TCP Connection

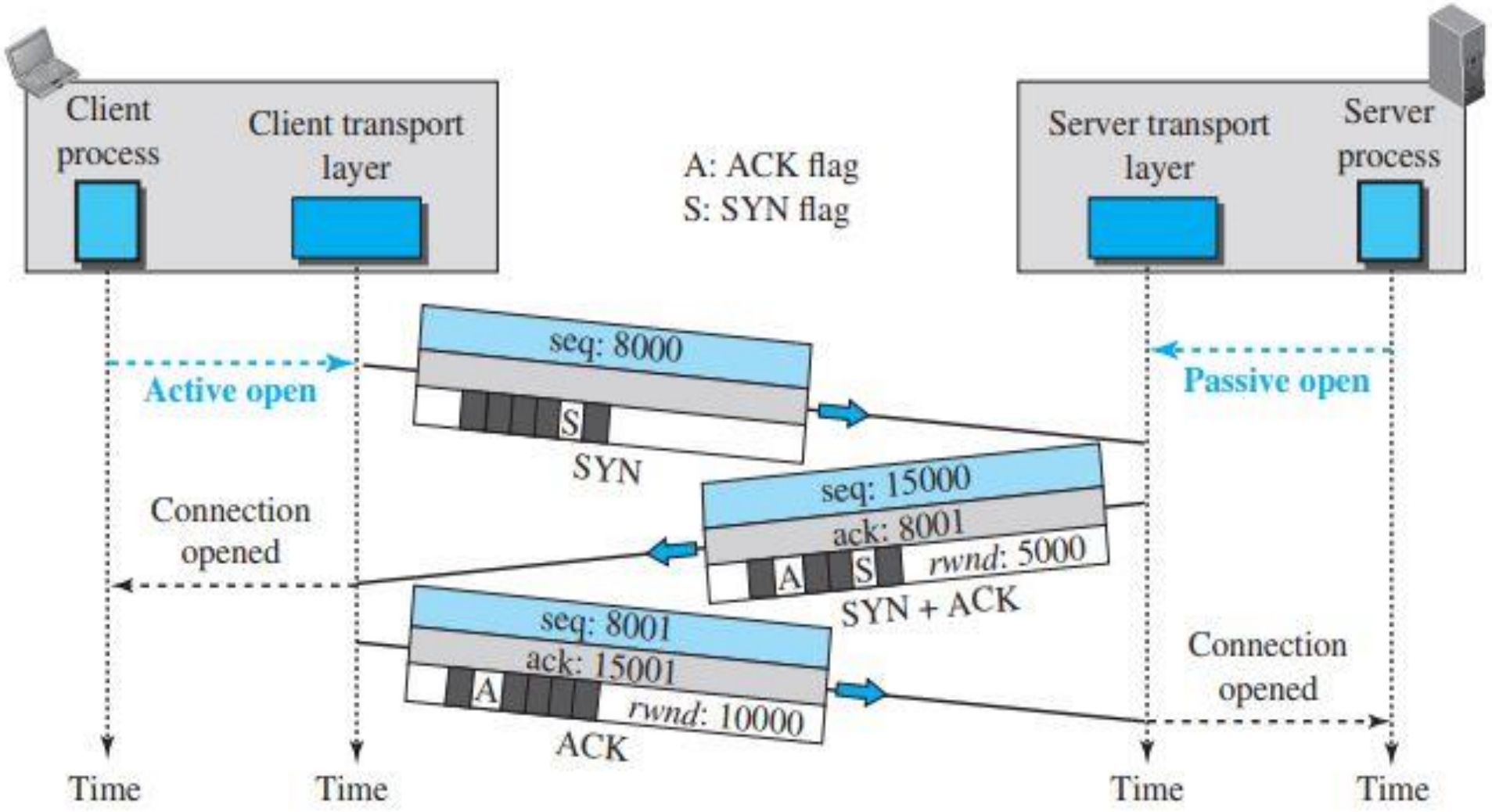

Fig 1. TCP Connection

TCP Data Transfer

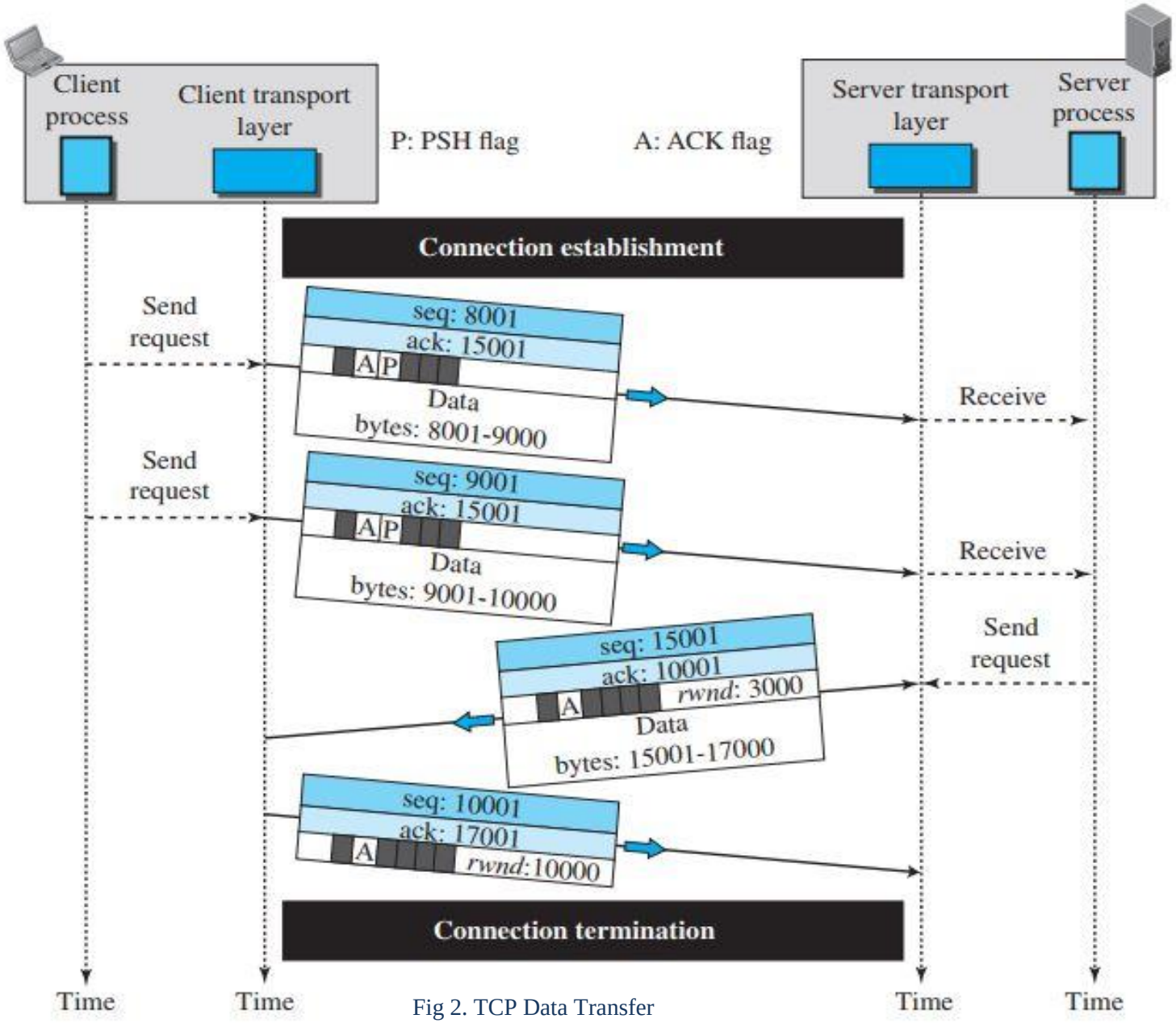

Fig 2. TCP Data Transfer

Pushing Data

- Application program from sender wants to receive immediate response
- Delayed transmission and delayed delivery is not allowed
- Application program at the sender request a push operation
- Sending TCP would not wait for the window to be filled
- It must create the segment and send it immediately
- Sending TCP must set the push bit to let the receiver know that the data has to be delivered as soon as possible and not to wait for more data to come

Urgent Data

- TCP is stream-oriented protocol which means data from application program to TCP are presented as stream of bytes
- There are occasions where application program needs to send urgent bytes
- Sending application program tells the sending TCP that the piece of data is urgent using ***URG*** bit as set
- TCP insert the urgent data at the beginning of the segment and rest of the segment can contain normal data from the buffer
- The urgent pointer field in the header defines the end of the urgent data
- ***For example, if the segment sequence number is 1200 and the value of urgent pointer is 300, the first byte of urgent data is 1200 and last byte will be 1499 and the rest of the bytes in the segment is non-urgent***

Urgent Data

- TCP wants some portion of the byte stream to be given special treatment
- Receiver TCP delivers bytes (urgent or non-urgent) to the application program in order
- Receiver TCP informs the application program about the beginning and end of the urgent data

Connection Termination

- **Client or server can close the connection, usually initiated by the client. It can be achieved by using *three-way handshaking* or *four-way handshaking with a half-close option***

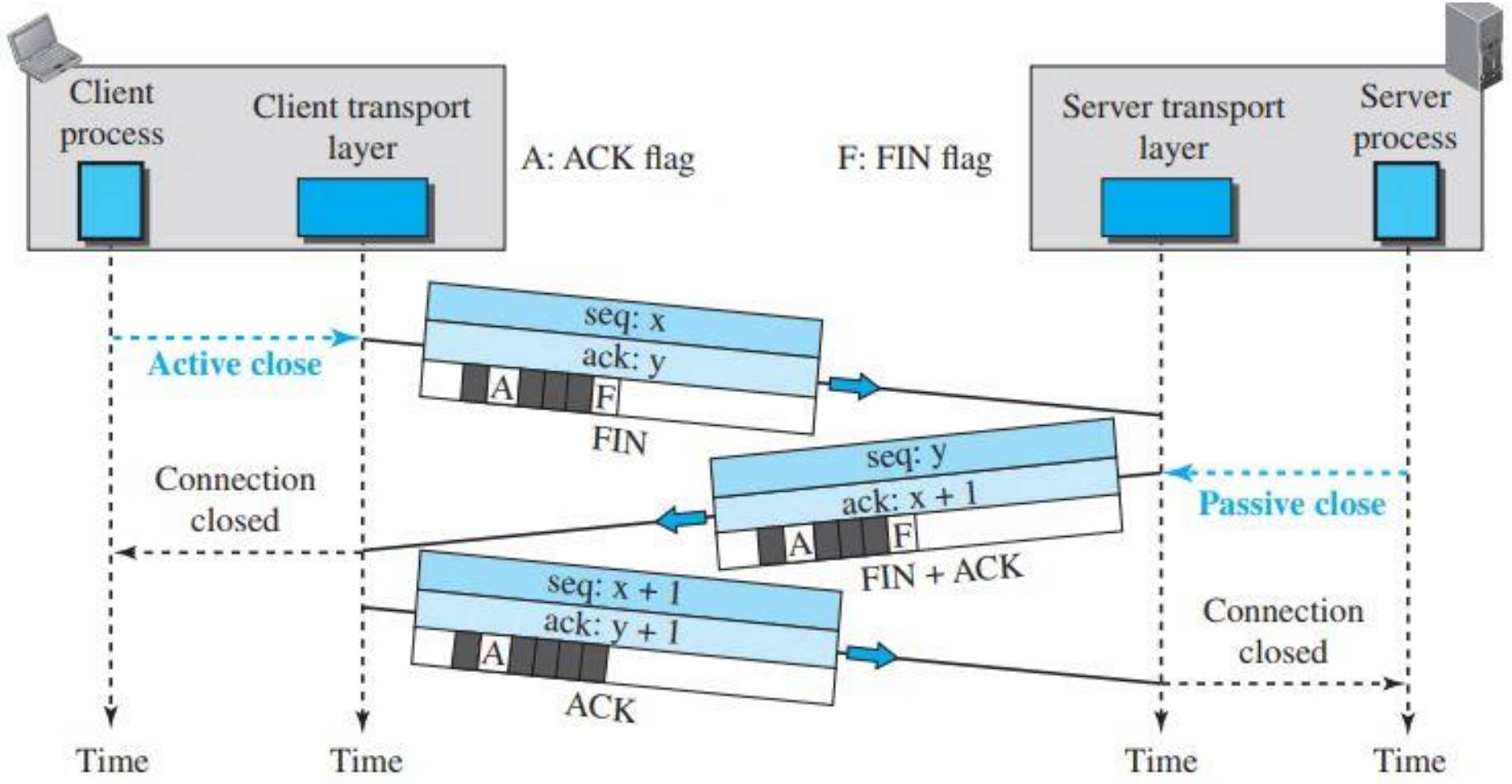

Fig 3. TCP Connection Termination using three-way handshaking

Connection Termination using three-way handshaking

- Client's TCP gets the close command from its process and sends the *FIN* segment
- A *FIN* segment can include last chunk of data, or it could be just a control segment
- If it is a control segment, then it consumes only one sequence number to be acknowledged
- After receiving *FIN* segment, the server's TCP inform its process and sends a *FIN + ACK* segment and announce closing of the connection in its end
- This segment may also contain last chunk of data from the server
- If it doesn't have data, then it consumes only one sequence number to be acknowledged
- Client's TCP sends *ACK* of the *FIN* segment received by the server

Connection Termination using half-close

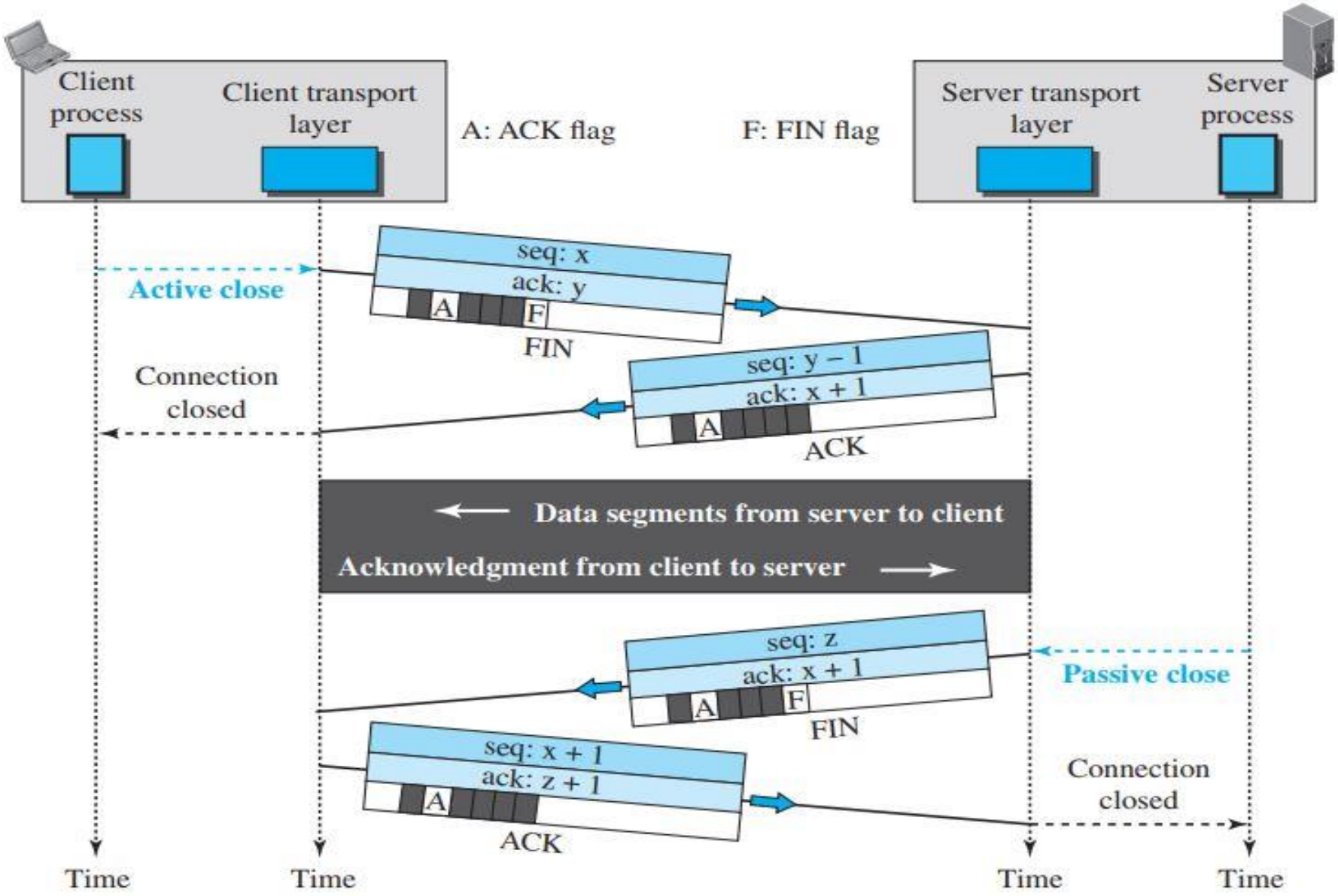

Fig 4. TCP Connection Termination using half-close

Connection Termination using half-close

- In TCP, one end can stop sending data while still receiving the data from another end. This condition is called a ***half-close***
- Either server or a client can issue a half-close request
- In Fig 2, data transfer from a client to server stops by sending a FIN segment
- Server accepts the half-close by sending the ACK; however, the server can still send data (Ex:- Sorting)
- Server will send FIN segment once it is done with sending data from its end
- After half-closing the connection, data can travel from server to client and acknowledgment travel from client to server
- ***Connection Reset*:-** TCP at one end may deny a connection request, may abort an existing connection, or may terminate an idle connection. All of these are done with the RST (reset) flag.

Data Traffic

- **Average Data Rate**
  - Amount of data/time
  - Defines the average bandwidth needed by the traffic

- **Peak Data Rate**
  - Maximum data rate of the traffic (maximum y-axis value)
  - It indicates the peak bandwidth that the network needs for traffic to pass through without changing its data flow

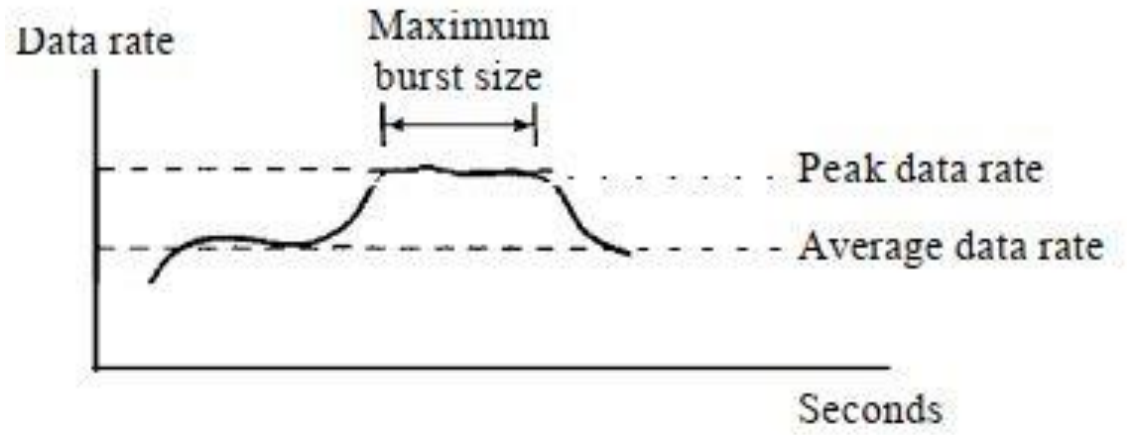

Fig 5. Traffic descriptors

Data Traffic

- **Maximum Burst Size**
  - Normally refers to the maximum length of time the traffic is generated at the peak rate

- **Effective bandwidth**
  - Bandwidth required by the network to allocate flow of traffic
  - It is a function of average data rate, peak data rate, and maximum burst size

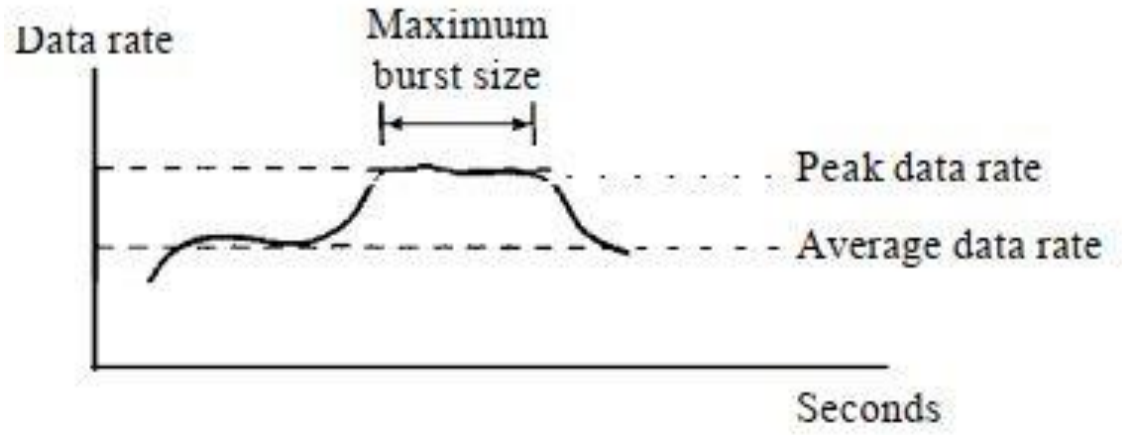

Fig 5. Traffic descriptors

Traffic Profiles

- **Constant Bit Rate (CBR)**
  - Data rate doesn't change
  - Average data rate equals peak data rate
  - Maximum burst size is not applicable
  - Easy to handle traffic
  - Network knows in advance how much traffic to allocate

- **Variable Bit Rate (VBR)**
  - Data rate changes with time
  - Average data rate not equals to peak data rate
  - Maximum burst size is small
  - Not easy to handle traffic as compared to CBR

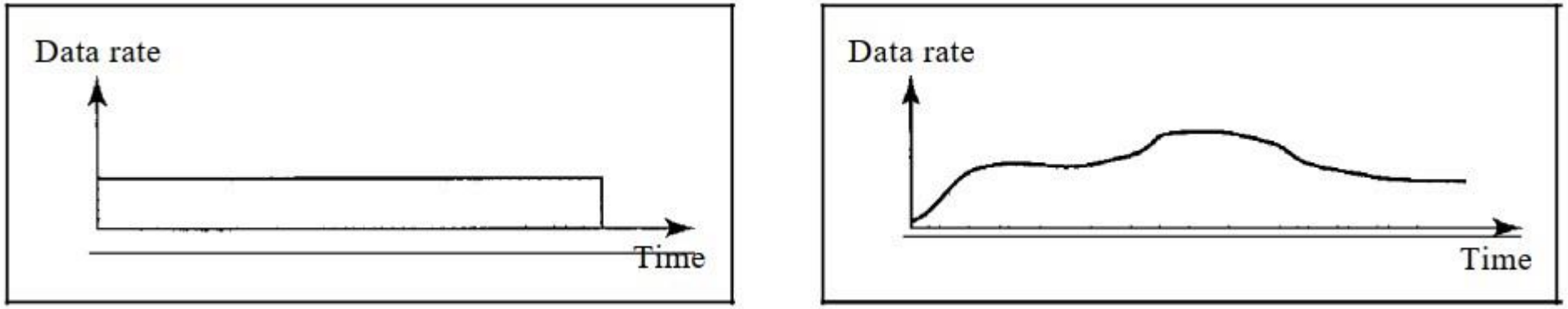

Fig 6. CBR and VBR

Traffic Profiles

- **Bursty**
  - Data rate suddenly changes
  - It may remain at the same value for a while
  - Average data rate is very different than peak data rate
  - Maximum burst size is significant
  - Most difficult traffic to handle
  - Main reason for congestion

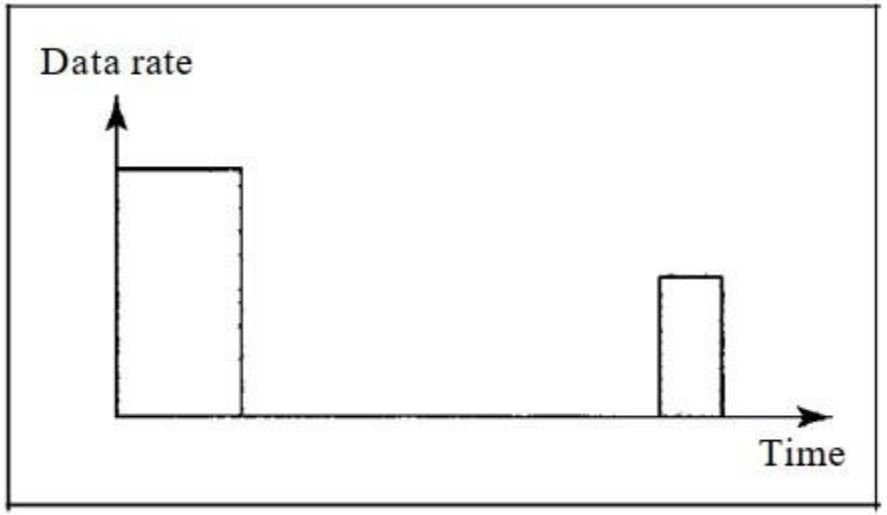

Fig 7. Bursty Traffic

Congestion

- **Congestion occurs if the number of packets sent to the network *(load on the network)* is greater than the number of packets network can handle *(capacity of the network)***
- **Routers and switches have queues**
- **Input queue become longer if the rate of packet arrival is higher than the packet processing rate**
- **Output queue become longer if the packet departure rate is less than the packet processing rate**
- **Open-loop congestion control is used before congestion**
- **Closed-loop congestion control is used after congestion**

Open-Loop Congestion Control

- **Retransmission policy**
  - Retransmission happens when packet is lost or corrupted
  - Retransmission increases congestion in the network
  - Good retransmission policy and timers must be designed

- **Window policy**
  - Selective repeat window is better than the Go-Back-N window for congestion control

- **Acknowledgment policy**
  - If the receiver doesn’t acknowledge each packet and sends cumulative acknowledgement
  - Receiver may send the acknowledgment only if it has data or certain timer expires

- **Discarding policy**
  - Discarding less sensitive packet may reduce congestion

- **Admission policy**
  - Check the resource requirement of the flow before admitting it in the network
  - Router can deny establishing a virtual-circuit connection if congestion is present

Closed-Loop Congestion Control

- **Backpressure**
  - Congested node stops receiving data from its upstream node
  - This in turn causes congestion in the upstream node and therefore, upstream nodes stops receiving data from its upstream node and so on
  - It can be applied to only virtual-circuit networks

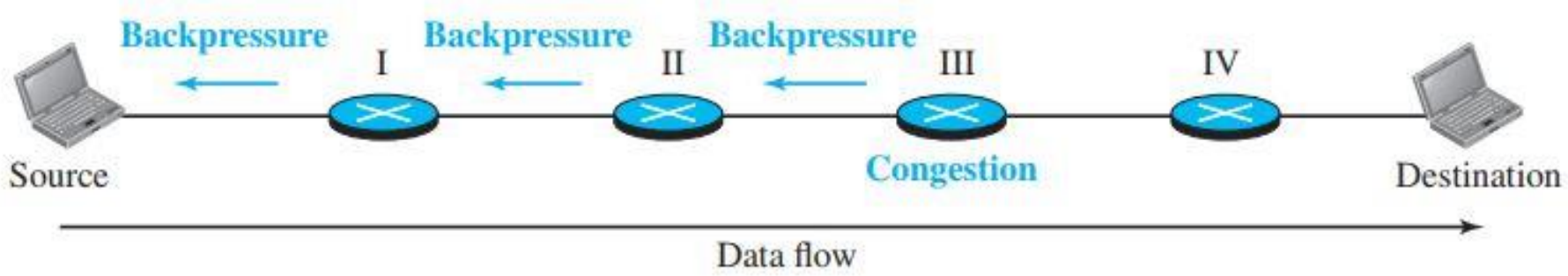

Fig 8. Backpressure

- **Choke packet**
  - A choke packet is sent from the congested node directly to source
  - Intermediate nodes through which the packet travels are not warned

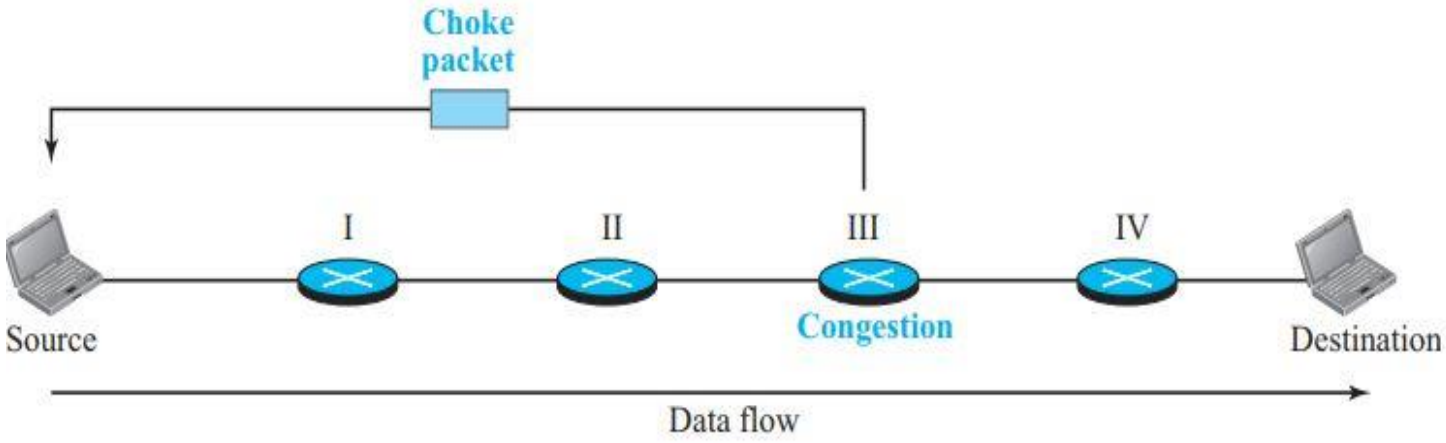

Fig 9. Choke packet

Closed-Loop Congestion Control

- **Implicit Signaling**
  - No communication between the congested node/nodes and the source
  - Source assumes congestion from some symptoms like delay on receiving ACKs

- **Explicit Signaling**
  - Signal is included in the packet that carries data
  - Backward signaling is used when a bit is set in a packet moving in the direction opposite to the congestion which warns source
  - Forward signaling is used when a bit is set in a packet moving in the direction of congestion which warns the destination about the congestion