計算機ネットワーク

開講クォーター: 1-2Q

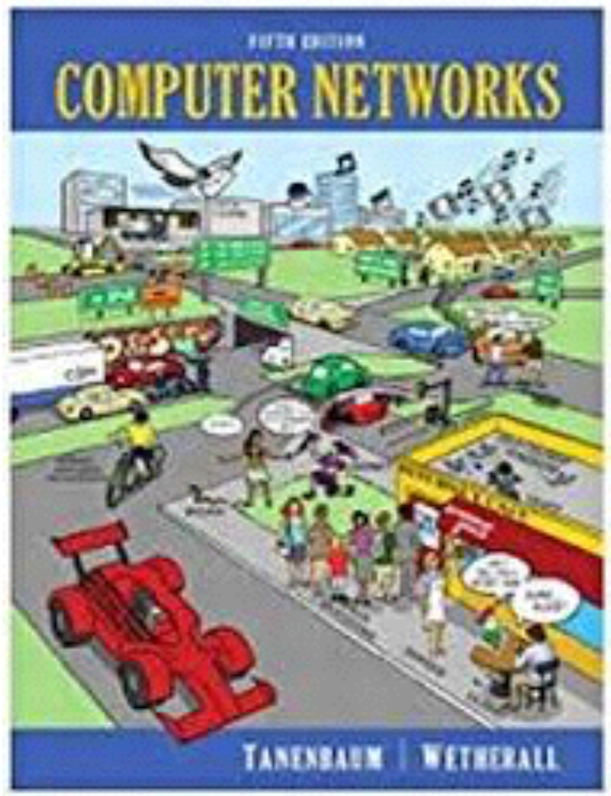
曜日・時限: 火7-8限

講義室: 1Q @ W834, 2Q @ W931

横田理央

rioyokota@gsic.titech.ac.jp

参考書



教科書

講義日程（2Q）

		授業計画		課題
06/14	第９回	ネットワーク層１ ルーティング・輻輳制御	５章	ルーティングの種類を理解し 輻輳制御手法を説明できる
06/21	第１０回	ネットワーク層２ インターネットとサービス品質	５章	インターネットの制御プロトコルを理解し ネットワーク間の接続について説明できる
06/28	第１１回	トランスポート層１ トランスポート・プロトコルの要素	６章	誤り制御とフロー制御を理解し 輻輳制御について説明できる
07/05	第１２回	トランスポート層２ UDP と TCP	６章	TCP の信頼性を理解し TCP のコネクション管理を説明できる
07/12	第１３回	アプリケーション層 DNS, 電子メール, www	７章	DNS, 電子メール, www のしくみを理解し ストリーミング，P2P について説明できる
07/26	第１４回	ネットワークセキュリティ１ 対称鍵暗号，公開鍵暗号	８章	暗号アルゴリズムを理解し SHA-1,2 と RSA について説明できる
08/02	第１５回	ネットワークセキュリティ２ デジタル署名，認証プロトコル	８章	電子メール,Web のセキュリティ の脅威について把握できる

Transmission Control Protocol (TCP)

Request for Comments (RFC): Internet standards

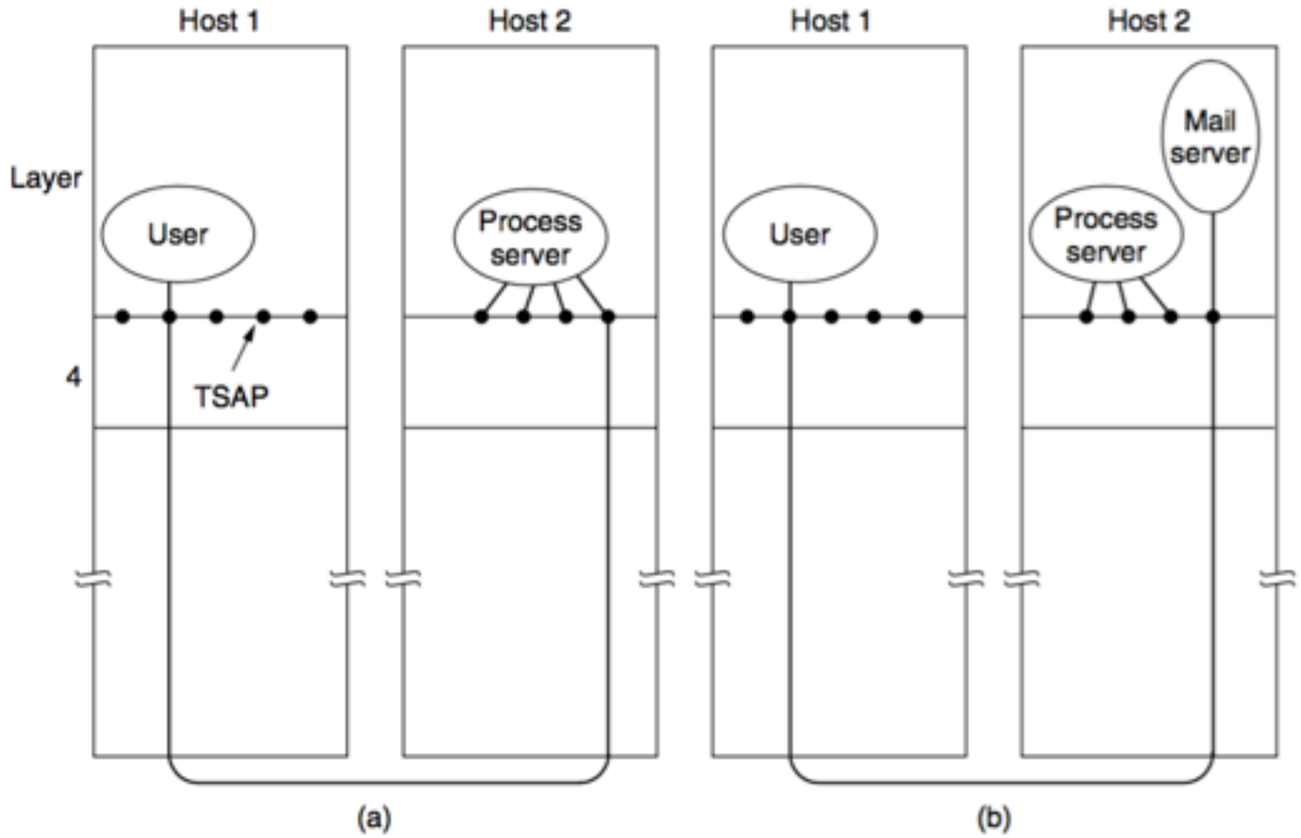
History

RFC 793: Formal definition
RFC1122: Clarification and bug fixes
RFC1323: Extensions for high performance
RFC2018: Selective acknowledgements
RFC 2581: Con-gestion control in
RFC 2873: Repurposing of header fields for quality of service
RFC 2988: Improved retransmission timers
RFC 3168: Explicit congestion notification

Ports

Port	Protocol	Use
20, 21	FTP	File transfer
22	SSH	Remote login, replacement for Telnet
25	SMTP	Email
80	HTTP	World Wide Web
110	POP-3	Remote email access
143	IMAP	Remote email access
443	HTTPS	Secure Web (HTTP over SSL/TLS)
543	RTSP	Media player control
631	IPP	Printer sharing

inetd (Internet daemon)



Transmission Control Protocol (TCP)

TCP Byte stream

There is no way for the receiver to detect the unit(s) in which the data were written.

TCP Push

To force data out, TCP has the notion of a PUSH flag that is carried on packets.
> TCP NODELAY

URGENT flag

This event causes TCP to stop accumulating data and transmit, or interrupt and read.
Its use is now discouraged because of implementation differences.

The TCP Protocol

Sequence number

32-bit

Every segment has a unique number

Segment size

Fixed 20-byte header (plus an optional part)

It can accumulate data from several writes into one segment or can split data from one write over multiple segments

TCP header, must fit in the 65,515- byte IP payload

Each link has an MTU (Maximum Transfer Unit)

Sliding Window Protocol

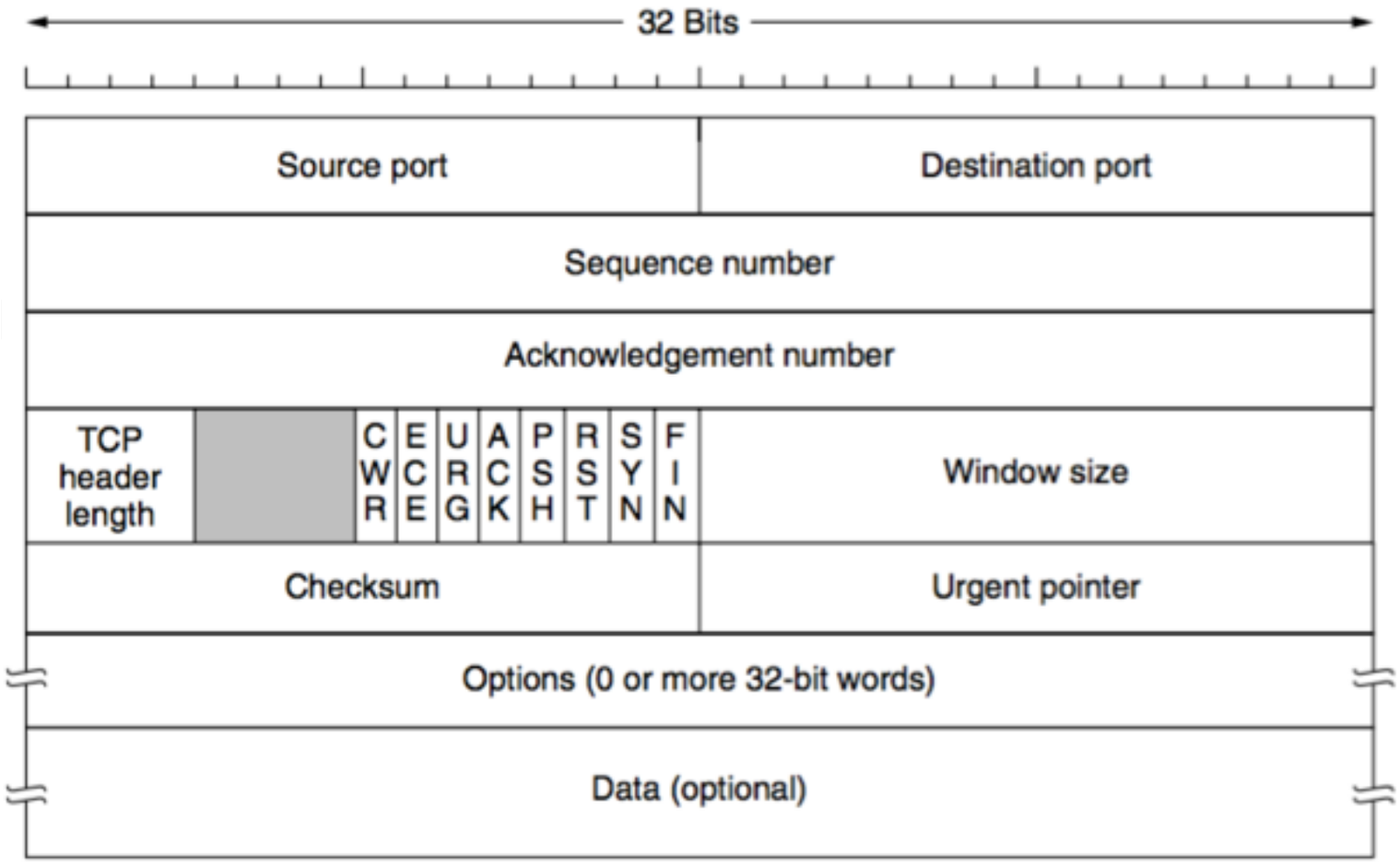
Dynamic window size

Segments can arrive out of order

Retransmissions may include different byte ranges

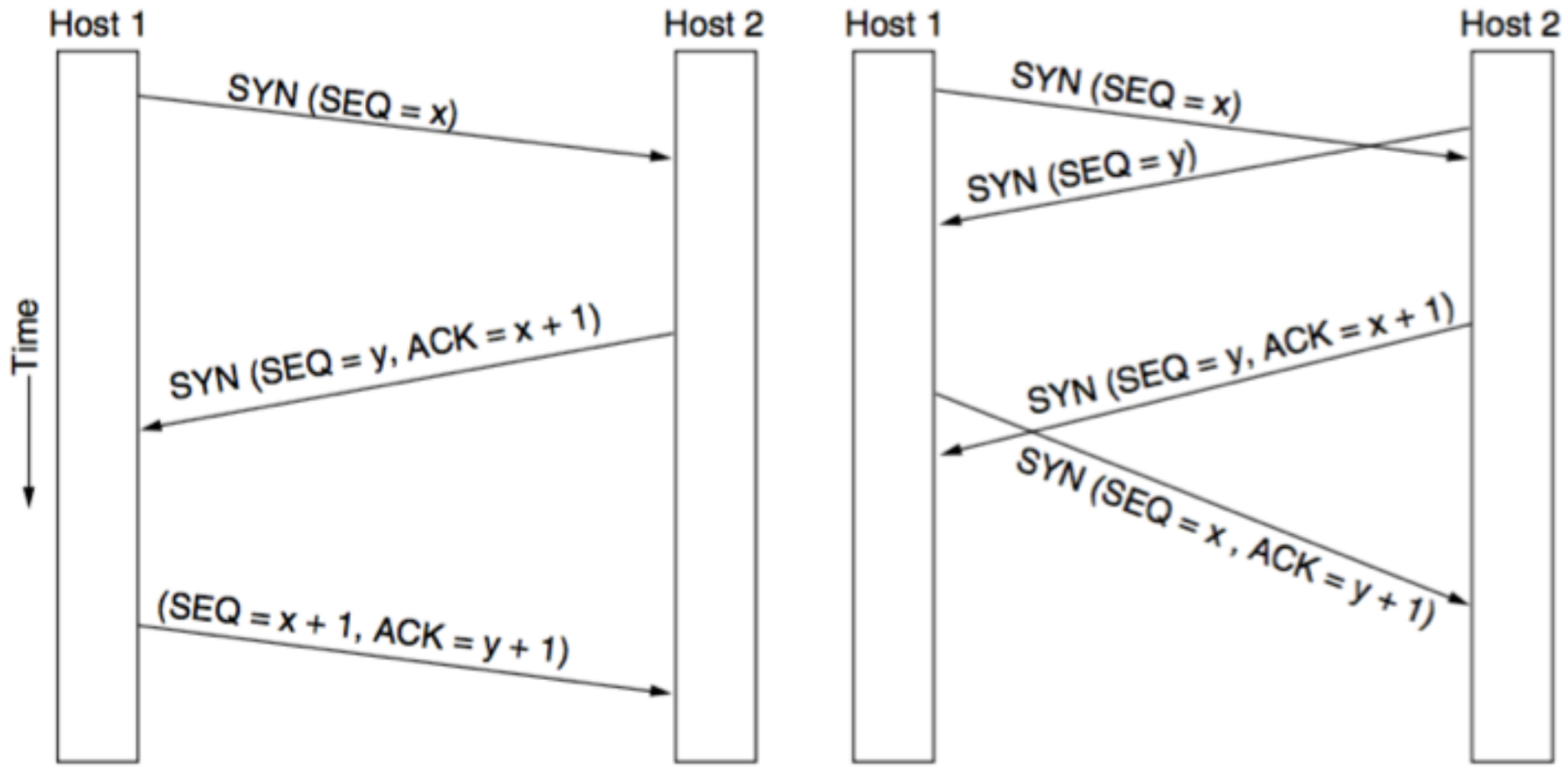
TCP Segment Header

ECE: Tell the sender to slow down
CWR: Congestion Window Reduced
URG: Urgent pointer
ACK: Acknowledgement number valid
PSH: Pushed data
RST: Reset connection
SYN: Establish connection
FIN: Release connection



Source/Destination port: A TCP port plus its host's IP address forms a 48-bit unique end point
Sequnce/Acknowledgement number: Every byte of data is numbered in a TCP stream
TCP header length: How many 32-bit words are contained in the TCP header
Window size: Tells how many bytes may be sent starting at the byte acknowledged
Checksum: Checksums the header (mandatory)
Urgent pointer: Used to indicate a byte offset from the current sequence number at which urgent data are to be found
Options: Provides a way to add extra facilities not covered by the regular header (Maximum segment size, Window scale, Timestamp, Selective ACKnowledgement)

TCP Connection Establishment



Three-way handshake

1. The CONNECT primitive sends a TCP segment with the SYN bit on and ACK bit off and waits for a response
2. If it accepts, an acknowledgement segment is sent back
3. In the event that two hosts simultaneously attempt to establish a connection, just one connection is established

SYN flood

A malicious sender can tie up resources on a host by sending a stream of SYN segments and never following through to complete the connection

TCP Connection Management Modeling

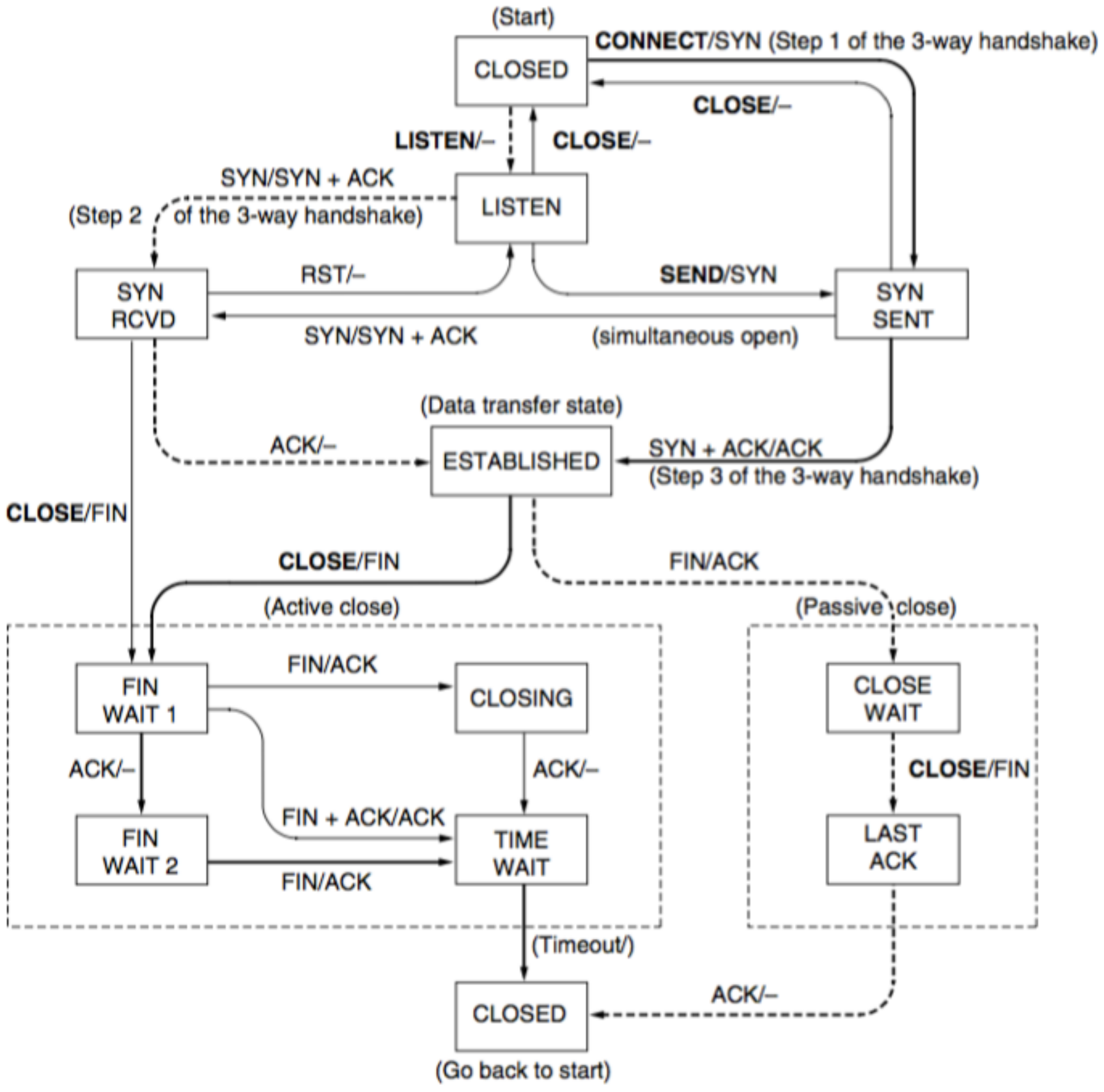
Connection Release

1. Each simplex connection is released independently
2. To release a connection, either party can send a TCP segment with the FIN bit
3. When the FIN is acknowledged, that direction is shut down for new data
4. When both directions have been shut down, the connection is released
5. To avoid the two-army problem, timers are used

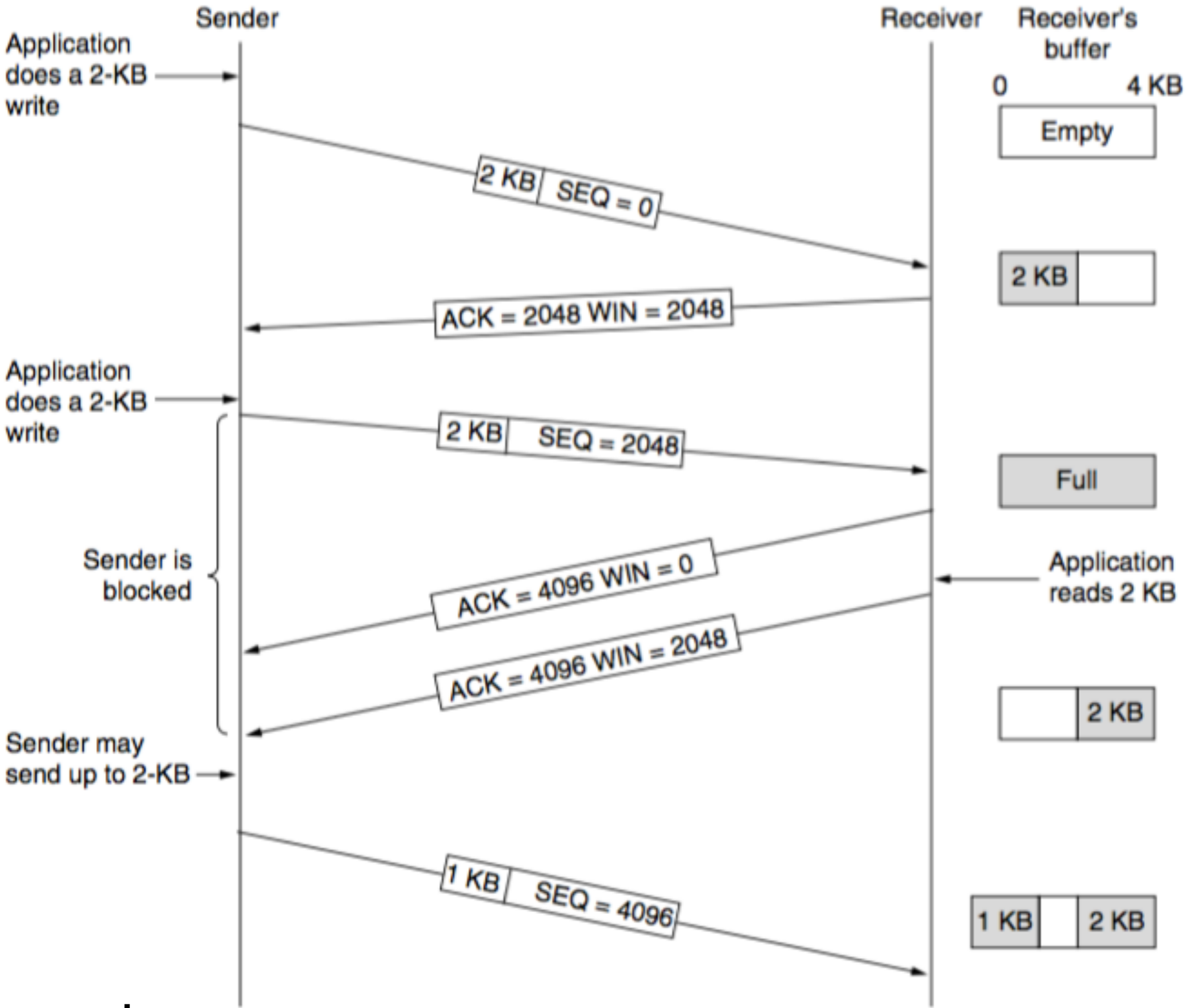
TCP connection management finite state machine

State	Description
CLOSED	No connection is active or pending
LISTEN	The server is waiting for an incoming call
SYN RCVD	A connection request has arrived; wait for ACK
SYN SENT	The application has started to open a connection
ESTABLISHED	The normal data transfer state
FIN WAIT 1	The application has said it is finished
FIN WAIT 2	The other side has agreed to release
TIME WAIT	Wait for all packets to die off
CLOSING	Both sides have tried to close simultaneously
CLOSE WAIT	The other side has initiated a release
LAST ACK	Wait for all packets to die off

TCP connection management finite state machine



TCP Sliding Window



Window probe

The sender may send a 1-byte segment to force the receiver to reannounce the next byte expected and the window size

Nagle's Algorithm

Delayed acknowledgements

Delay acknowledgements and window updates for up to 500 msec in the hope of acquiring some data on which to hitch a free ride

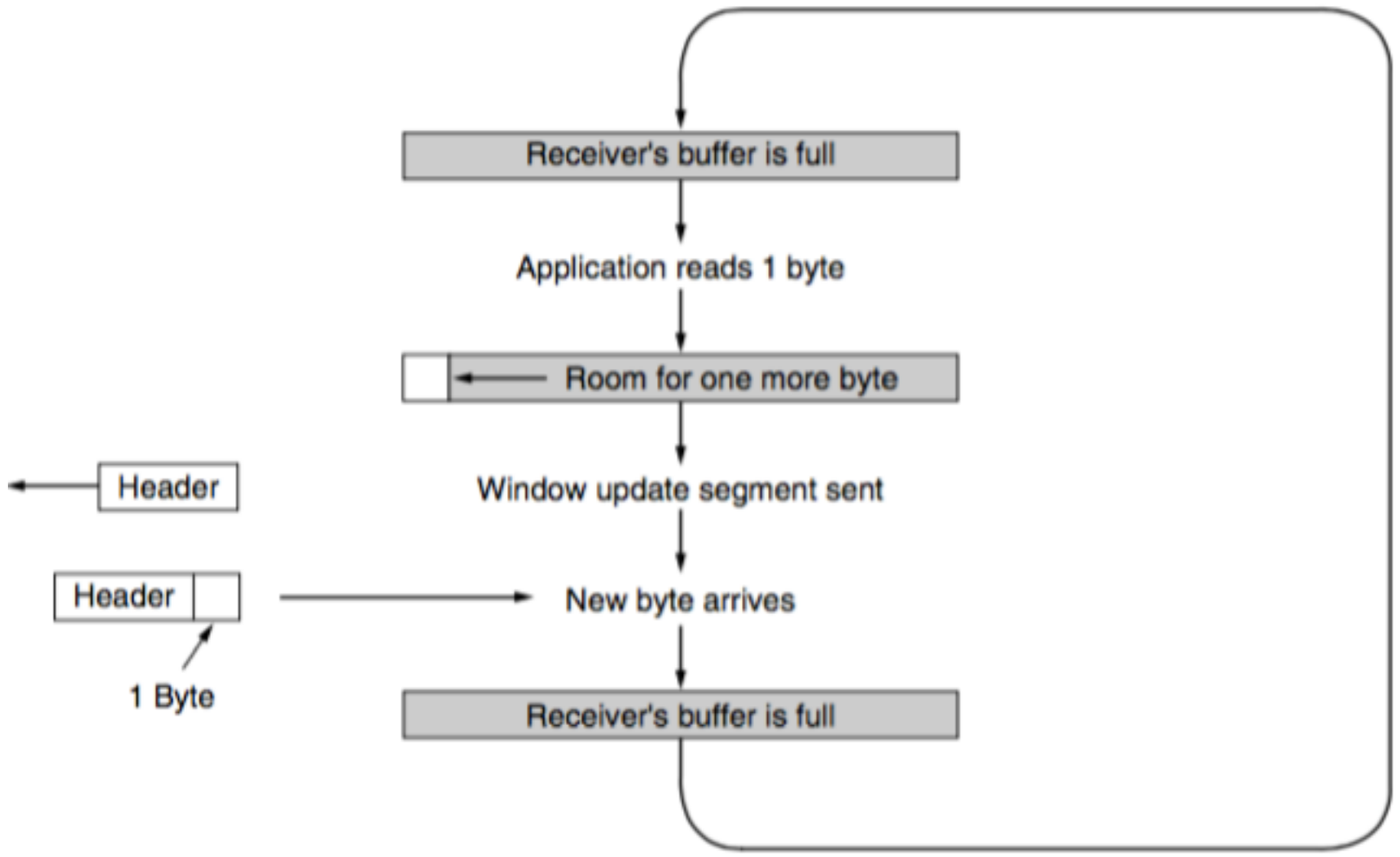
Nagle's algorithm

1. When data come into the sender in small pieces, just send the first piece and buffer all the rest until the first piece is acknowledged
2. Then send all the buffered data in one TCP segment and start buffering again until the next segment is acknowledged
3. A new segment should be sent if enough data have trickled in to fill a maximum segment

To disable Nagle's algorithm

Set TCP_NODELAY

Silly window syndrome

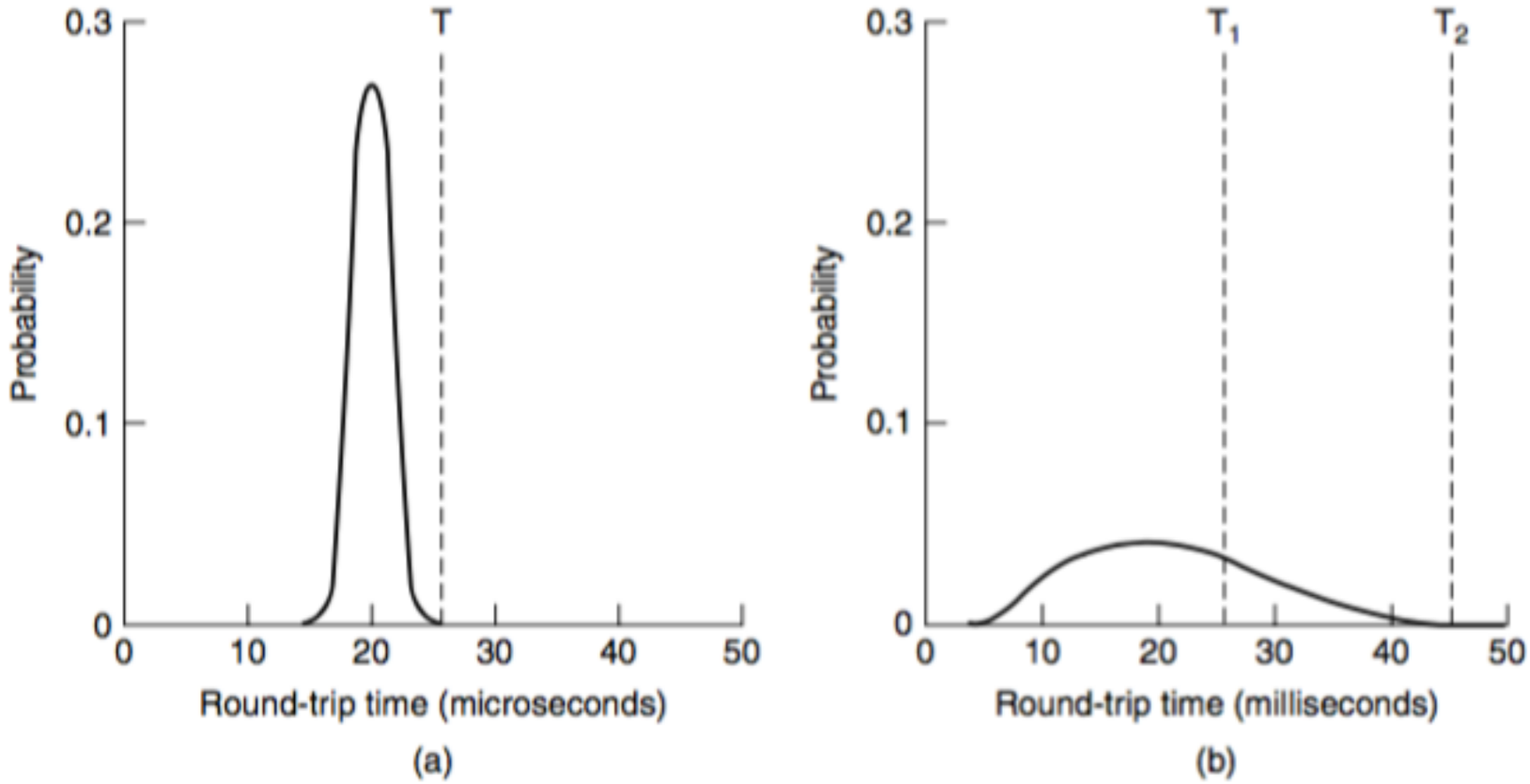


This problem occurs when data are passed to the sending TCP entity in large blocks, but an interactive application on the receiving side reads data only 1 byte at a time.

<u>Clarke's solution</u>

The receiver should not send a window update until it can handle the maximum segment size it advertised when the connection was established or until its buffer is half empty, whichever is smaller.

TCP Timer Management



RTO (Retransmission TimeOut)

When a segment is sent, a retransmission timer is started. If the segment is acknowledged before the timer expires, the timer is stopped. If, on the other hand, the timer goes off before the acknowledgement comes in, the segment is retransmitted.

How long should the timeout be?

If the timeout is set too short, unnecessary retransmissions will occur, clogging the Internet with useless packets. If it is set too long, performance will suffer due to the long retransmission delay whenever a packet is lost.

TCP Timer Management

EWMA (Exponentially Weighted Moving Average)

Use a dynamic algorithm that constantly adapts the timeout interval, based on continuous measurements of network performance

SRTT (Smoothed Round-Trip Time)

The best current estimate of the round-trip time to the destination in question.

$$SRTT = \alpha\, SRTT + (1 - \alpha)\, R$$

R: how long the acknowledgement took

α: how quickly the old values are forgotten

Using a constant value was too inflexible because it failed to respond when the variance went up

RTTVAR (Round- Trip Time VARiation)

$$RTTVAR = \beta\, RTTVAR + (1 - \beta)\ |SRTT - R|$$

RTO (retransmission timeout)

$$RTO = SRTT + 4 \times RTTVAR$$

TCP Timer Management

Karn's algorithm

Do not update estimates on any segments that have been retransmitted

Persistence timer

To avoid deadlock the sender transmits a probe to the receiver. The response to the probe gives the window size.

Keepalive timer

When a connection has been idle for a long time, the keepalive timer may go off to cause one side to check whether the other side is still there.

TCP Congestion Control

It is up to the transport layer to receive congestion feedback from the network layer and slow down the rate of traffic that it is sending into the network.

TCP congestion control is based on implementing AMID using a window and with packet loss as the binary signal.

Congestion window (network capacity)

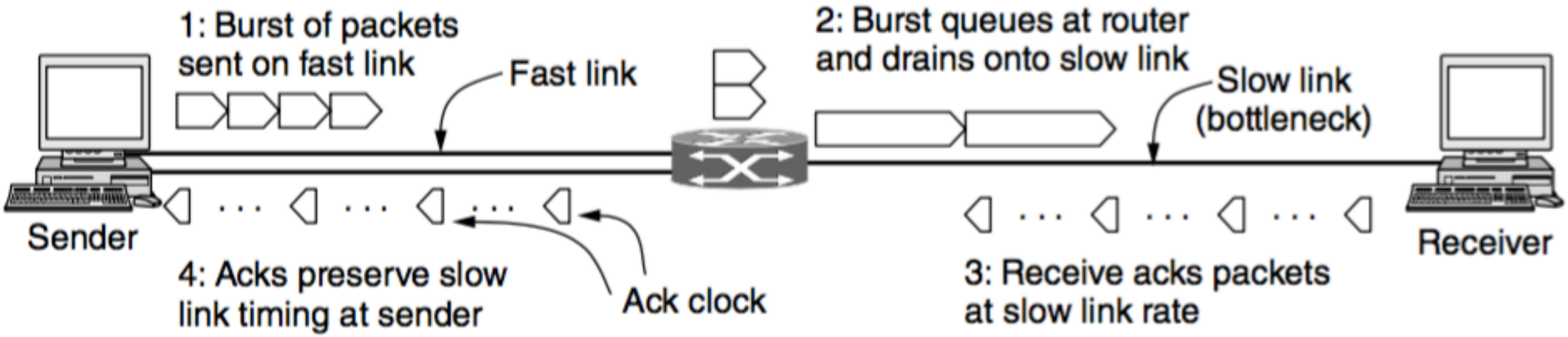
Size is the number of bytes the sender may have in the network at any time.

Flow control window (receiver capacity)

Specifies the number of bytes that the receiver can buffer

All the Internet TCP algorithms assume that lost packets are caused by congestion and monitor timeouts

ACK Clock

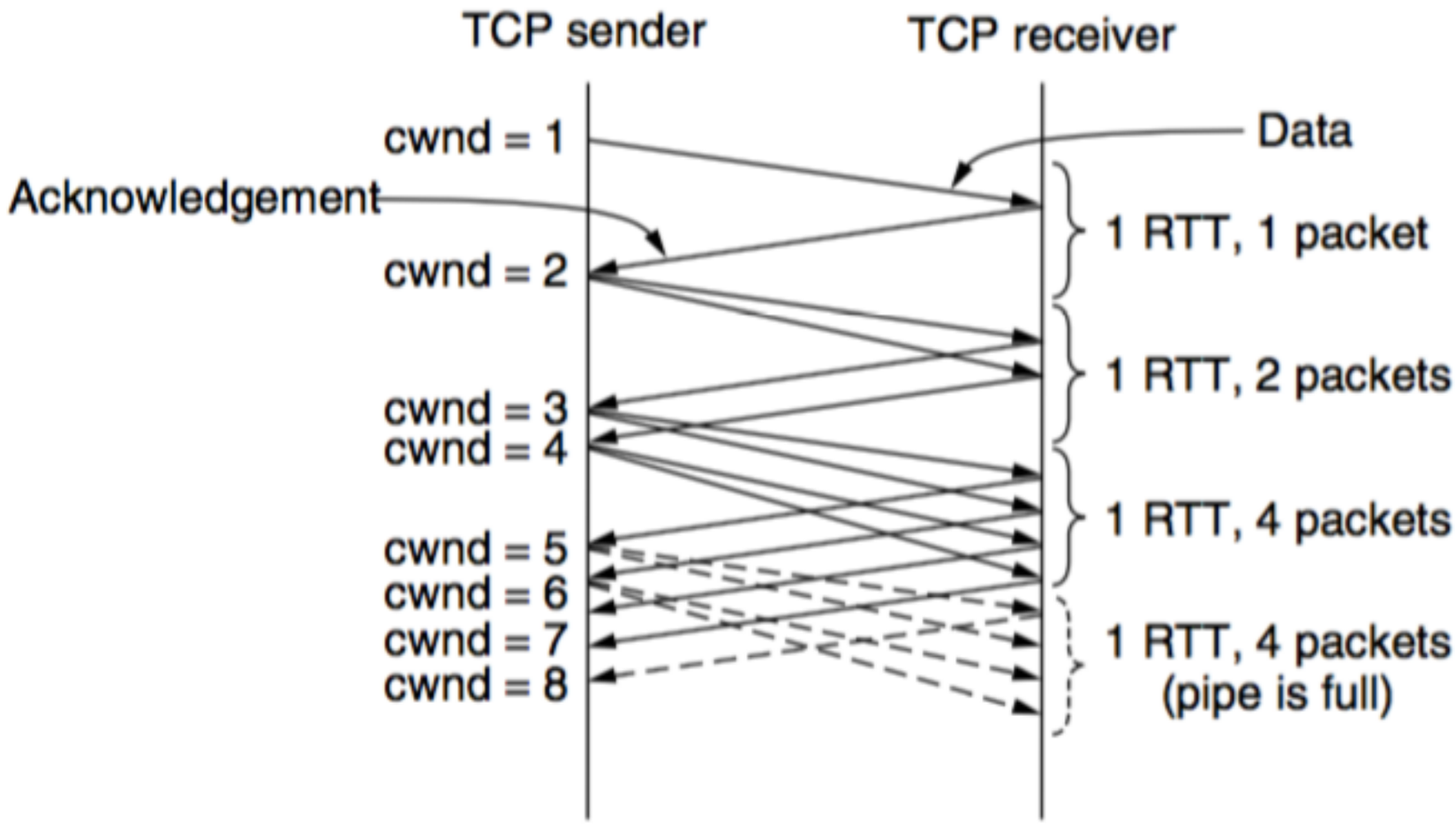


The acknowledgements return to the sender at about the rate that packets can be sent over the slowest link in the path.

If it injects new packets into the network at this rate, they will be sent as fast as the slow link permits, but they will not queue up and congest any router along the path.

Slow Start

The AIMD rule will take a very long time to reach a good operating point on fast networks if the congestion window is started from a small size.

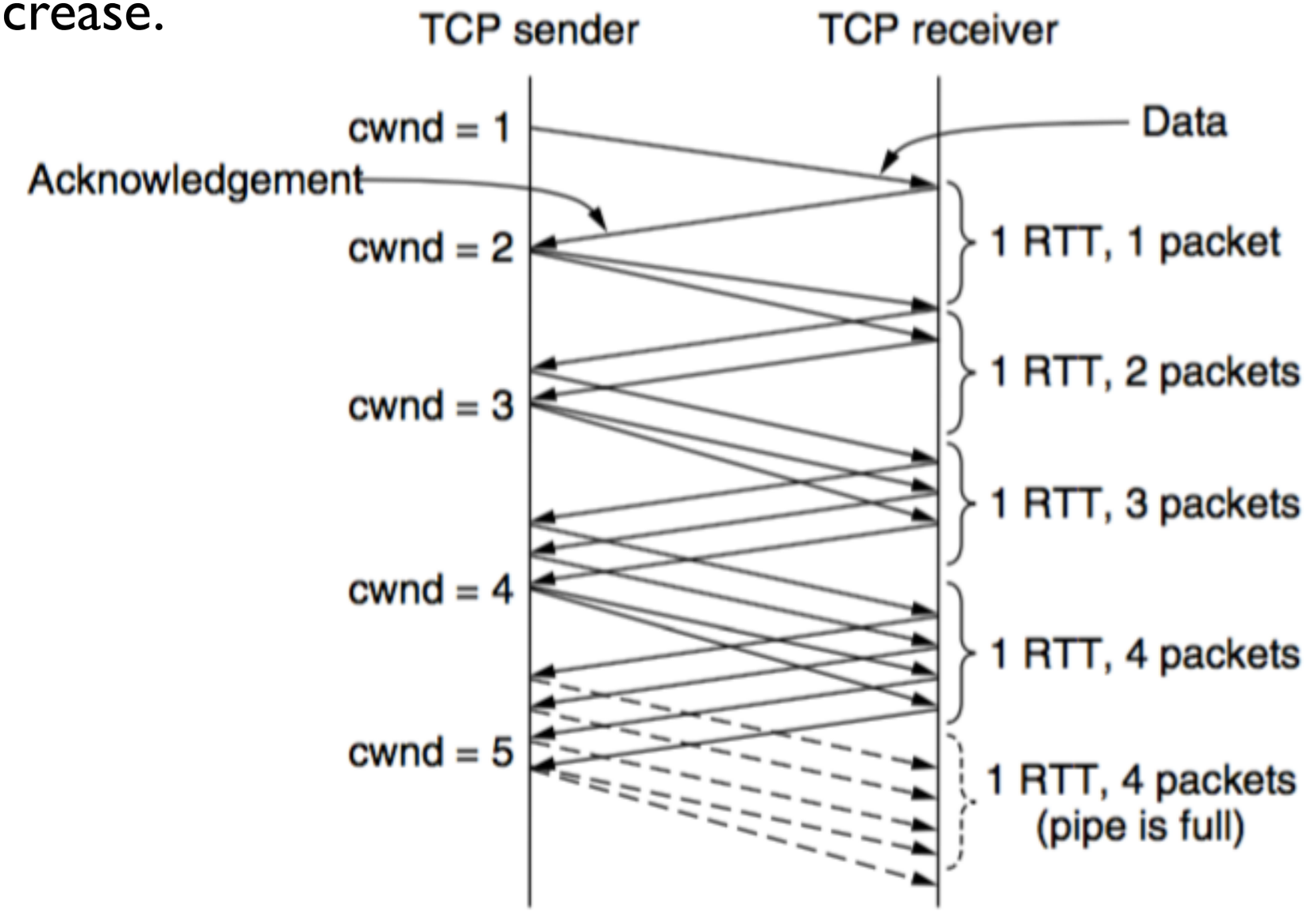


The AIMD rule will take a very long time to reach a good operating point on fast networks if the congestion window is started from a small size.

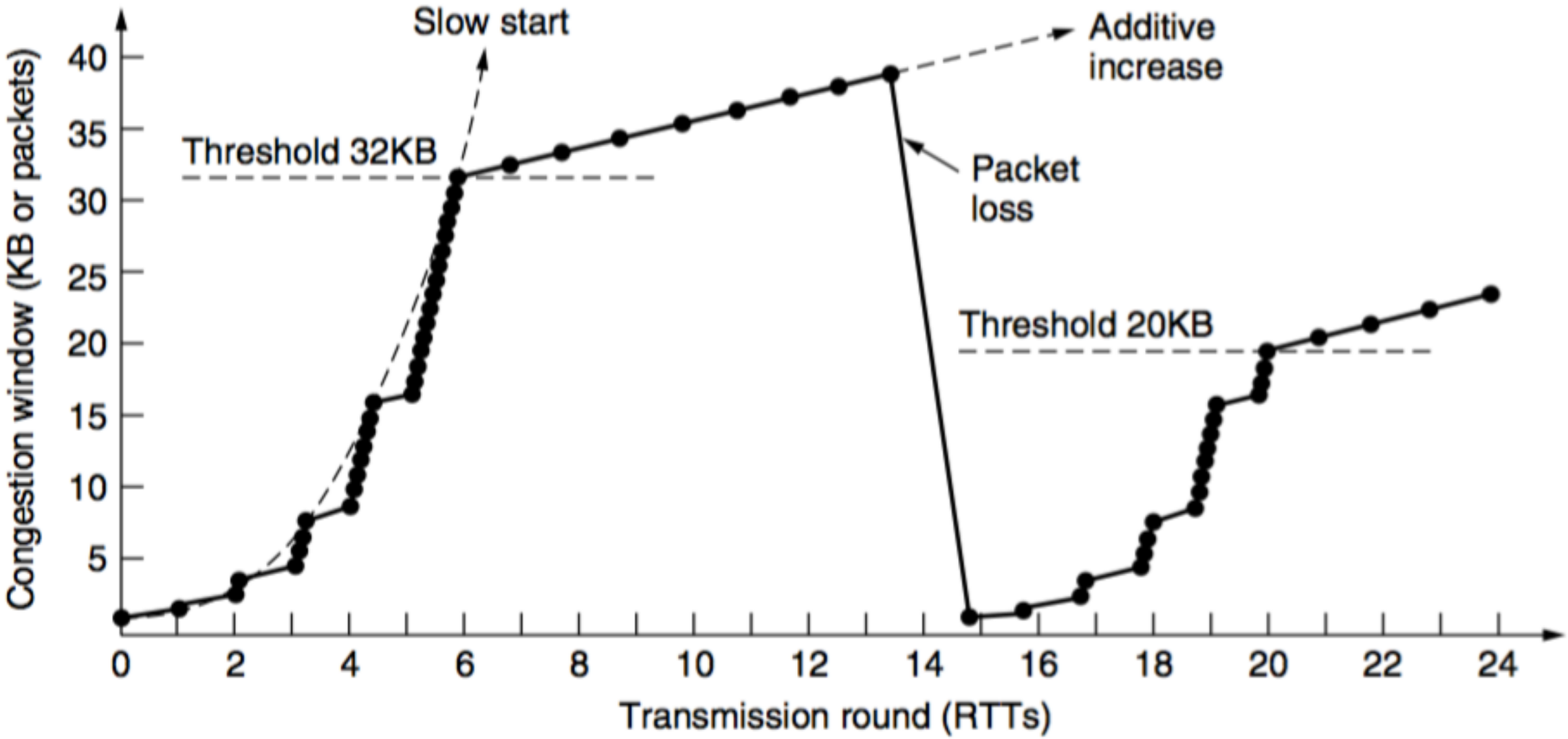
Because slow start causes exponential growth, eventually it will send too many packets into the network too quickly.

Slow Start Threshold

1. Initially this value is set arbitrarily high, to the size of the flow control window, so that it will not limit the connection.
2. Whenever a packet loss is detected, for example, by a timeout, the slow start threshold is set to be half of the congestion window and the entire process is restarted.
3. Whenever the slow start threshold is crossed, TCP switches from slow start to additive increase.



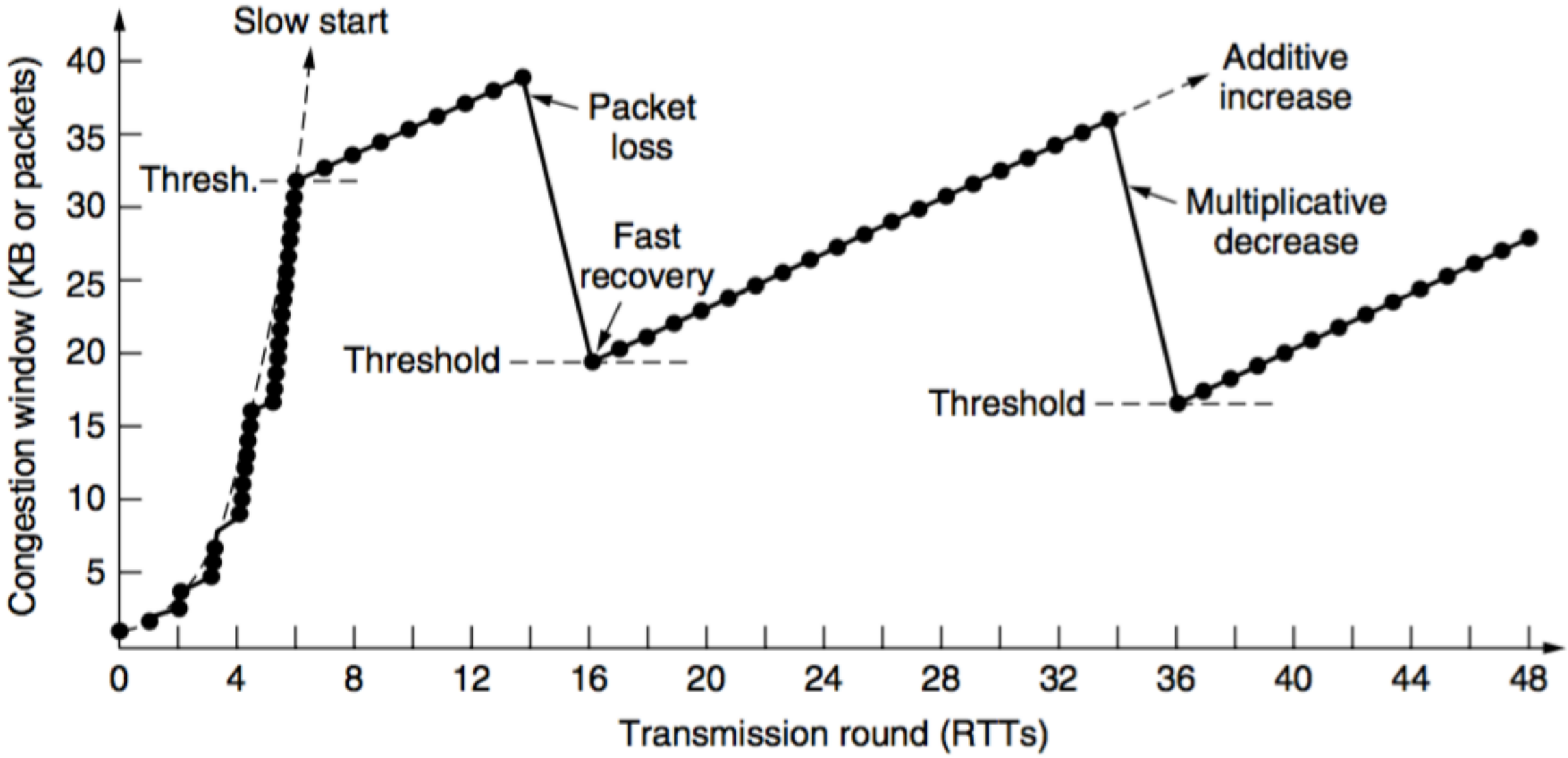
Slow Start Threshold



TCP Tahoe

Slow start followed by additive increase

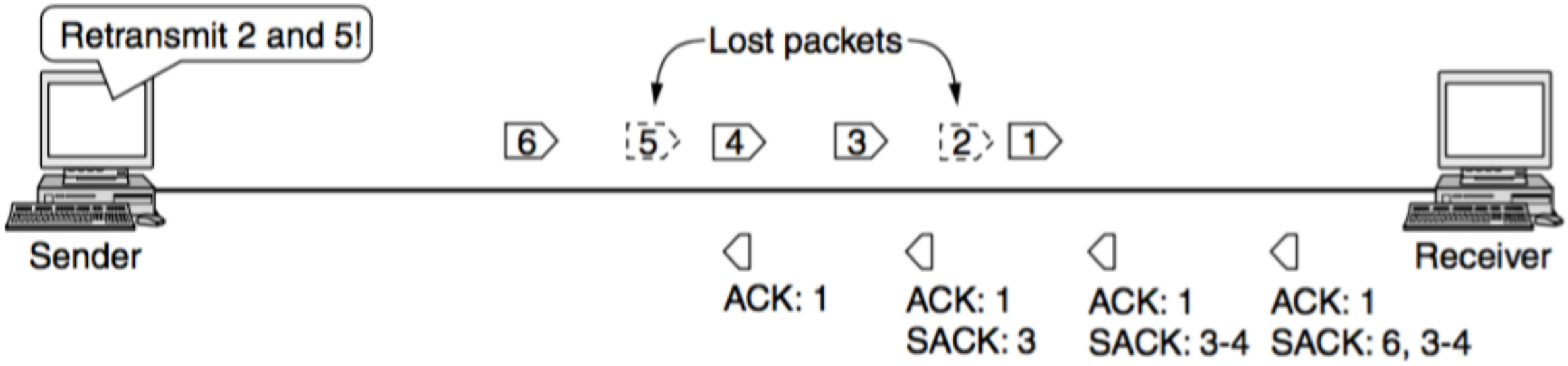
Slow Start Threshold



TCP Reno

Slow start followed by additive increase + fast recovery

SACK (Selective ACKnowledgements)

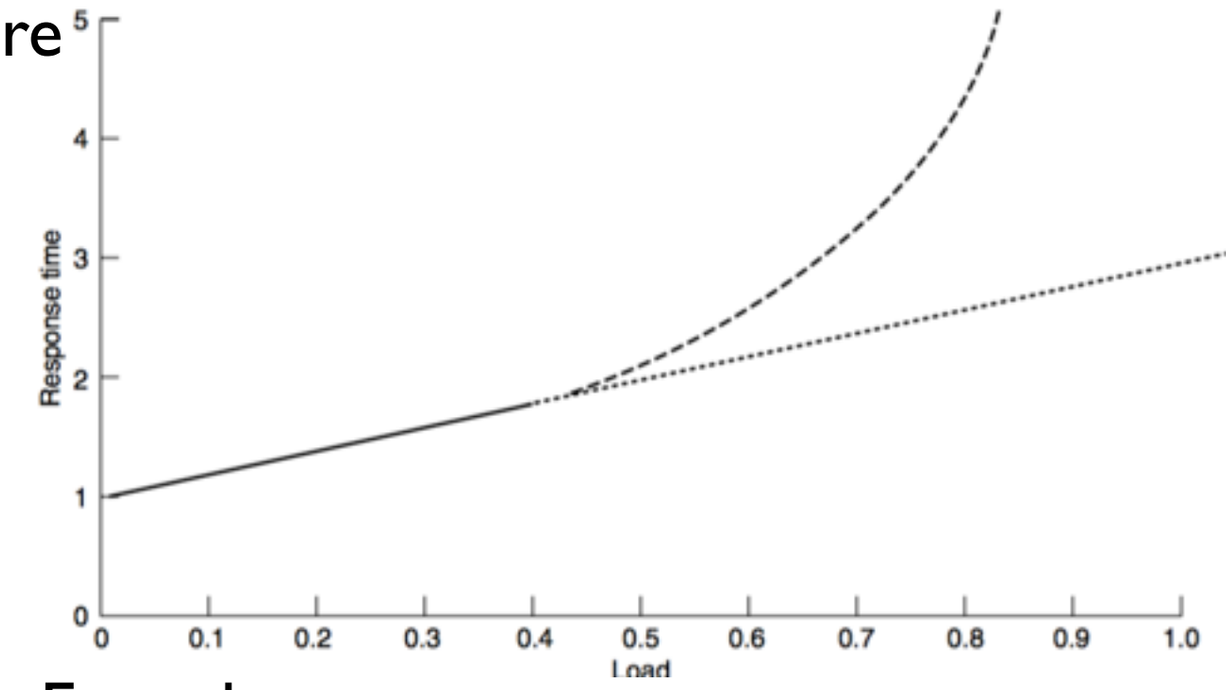


The use of ECN is enabled for a TCP connection when both the sender and receiver indicate that they are capable of using ECN by setting the ECE and CWR bits during connection establishment.

Performance Problems in Computer Networks

Causes of performance degradation

1. Congestion
2. Resource imbalance
3. Broadcast storm (caused by error notification)
4. Synchronous overload after power failure
5. Lack of system tuning
6. Setting timeouts
7. Jitter



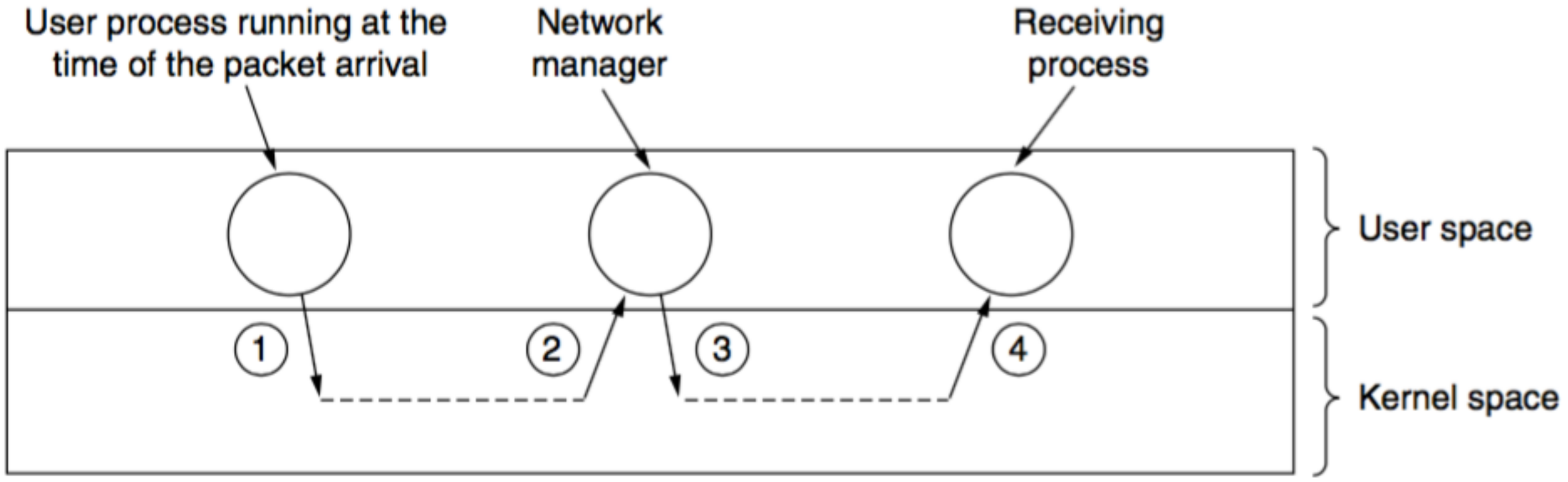
Performance measurement

1. Make Sure That the Sample Size Is Large Enough
2. Make Sure That the Samples Are Representative
3. Caching Can Wreak Havoc with Measurements
4. Be Sure That Nothing Unexpected Is Going On during Your Tests
5. Be Careful When Using a Coarse-Grained Clock
6. Be Careful about Extrapolating the Results

Performance Problems in Computer Networks

Host Design for Fast Networks

1. Host Speed Is More Important Than Network Speed
2. Reduce Packet Count to Reduce Overhead
3. Minimize Data Touching
4. Minimize Context Switches
5. Avoiding Congestion Is Better Than Recovering from It
6. Avoid Timeouts

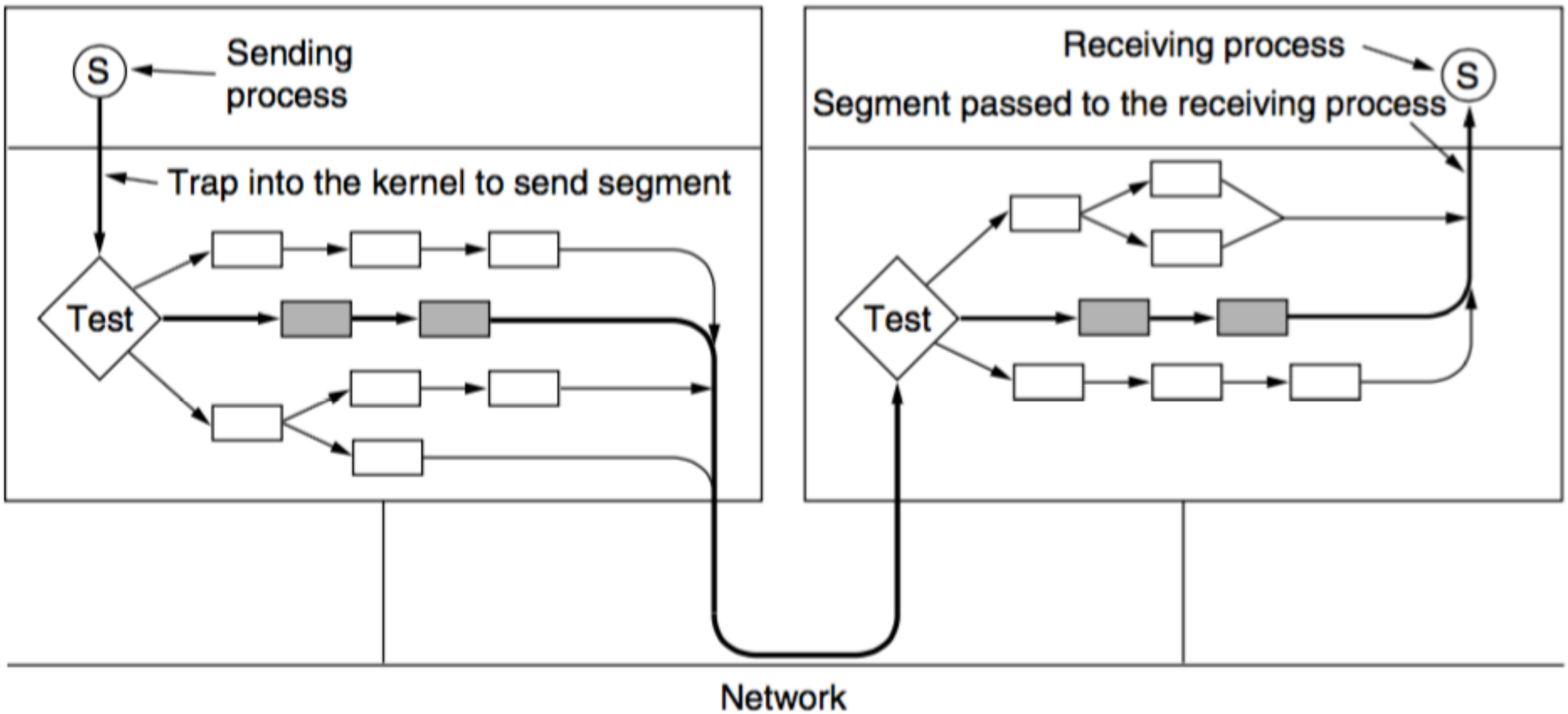


Fast Segment Processing

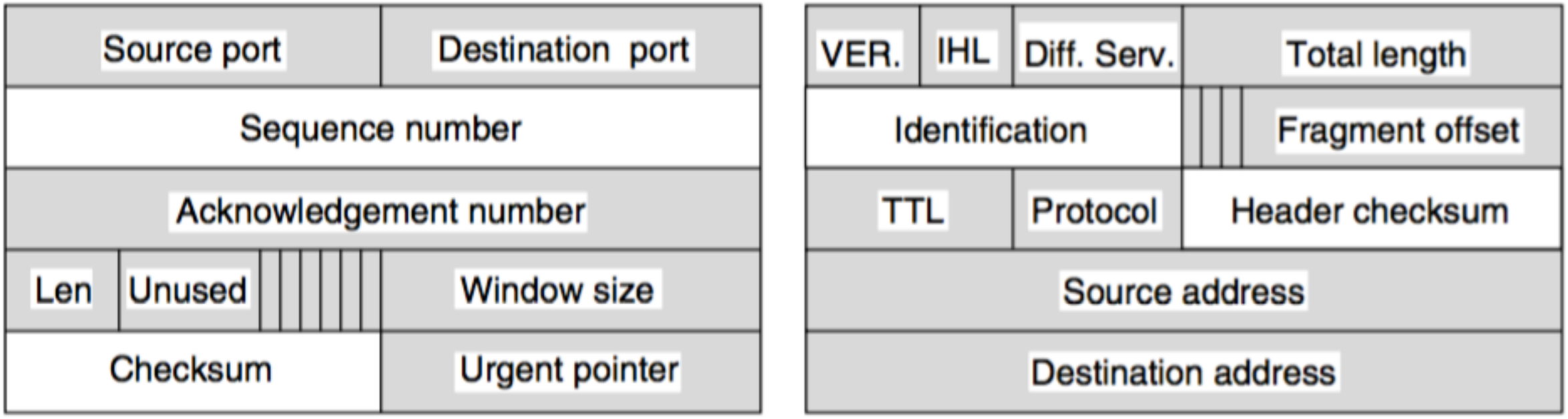
Segment processing overhead

1. Overhead per segment
2. Overhead per byte

The key to fast segment processing is to separate out the normal, successful case (one-way data transfer) and handle it specially.



Fast Segment Processing



TCP IP

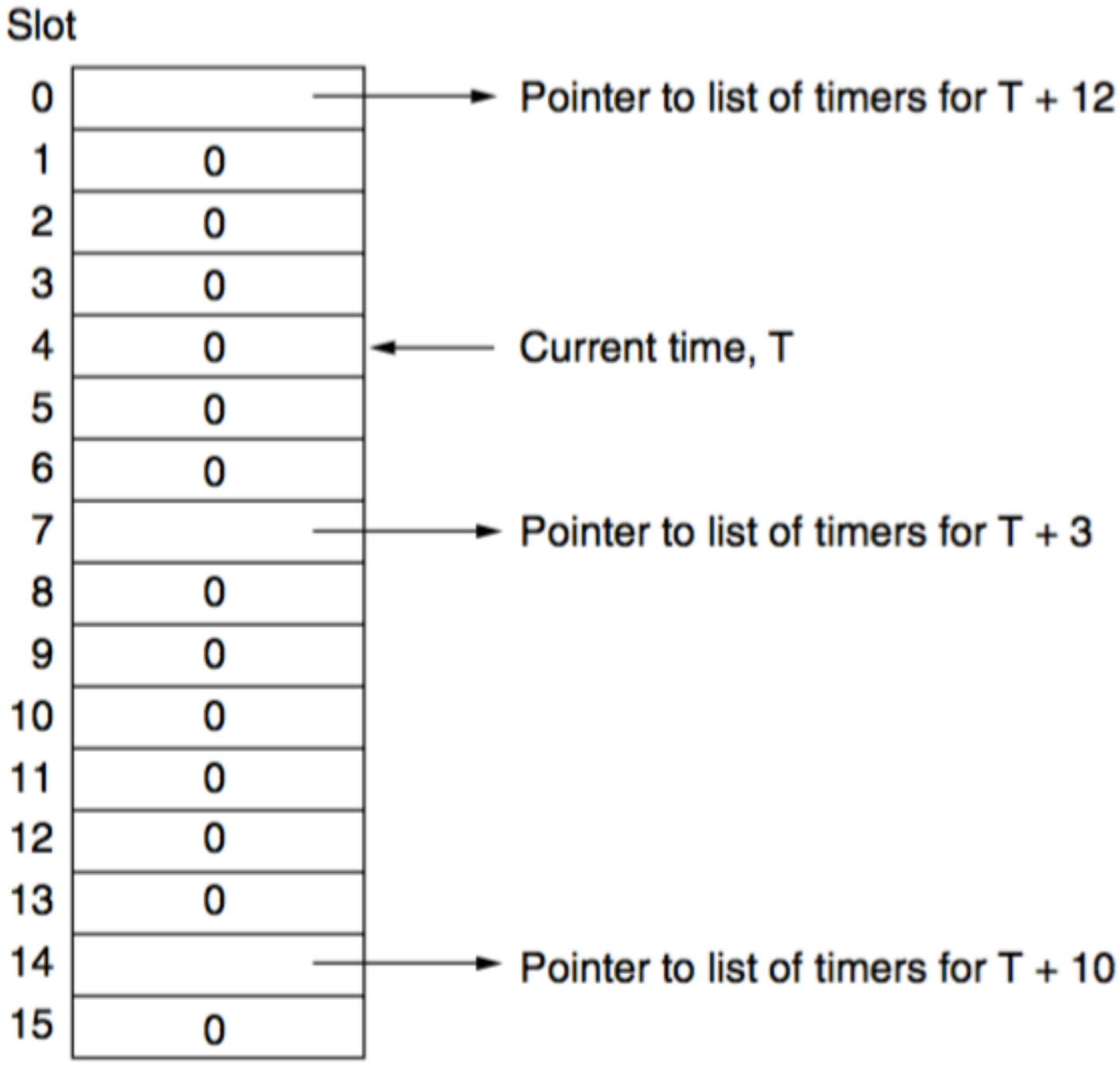
Shaded parts are the same between consecutive segments on a one-way flow.

1. Connection record can be stored in a hash table
2. Function of the two IP addresses and two ports is the key
3. The segment is checked to see if it is a normal one
4. The fast path updates the connection record and copies the data to the user.

Timing Wheel

Timers are scheduled to expire at 3, 10, and 12 ticks from now

If the timer set for T + 10 has to be canceled, the list starting in slot 14 has to be searched and the required entry removed



Header Compression

Header compression obtains large gains by using knowledge of the protocol format

Van Jacobson (1990)

Able to compress a typical TCP/IP header of 40 bytes down to an average of 3 bytes

ROHC (RObust Header Compression)

Designed to tolerate the loss that can occur on wireless links

Header compression can help by reducing the amount of data that is sent, and hence reducing transmission delay

Protocols for Long Fat Networks

Wrapping time of 32-bit sequence numbers

56-kbps: 1 week
10-Mbps: 57 minutes
1-Gbps: 34 seconds (< maximum packet lifetime 120 seconds)

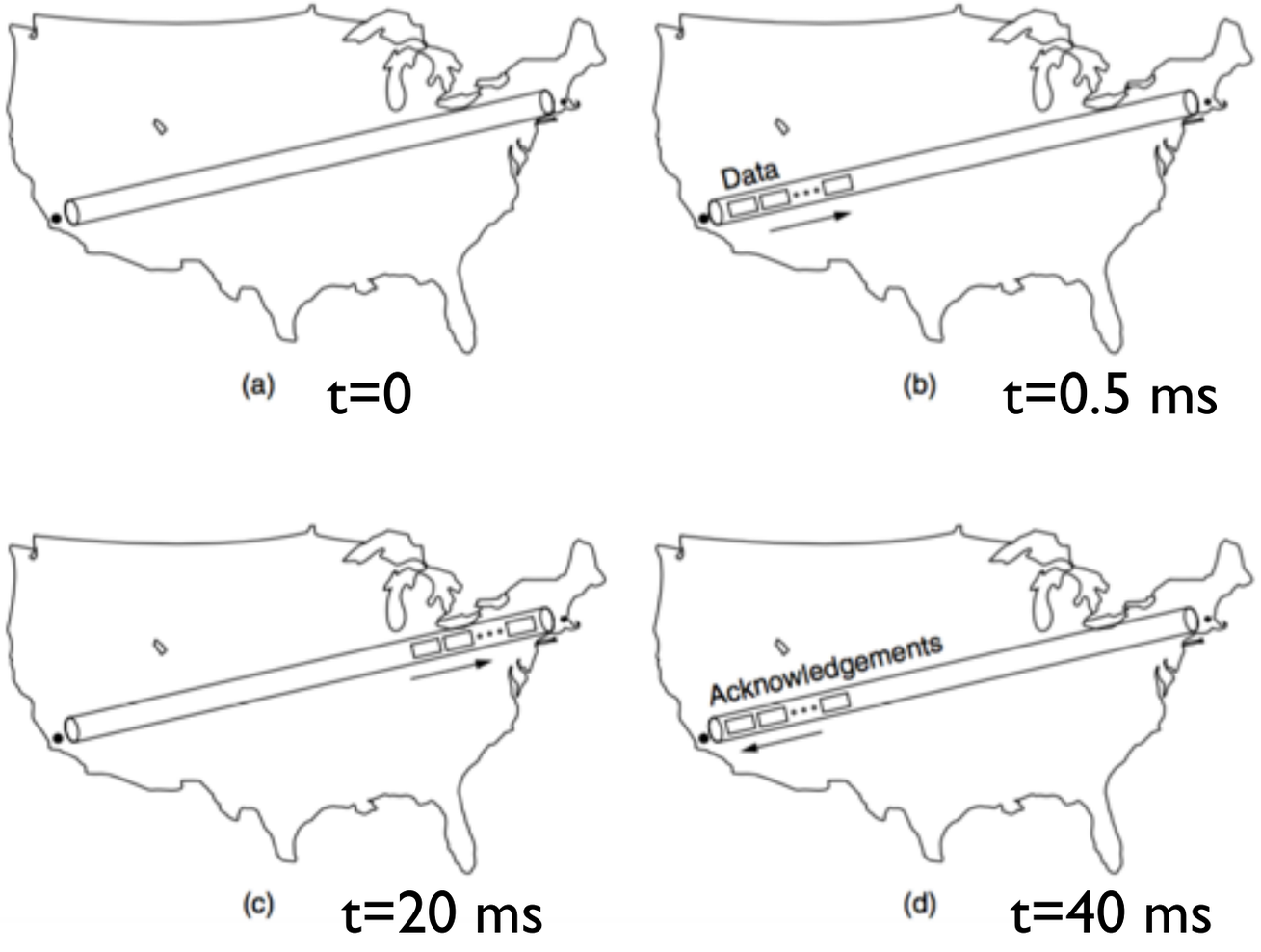
PAWS (Protection Against Wrapped Sequence numbers)

Use timestamp to extend the sequence number

Bandwidth-delay product

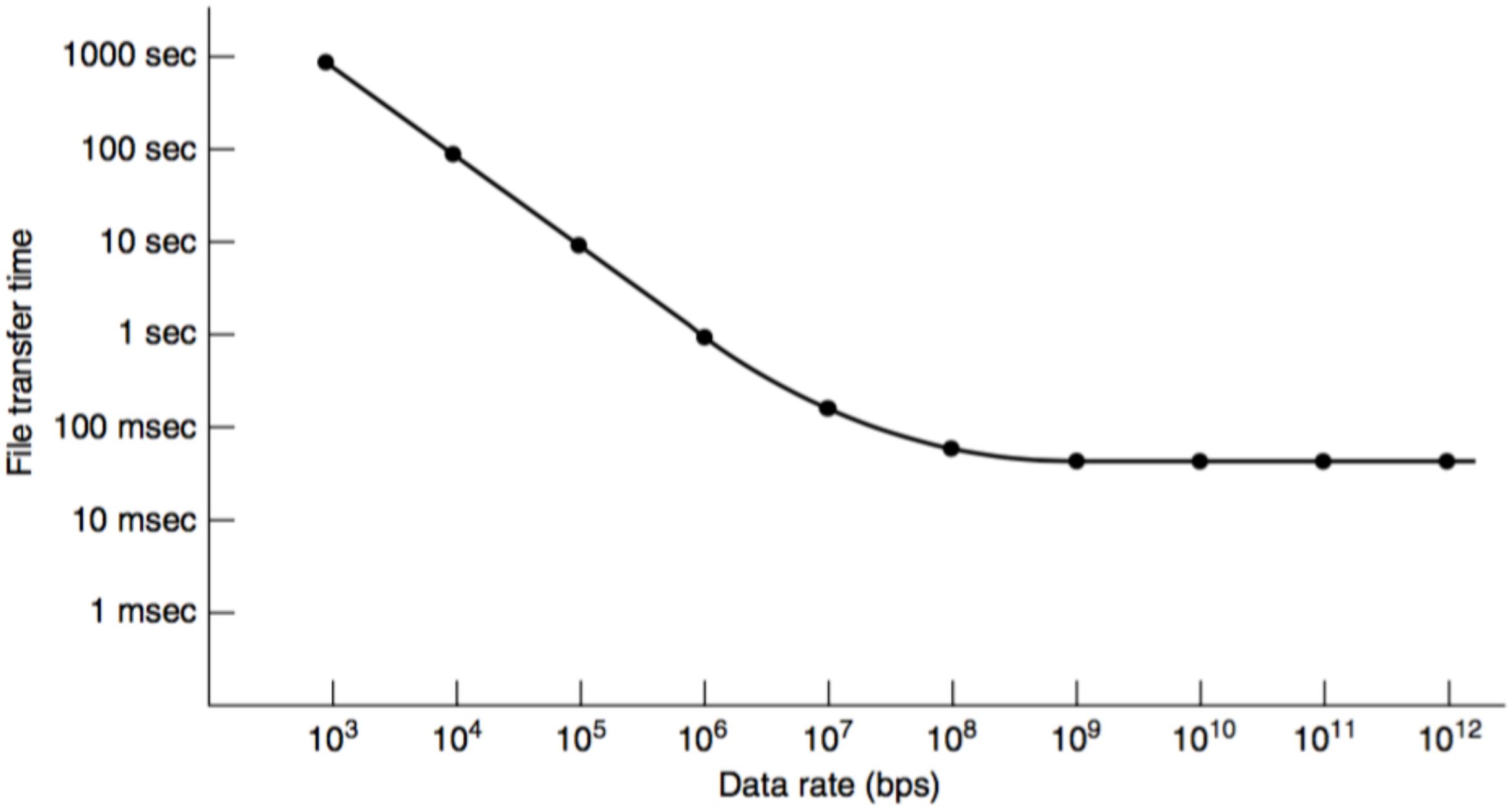
(bandwidth) x (round-trip delay)
=40 Mbits

The receiver's window must be at least as large as the bandwidth-delay product



Protocols for Long Fat Networks

Gigabit lines are delay limited rather than bandwidth limited



DTN (Delay-Tolerant Networking)

Satellites, mobile devices

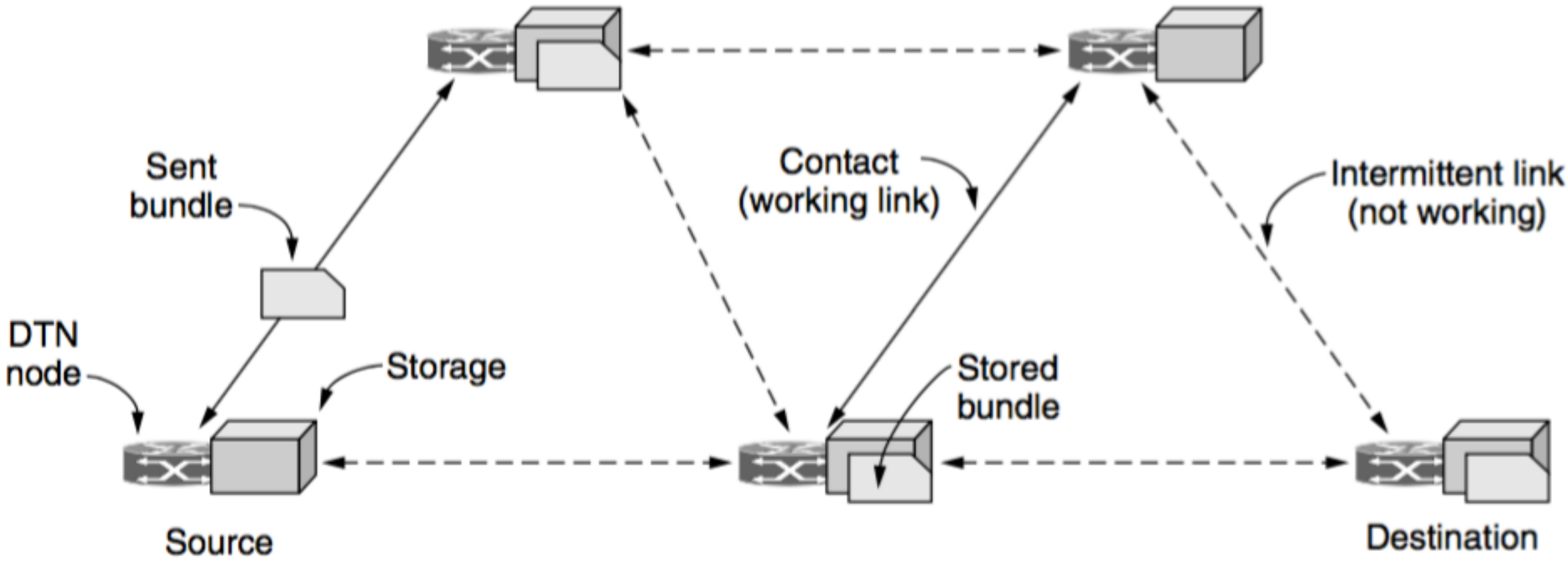
Message switching

Store data at nodes and forward them later when there is a working link

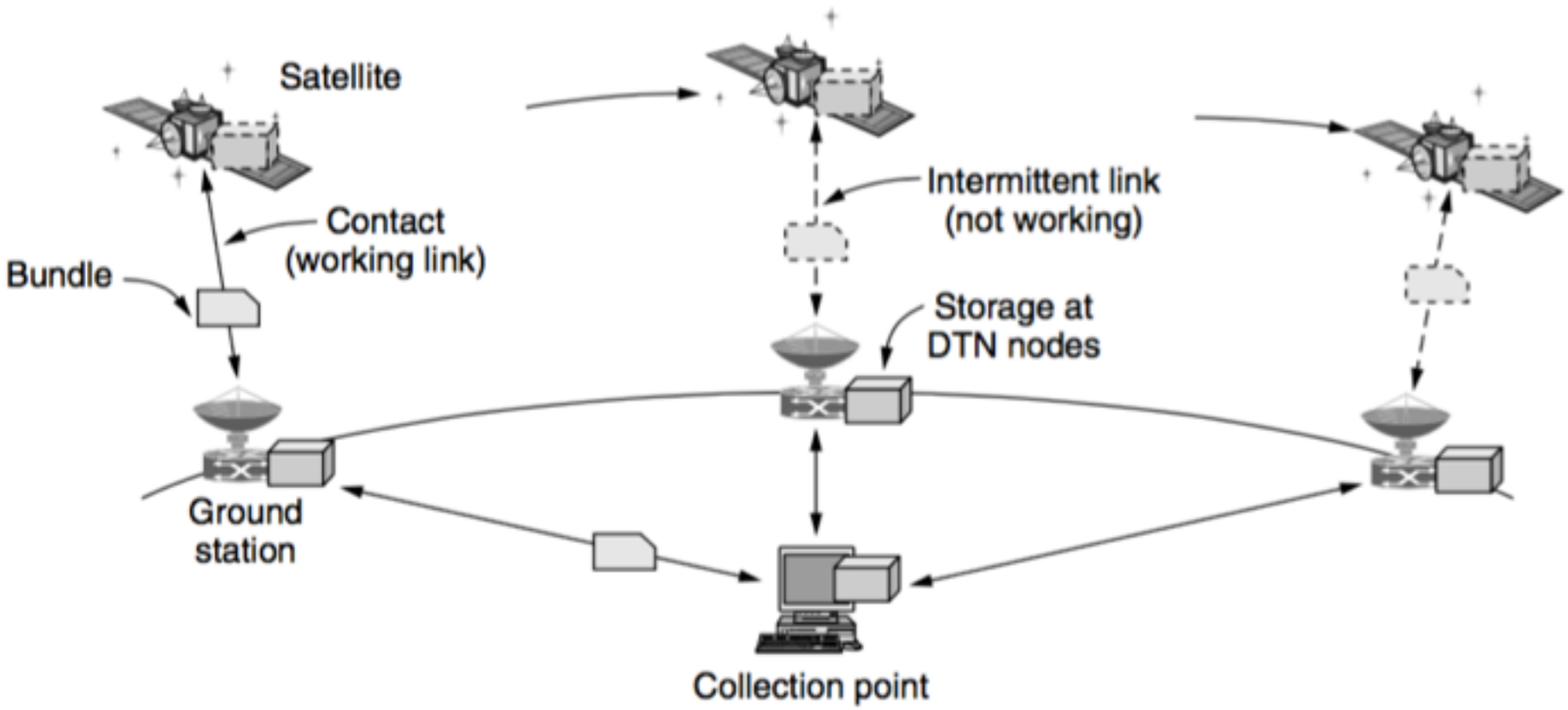
DTN Architecture

At DTN nodes, bundles may be stored for hours

Nodes are allowed to move

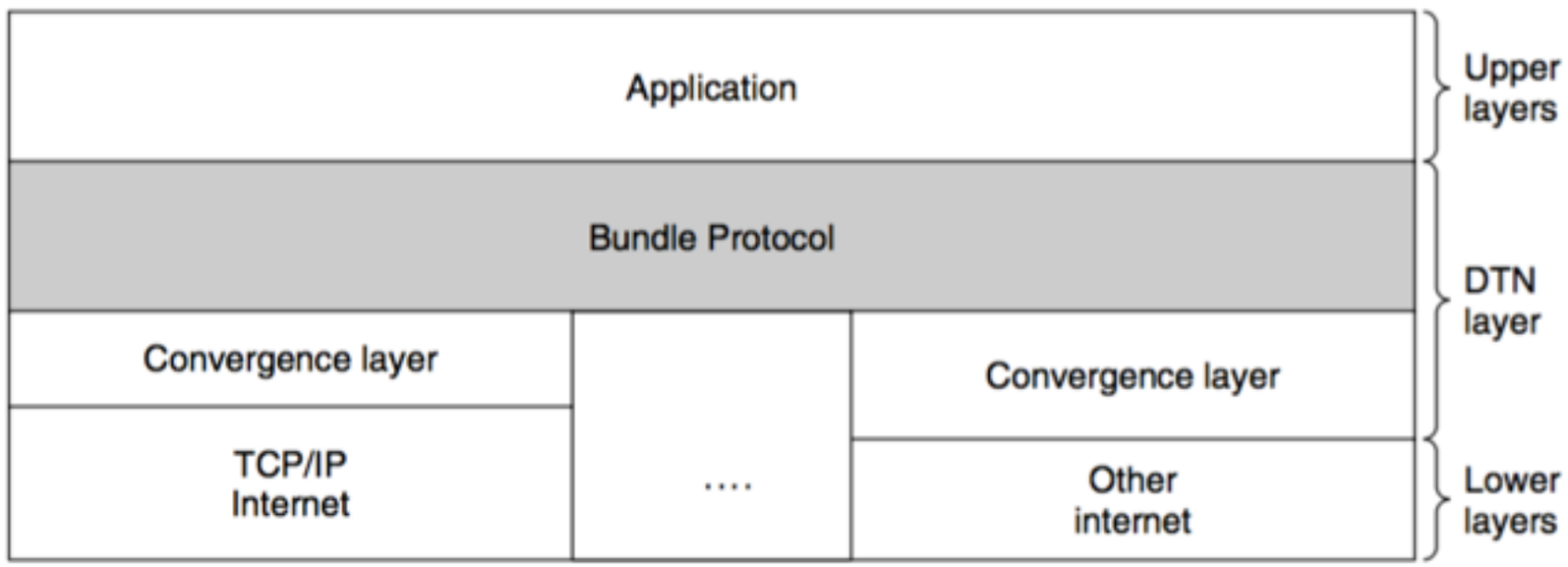


DTN (Delay-Tolerant Networking)



Bundle protocol

Runs over TCP/IP or other internet protocols



講義日程（2Q）

		授業計画		課題
06/14	第９回	ネットワーク層１ ルーティング・輻輳制御	５章	ルーティングの種類を理解し 輻輳制御手法を説明できる
06/21	第１０回	ネットワーク層２ インターネットとサービス品質	５章	インターネットの制御プロトコルを理解し ネットワーク間の接続について説明できる
06/28	第１１回	トランスポート層１ トランスポート・プロトコルの要素	６章	誤り制御とフロー制御を理解し 輻輳制御について説明できる
07/05	第１２回	トランスポート層２ UDP と TCP	６章	TCP の信頼性を理解し TCP のコネクション管理を説明できる
07/12	第１３回	アプリケーション層 DNS, 電子メール, www	７章	DNS, 電子メール, www のしくみを理解し ストリーミング，P2P について説明できる
07/26	第１４回	ネットワークセキュリティ１ 対称鍵暗号，公開鍵暗号	８章	暗号アルゴリズムを理解し SHA-1,2 と RSA について説明できる
08/02	第１５回	ネットワークセキュリティ２ デジタル署名，認証プロトコル	８章	電子メール,Web のセキュリティ の脅威について把握できる