컴퓨터 네트워크 – 11 주차

Jong-Kyou Kim, PhD

2014-05-14

Packets to signals

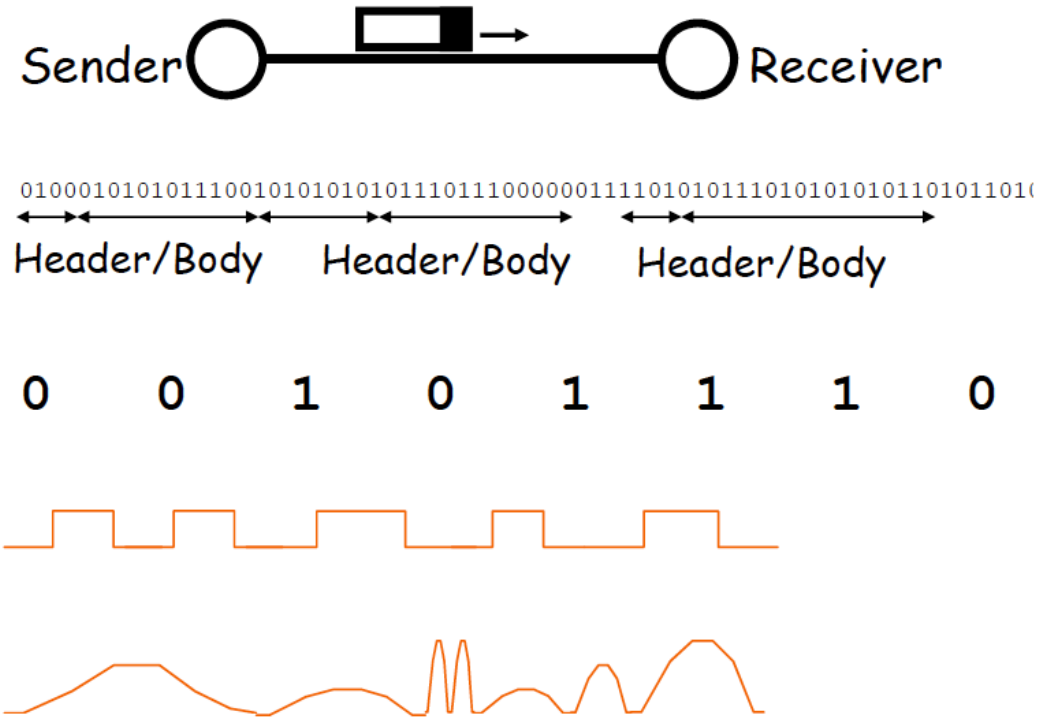

그림 : Packets to signals

Jong-Kyou Kim, PhD

Challenges of Signal Transmission

- Noise, Attenuation, Dispersion

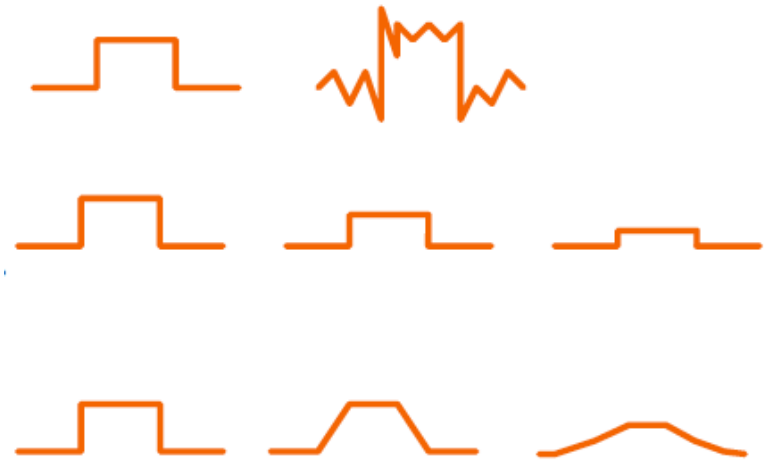

그림 : Signal transmission

Jong-Kyou Kim, PhD

Challenges of Signal Transmission

- 신호전달에는 반드시 오류가 발생한다

  $\longrightarrow$ 어떻게 오류를 검증할까?

  - 오류검증을 위한 추가 정보를 전달한다
  - 추가정보의 검증에는 코딩이론이 사용된다.

코딩이론의 응용: 자기디스크 (Magnetic Disk)

- 헤드, 트랙, 섹터로 구성
- 운영체제가 컨트롤러에 명령을 내려 작동

그림 : magnetic disk

코딩이론의 응용: Rotary encoder

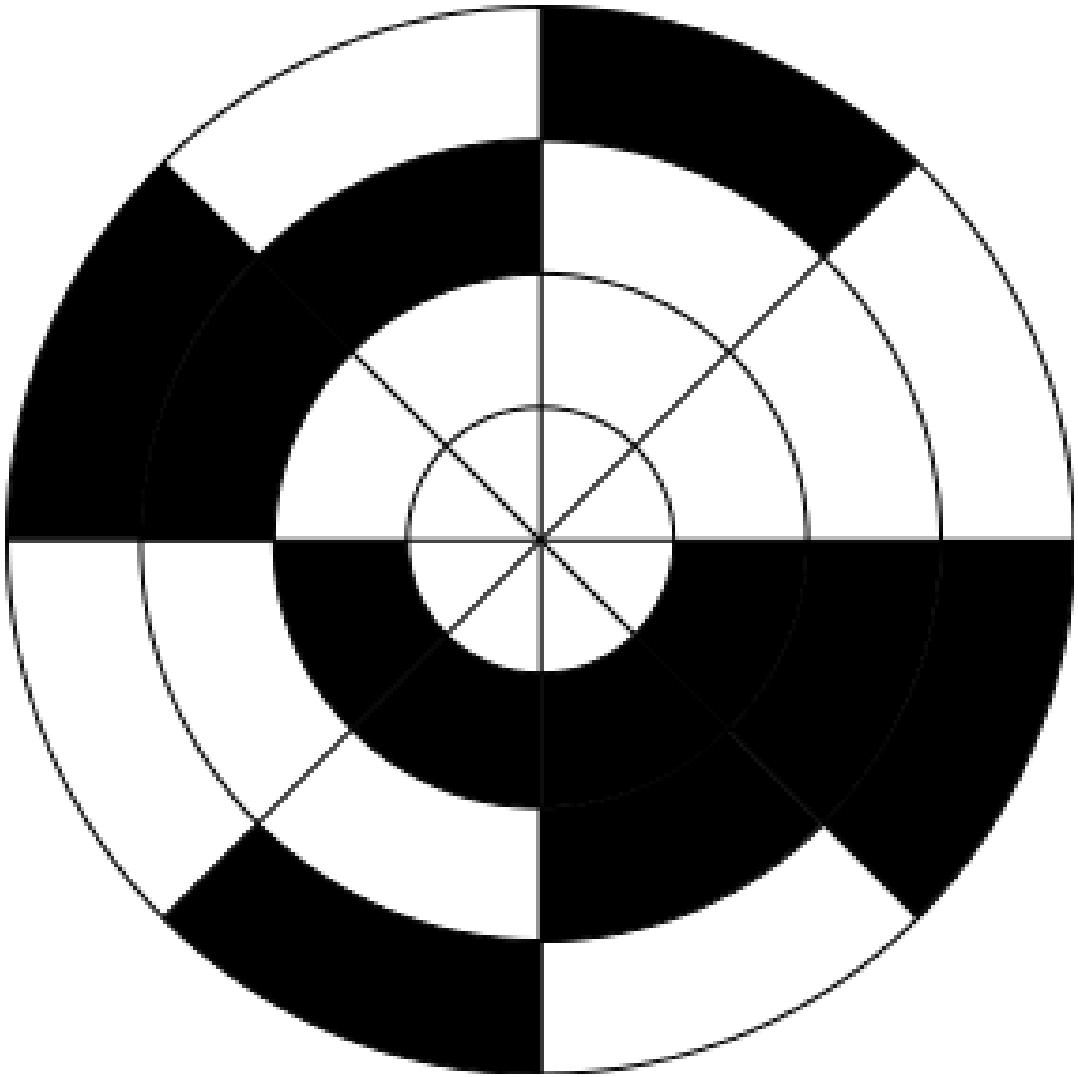

그림 : Rotary encoder

코딩이론의 응용: 오류검증

- 000 $\longrightarrow$ 000
- 001 $\longrightarrow$ 001
- 010 $\longrightarrow$ 011
- 011 $\longrightarrow$ 010
- 100 $\longrightarrow$ 110
- 101 $\longrightarrow$ 111
- 110 $\longrightarrow$ 101
- 111 $\longrightarrow$ 100

$\longrightarrow$ 왼쪽에 비해서 오른쪽의 장점은?

코딩이론의 응용: Gray code

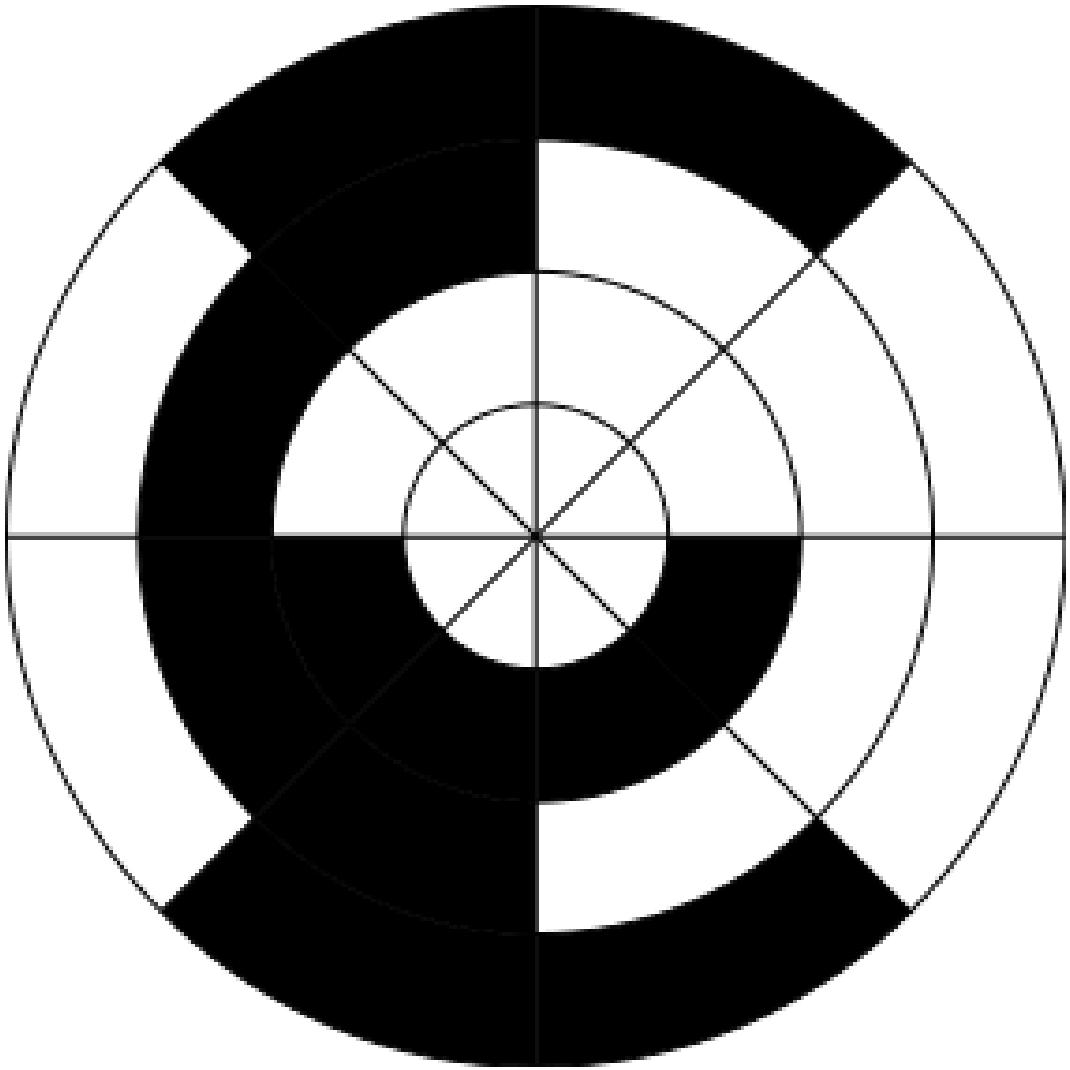

그림 : Gray code

코딩이론의 응용: 오류검출/수정

- 오류검출: 패리티
  - 1 비트 추가
  - 짝수, 홀수
- 오류수정: 반복코드 ($n = 3$): 다수결
  - Reference: $000 \longrightarrow 0$, $111 \longrightarrow 1$
  - $101 \longrightarrow 1$, $100 \longrightarrow 0$

오류검출/수정 (해밍코드)

- 000 111 로 0/1 을 encode
  - 001 을 보면 $\longrightarrow$ 000
  - 011 을 보면 $\longrightarrow$ 111
- 4 비트를 다음과 같은 codebook 으로 만들면?

0000000	0001101	0010111	0011010
0100011	0101110	0110100	0111001
1000110	1001011	1010001	1011100
1100101	1101000	1110010	1111111

  - 1001111 은? 1001011

Hamming distance 와 error correction

- codebook 내의 모든 비트를 비교하여 서로 다른 위치의 갯수
  - 예: D(000,111) = 3, D(000,001) = 1
  - D(1001111, 1001011) = 1
  - D(0000000,0001101) = 3
- n 비트를 보낼 때 m 비트만큼만 보내고 $n - m$ 비트는 hamming distance 를 3 으로 유지하는데 사용한다면? $\longrightarrow$ error correction
- n 이 커지면 $\frac{m}{n} \rightarrow 1$

  $\longrightarrow$ 검증은 블럭단위로 해야 $\longrightarrow$ Frame

Frame

- 오류탐지 및 교정을 위한 최소한의 단위
  - 각각의 비트에 의미가 부여됨
- 최소한의 기능
  - 시작점과 끝점을 분간할 수 있어야 한다
  - 시작주소와 끝을 알 수 있어야 한다.
- 예: Ethernet frame

그림 : Ethernet frame

- Preamble: 7 bytes 10101010 + 10101011

Virtual communication

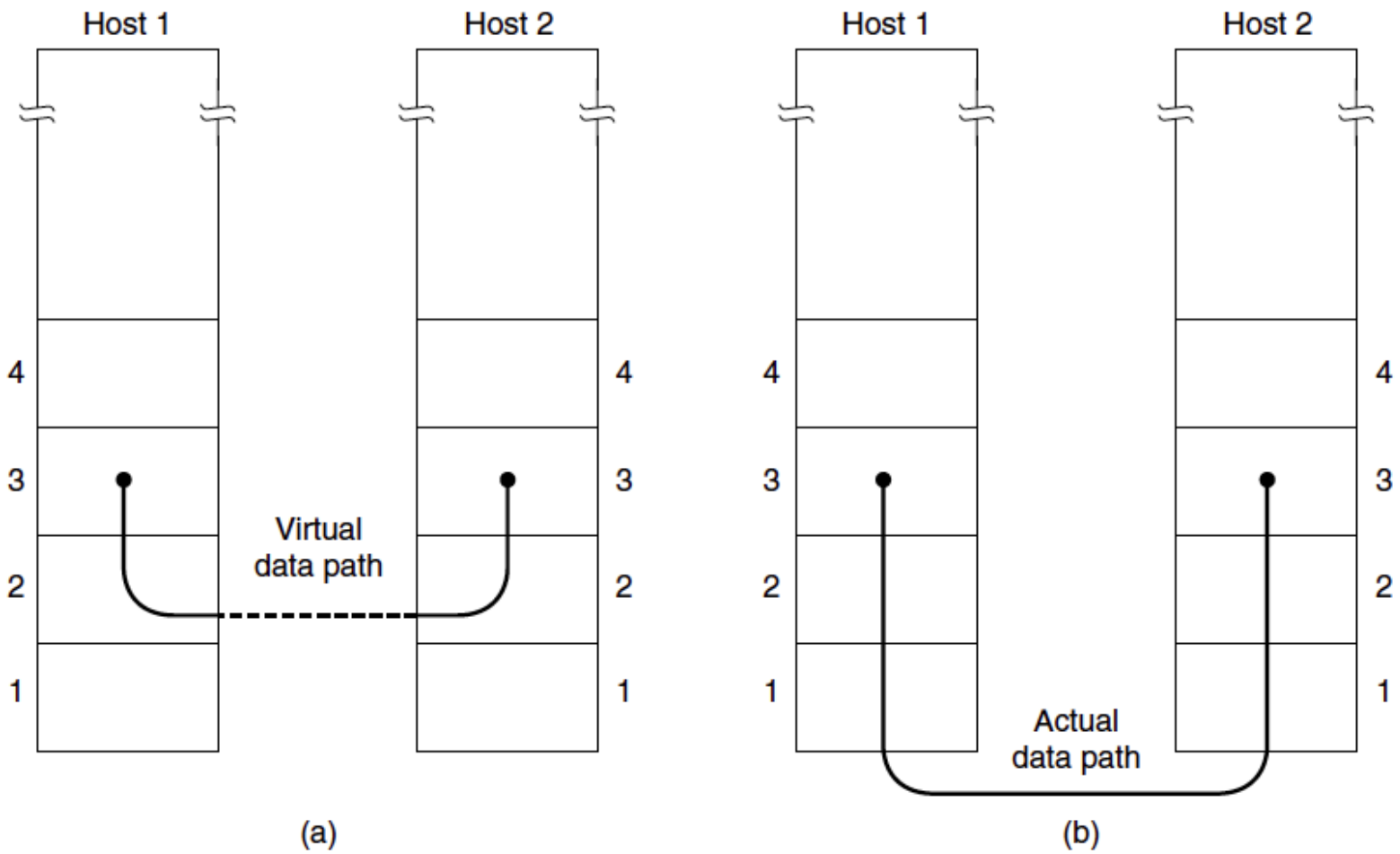

그림 : Virtual communication

Jong-Kyou Kim, PhD

Byte count framing

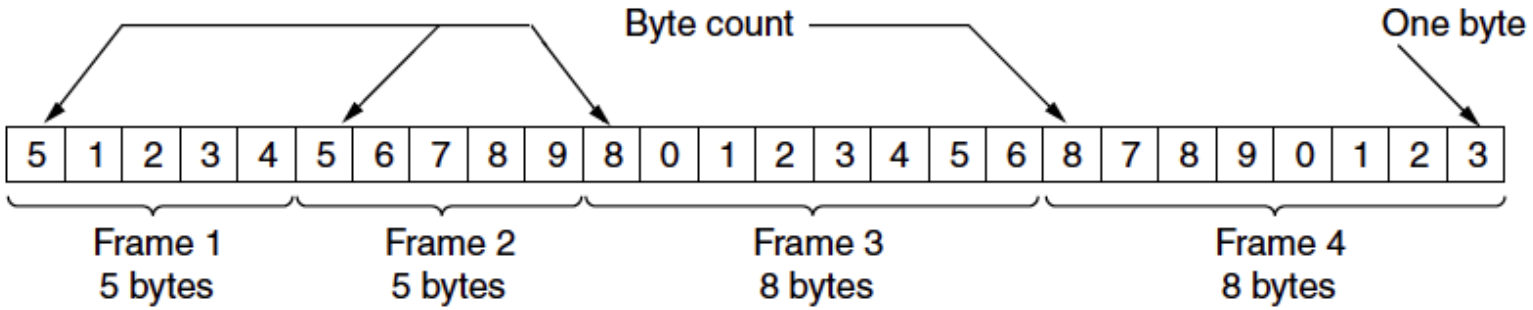

그림 : Byte count

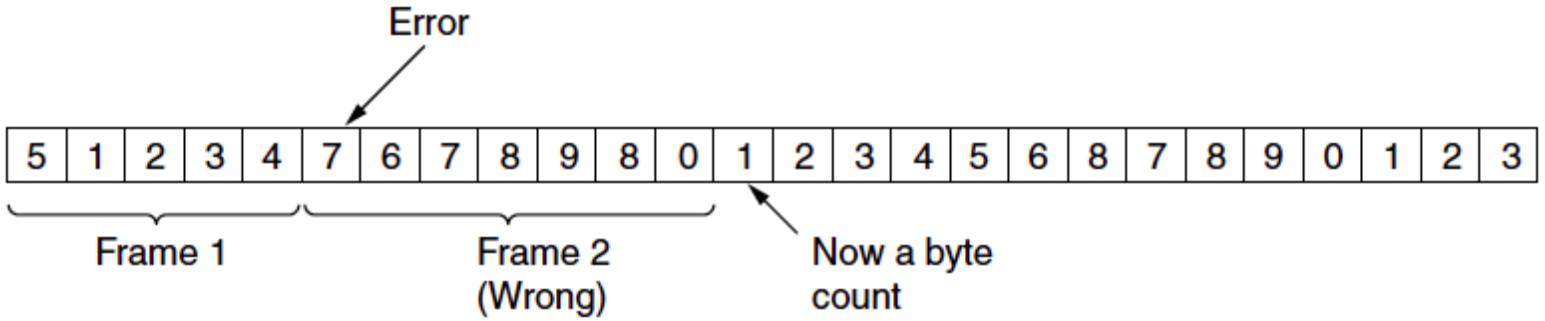

그림 : Byte count with errors

Jong-Kyou Kim, PhD

Flag byte

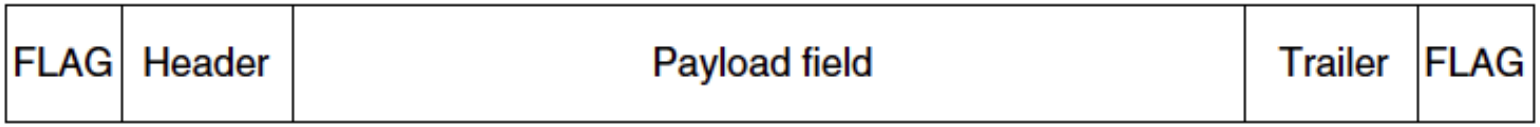

그림 : Flag bytes

Byte stuffing

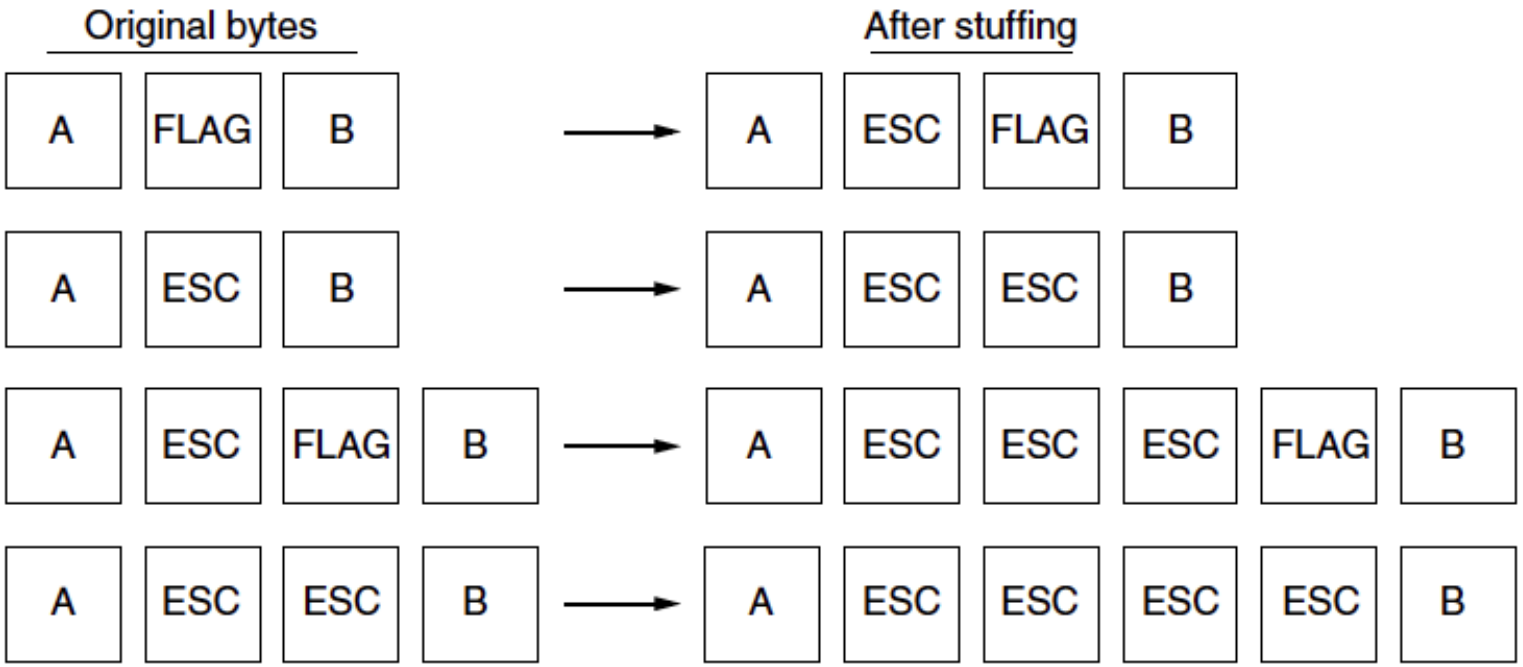

그림 : Byte stuffing

Bit stuffing

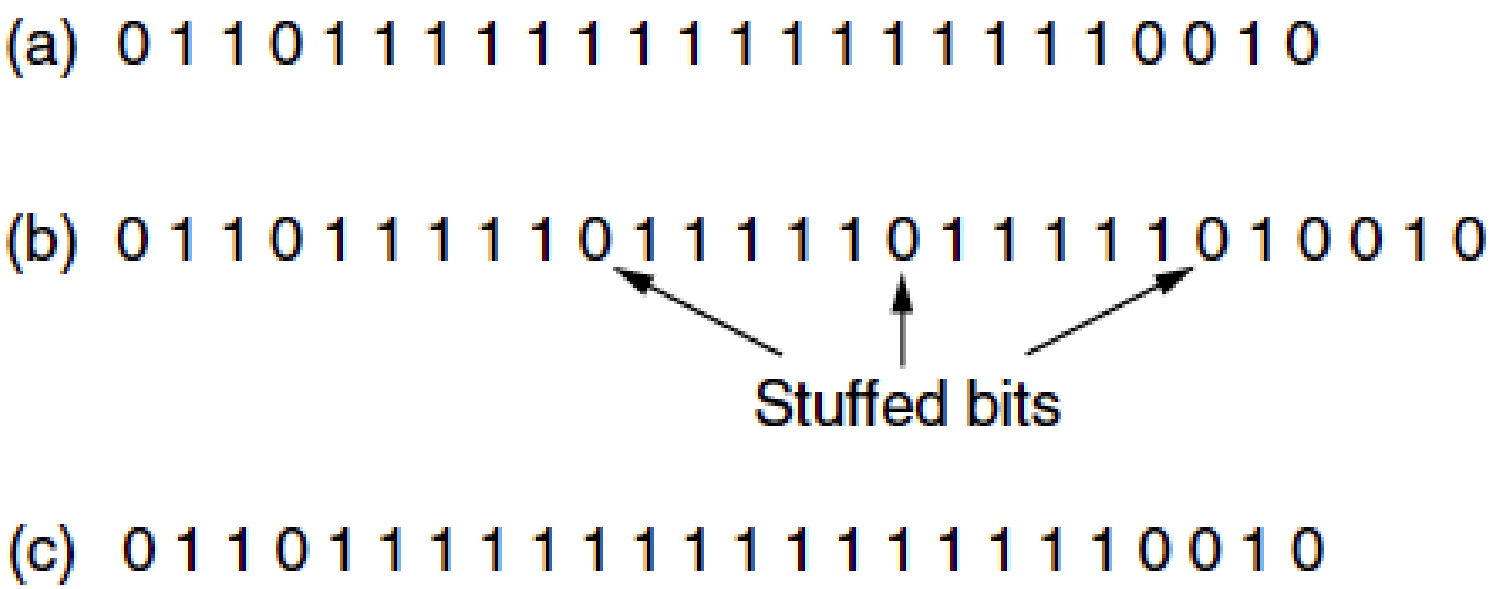

그림 : Bit stuffing

브로드캐스트 채널

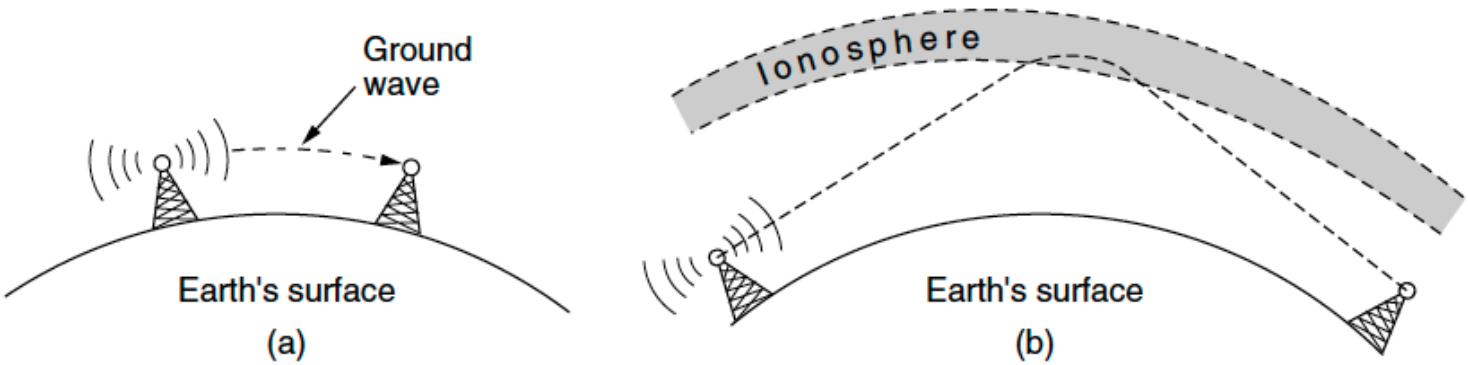

그림 : 단순화된 통신 개념도

Jong-Kyou Kim, PhD

브로드캐스트 채널

- 이제까지의 논의: 하나의 sender $\longrightarrow$ 하나의 receiver
- 현실: 여러 sender 와 여러 receiver

브로드캐스트 채널

- 한 사람이 이야기하면 모든 사람이 들을 수 있다
- 두 사람이 한꺼번에 이야기하면 이야기가 섞여서 알아 들을 수 없다
- 음성의 경우 $\longrightarrow$ 독립적인 채널을 주어야 한다
  - 한 노드가 채널을 점유하면 정보 전달이 없어도 유지된다

  $\longrightarrow$ 다른 방에서 이야기를 나누는 것
- 데이터의 경우 $\longrightarrow$ 채널을 같이 사용할 수 있다
  - 채널을 독점하지 않고 다른 노드가 채널을 사용할 수 있도록 한다

  $\longrightarrow$ 같은 방에서 순서대로 이야기 한다

Pure ALOHA

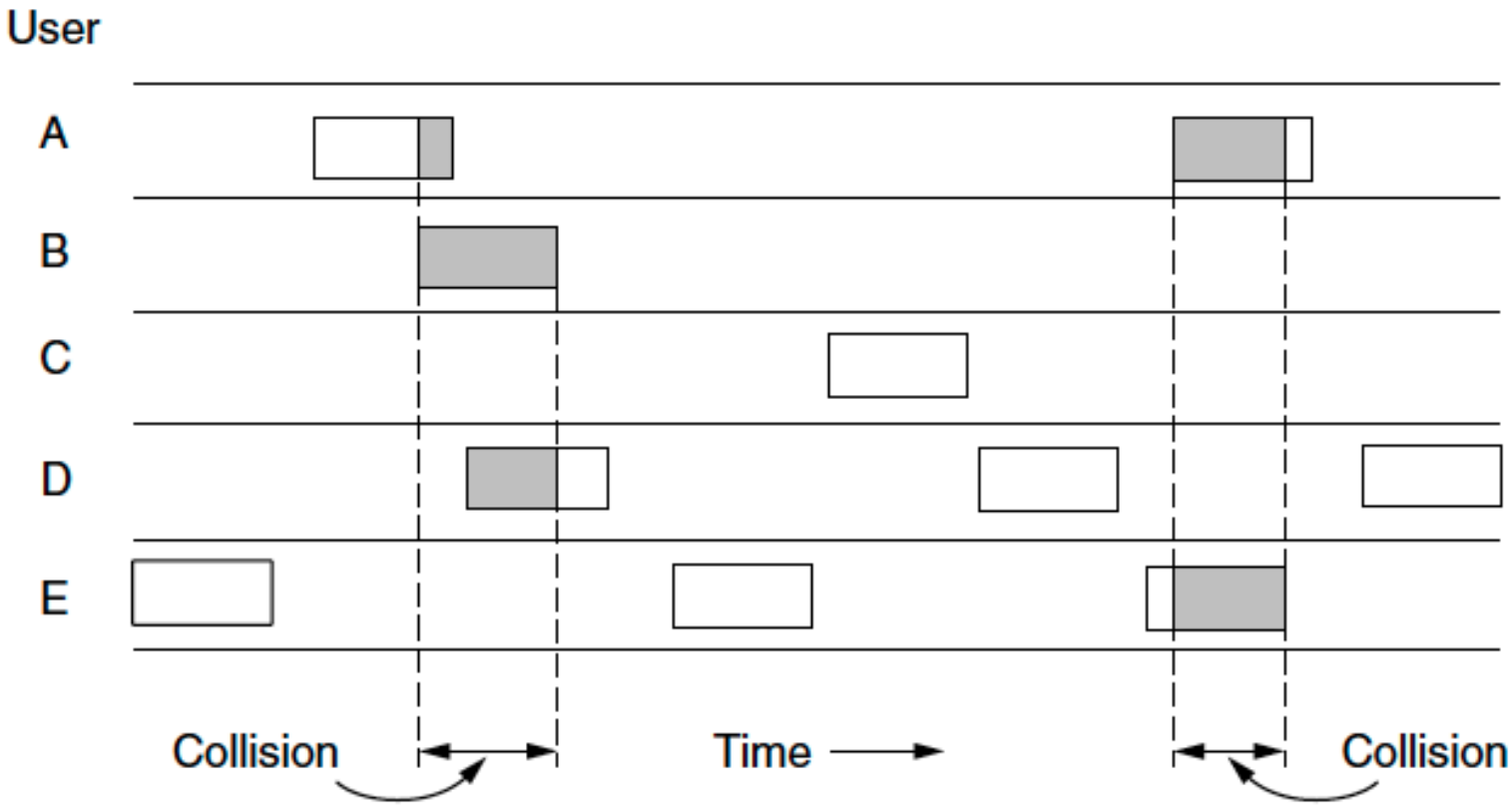

그림 : Pure ALOHA

CSMA/CD

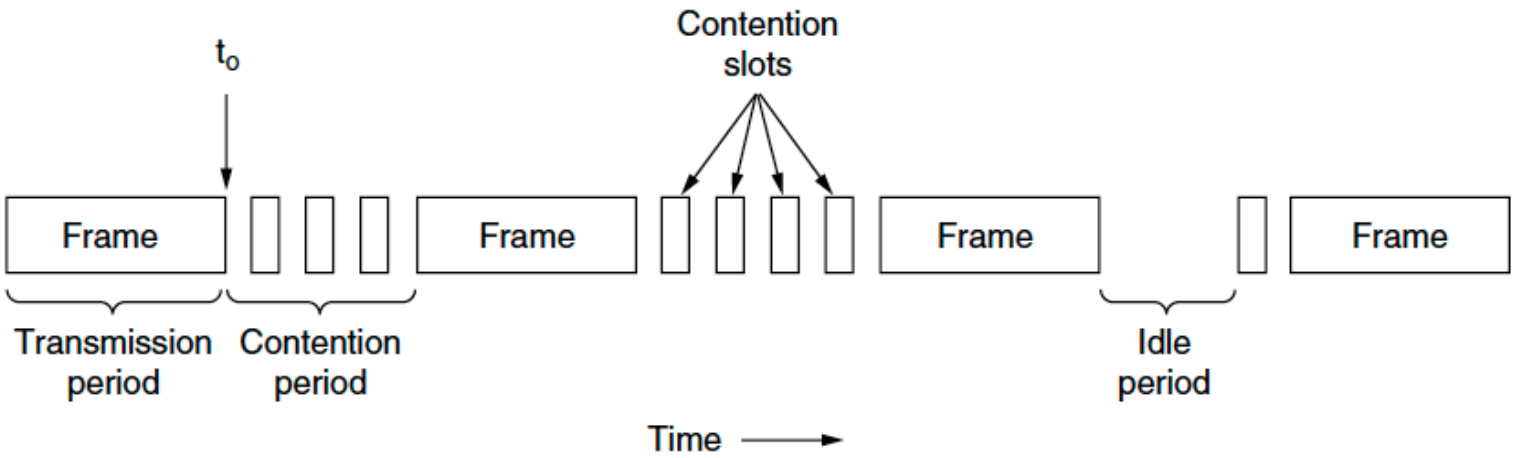

그림 : CSMA/CD

Ethernet: classic architecture

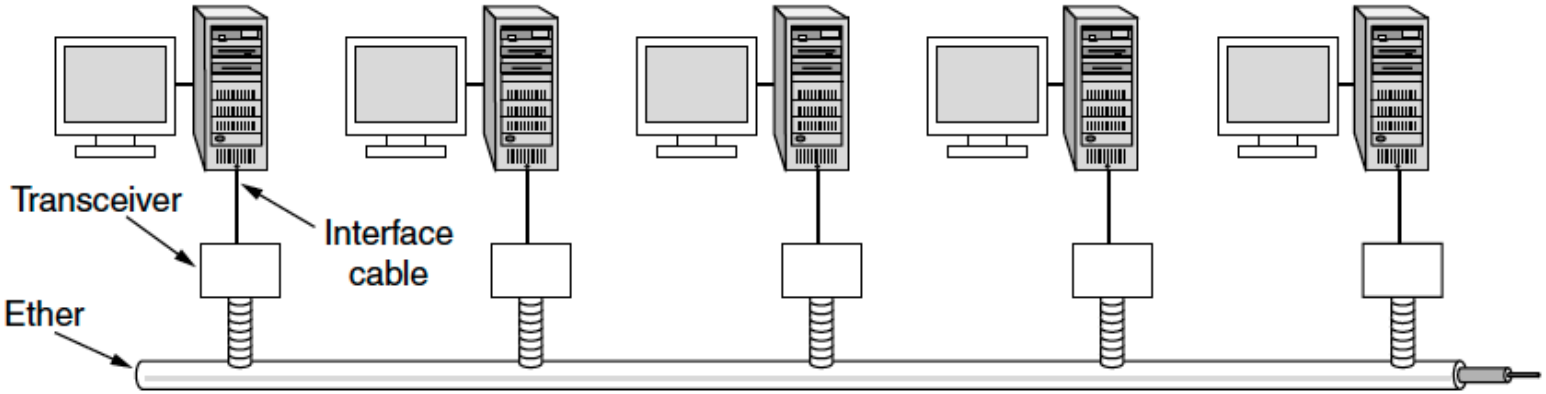

그림 : Ethernet

Jong-Kyou Kim, PhD

Ethernet frame

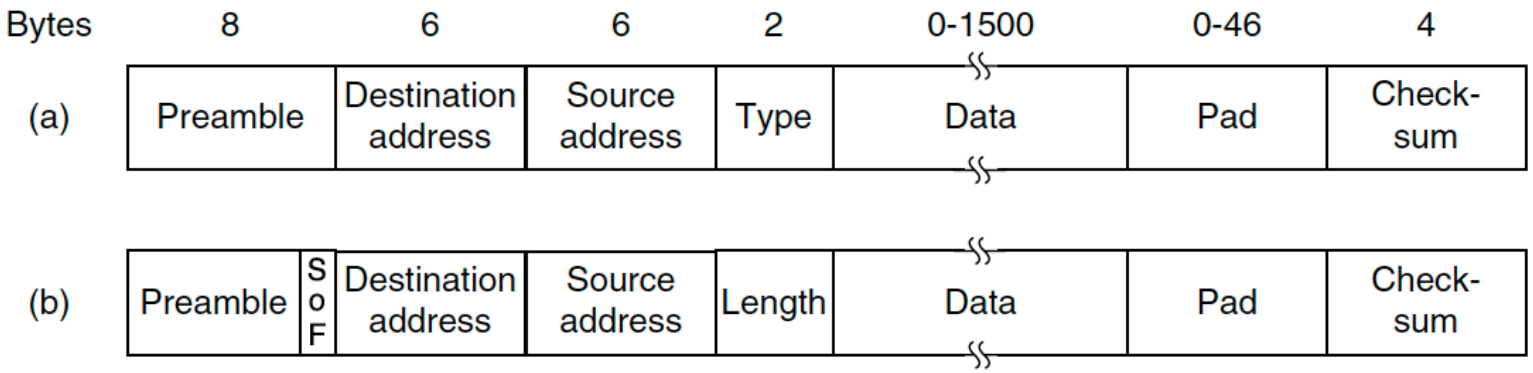

그림 : Ethernet frame

Jong-Kyou Kim, PhD

Ethernet: modern architecture

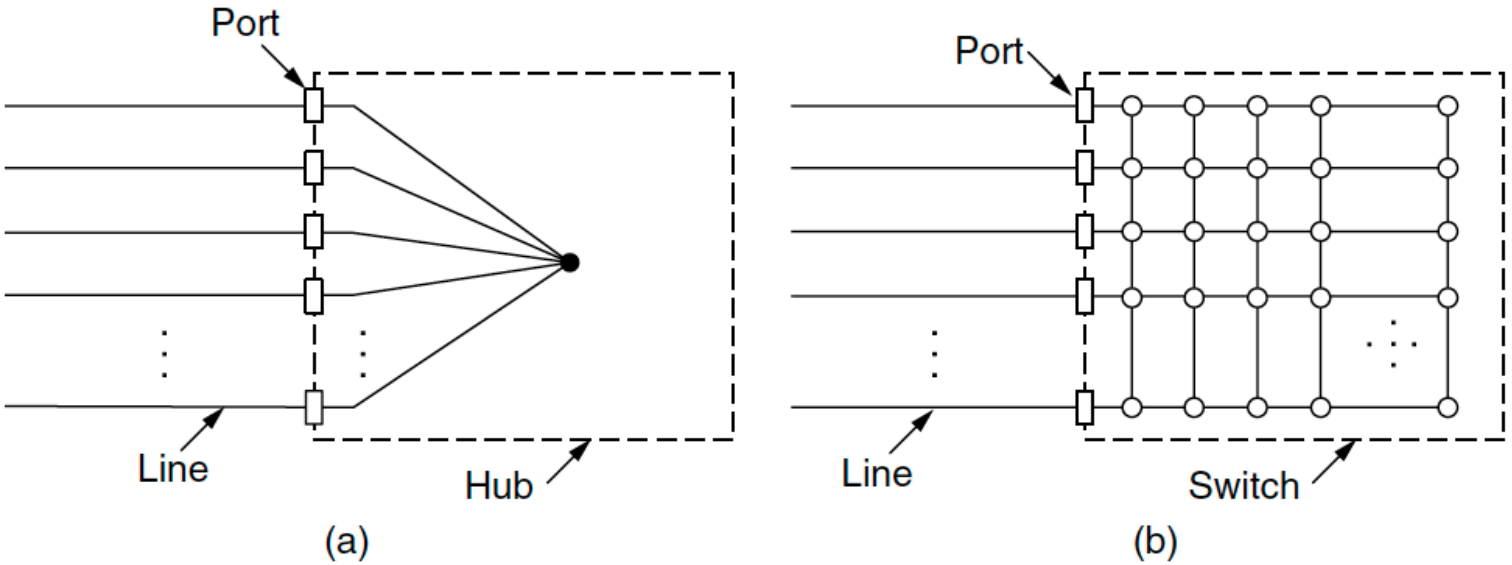

그림 : Modern Ethernet

WiFi (Wireless LAN)

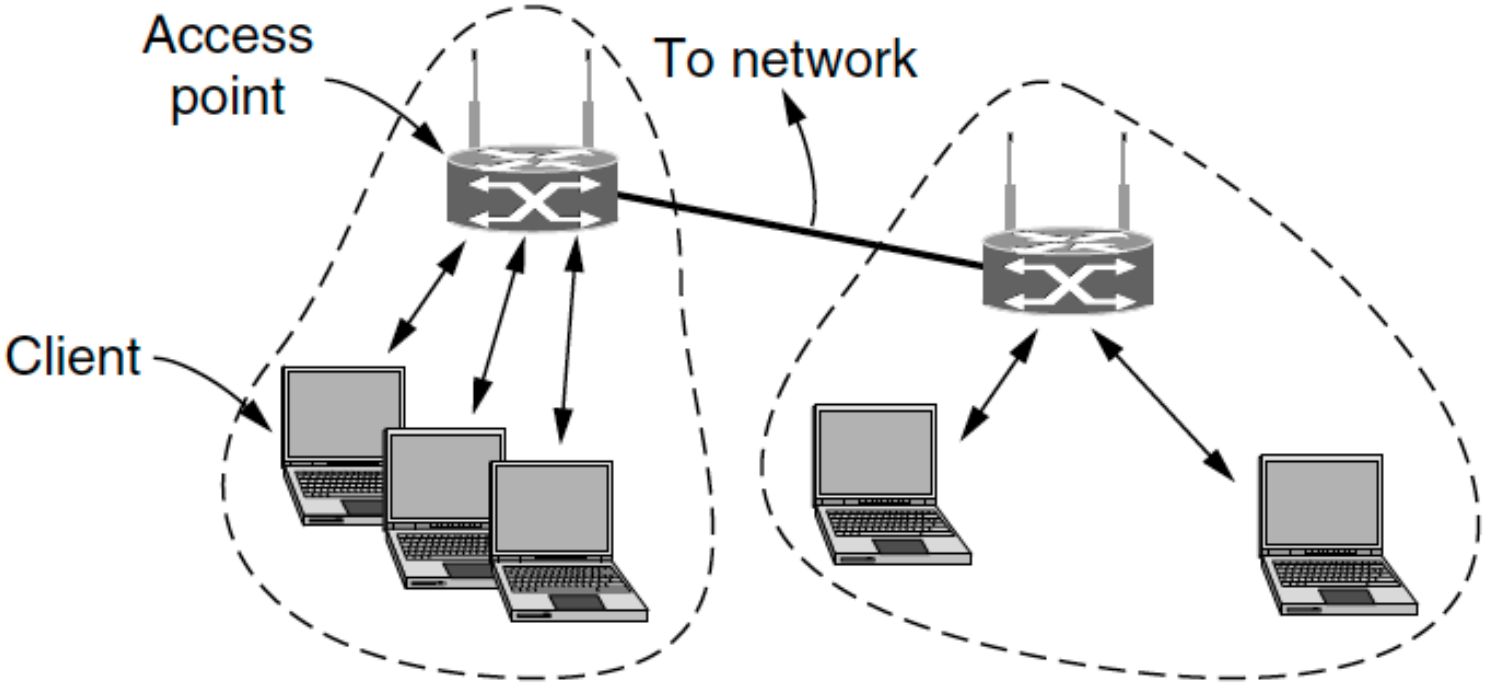

그림 : Infrastructure mode

WiFi (Wireless LAN)

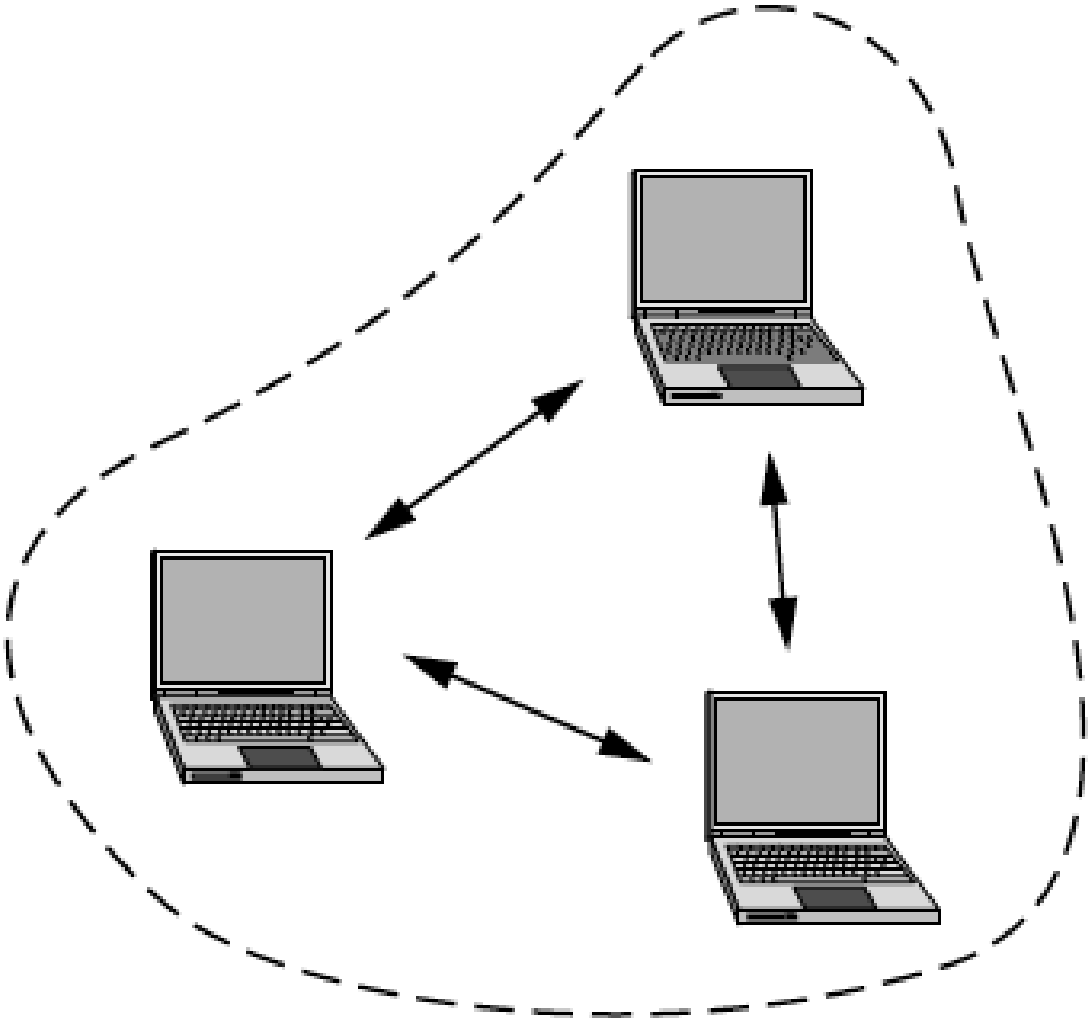

그림 : Ad-hoc mode

Jong-Kyou Kim, PhD

WiFi: Hidden terminal problem

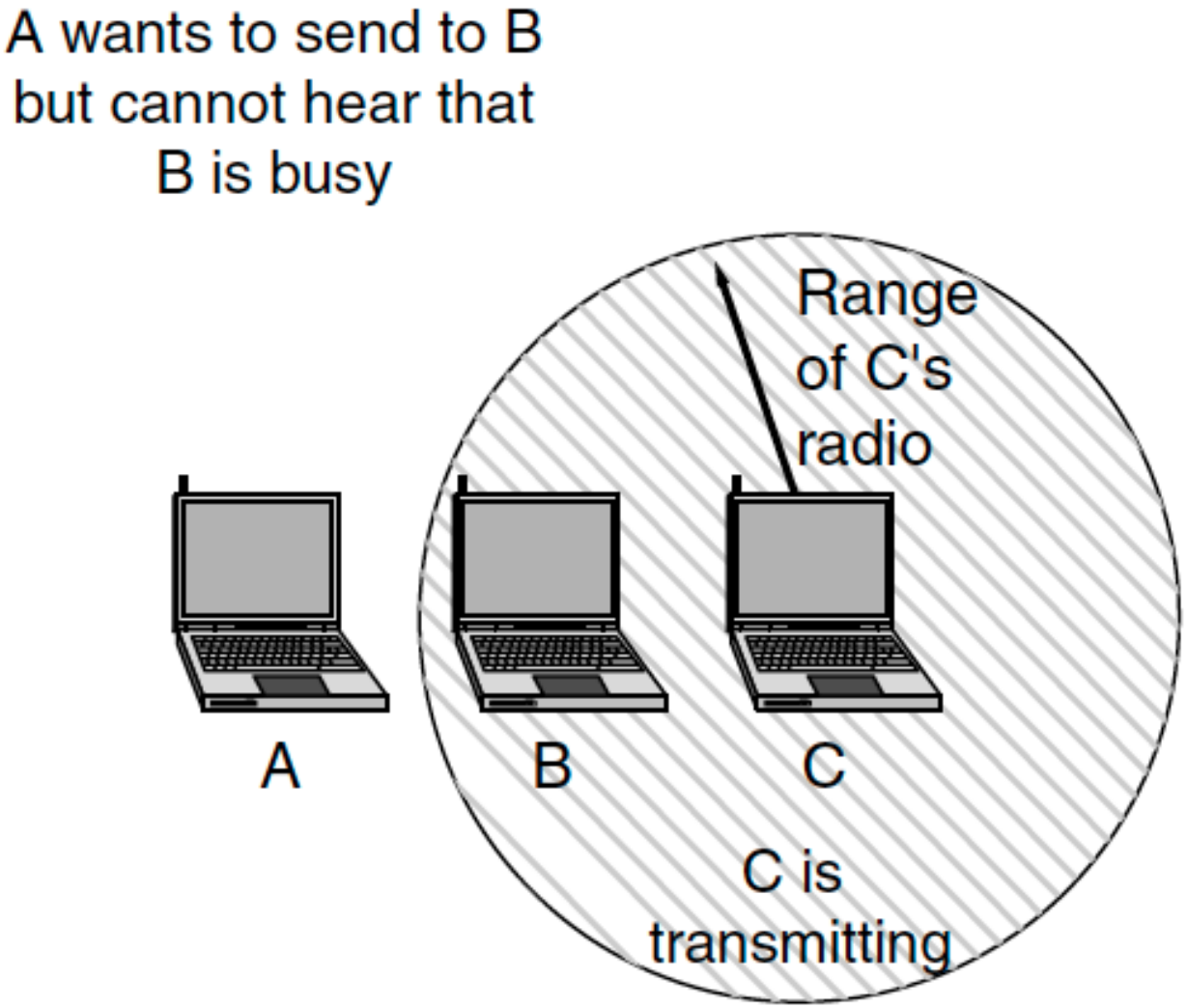

그림 : Hidden terminal

WiFi: Exposed terminal problem

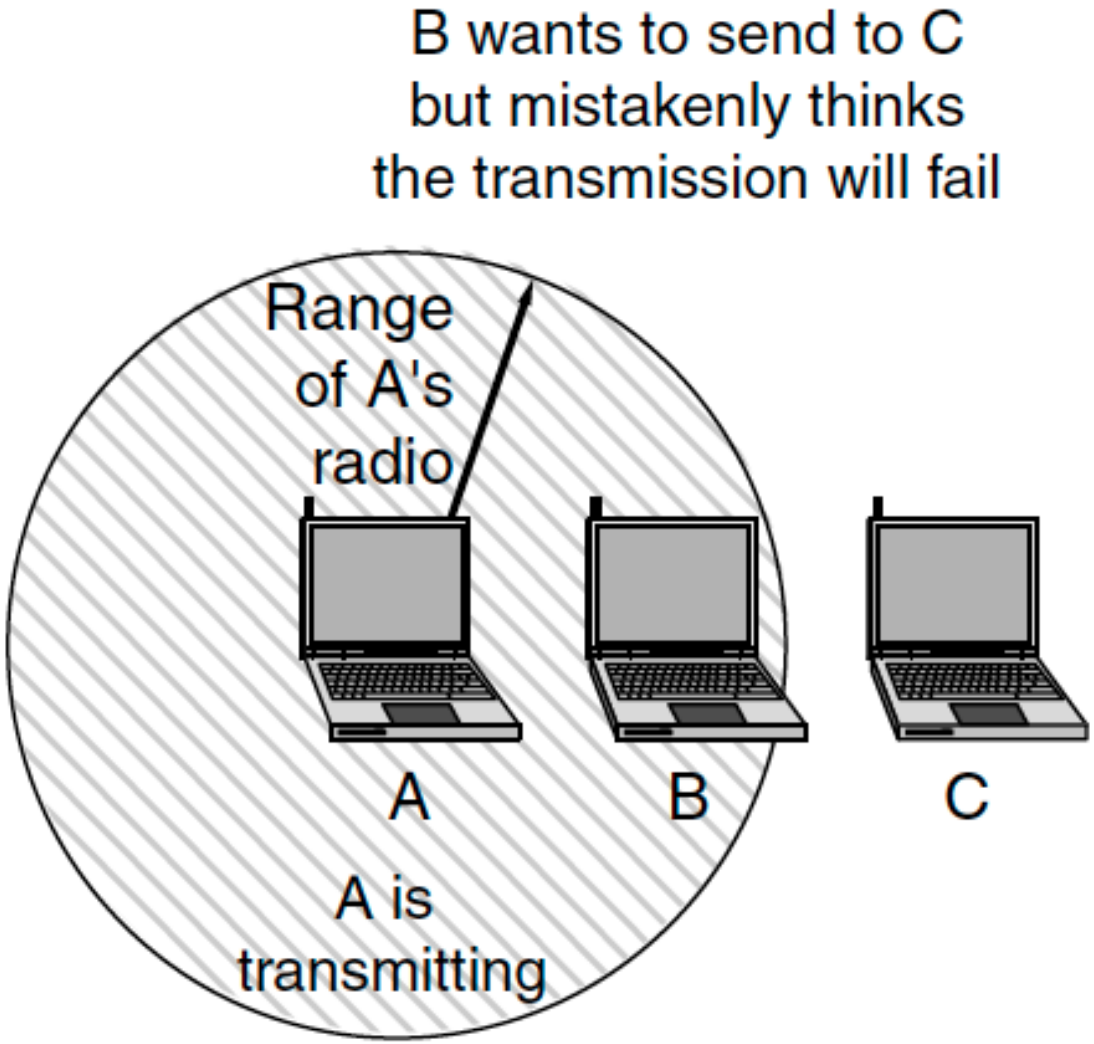

그림 : Exposed terminal

WiMAX (Broadband Wireless)

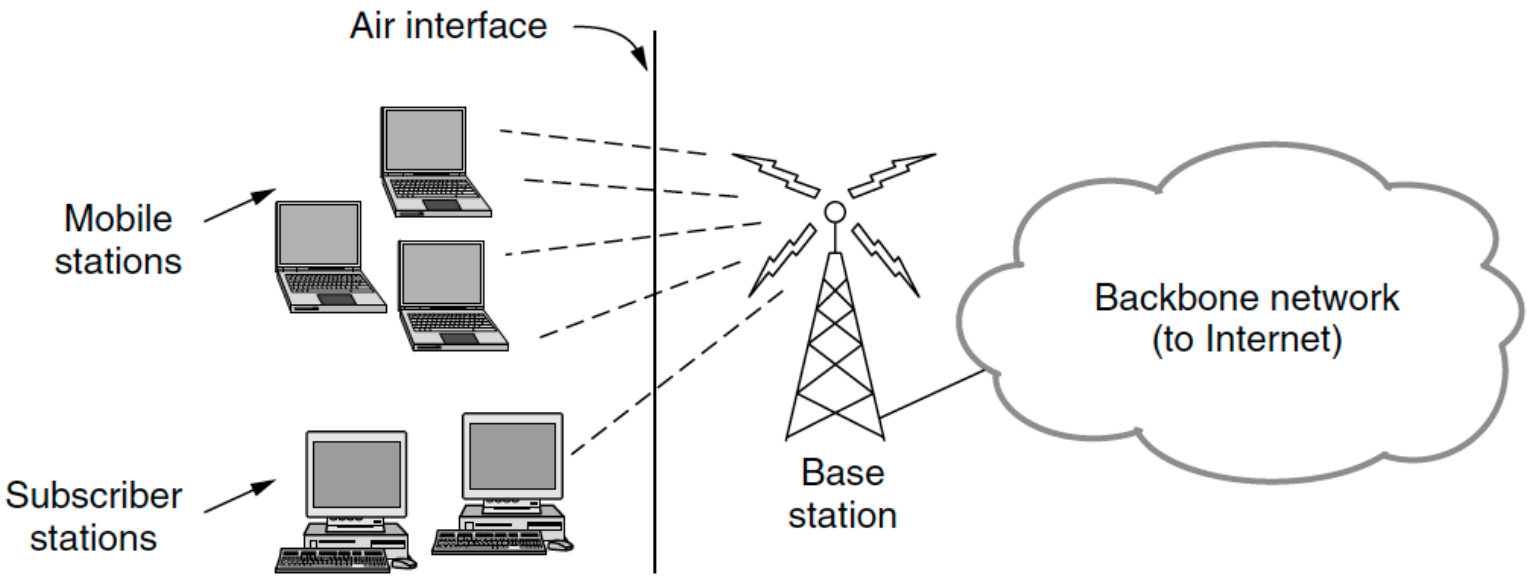

그림 : WiMAX

단방향, 양방향

- 단방향: 데이터 전송시 수신 불가
  - 장점: 저비용
  - 단점: 보내고 받기 위한 규약 필요
- 양방향: 데이터 전송과 수신 동시에 가능
  - 장점: 언제든 데이터를 보내고 받을 수 있음
  - 단점: 고비용
- 대부분의 네트웍 응용: 요청-응답 (CS, P2P)

$\longrightarrow$ 단방향으로 충분

단방향 통신

- 이상적인 경우
  - 요청 (데이터 전송) $\longrightarrow$ 응답 (ACK/요청)
  - piggy back
- 요청데이터 전송에서 오류
  - 데이터가 다 도착했는지
- 응답데이터 전송에서 오류
  - 새로 요청해도 되는지