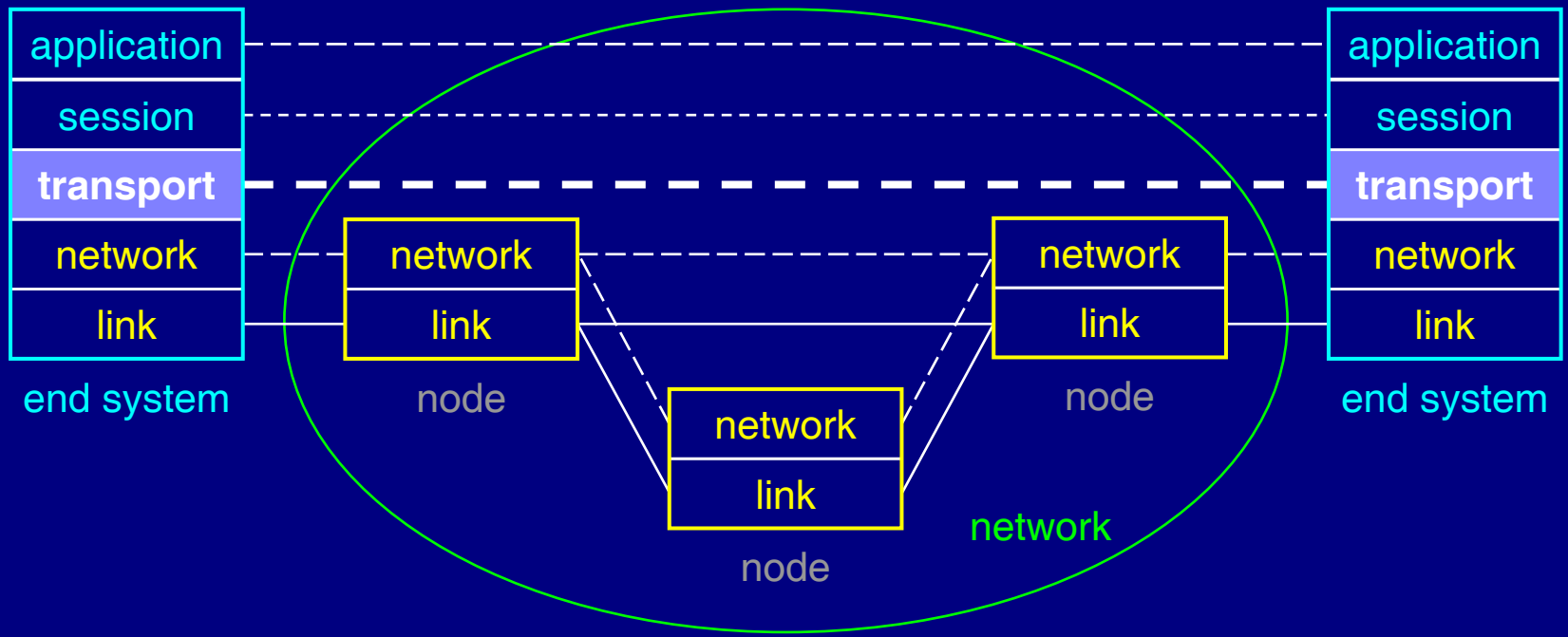
End-to-End Protocols

1. Introduction
2. Fundamentals and design principles
3. Network architecture and topology
4. Network control and signalling
5. Network components
   - 5.1 links
   - 5.2 switches and routers
6. End systems
7. End-to-end protocols
8. Networked applications
9. Future directions

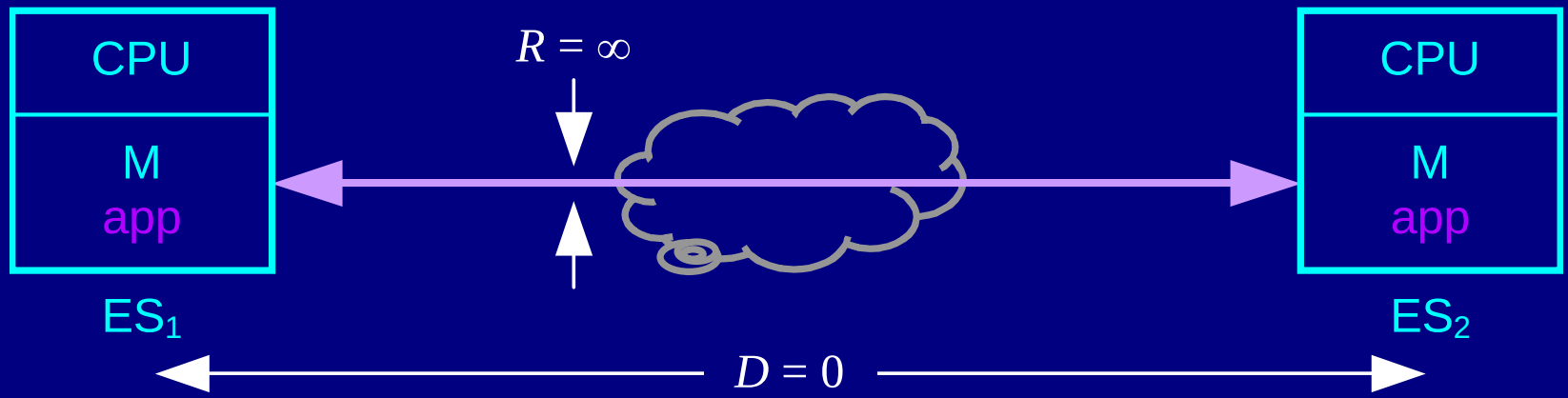
End-to-End Protocols



7.1. Functions and mechanisms
7.2. State management
7.3. Framing and multiplexing
7.4. Error control
7.5. Flow & congestion control
7.6. Security & info assurance

Ideal End-to-End Network Model

Bandwidth and Delay



- Zero end-to-end delay
- Unlimited end-to-end bandwidth

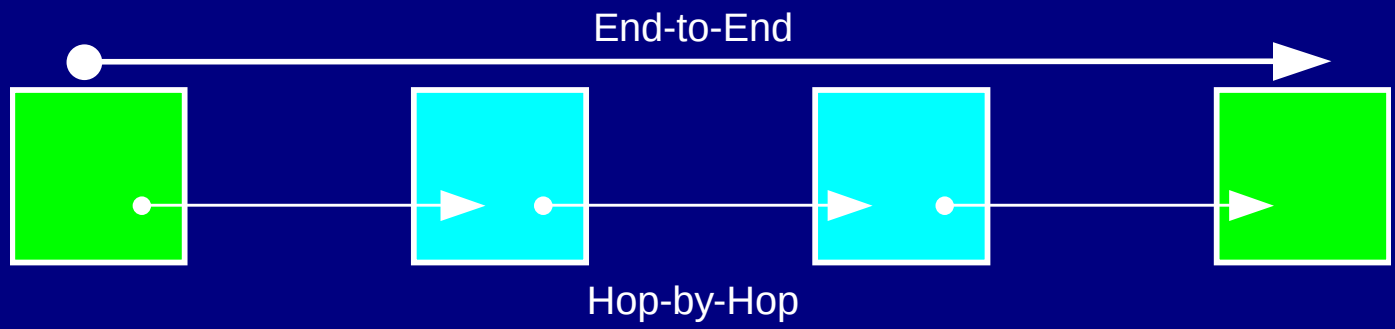
End-to-End Protocols
Functions and Mechanisms

7.1 Functions and mechanisms
- 7.1.1 End-to-end semantics
- 7.1.2 End-to-end mechanisms
- 7.1.3 Transport protocols
- 7.1.4 Control of state

7.2 State management
7.3 Framing and multiplexing
7.4 Error control
7.5 Flow and congestion control
7.6 Security and information assurance

E2E Functions and Mechanisms

End-to-End Semantics



- Hop-by-hop functions
  - link layer protocols
  - link compression and FEC
  - network forwarding
  - link / subnet error control
  - embedded protocols (e.g. protocol boosters)
- End-to-end functions
  - transport protocols
  - source routing
  - end-to-end encryption
  - session protocols
  - application protocols

E2E Functions and Mechanisms
End-to-End Argument

- Hop-by-hop function composition ≠ end-to-end
- Examples
  - HBH encryption has data in clear inside network nodes
  - HBH link error control doesn't cover network layer errors

End-to-End Argument T-3

Functions required by communicating applications can be correctly and completely implemented only with the knowledge and help of the applications themselves. Providing these functions as features within the network itself is not possible.

E2E Functions and Mechanisms
Hop-by-Hop Performance Enhancement

- E2E functions replicated HBH if performance benefit
- Example
  - per link error control in high bandwidth-×-delay networks reduce E2E control loop when error

Hop-by-Hop Performance Enhancement Corollary T-3A

It is beneficial to duplicate an end-to-end function hop-by-hop if the result is an overall (end-to-end) improvement in performance.

E2E Functions and Mechanisms
E2E Argument Misinterpretations

- E2E-only
  - do not replicate E2E services or features HBH
  - violated HBH performance enhancement corollary
- Everything E2E
  - implement as many services or feature E2E as possible
  - misstatement of Internet design philosophy:
    simple stateless network for resilience and survivability

E2E Functions and Mechanisms
Meaning of "In the Network"

- Functional (general use in this tutorial)
  - layers 1 – 3
    - physical
    - MAC
    - link
    - network
- Topological
  - colocated with network nodes
- Administrative
  - owned by network service provider

E2E Functions and Mechanisms

End-to-End Mechanisms

- Framing
- Multiplexing
- End-to-end state and connection management
- Error control
- Flow control
- Congestion control

E2E Functions and Mechanisms
Transport Protocol Service Categories

- Transfer mode
  - connectionless datagram
  - connection-oriented
  - transaction
  - continuous media streaming
- Reliability
  - loss tolerance
- Delivery order
  - ordered
  - unordered
- Traffic characteristics

E2E Functions and Mechanisms
Types of Transport Protocols

- Spectrum of transport protocol specialisation
  - general purpose
  - functionally partitioned
  - application-oriented
  - special purpose
- Software engineering and implementation choice

Transport Protocol Functionality T-IV

Transport protocols must be organised to deliver the set of end-to-end high-bandwidth, low latency services needed by applications. Options and service models should be modularly accessible, without unintended performance degradation and feature interactions.

E2E Functions and Mechanisms

Example$_{7.2}$ End-to-End Internet Protocols

- Transport protocols

Protocol	Name	Function	Status	Ref
TCP	transmission control protocol	reliable data transfer with congestion control	standard	RFC 793 STD 007
UDP	user datagram protocol	socket access to unreliable IP datagrams	standard	RFC 768 STD 006
RTP	real-time protocol	file and document transfer (typically over UDP)	standards track	RFC 1889
T/TCP	TCP for transactions	remote login	experimental	RFC 1644
RDP	reliable data protocol	reliable data transfer with no congestion control	experimental	RFC 908
SCTP	stream control transmission protocol	signalling proposed for wireless	proposed standard	RFC 2960

E2E Functions and Mechanisms

Example 7.2 End-to-End Internet Protocols

- "Application layer" protocols

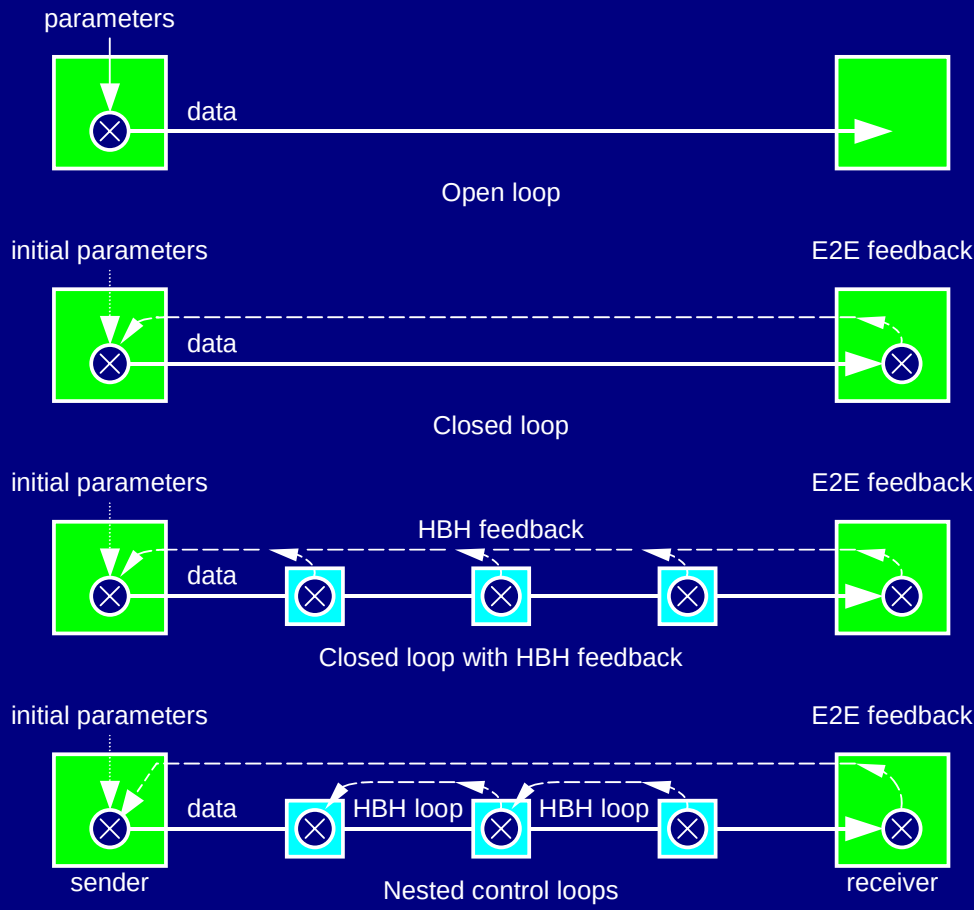
Protocol	Name	Function/use
HTTP	hypertext transfer protocol	Web browsing
SMTP	simple mail transfer protocol	email relay and delivery
FTP	file transfer protocol	file and document transfer
Telnet	telnet	remote login
NFS	network file system	remote access to files
RTSP	real-time streaming protocol	control of multimedia streaming

"application layer" only because they run over TCP or UDP

E2E Functions and Mechanisms

Control of State

- Open loop
  - control parameters
  - no feedback
- Closed loop
  - initial parameters
  - feedback
    - dynamic adaptation
- Closed loop with HBH feedback
- Nested control loops



E2E Functions and Mechanisms
Control of State

- Open-loop control
  - use when possible to eliminate control loop latency
    - particularly for high bandwidth-×-delay product networks
- Closed-loop feedback control
  - use when necessary, e.g. reliability

Open- vs. Closed-loop Control T-6D

Use open-loop control based on knowledge of the network path to reduce the delay in closed loop convergence. Use closed-loop control to react to dynamic network and application behaviour; intermediate per hop feedback can sometimes reduce the time to converge.

E2E Functions and Mechanisms
Control of State

- Aggressiveness of closed-loop control
  - rapid enough to system converges
  - not too aggressive avoid instability and oscillations

Aggressiveness of Closed-Loop Control T-6D*a*

The response to closed-loop feedback must be rapid enough so the system converges quickly to a new stable state – but not so aggressive that oscillations occur due to overreaction to transients.

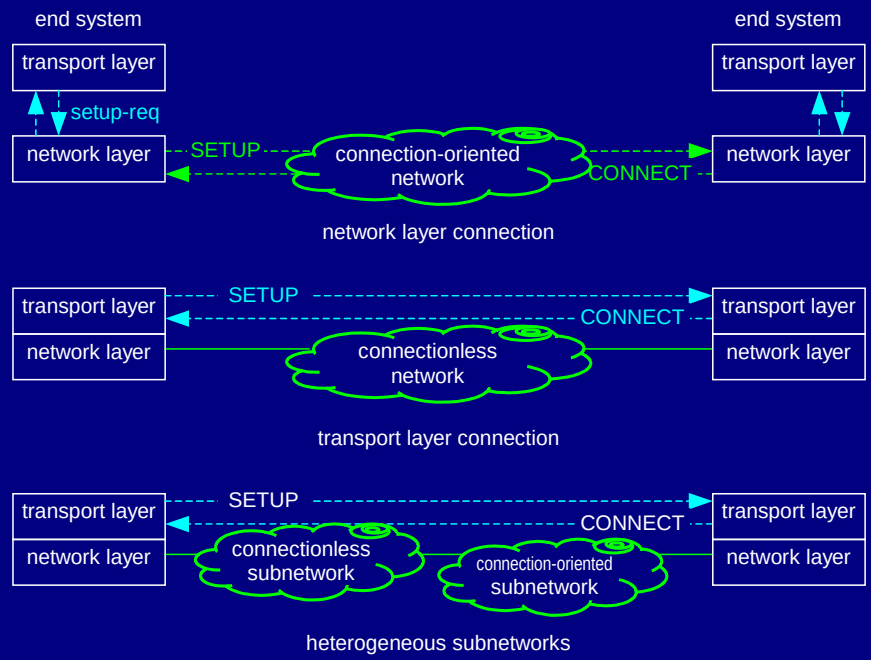
End-to-End Protocols
State Management

7.1 Functions and mechanisms
7.2 State management
- 7.2.1 Impact of high speed
- 7.2.2 Transfer modes
- 7.2.3 State establishment and maintenance
- 7.2.4 Assumed initial conditions

7.3 Framing and multiplexing
7.4 Error control
7.5 Flow and congestion control
7.6 Security and information assurance

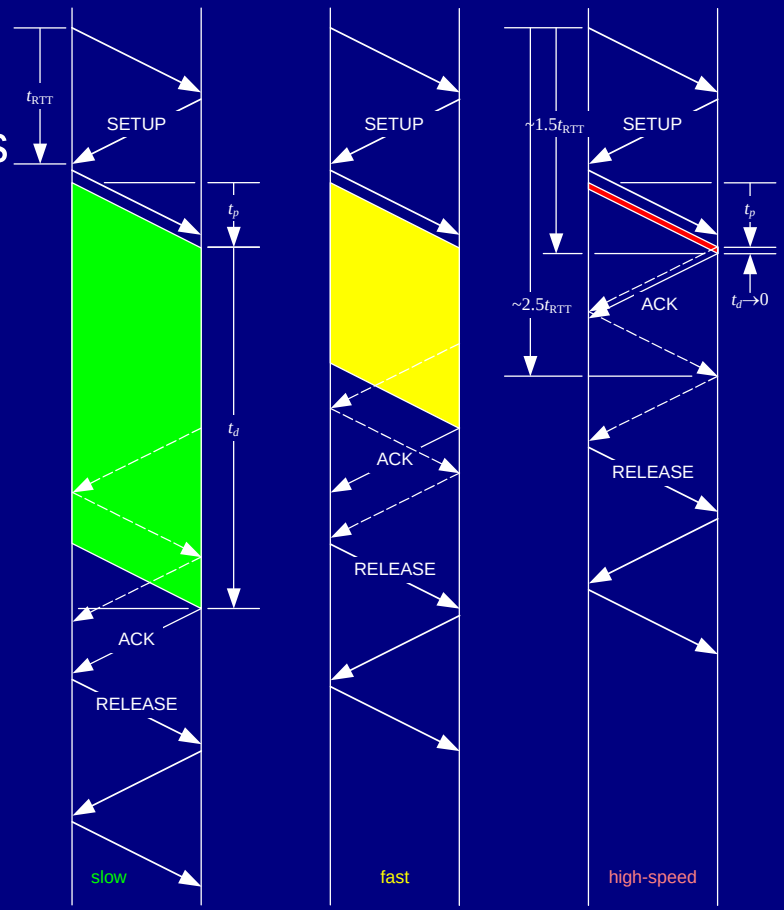
State Management
Transport and Network Connections

- L4 connections
  - required over
    - connectionless L3
    - path with any connectionless subnetwork
  - may exploit
    - connection-oriented L3



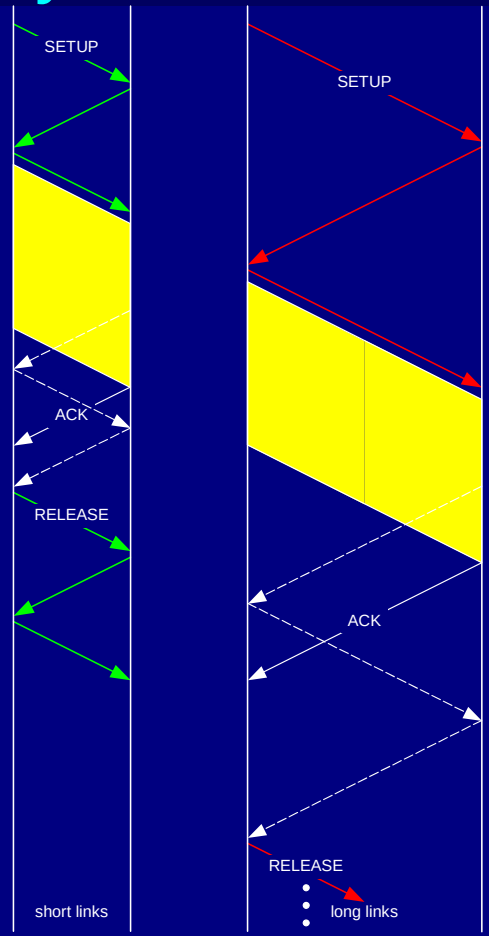
E2E State Management
Impact of High Speed

- Connection shortening
  - transmission delay decreases
  - signalling delay constant
- E2E signalling
  - significant fraction of time



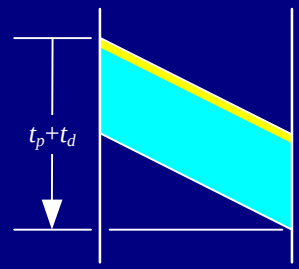
E2E State Management
Impact of Long Latency

- Connection lengthening
  - transmission delay increases
  - signalling delay increases
- E2E signalling
  - open state longer
  - denial of resource to others



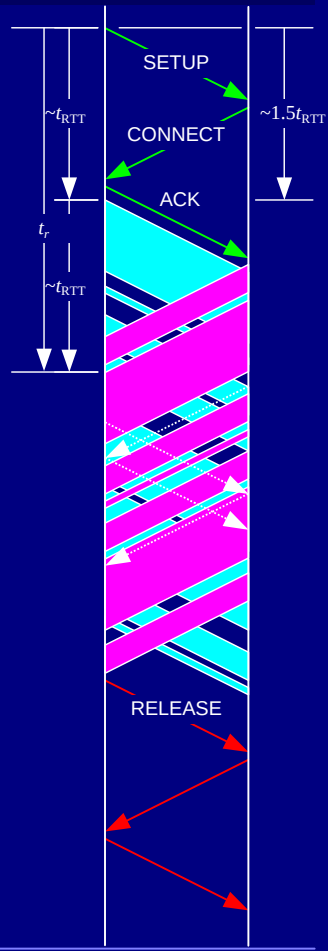
Transfer Modes
Datagram

- Independent packets
  - each has fully self-describing header
  - no network state needed to forward



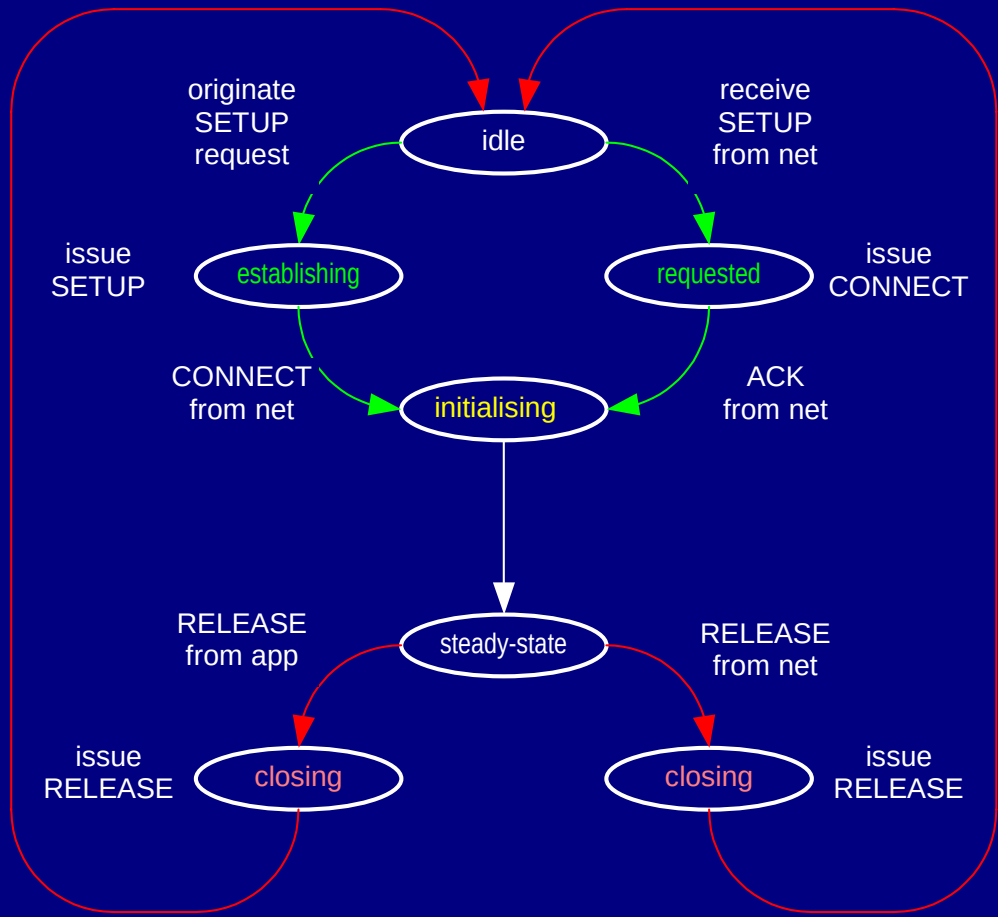
Transfer Modes
Connection

- Explicit connection setup
  - 3-way handshake
    SETUP / CONNECT / ACK
- Data flow
  - packets need connection or flow identifier
    - used by nodes to look up state
- Release of resources and state
  - explicit RELEASE
  - time out state



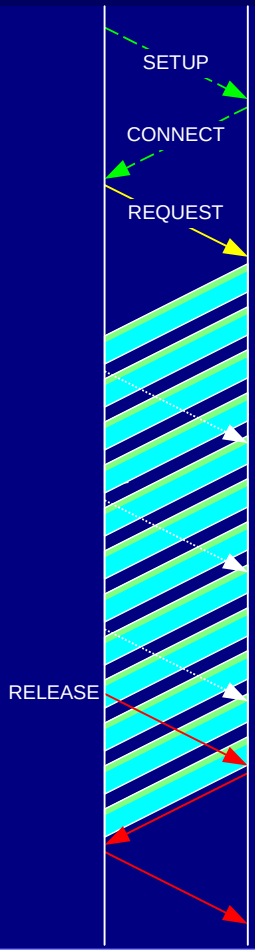
Transfer Modes
Connection

- Connection State Machine
  - idle
  - establishing
    - install state
  - initialising
    - state convergence
  - steady state
  - closing
    - release state

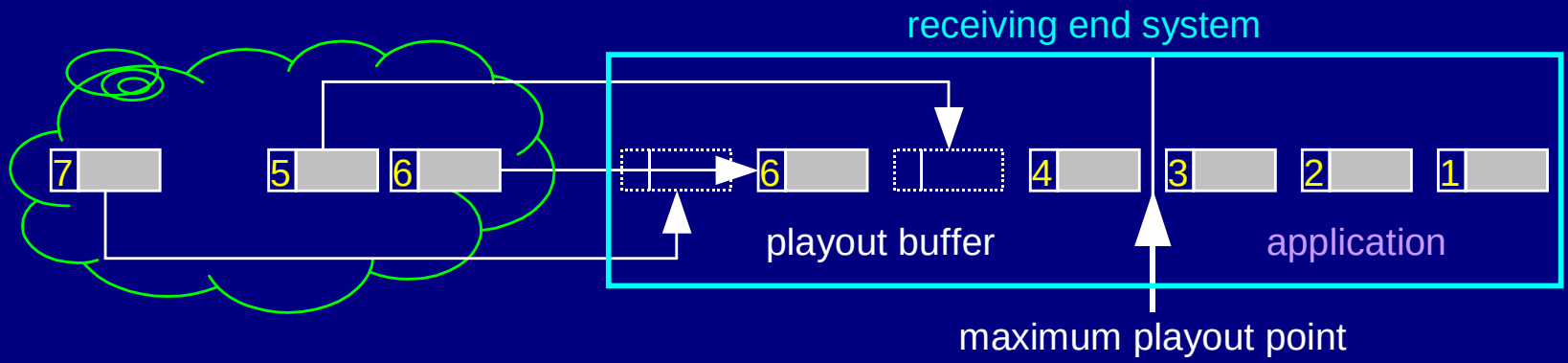


Transfer Modes
Stream

- Various mechanisms to start stream
  - explicit client request
  - server push
  - may or may not establish connection state
- Data flow
  - synchronisation and control
    - embedded or
    - out-of-band



Transfer Modes
Stream



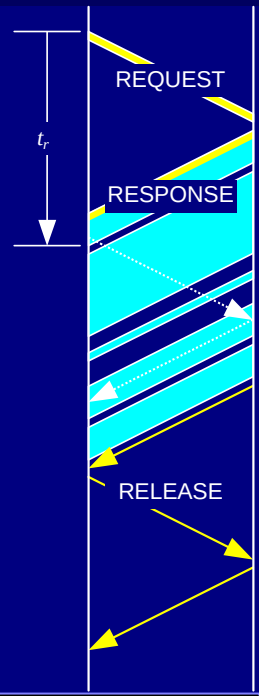
- Playback buffer to reorder and absorb jitter
  - adds delay

Continuous Media Streaming T-I.2

Timeliness of delivery should not be traded for reliable transfer in real-time continuous media streams. Use a playback buffer to reorder and absorb jitter.

Transfer Modes
Transaction

- Transaction request
  - may or may not establish connection state
  - explicit release of connection state optional
- Data returned
- Overlap to reduce latency
  - request/response with control



Minimise Transaction Latency T-6A

Combine connection establishment with request and state transfer to reduce end-to-end latency for transactions.

E2E State Management
Hard vs. Soft

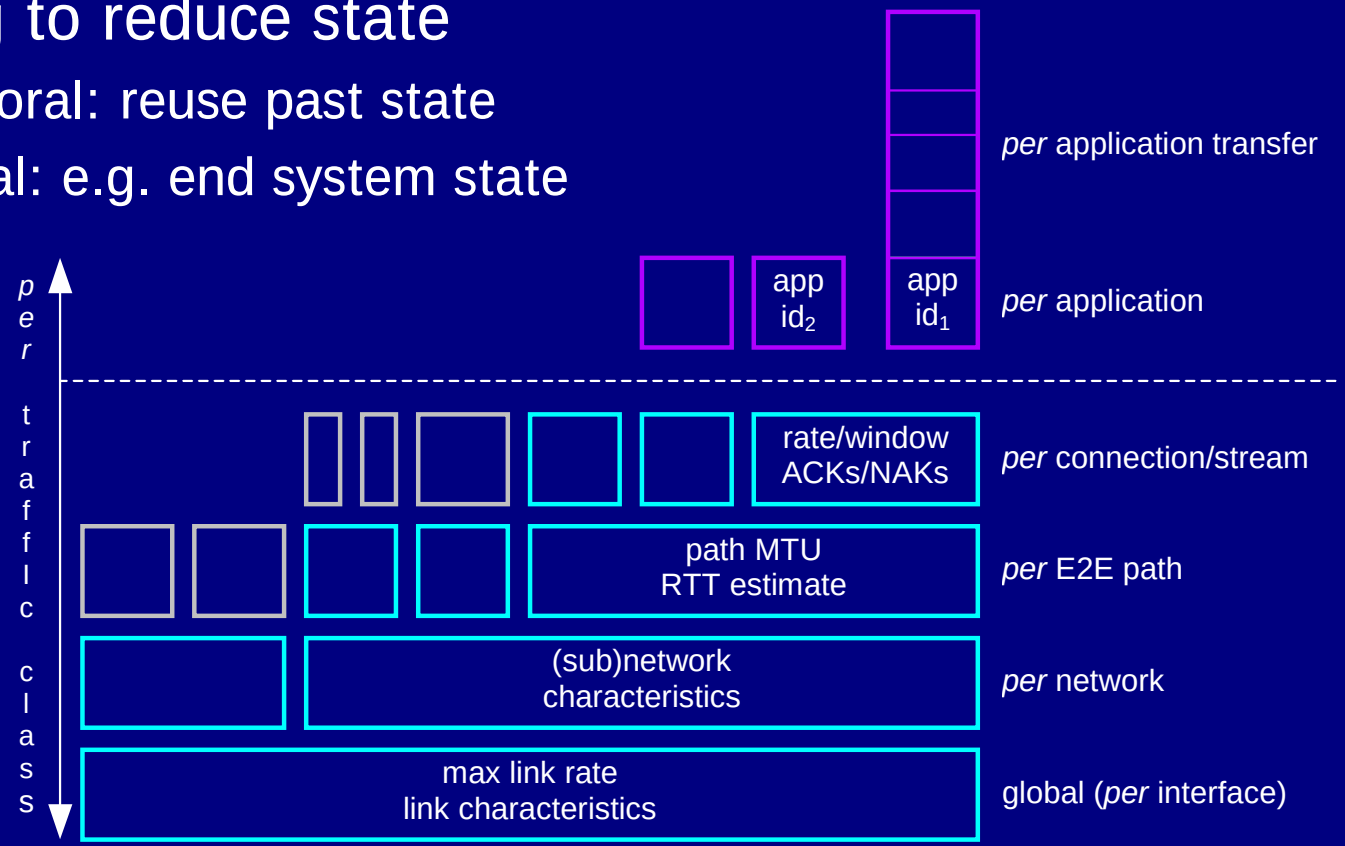
- Hard state
  - explicitly established and released
  - deterministic
  - delay to establish and memory to maintain
- Soft state
  - established and removed as needed and memory allows
  - robust to failures
  - if not explicitly established, delay to accumulate

Hard vs. Soft State T-5A

Balance the determinism and stability of hard state against the adaptability and robustness to failure of soft state.

E2E State Management
State Aggregation and Sharing

- Sharing to reduce state
  - temporal: reuse past state
  - spatial: e.g. end system state



E2E State Management
State Aggregation and Sharing

- Sharing of state
  - reduces granularity of individual entities
  - state shared is fate shared state

State Aggregation T-5B

Spatially aggregate state to reduce complexity, but balance against loss in granularity. Temporally aggregate to reduce or eliminate establishment and initialisation phase.
State shared is fate shared.

E2E State Management
Assumed Initial Conditions

- Rapid convergence to steady state
  - assume initial conditions
    - normal or common case
    - past behaviour
    - current network state

Use Initial Assumptions to Converge Quickly T-5E

The time to converge to steady state depends on the accuracy of the initial conditions, which depends on the currency of state information and the dynamicity of the network. Use past information to make assumptions that will converge quickly.

End-to-End Protocols
Framing and Multiplexing

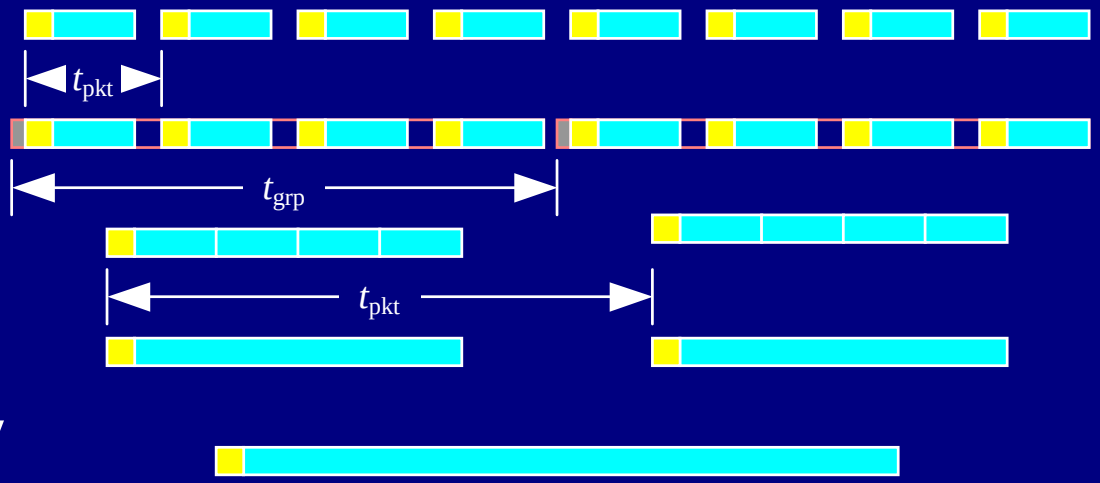
7.1 Functions and mechanisms
7.2 State management
7.3 Framing and multiplexing
- 7.3.1 Framing and fragmentation
- 7.3.2 Application layer framing
- 7.3.3 Multiplexing

7.4 Error control
7.5 Flow and congestion control
7.6 Security and information assurance

E2E Framing and Multiplexing

Packet Size and Control Overhead

- Small packets
  - large overhead
  - short interarrival
- Grouped
- Blocked
- Large
  - increased delay
- Jumbogram

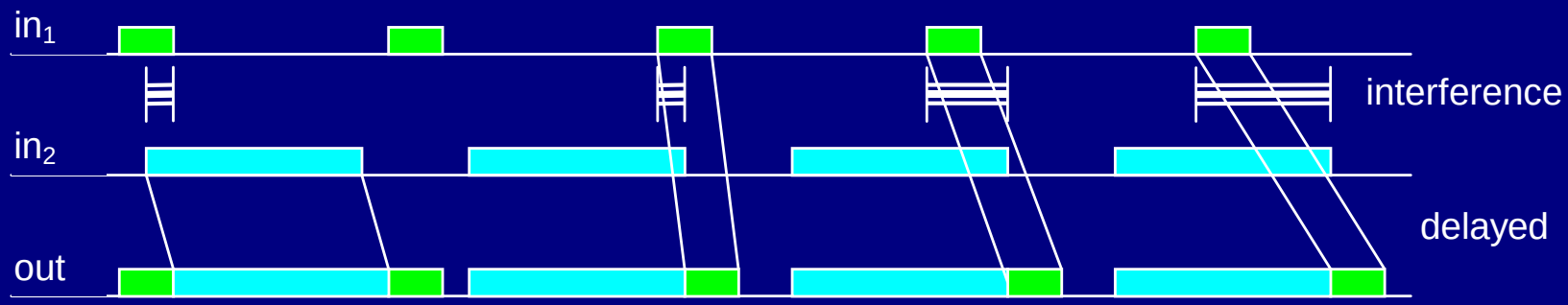


Balance Packet Size T-8A

Trade between control overhead and fine enough grained control. Choose size and aggregation methods that make sense end-to-end.

E2E Framing and Multiplexing

Packet Size and Control Overhead



- Large packets
  - impose jitter and delay on one another

E2E State Management
MTU and per Hop Fragmentation

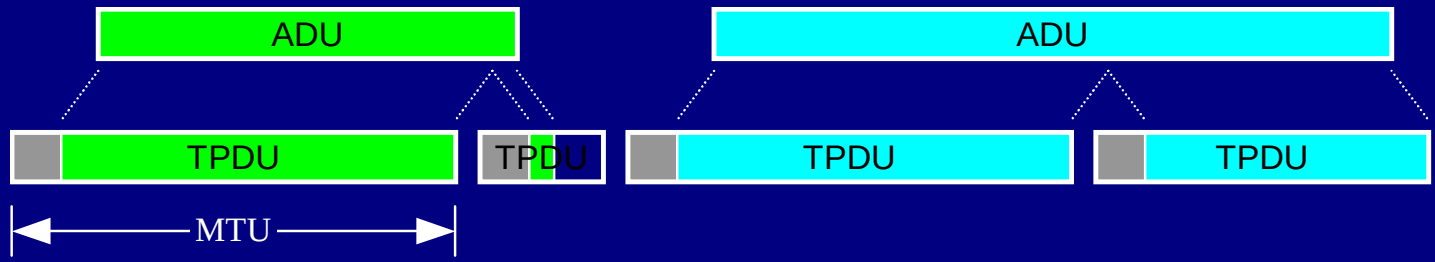
- Subnetworks have limits on packet size
  - MTU: maximum transfer unit
- If |TPDU| > min(MTU)
  - fragmentation will occur in the network
  - significant impact on packet processing rate and delay

Avoid Hop-by-Hop Fragmentation T-5F

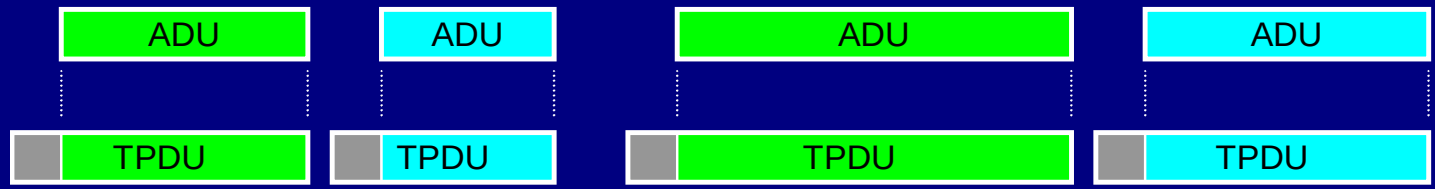
The transport layer should be aware of or explicitly negotiate the maximum transfer unit of the path to avoid the overhead of fragmentation and reassembly.

E2E Framing and Multiplexing

Packet Size and Control Overhead



transport layer fragmentation



application layer framing

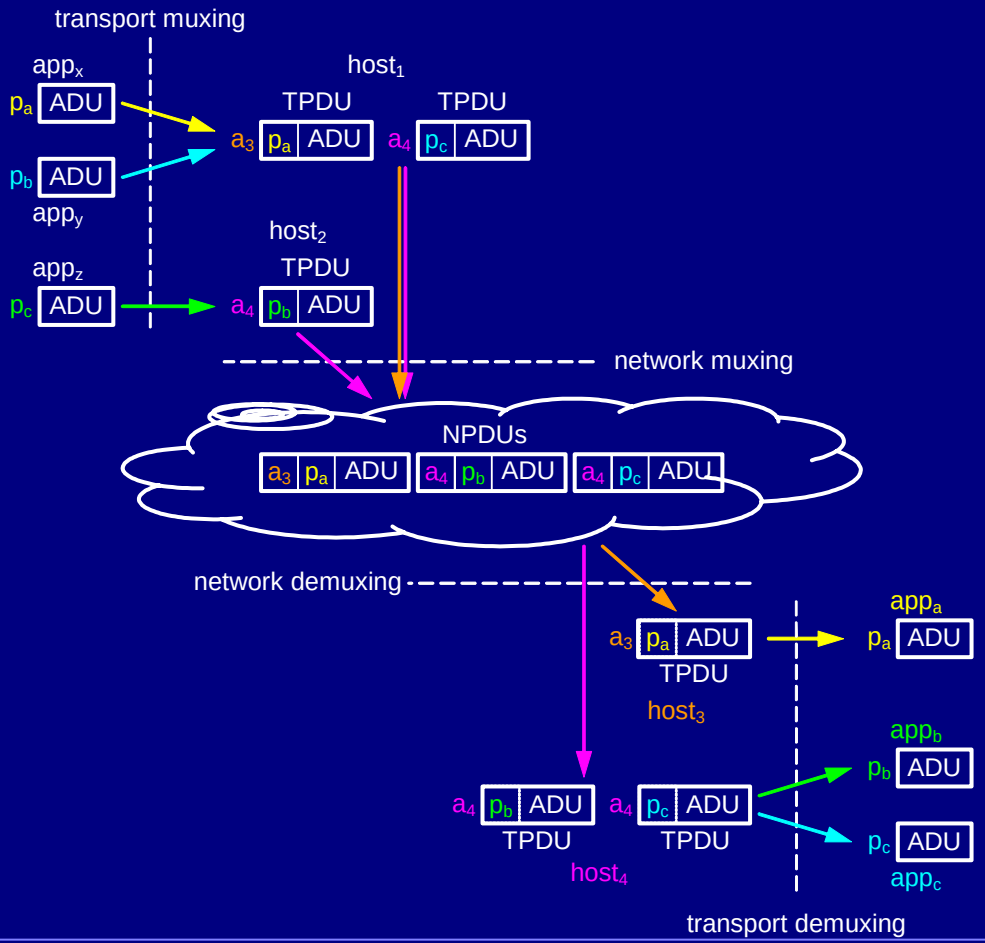
Application Layer Framing T-8A

To the degree possible, match ADU and TPDU structure. In cases where the application can better determine structure and react to lost and misordered ADUs, ALF should be employed.

E2E Framing and Multiplexing

Layered Multiplexing

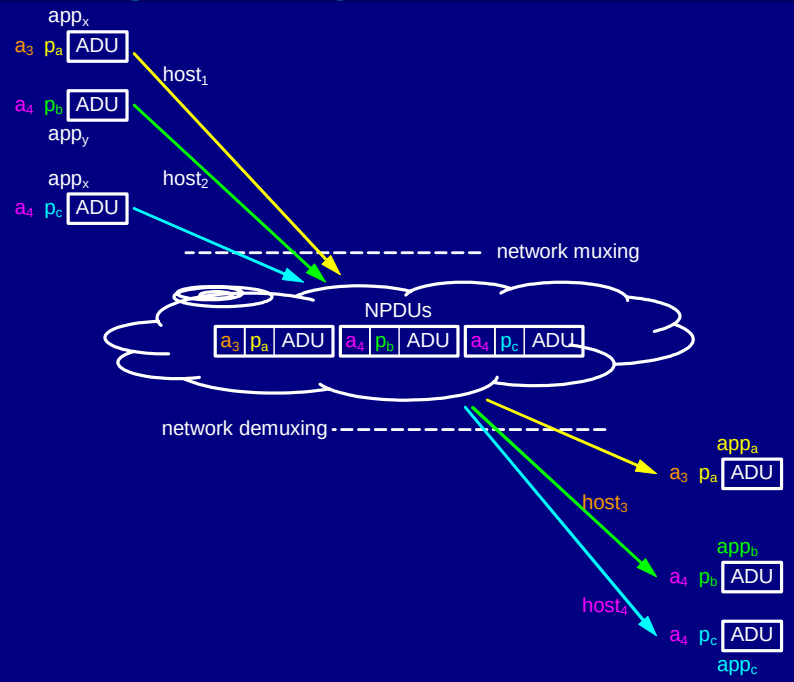
- Multiplexing
  - application
    - interapplication coordination
  - transport
    - applications sharing network interface
  - network
    - end systems sharing switch
  - MAC and link
    - interfaces sharing medium



E2E Framing and Multiplexing

Integrated Multiplexing

- Multiplexing
  - significant overhead
    - processing limits
    - delay
  - undesirable fate sharing
    - QOS
  - perform all layers at once
    - network layer for wired
    - MAC layer for wireless



Limit Layered Multiplexing T-4A

Layered multiplexing should be minimised and performed in a single integrated manner for all layers at the same time.

End-to-End Protocols
Error Control

7.1 Functions and mechanisms
7.2 State management
7.3 Framing and multiplexing
7.4 Error control
- 7.4.1 Types and causes of errors
- 7.4.2 Impact of high speed
- 7.4.3 Closed-loop retransmission
- 7.4.4 Open-loop error control

7.5 Flow and congestion control
7.6 Security and information assurance

E2E Error Control
Types and Causes of Errors$_1$

- Bit errors
  - wireless channels
  - buggy hardware
- Packet loss
  - network queue overflow
  - transport protocol packet discard when bit error
  - burst errors
  - drops due to checksum/CRC mismatch (result of bit error)
- Packet misordering
  - multiple paths through switch or network
  - path reconfiguration

E2E Error Control
Types and Causes of Errors$_2$

- Packet duplication
  - retransmission
- Packet insertion
  - undetectable header error
  - long packet life (two packets with same sequence number)
- Fragment loss
  - error multiplication:
    fragment loss requires entire frame loss or discard

E2E Error Control
Impact of High Speed

- Bandwidth-×-delay product increases
  - fixed duration error event larger relative impact
  - reaction to errors decreases in terms of number of bits sent
- Sequence numbers
  - sequence numbers wrap if sequence space too small
    - causes packet misinsertion

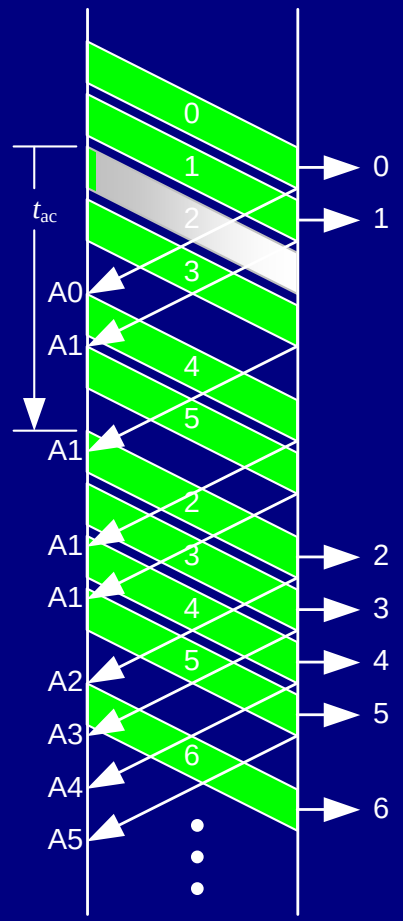
Packet Control Field Values T-8C

Optimise header control field values to trade efficiency against expected future requirements. Fields that are likely to be insufficient for future bandwidth-×-delay products should contain a scale factor.

E2E Error Control

Closed Loop Retransmission: Go-Back-*n*

- Sliding window
  - o packets are *sequentially* acknowledged
  - o previous ACK used for
    - out of sequence packet
    - next packet after loss
  - o sender timer fires if ACK not received
    - reset transmission beginning at lost packet
  - – significant loss penalty for high bw-×-delay
    - go back and retransmit *all* since loss
    - many unneeded retransmissions
    - significant additional delay



E2E Error Control

Closed Loop Retransmission: Go-Back-*n*

- Fast retransmit
  - o assume several duplicate ACKs are loss
  - o tell sender to go-back-*n*
  - \+ recovers more quickly (don't need to wait for timer to fire)
- Delayed ACK
  - o aggregate several ACKs
  - \+ reduces ACK traffic
  - \+ allows some receiver resequencing (within ACK aggregation)

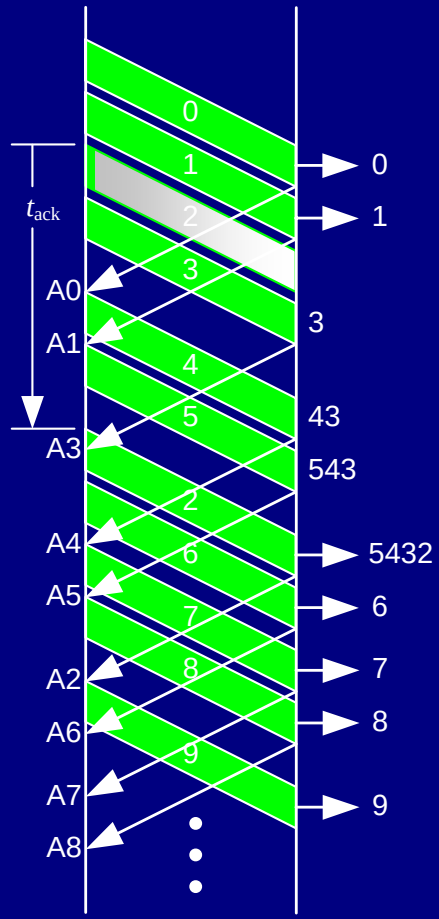
Decouple Error, Flow, and Congestion Control T-6E

Exploit selective acknowledgements and open-loop rate control to decouple error, flow, and congestion control mechanisms.

E2E Error Control

Closed Loop Retransmission: Selective Repeat

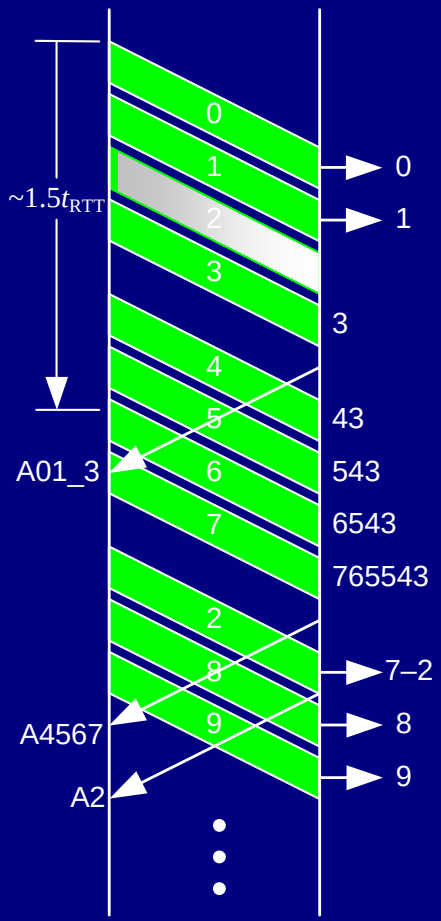
- All packets acknowledged
- Missequenced packets
  - o reordered at receiver
  - – increases receiver complexity
- Lost packets
  - o selectively retransmitted
  - + no go-back-*n* latency penalty
  - – requires more receiver buffer space
- Latency and Bandwidth reduced



E2E Error Control

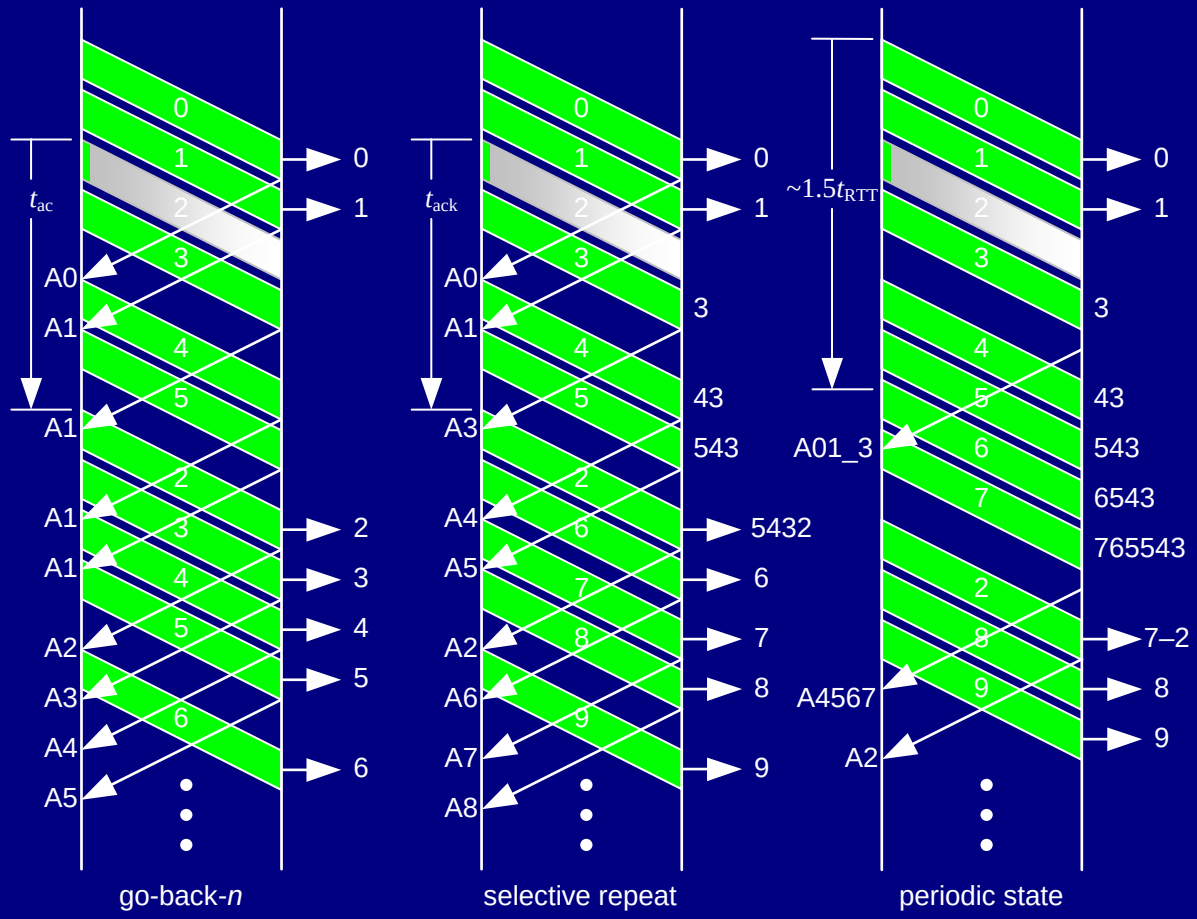
Closed Loop Retransmission: Periodic State

- Selective ACK blocks
  - o bit vectors can aggregate ACKs
  - \+ significant reduction in control packets
  - \+ simple receiver implementation possible



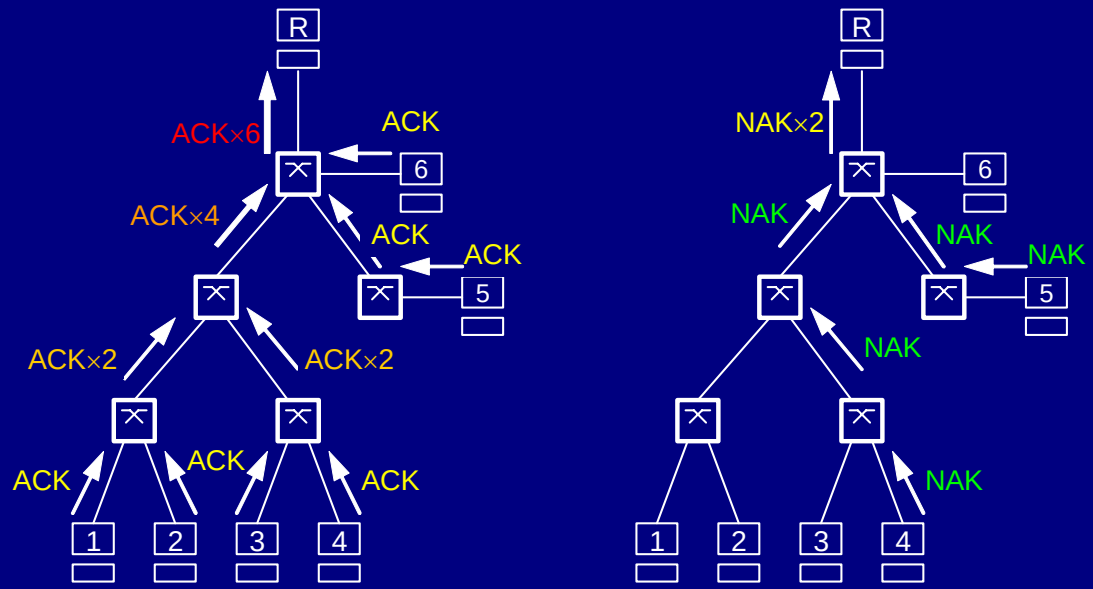
E2E Error Control

Closed Loop Retransmission: Comparison



E2E Error Control

Closed Loop Retransmission: Multicast



- ACK implosion: ACK suppression needed
  - o NAKs (negative ACKs) with liveness polling
  - o localised hop-by-hop buffering and retransmission
  - – significant reduction of traffic on relatively reliable links

Error Control

Example: 7.6 SNA Error Control

- SNA (IBM Systems Network Architecture)

Scope	Layer	Name	Reliability
hop-by-hop	link	data link control	yes
end-to-end	transport	path control	no

- Not in agreement with end-to-end arguments, but...
  - engineered reliability of store-and-forward hosts
    - no part of data path not covered by ECC: processor + memory

E2E Error Control
Open Loop

- Open loop error control
  - o some applications do not need guaranteed reliability
    - applications that are tolerant to some loss
    - real-time or interactive delay requirements
  - – eliminates control loop latency for recovery
  - \+ requires additional end-systems bandwidth & processing
  - \+ requires additional network bandwidth
- Forward error correction
  - – block codes: per block recovery
  - – convolutional codes: continuous over window

E2E Error Control
Hybrid Open/Closed-Loop Control

- Hybrid open loop with closed loop when necessary
  - statistical error bound with FEC
  - retransmission only when FEC doesn't allow correction
  - appropriate for
    - high latency and bandwidth-×-delay paths
    - asymmetric bandwidth paths

Forward Error Correction T-6D*e*

Use FEC for low-latency, open-loop flow control when bandwidth is available and statistical loss tolerance bounds are acceptable to the applications.

End-to-End Protocols
Flow and Congestion Control

7.1 Functions and mechanisms
7.2 State management
7.3 Framing and multiplexing
7.4 Error control
7.5 Flow and congestion control
- 7.5.1 Impact of high speed
- 7.5.2 Open-loop rate control
- 7.5.3 Closed-loop flow control
- 7.5.4 Closed-loop congestion control
- 7.5.5 Hybrid flow and congestion control

7.6 Security and information assurance

E2E Flow and Congestion Control
Definitions

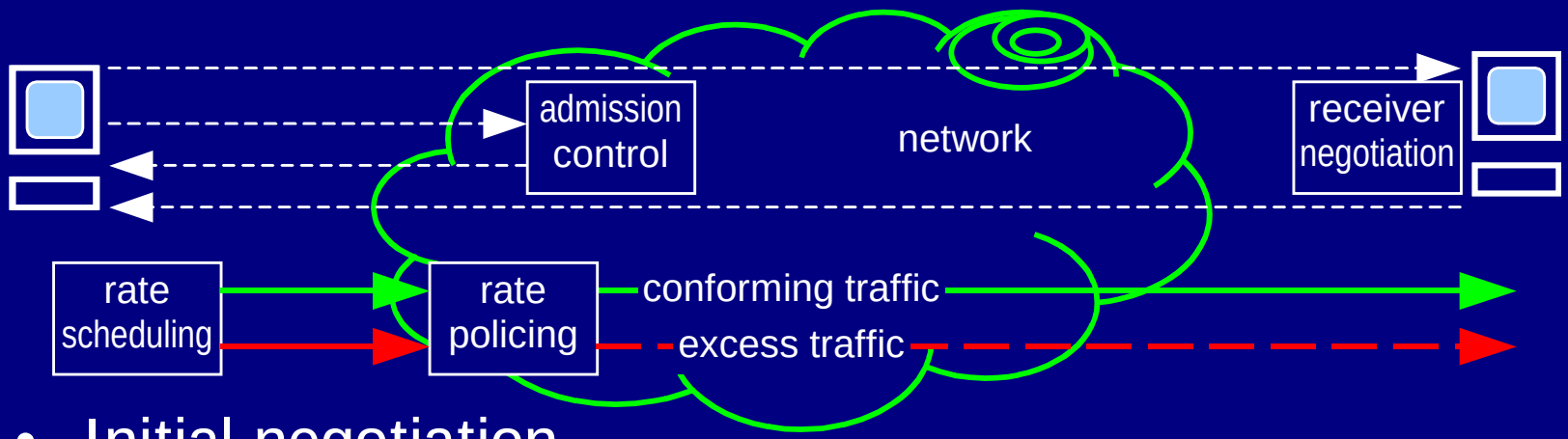
- Flow control
  - control transmission to avoid overwhelming *receiver*
- Congestion control
  - control transmission to avoid overwhelming *network* paths
- Types
  - *explicit* relies on congestion or flow information from nodes
    - allows decoupling of error, flow, and congestion mechanisms
  - *implicit* assumes loss is congestion
    - bad assumption in wireless networks (see [Krishnan 2004])

E2E Flow and Congestion Control
Impact of High Speed

- Bandwidth-×-delay product increases
  - response to an event happens after more bits transferred
  - it may be too late to react
    - all bits causing problem may already be transmitted
    - other cause of problem may have gone away

E2E Flow and Congestion Control

Open-Loop Rate Control



- Initial negotiation
  - *admission control* limits traffic to available resources
  - receiver negotiates what it can accept (flow control)
- Steady state
  - *policing* enforces traffic contract from transmitter
    - excess traffic may be marked and passed if capacity available

E2E Flow and Congestion Control
Open-Loop Rate Control

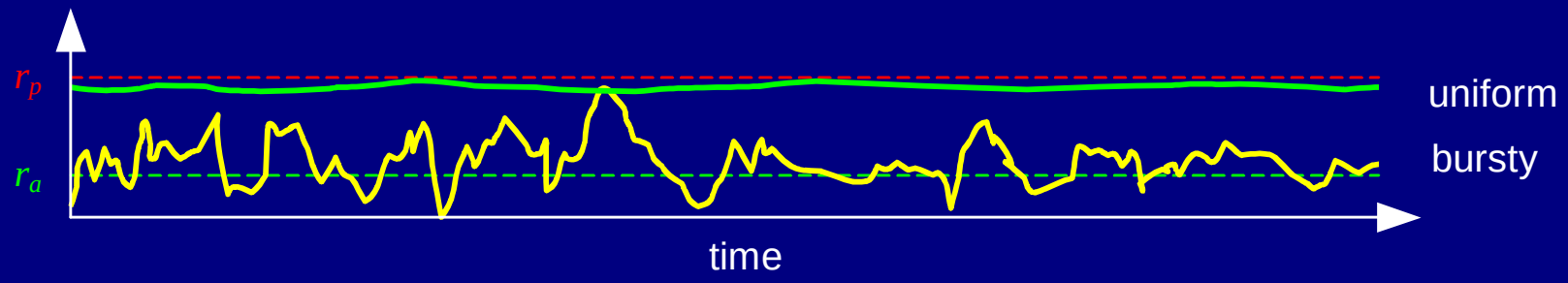
- Requires connection establishment
  - – overhead and delay for connection setup
    - optimistic establishment or fast reservations ameliorate
  - \+ QOS guarantees possible in steady state
  - \+ feedback control loops not needed during transmission
  - \+ network stability less dependent on congestion control
    - unless resources significantly overbooked

Use Knowledge of Network Paths for Open-Loop Control T-6D*o*

Exploit open-loop rate and congestion control based on a priori knowledge to the degree possible to reduce the need for feedback control.

E2E Flow and Congestion Control

Open-Loop Rate Control: Parameters



- Main parameters
  - peak rate r_p
  - average rate r_a
  - burstiness (peak to average ratio or max burst size)
- Derived parameters
  - jitter

Open Loop Rate Control

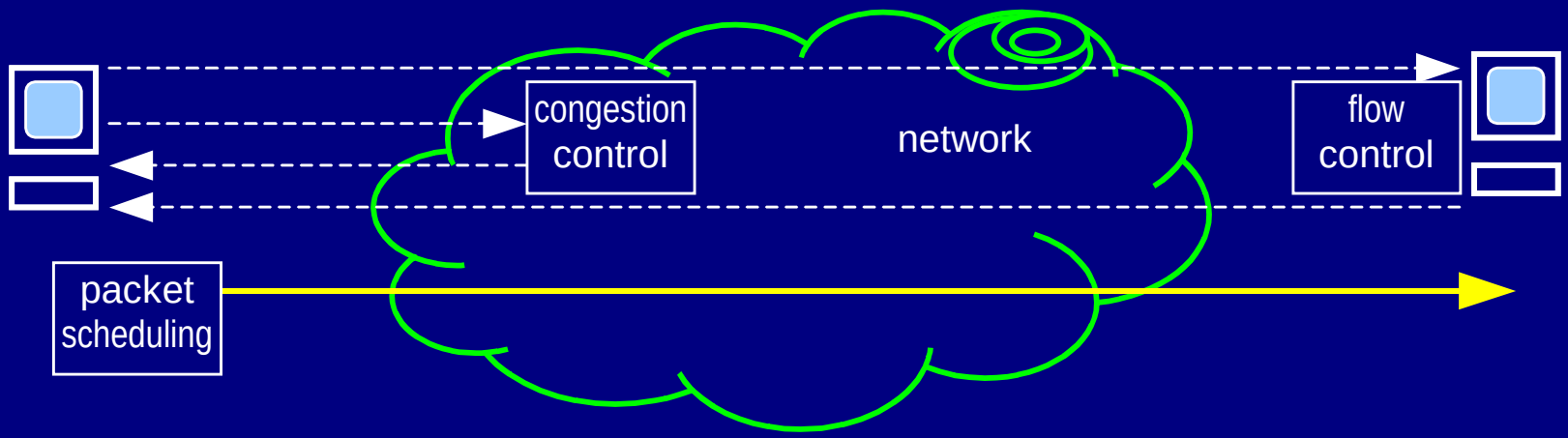
Example: ATM Traffic Classes 7.7

Class	Name	Control	Traffic	Parameters
CBR	constant bit rate	open loop	real time	peak rate
rt-VBR	real-time variable bit rate	open loop	data	peak and average rate burst, jitter, latency
nrt-VBR	non-real-time variable bit rate	open loop	data	peak and average rate burst
ABT	ATM block transfer	fast reservation open loop	data	peak and average rate burst
ABR	available bit rate	hybrid	data	minimum rate peak rate*
GFR	guaranteed frame rate	open loop	data	minimum rate peak rate*
UBR	unspecified bit rate	none	best effort	peak rate*

* specified in SETUP but not guaranteed

E2E Flow and Congestion Control

Closed-Loop Flow and Congestion Control



Use Closed-Loop Congestion Control to Adjust to Network Conditions

T-6D*c*

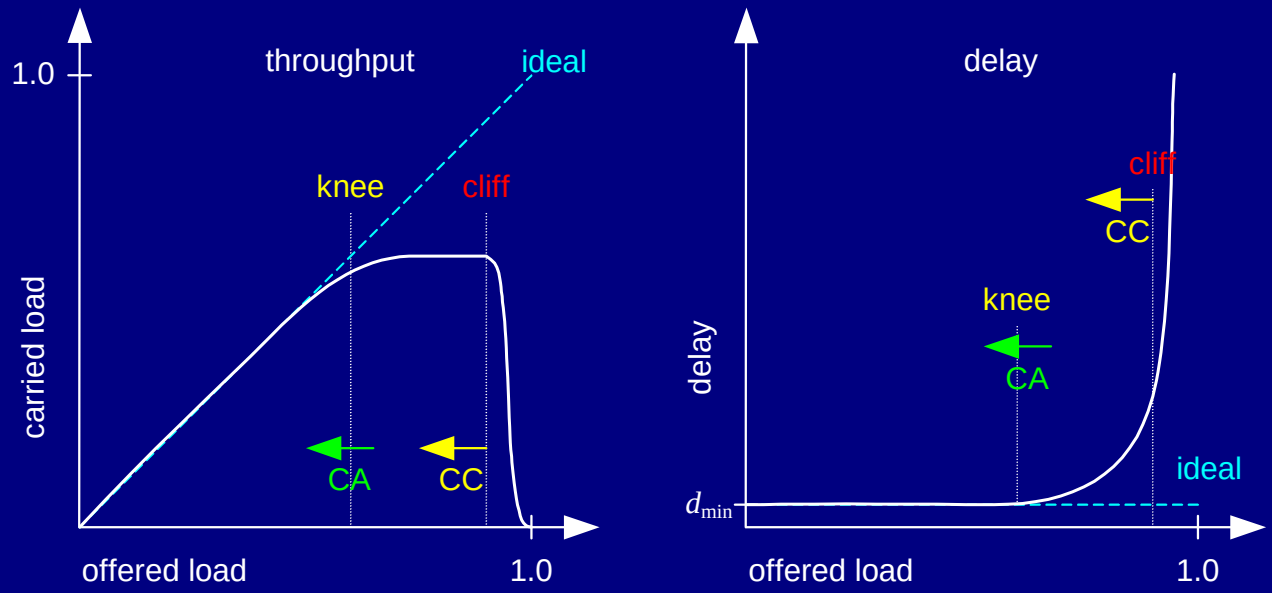
Closed-loop feedback is needed to adjust to network conditions when there are no hard reservations.

E2E Flow and Congestion Control
Closed-Loop Flow Control

- Feedback from *receiver* to limit rate
- Window to limit amount of unacknowledged data
  - static window
  - dynamic window
    - combined with congestion control

E2E Flow and Congestion Control

Congestion Avoidance and Control



Congestion Avoidance

T-II.4*c*

Congestion should be avoided before it happens. Keep queues from building and operate just to the left of the knee to maximise throughput and minimise latency.

E2E Flow and Congestion Control
Closed-Loop Congestion Control

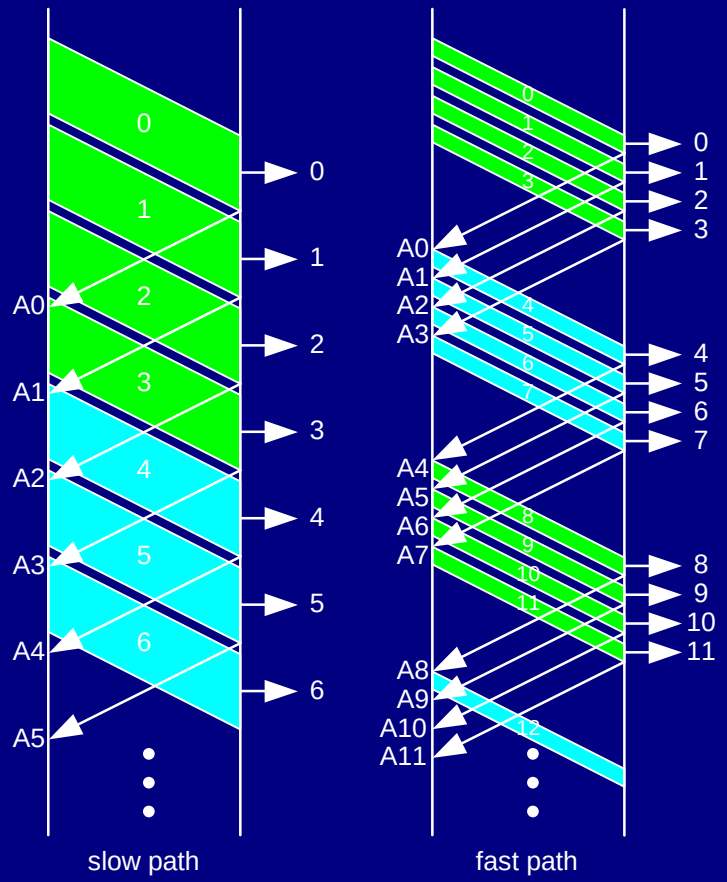
- Feedback from *network* to limit rate
- Window to limit amount of unacknowledged data
  - dynamic window
  - conservation of packets in the network
  - self clocking
- Required unless open loop with hard reservations

Use Closed-Loop Congestion Control to Adjust to Network Conditions T-6D*c*

Closed-loop feedback is needed to adjust to network conditions when there are no hard reservations.

E2E Flow and Congestion Control
Closed-Loop Congestion Control

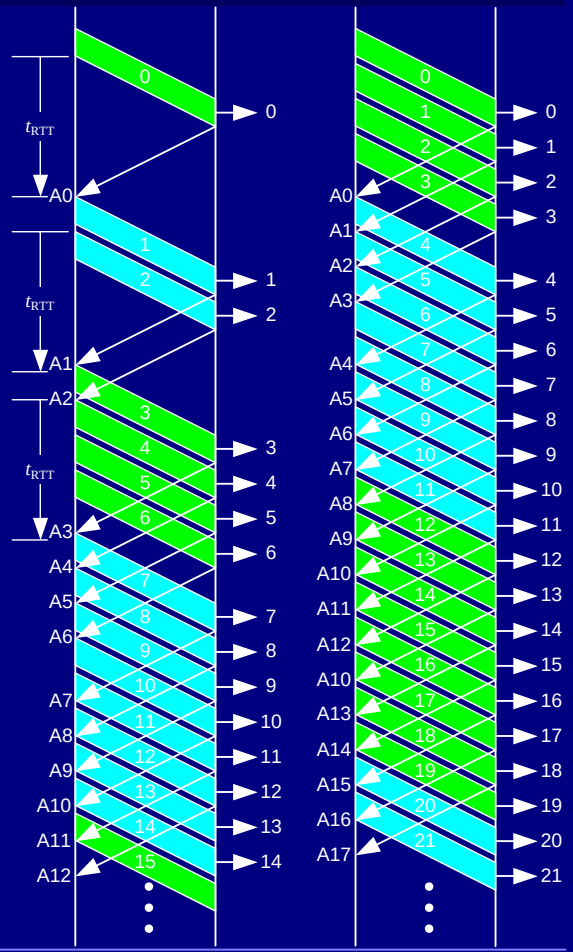
- Window size critical
  - big enough to fill pipe



E2E Flow and Congestion Control

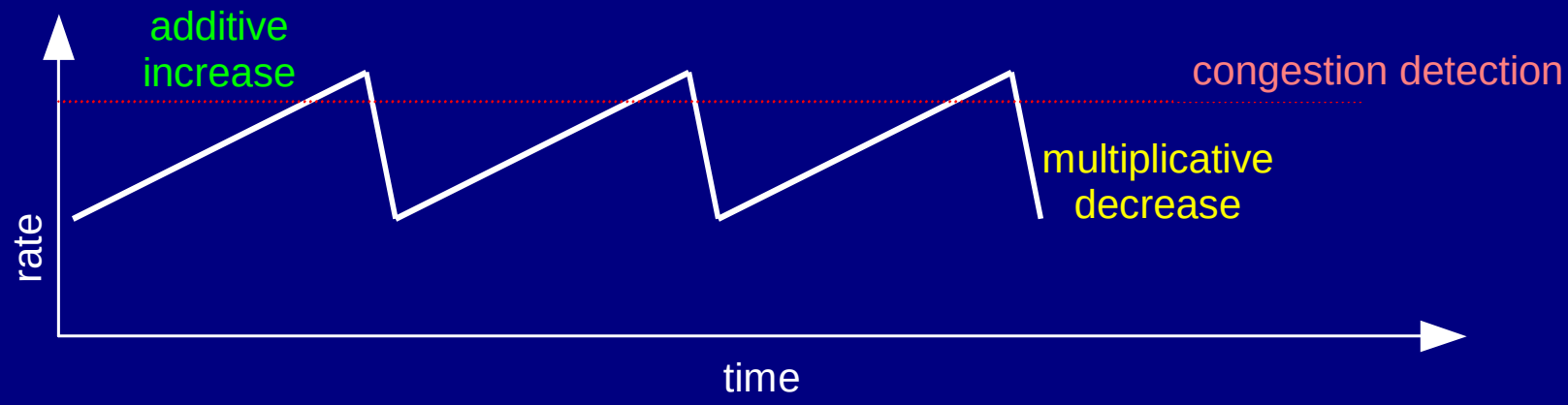
Closed-Loop Congestion Control: Initialisation

- Slow start initialisation
  - increase window until path loaded
- Critical parameters
  - initial window size
  - rate of increase
- Tradeoffs
  - conservative on high bw-×-delay:
    - multiple round trips
    - never get to full rate for transactions
  - aggressive:
    - induced congestions



E2E Flow and Congestion Control

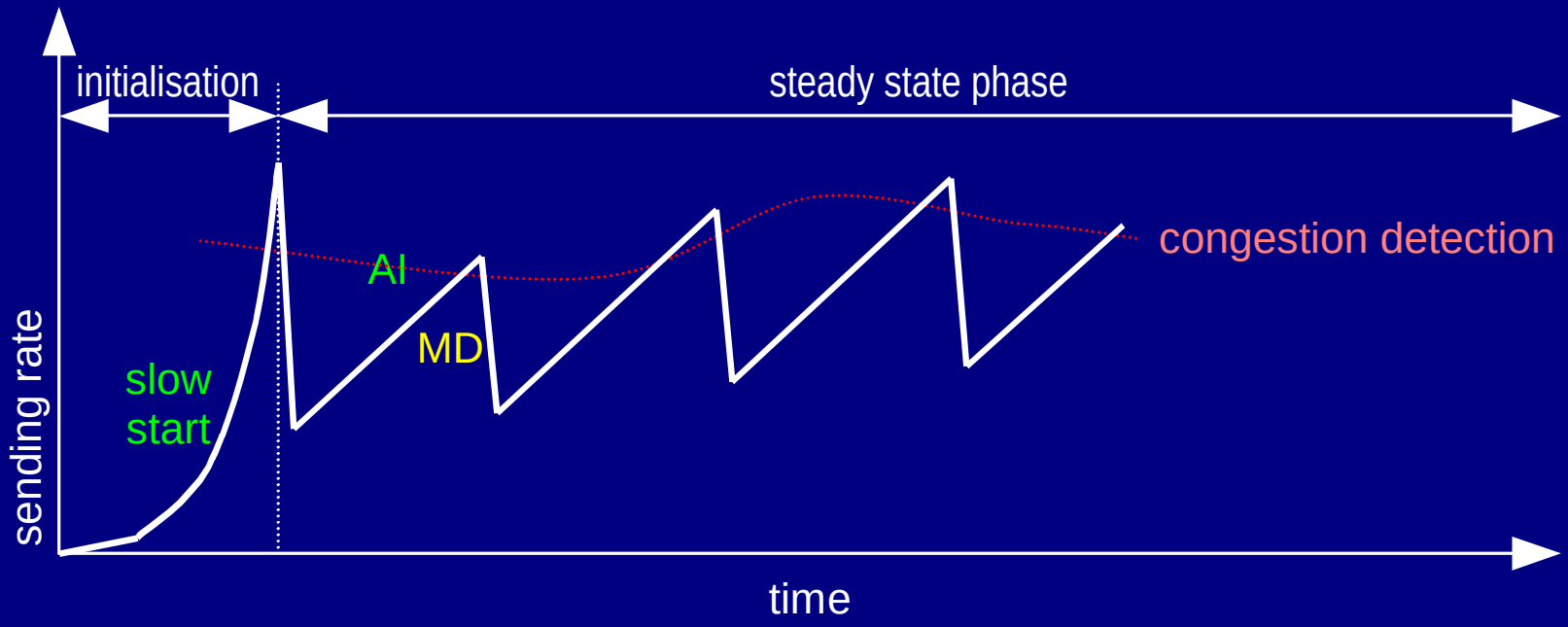
Closed-Loop Congestion Control: Steady State



- AIMD steady state
  - additive-increase slowly increases rate
    - increment window
  - multiplicative-decrease quickly throttles with congestion
    - divide window
  - RED attempts to reduce when congestion impending

E2E Flow and Congestion Control

Closed-Loop Congestion Control



- Initialisation phase: slow start
- Steady-state phase: AIMD

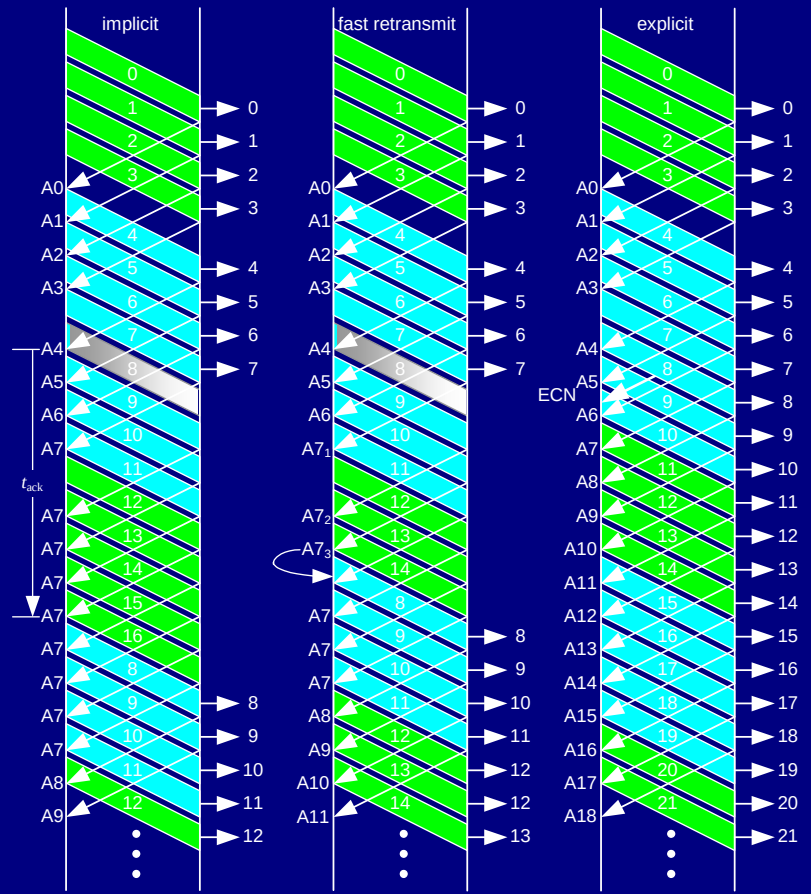
E2E Flow and Congestion Control
Closed-Loop Congestion Control: Steady State

- Implicit control
  - missing ACK assumed to be congestion
  - reasonable when all losses are due to congestion
    - fiber optic channels connected by reliable switches
  - performs poorly when significant losses in channel
    - mobile wireless links
    - under-provisioned CDMA (including optical CDMA)
- Explicit control
  - explicit congestion notification (ECN)
    - throttle with ECN message
  - some decoupling of error and congestion control
    - throttle before packet loss
    - not sufficient for lossy wireless link

E2E Flow and Congestion Control

Closed-Loop Congestion Control: Steady State

- Implicit control
  - standard
  - with fast retransmit
- Explicit control
  - ECN



E2E Flow and Congestion Control

Example$_{7.8}$ TCP Congestion Control

- Fast recovery
  - 1/2 congestion window after 3dupACK rather than slow start
- Partial acknowledgement response [Hoe 1996]
  - 1/2 window reduction only once with partial retransmit ACK
  - rather than per packet
- RED: random early detection (discard) [Floyd 1993]
  - discard packets when router queue threshold exceeded
  - throttle TCP source earlier before congestion occurs
- ECN: explicit congestion notification [Floyd 1994]
  - use IP ECN to trigger multiplicative decrease

E2E Flow and Congestion Control

Example$_{7.8}$ TCP Congestion Control

- Research optimisations [jury still out]
  - Pacing [Visweswaraiah 1997]
    - spread segment transmissions
    - rather than burst
  - Rate control [Padhye 1998]
    - rate based on equations that describe TCP behaviour
    - can also make UDP transmission “TCP friendly”
  - ETEN: explicit transport error notification [Samaraweera 1997]
    - signal loss due to channel bit errors
    - loss of ACK is not misconstrued as congestion

E2E Flow and Congestion Control

Example 7.8 TCP Implementations

- Tahoe
  - slow start and congestion avoidance algorithms
  - fast retransmit after triple duplicate ACKs
- Reno – widely implemented
  - Tahoe +
  - fast recovery
- NewReno [Floyd 1999]
  - Reno +
  - partial ACK recovery
- Vegas [Brakmo 1995] (research implementation)
  - RTT estimate rather than AIMD

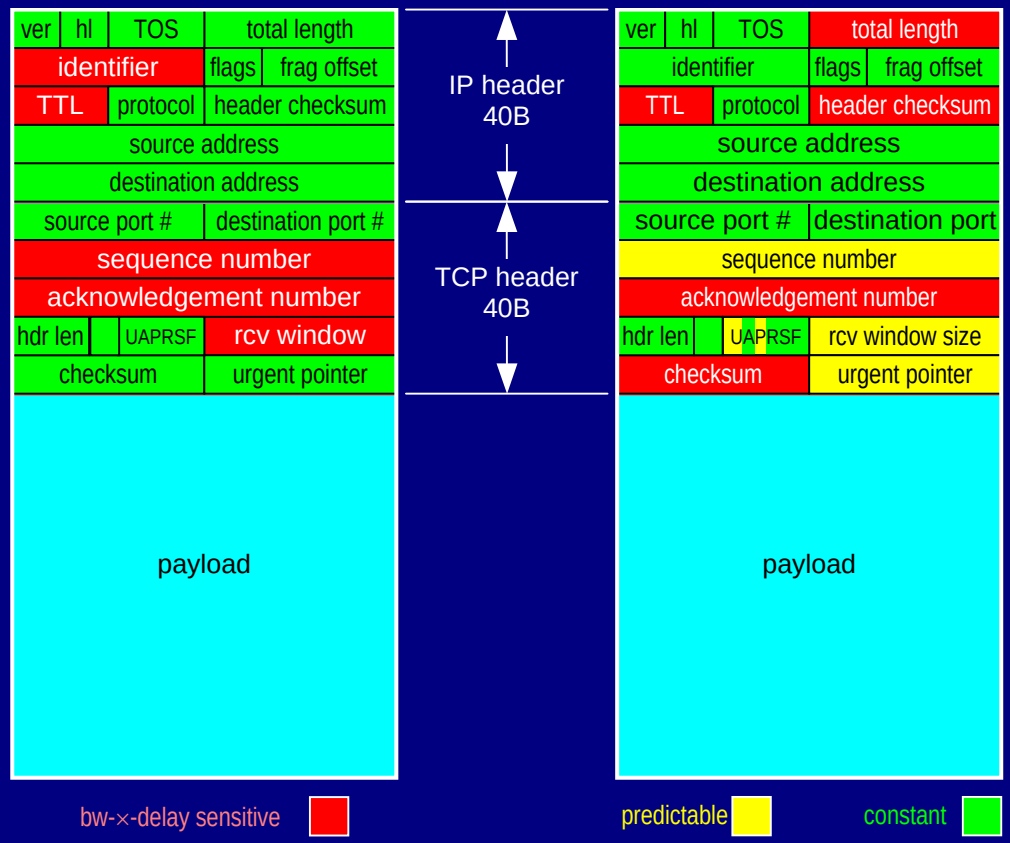
E2E Flow and Congestion Control
Hybrid Control

- Dynamic rate control
  - open-loop rate control modified by network feedback
    - example: ATM ABR
- Pacing to reduce burstiness
  - sender base rate control augments closed-loop control
  - window transmission spread over RTT
  - options
    - pace initial window, allow ACK self clocking to take over
    - periodic repacing to compensate for ACK compression
    - continuous pacing

E2E Protocols

Example 7.9 TCP High-Speed Optimisations

- TCP/IP headers
- bw-×-delay
  - fields that limit
    - sequence space
    - timer related
    - window size
- Field predictability
  - use template for
    - constant
    - predictable
  - must compute
    - highly variable



E2E Protocols

Example 7.9 TCP High-Speed Optimisations

- Common optimisations [RFC 1323]
  - Window scale option [RFC 1323]
    - SYN option power-of-2 multiplier to initial window
  - RTTM (round-trip time measurement) [RFC 1323]
    - timestamp to allow RTT measurement
  - PAWS (protection against wrapped seq. nos.) [RFC 1323]
    - 32-bit timestamp augments 32-bit sequence number
  - SACK (selective acknowledgement) [RFC 2018]
    - byte range header options for 3 – 4 selective ACK ranges
  - Fast retransmit [RFC 2581]
    - retransmit on triple duplicate ACKs without waiting for timer

E2E Protocols

Example$_{7.9}$ TCP High-Speed Optimisations

- Experimental optimisations
  - larger initial window ≈ 4KB [RFC 3390]
  - concern about aggressiveness
- Protocol Research
  - trailer checksum [Bridges 1994]
    - hotly debated ID
    - dismissed by IETF community
    - question of performance gain vs. implementation pain
  - pacing [Partridge 2001]
    - spreads window transmission within RTT
    - bandwidth estimation and RTT measurement

E2E Protocols

Example$_{7.9}$ TCP High-Speed Optimisations

- Implementation optimisations
  - header prediction [Jacobson 1990, Pink 1994]
- Implementation research
  - TCP/IP ILP [Clark 1990]
    - complex interactions with cache

End-to-End Protocols
Security and Information Assurance

7.1 Functions and mechanisms
7.2 State management
7.3 Framing and multiplexing
7.4 Error control
7.5 Flow and congestion control
7.6 Security and information assurance
- 7.6.1 End-to-end security
- 7.6.2 High-speed security

E2E Security
HBH vs. E2E

- Security and information assurance *must* be E2E
  - information in the clear inside network nodes *not* secure
- Justification for HBH security mechanisms
  - link security may be good enough for *some*
    - wireless link encryption for WEP (wire equivalent protection)
      note: 802.11 WEP not strong enough
  - subnetwork or edge-to-edge security for VPNs
    - assures enterprise security across open network...
      ...but not individual flow security

E2E Security
High-Speed Security

- Session establishment
  - significant overhead
  - may involve third party: CA (certificate authority)
  - application dictates startup delay bounds
    - interactive: 100 ms – 1 s target
- Encryption

E2E Security
High-Speed Security

- Session establishment
- Encryption
  - per byte critical path operation with significant complexity
  - algorithm and mode has significant impact on performance
  - end-to-end cryptographic synchronisation may be required

Critical Path Optimisation of Security Operations T-6D*c*

Encryption and per packet authentication operations must be optimised for the critical path.