CSC0056: Data Communication

Lecture 04: Hamming Code and Data Retransmission Strategies

Reading assignment: Sections 2.4-2.4.1

Instructor: Chao Wang 王超

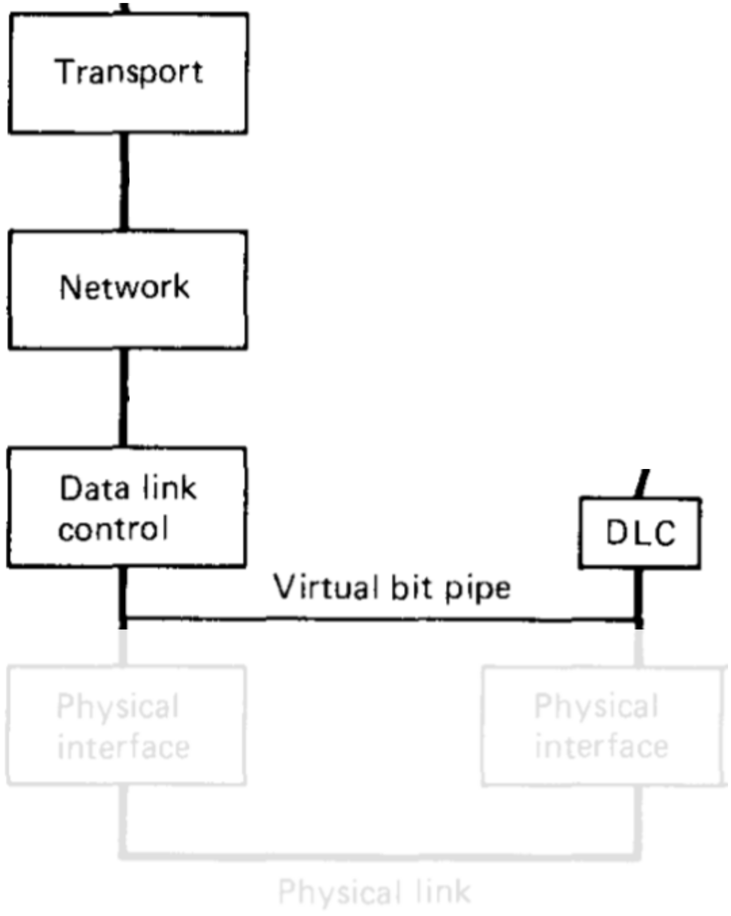
Department of Computer Science and Information Engineering

NATIONAL TAIWAN NORMAL UNIVERSITY

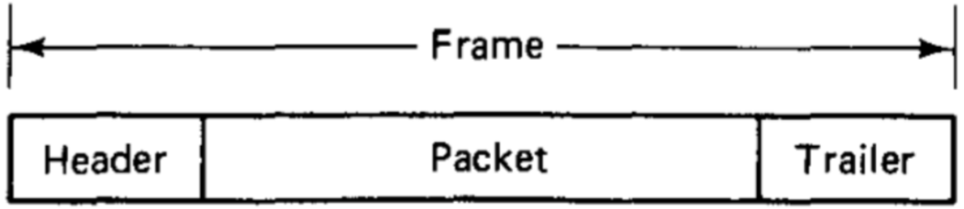
Outline of lecture04

- Hamming code for error detection and correction
    - For this part, we will use the whiteboard.
- Data retransmission strategies and analysis (data link layer)
    - ARQ basics
- Retransmission strategies at a higher layer

Data transmission viewed from the data link layer



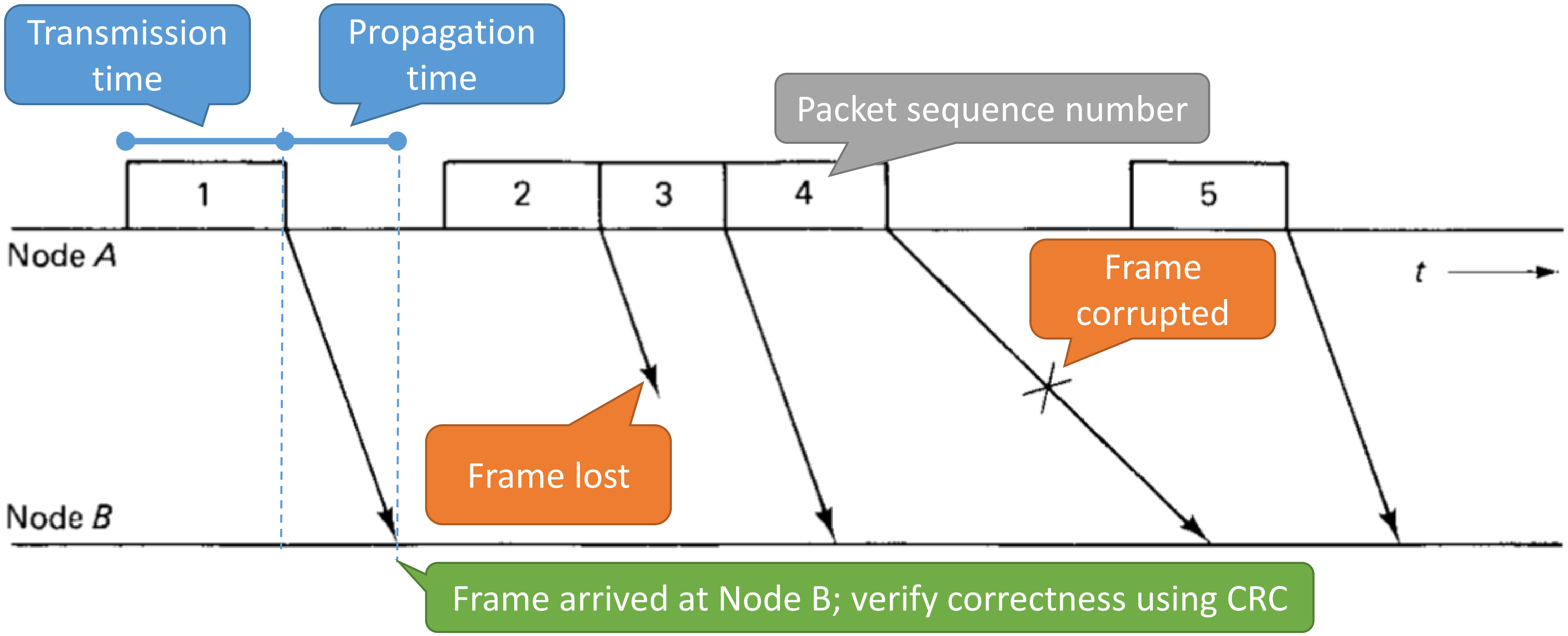
- Data link layer creates a *frame* by appending a header and trailer to each of the *packet* from the network layer



- Data link layer views the underlying point-to-point channel as an *unreliable* virtual bit pipe
- A frame could be lost or corrupted (i.e., contains errors) in the virtual bit pipe

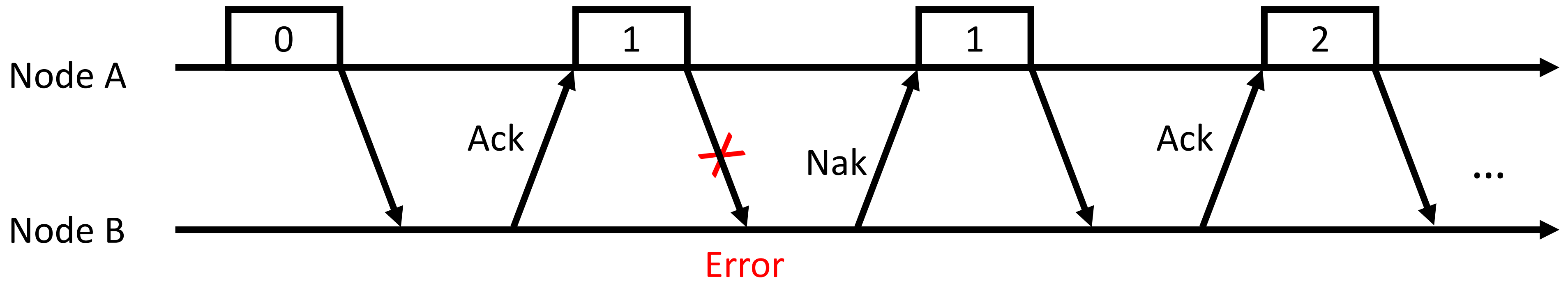
ARQ: Automatic Repeat Request

- Model and definition, assuming Node A sends frames to Node B



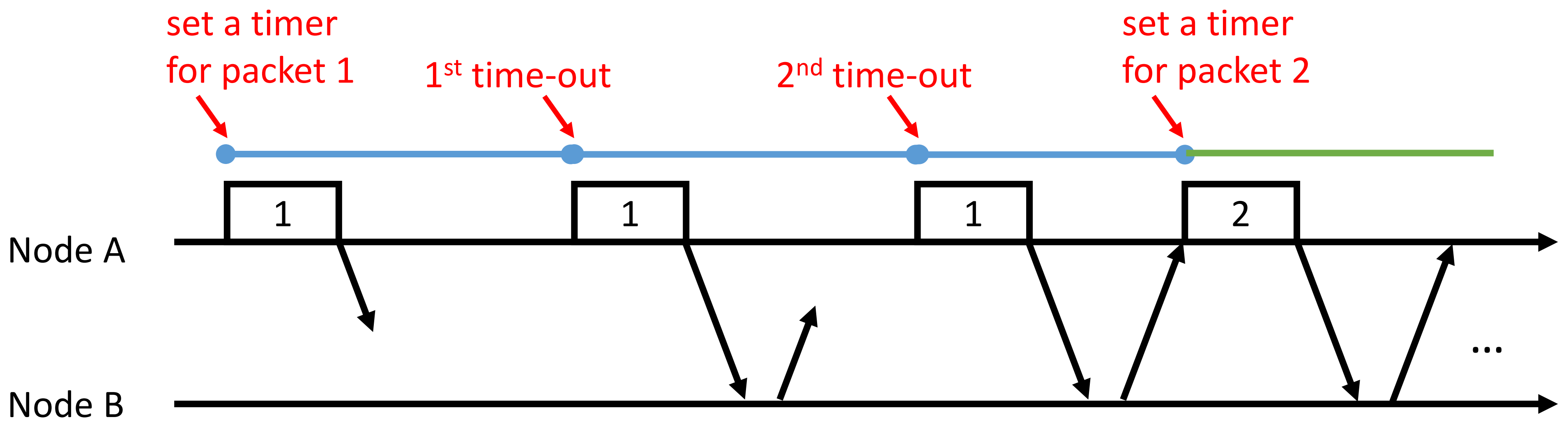
An initial idea of Stop-and-Wait ARQ

- Node A sends a new packet only if it received an acknowledgement from Node B (called an Ack)
- Node B sends an Ack if the received frame is error-free; otherwise, it sends a negative acknowledgement (called an Nak)



The need for time-out and re-transmission

- With no time-out, Node A may wait forever (for either Ack or Nak) if a frame is lost in either direction during transmission…
- Upon a time-out, Node A will re-transmit the same packet
- Design issue: what is a desirable length of a timer? (see later slides)

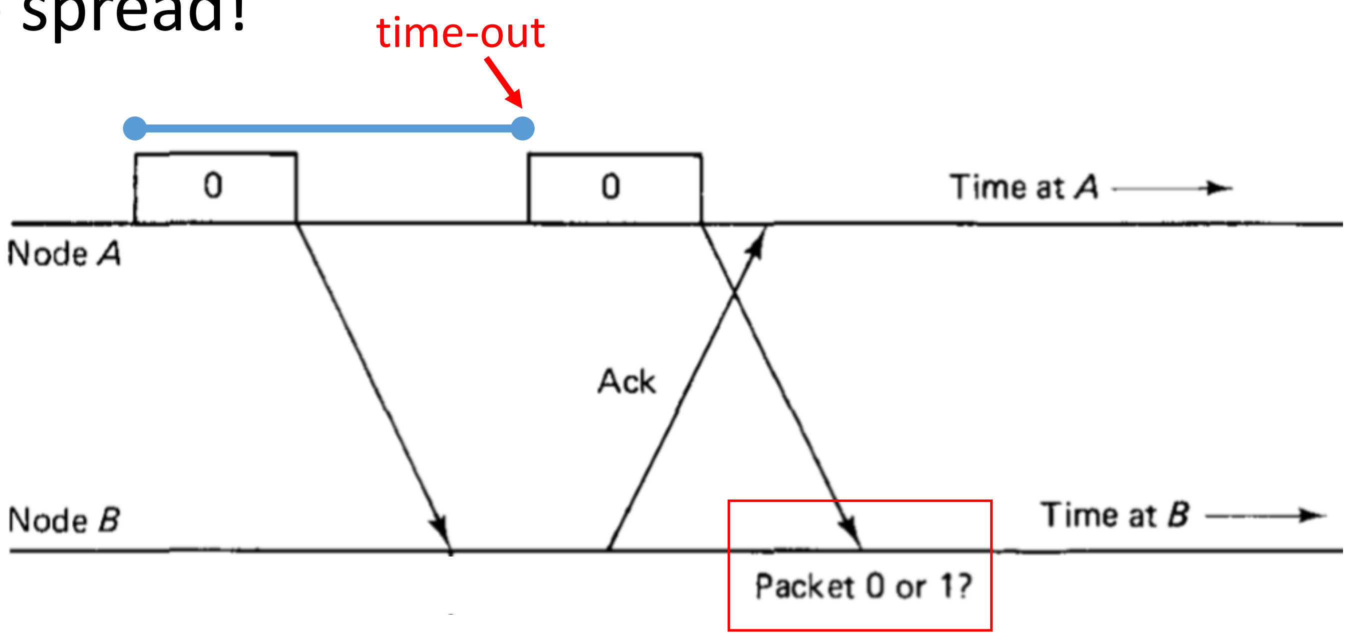


The need for packet sequence number

- Node B otherwise may not be able to tell if a frame contains a new packet or if a frame contains a previous packet

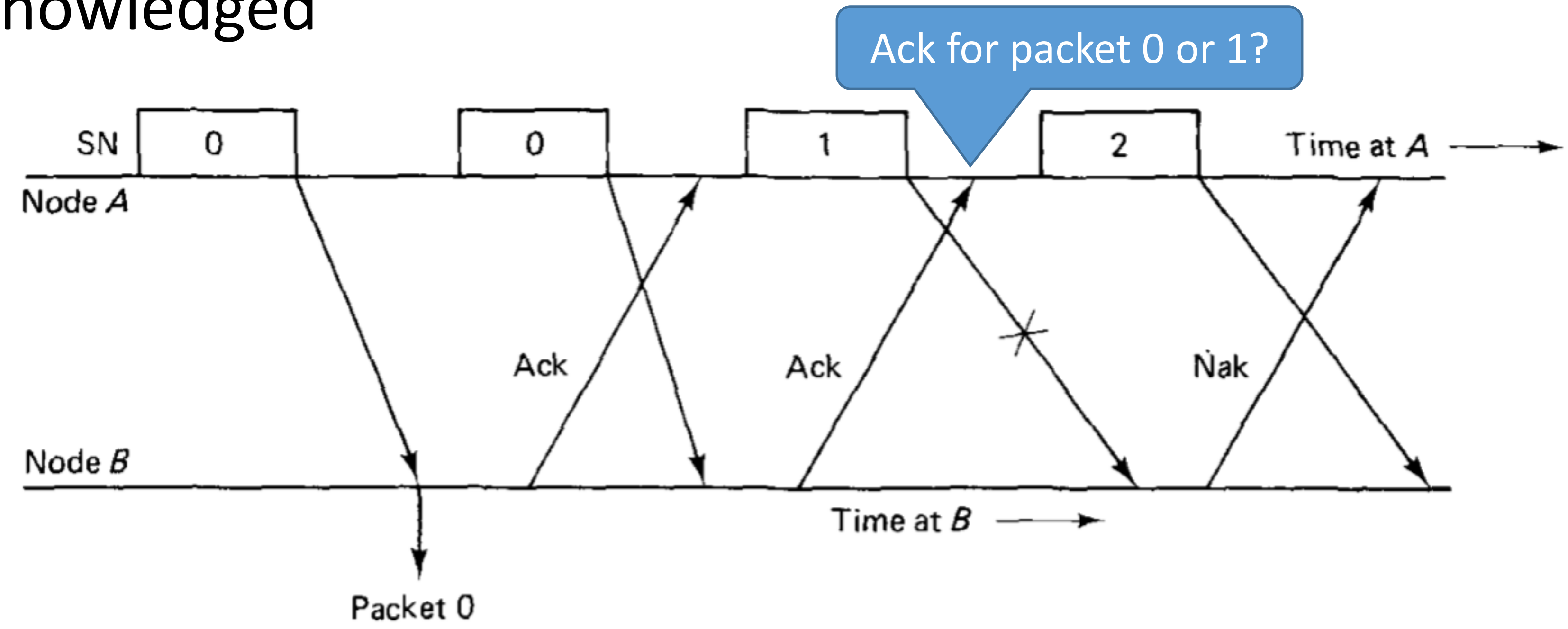
✓ In general, in designing distributed algorithms, a challenge is that information takes time to spread!

Correctness and efficiency are two critical aspects in design of distributed algorithm.



The need for distinct acknowledgements

- With simple Ack/Nak, Node A might not be able to tell which packet Node B acknowledged

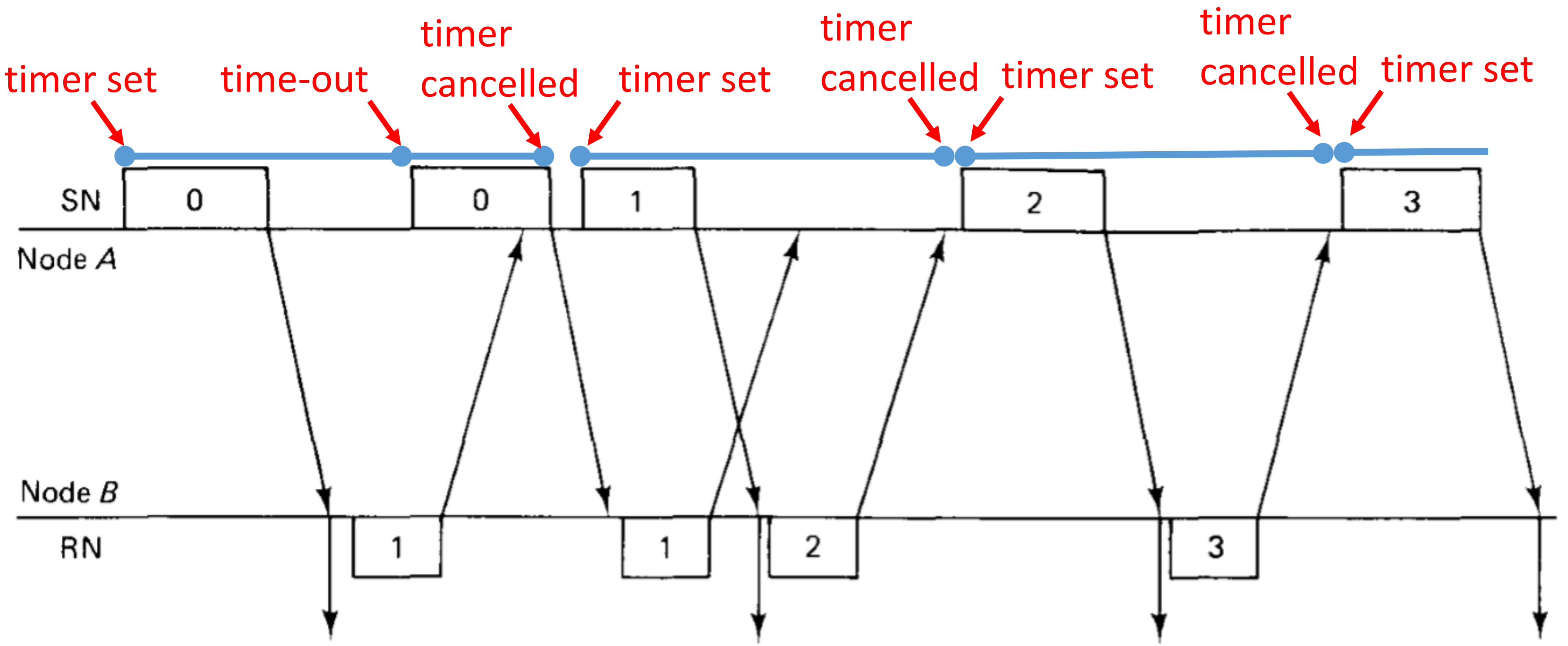


- Solution: Node B sends the number of frame currently *awaited*

A working version of stop-and-wait ARQ

- Using time-out, sequence number (SN), and request number (RN) to coordinate Nodes A and B

(Read the textbook for the assumptions and the algorithm)

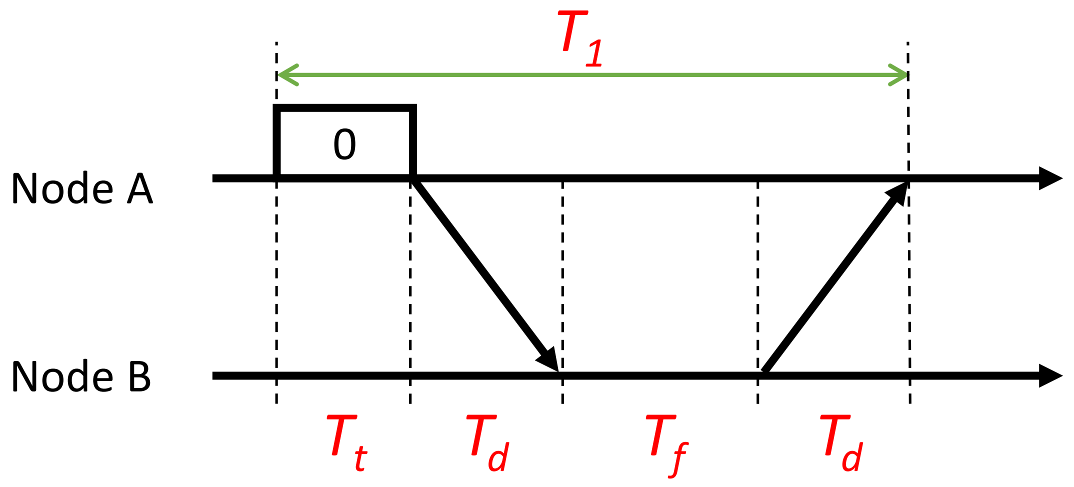


Analyzing efficiency of the stop-and-wait

- There are several ways to look at efficiency
- Way #1 (latency): let T be the total time between the transmission of a packet and reception of its Ack

✓ If error-free, $T = T_1 = T_t + 2T_d + T_f$

T_t : packet transmission time
T_d : delay in propagation and processing
T_f : feedback transmission time



Efficiency in terms of latency

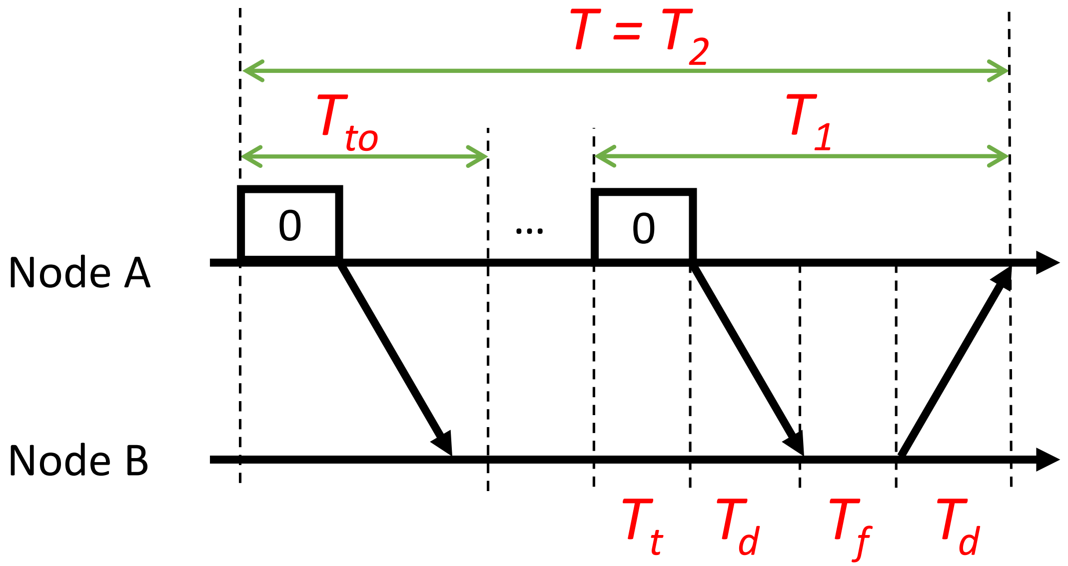
- ✓ With chances of error,

  $T = T_2 = T_1 + T_{to}(EX\text{-}1)$

  $EX = q^{-1}$

  (Refer to the note for a proof)

- ✓ **Latency depends on T_{to} and q**

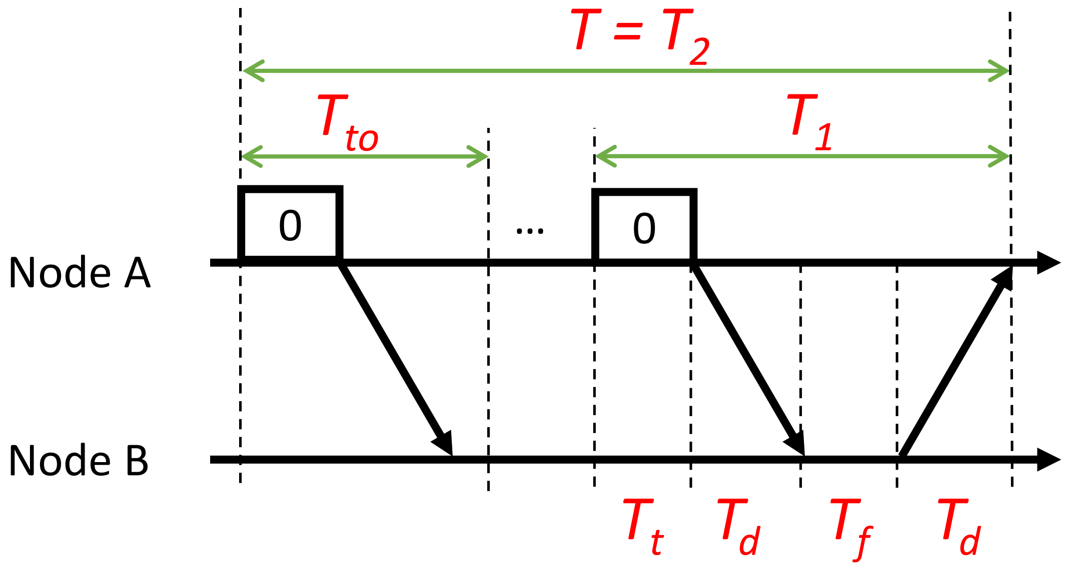


EX : the expected number of times a packet must be transmitted for a stop-and-wait system
q : the probability that a packet is correctly received and acked on a given transmission
T_{to} : length of a timer (i.e., duration before a time-out)
T_t : packet transmission time
T_2 : total time between the transmission of a packet and reception of its Ack

Effects of the length of timer

- Setting $T_{to} > T_1$, we have

  $$T_2 = T_1 + T_{to}(q^{-1}-1) > q^{-1}T_1$$



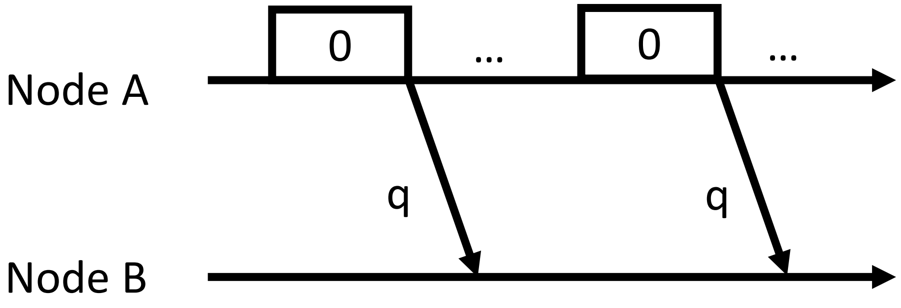
- setting $T_{to} < T_1$, make sense?
  - will cause additional redundancy
  - but then we have $T_2 < q^{-1}T_1$, which gives us a way to bound the latency.
  - The percentage of increase in latency is $(T_2 - T_1)/T_1 < (1 - q)/q$.

- Exercise: plug in different values of q and see how T changes

Another way to look at efficiency of the stop-and-wait : link goodput

- Way #2 (link goodput): (way #1 at slide #11)

  Define efficiency *E* as the expected number of packets delivered to node B per frame from A to B



✓ **Efficiency E = $\underline{1/EX = q}$ = $1-p$**

(Similar to the proof that we used for analyzing latency)

✓ **for all ARQ protocols, $E \leq 1-p$**

EX : the expected number of transmissions before the packet is correctly received *p* : the probability of frame error *q* : the probability of no frame error (***q=1-p***)

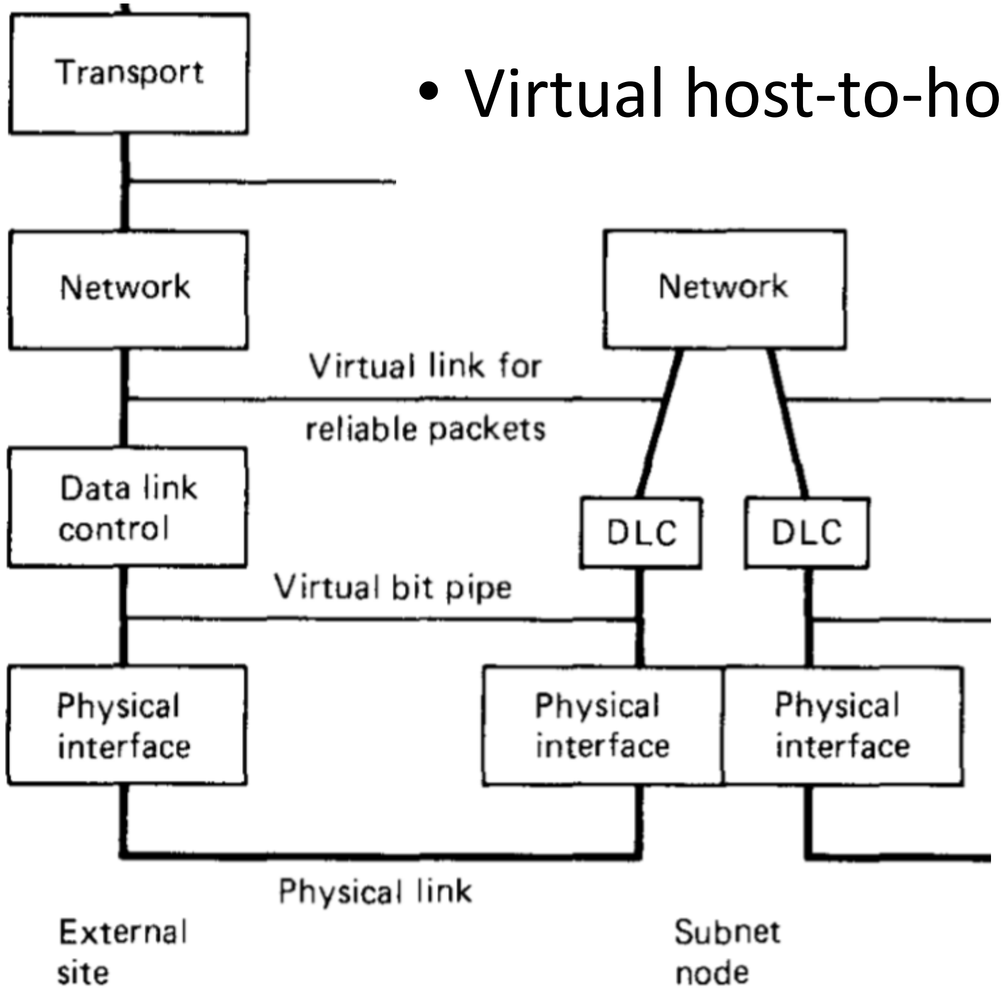
Outline of lecture04

- Hamming code for error detection and correction
  - For this part, we will use the whiteboard.

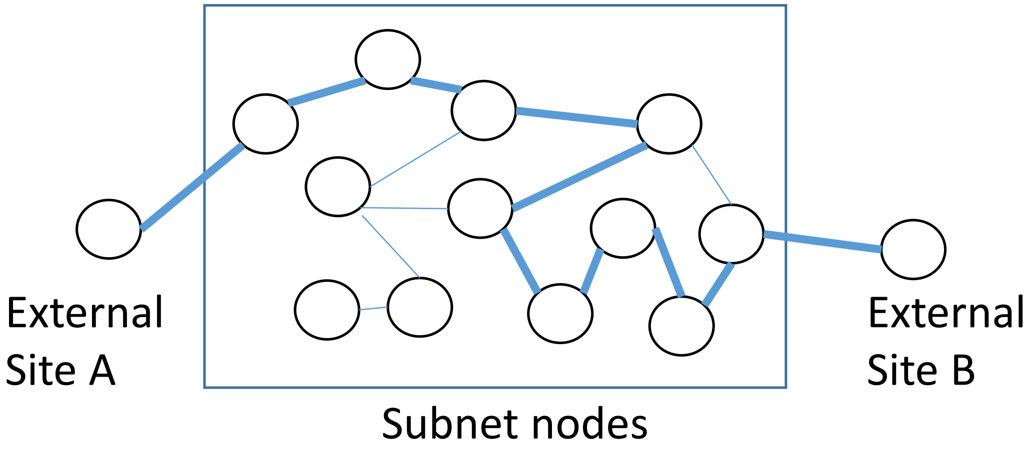
- Data retransmission strategies and analysis (data link layer)
  - ARQ basics

- **Retransmission strategies at a higher layer**

Concept review of the network layer



- Virtual host-to-host packet service to the higher layers
- Each host (node) contains one network layer module
- Provide *routing* for the network:



Error recovery at higher layers

- Conceptually very similar to ARQ at the data link layer
  - Point-to-point error recovery: two nodes and a link in between
  - End-to-end error recovery: two sites and a subnet in between
    - May operate on a go-back-n/selective repeat basis
- Key difference: at higher layers, packets might arrive out of order
  - Solution candidates:
    - Enforce ordering
    - Use a large modulus number for packet numbering
    - Destroy a packet after some time
- Example: error recovery in TCP
  - You may refer to Sec. 2.9.3 in the textbook