggressive for network with short RTT or low speed network

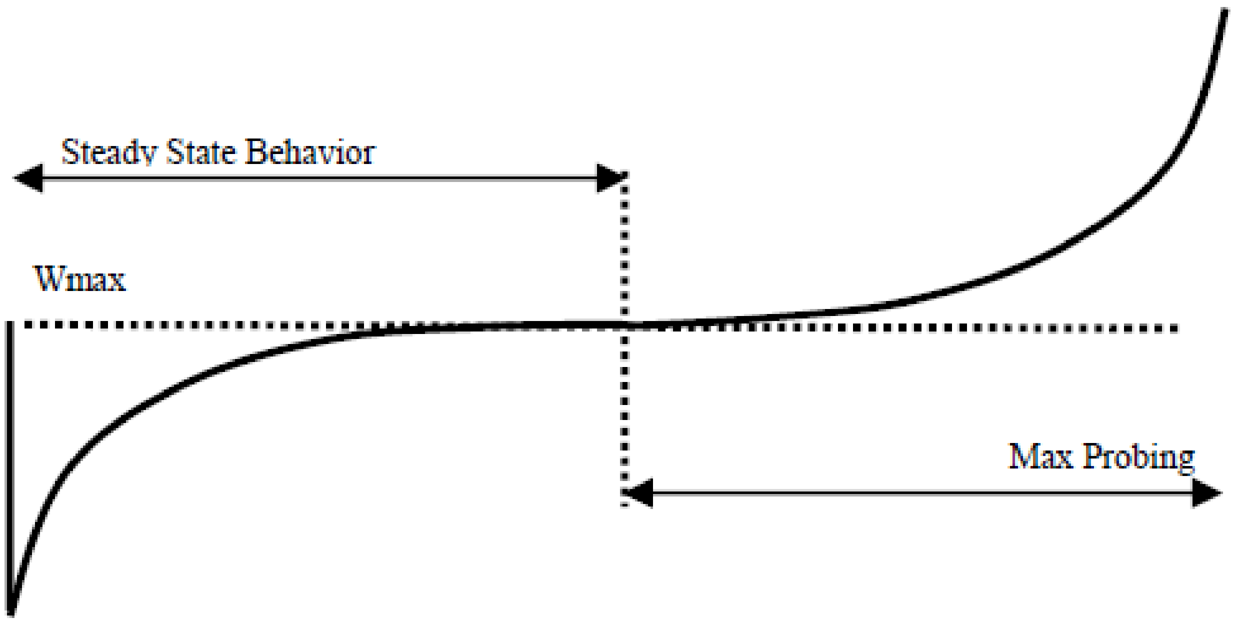


Fig. 2: The Window Growth Function of CUBIC

$$W_{cubic} = C(t-K)^3 + W_{\max}$$

Window increment is no more than Smax per second

Compound TCP

- Runs a loss based and a delay based TCP together

$$win = \min(cwnd + dwnd, awnd)\,, \qquad (1)$$

$$cwnd = cwnd + 1/win\,,$$

where win is defined with equation (1).

Compound TCP

Delay Measurements:

$$Expected = win / baseRTT$$

$$Actual = win / RTT$$

$$Diff = (Expected - Actual) \cdot baseRTT$$

Window Increase/decrease:

$$dwnd(t+1) = \begin{cases} dwnd(t) + (\alpha \cdot win(t)^k - 1)^+, \text{if } diff < \gamma \\ (dwnd(t) - \zeta \cdot diff)^+, \text{if } diff \geq \gamma \\ (win(t) \cdot (1-\beta) - cwnd/2)^+, \text{if loss is detected} \end{cases}$$

Some typical values:

- k = 0.8 or ¾, α = 1/8, β = ½