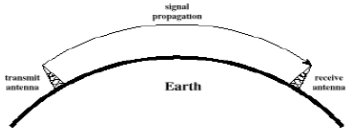
CS4222 Cheatsheet

Introduction to Wireless Networking

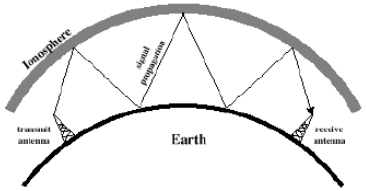
Wave Propagation

1. Ground-Wave Propagation



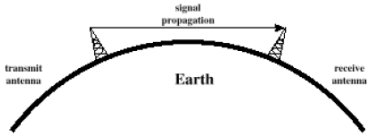
- Follow contour of earth, can propagate considerable distances, frequencies up to 2MHz. Suitable for low frequency, long range transmission.
- E.g: AM Radio (Medium Wave: 520 kHz - 1,610 kHz)

2. Sky Wave Propagation



- Signal "reflected" from ionized layer of atmosphere back down to earth
- Signal travel a number of hops, back and forth between ionosphere and earth's surface
- Reflection effect caused by refraction, 3 to 30MHz??
- Examples: Short wave AM (1.711MHz – 30.0MHz), amateur/CB radio

3. Line of Sight propagation



- Transmitting and receiving antennas **must be within** line of sight
- Note: For transmission, longer distances usually mean lower bitrate

Line of Sight Equations

- Maximum distance between 2 antennas for LOS propagation:

$$3.57\left(\sqrt{Kh_1}+\sqrt{Kh_2}\right)$$

- h_1 = height of antenna one
- h_2 = height of antenna two
- K = adjustment factor to account for refraction (default value is 4/3)

→ The distance returned by answer is in km

Some Examples

- Transmitter is 50m high, receiver is at ground level (0m), what is the maximum distance between receiver and transmitter using LOS?
  - $D = 3.57\ (\ 4/3 * 50\ )^{1/2} = 29.15km$
- Raise the receiver by 10m, how high does the transmitter antenna needs to be to maintain same transmission distance?
  - $29.15 = 3.57\ (\ (4/3 * H)^{1/2} + (4/3 * 10\ )^{1/2}\ )$
  - $H = 15.26m$
- Raise the transmitter and receiver are placed at the same height and the transmission needs to be at least 30km, how high do the antennas need to be placed?
  - $30 = 3.57(2(4/3 * H)^{1/2})$
  - $H = 13.24m$

Free Space Propagation Model

- For line-of-sight path

$$P_r = G_r G_t \left(\frac{c}{4\pi f_c d}\right)^{\alpha} P_t$$

  - f_c is the center frequency in Hz
  - c is speed of light
  - d is the distance between transmitter and receiver
  - α is the path loss component
  - G is antenna gain

→ With this formula we can estimate power at the receiver based on the power at the transmitter

→ power and frequency (fc) are inversely correlated. Higher frequency → lower power → lower bitrate

→ higher power at receiver → higher bitrate → higher quality at receiver

Types of Antennas

Dipole antenna, half-wave dipole

Dipole antenna - simplest practical antenna, two wires pointed in opposite directions arranged either horizontally or vertically, with one end of each wire connected to the radio and the other end hanging free in space
Half-wave dipole - A more efficient variation is the half-wave dipole, which radiates with high efficiency when the signal wavelength is twice the electrical length of the antenna

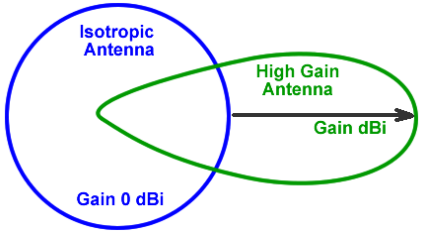
→ So, for an antenna like 6cm, it is good at producing waves with 12cm wavelength, i.e. speed(light)/lambda = ~2.4GHz frequency

Half Wave Dipole Antenna Length

Frequency	Half Wavelength
1 MHz	150m
10 MHz	15m
100 MHz	1.5m
433 MHz	~35cm
900 MHz	~17cm
2.4 GHz	~6cm
5 GHz	~3cm
60 GHz	~2.5mm

Antenna Gain

- increase power output in a particular direction.



→ Isotropic Antenna (theoretical antenna where intensity radiated in all direction is equal)

→ Depending on physical size and shape of antenna, we can affect the effective area of transmission(?)

$$G = \frac{4\pi A_e}{\lambda^2} = \frac{4\pi f^2 A_e}{c^2}$$

- G = antenna gain
- A_e = effective area
- f = carrier frequency
- c = speed of light (≈ 3×10^8 m/s)
- λ = carrier wavelength

Path Loss

- Power at receiver is proportional to power at transmitter and inversely proportional to distance between the sender and receiver. Does this mean that radio propagation is purely circular (should fall equally across all distances)?

- Path Loss in dB $10\log(\frac{P_t}{P_r})$
- Also simplifies to $P_r \sim \left(\frac{1}{d}\right)^{\alpha} P_t$
- Implies a circular disk model for radio propagation. True in practice?

→ No, when the orientation of the antenna changes, it also affects the power at receiver in practice

→ Estimate path loss using the first formula above (we x10 just to make the number a bit bigger)

→ Second formula is simply from FSPM

Path Loss Exponent

Environment	Path Loss Exponential (α)
Free Space	2
Urban area cellualr radio	2.7 to 3.5
Shadowed urban cellular radio	3 to 5
In building LOS	1.6 to 1.8
Obstructed in building	4 to 6
Obstructed in factory	2 to 3

Line of Sight Transmission

- Follow free space loss model

→ Signal disperses with distance, larger loss for higher frequencies, loss at higher frequencies can be compensated with antenna gain.

- Atmospheric Absorption

→ Water vapour and oxygen absorb radio signals

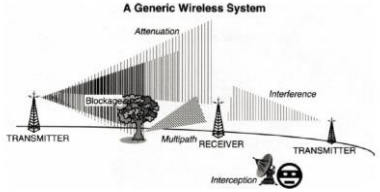
→ Water (absorb?) greatest at 22GHz, less below 15 GHz

→ Oxygen absorb greatest at 60 GHz, less below 30 GHz

→ Rain and fog scatter radio wave

Mobile Radio Propagation

- Blockage, attenuation, multipath, interference, interception



→ Attenuation is the loss of signal strength in a connection

→ Depending on frequency of transmission, building/obstacle penetration strength differs. Low frequency penetrates better

→ Receiver can receive multiple copies of a transmission with different delays due to multiple path transmission, causing signal to arrive at more than 1 different timings.

Multipath Propagation

1. Reflection (off earth, buildings)
2. Diffraction (radio wave bend around obstacles)
3. Scattering (Due to obstacles smaller than wavelength of a wave, like foliage, signal is scattered into several weaker signals)

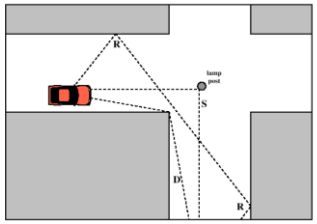


Figure 5.5 of Beard and Stallings (R – Reflection, S – Scattering, D - Diffraction)

→ examples of reflection, diffraction and scattering

Effects of Multipath Propagation

→ **Multiple copies of a signal may arrive at different phases. If phases add destructively, the signal level relative to the noise declines, making detection even more difficult**

→ Intersymbol interference. One or more delayed copies of a pulse may arrive at the same time as the primary pulse for a subsequent bit. Basically, one or more symbols interfere with other symbols as noise or a less reliable signal, usually caused by multipath propagation

Fading

- Time variation of received signal power caused by changes in transmission medium or paths

- **Large scale fading**

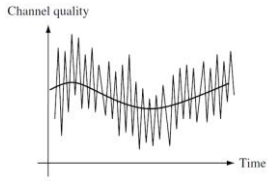
→ Over longer distances (much longer than wavelength), large scale fading represents the average signal-power attenuation or path loss due to motion over large areas, impacted by environmental changes

- **Small scale fading**

→ Doppler spread causing signal performance to change across a short time due to movement of users and obstacles.

→ Multipath fading causes the signal to vary due to combination of delayed signal arrivals.

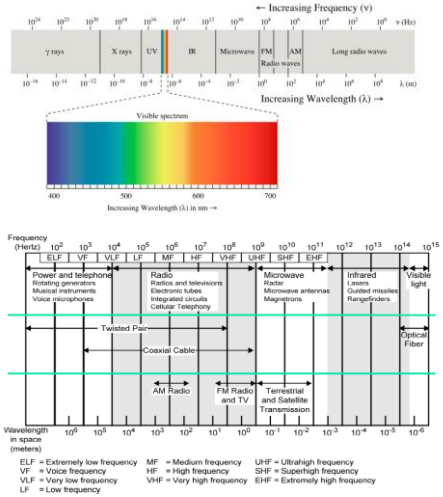
Channel quality varies over multiple time-scales.



→ I think the jagged one is representing short scale fading, the curved line is large scale fading

→ Over long time the quality changes slowly, over short time can have sharp variations

Electromagnetic Spectrum



Signal Frequency Bands

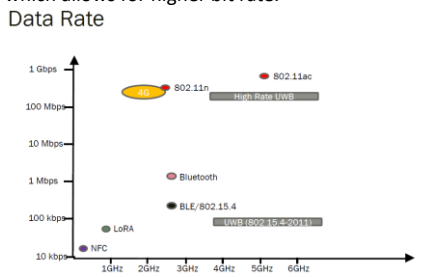
Usage	Frequency Band
AM Radio	(Medium Wave) 500 - 1600 kHz with 9-10 kHz spacing
FM Radio	87.5 - 108.0 MHz, channel spacing varies, usually > 100kHz
VHF (TV)	54 - 216MHz
UHF (TV)	470 - 806 MHz, 6MHz channel spacing
Data Transmission (e.g. Bluetooth, Wifi, cellular)	300MHz – 6GHz, 60GHz 802.11ac defines use of 160MHz band 802.11ad runs in the 60GHz band

Shannon Capacity
→ Tight upper bound on the rate at which information can be reliably transmitted over a communication channel
- Bandwidth available (B)
- Quality of channel (Level of noise, SNR)
$C = B * \log_2(1 + SNR)$
→ Intuitively, capacity (C) increases with available bandwidth (linearly) and increases with SNR(log scale)

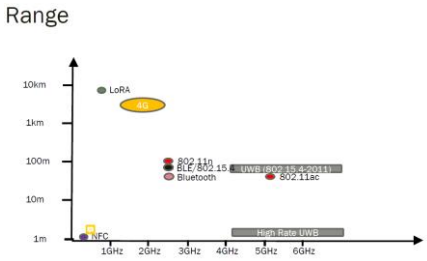
Free Space Propagation Model + Shannon Capacity
→ The main relationship is in the Pr of FSPM and SNR of Shannon. The higher the Pr, the greater the signal strength and higher the SNR.
→ By increasing the power, we can result in higher range and data rate.
→ By increasing B, we can increase data rate (C).
→ By increasing frequency of transmission, the range of the transmission decreases.
→ We can increase data rate by increasing antenna gain with better antenna design

Types of Wireless Networks
- Wireless Personal Area Network (WPAN)
  - *NFC (Near-Field Communication), RFID, Bluetooth, ZigBee*
- Wireless Local Area Network (WLAN)
  - *IEEE 802.11 (WiFi)*
- Wireless Metropolitan Area Network (WMAN)
  - *IEEE 802.16 (WiMAX)*
- Wireless Wide Area Network
  - *GSM, 3G/4G/LTE*

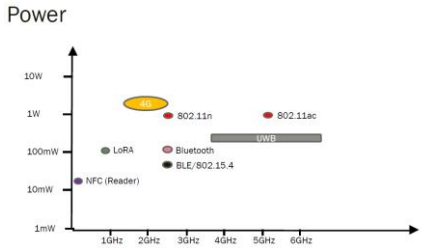
Characteristics of Wireless Networks
- Range, Data Rate, Power, Frequency Band.
→ Devices that require less power tend to require lower data rate (not much things to transmit), so, we can send these at lower frequencies for a longer range.
→ Technically, bit rate does not depend on carrier frequency. However, with higher frequencies, we can have higher bandwidth, which allows for higher bit rate.



→ We can see that low bit rate protocols like nfc and lora transmit at low frequencies. 4G (2.4GHz and 5GHz) transmit at much higher frequencies. 5GHz has shorter range but higher bit rate.

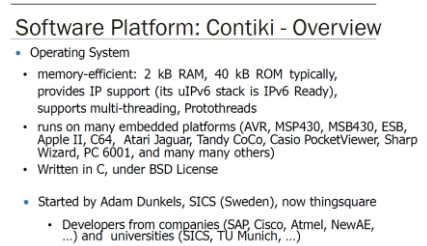


→ Why is the range on NFC so short when frequency is so low? We achieve this by having very very low power on NFC



→ 4G – high data rate, high range. Done with high power

Contiki
- CC2650 Sensortag, ARM cortex M3 32-bit processor, 48MHz clock speed
  - Memory: 8KB of SRAM for Cache, 20KB of SRAM, 128KB of In-System Programmable Flash
  - Network: 6LoWPAN, Bluetooth Low energy, RF4CE, ZigBee
  - 10 sensors including support for light, digital microphone, magnetic sensor, humidity, pressure, accelerometer, gyroscope, magnetometer, object temperature, and ambient temperature



→ Event driven at kernel level with optional threading
→ Usually for programming, we are more used to having some kind of multi-threaded model where the kernel can block and unblock and perform some kind of context switching. To achieve this, each thread will have its own stack to store information, but its quite expensive on embedded systems
→ For Contiki, we program in a "multithreaded" model but under the hood it compiles into event driven model which works much better for hardware devices
→ Even so, you cant write functions that are too "long" else when it compiles, other programs might get starved
→ No memory protection in Contiki
→ can use memb_alloc and memb_free for dynamic memory management
→ Provides MAC protocol right off the box, ContikiMAC

Mobile Devices/IoT & Energy Consumption
→ Advancement in MEMs technology has made it possible to build very small sensors, with low computation power, with relatively low bit-rate
→ Designed to execute a specific task (or small number of tasks)
→ Deployed in large quantities (each node is small and cheap), densely and in a highly redundant fashion. Each sensor node can hear many other sensor nodes.
→ Why deploy so many? Make up for the lack of generality of the node functions, increase reliability/coverage of entire network
→ Challenge? We need to design network protocols that can coordinate these nodes, and they must run on very small network(??)

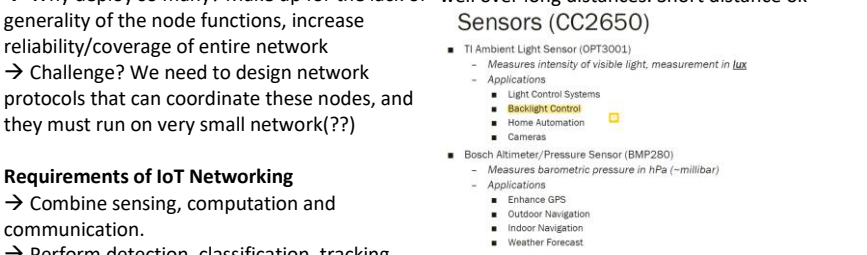
Requirements of IoT Networking
→ Combine sensing, computation and communication.
→ Perform detection, classification, tracking

Benefits of Wireless Networking?
- Lower cost of wiring, retrofitting (install new part to system that was once unavailable to it) of existing systems, can withstand harsh conditions.

What is IoT?
- The interconnection of uniquely identifiable embedded computing devices within the existing Internet infrastructure.
→ Reachable through internet protocols so that any device can be contacted.
→ Any thing can be a device and the device is smart (can compute, sense and communicate)
→ Can perform many tasks and can be shared.

→ Raspberry Pi, Arduino, Microbit, CC2650. First 3 has more storage, memory, higher computation power than CC2650, but more power hungry

CC2650 Sensors
→ Gyroscope: Detect rotation about x,y,z axes in degree per seconds (dps)
→ Accelerometer: output for 3-axis in g, when placed on flat surface, will output 0g on x and y axes but 1g on z (g as in acceleration due to gravity, 9.81ms^{-2})
→ Magnetometer: Detect terrestrial magnetism in 3-axis. Unit is in μT
Applications: Localization, dead reckoning (process of calculating current position of some moving object by a previously determined position), motion detection, gaming, health and fitness.
→ How to perform localization? One possible way is the use of accelerometer to estimate, by getting the speed. However, by integrating acceleration into speed, we miss out on the "constant" C, so the estimation may not work well over long distances. Short distance ok



→ Light, pressure sensors.
→ Sensor readings are generally very noisy and need to be cleaned.
- Some advice:
  - *Take note of the "highest" sampling rate supported by the device, higher does not always means better*
  - *Printf are good for debugging, but too much I/O can slow down the processing significantly*
  - *Some smoothing/averaging can be useful (low pass filter)*
  - *Do not rely your decision only on a small number of samples*

Some quick revision on circuits
Voltage (V): SI unit for electromotive force
Current: Ampere (A), SI unit of electric current
Power (W) = IV
Energy (J) = Pt
→ Energy is like "distance" and power is like "speed"

Battery
→ Battery Lifetime measured using watt-hour (or Wh), which is a unit of energy equivalent to 1W of power expended for 1hour of time.
E.g. Ipad2: 25Wh
→Ampere-hour (Ah) is a unit of electric charge equal to the charge transferred by a steady current of 1A flowing for 1hour.
E.g. Samsung S4: 2600mAH (3.7V)

Energy Consumption of Components in Mobile Nodes
- Mobile Nodes are battery powered, and energy constraints is fundamental to the design of low power network protocols.
→ System components: Network, screen/monitor, computation, sensing

Network Energy Consumption

TABLE IV
CURRENT CONSUMPTION OF CHIPSETS FOR EACH PROTOCOL

Standard	Bluetooth	UWB	ZigBee	Wi-Fi
Chipset	BlueCore2	XS110	CC2430	CX83111
VDD (volt)	1.8	3.3	3.0	3.3
TX (mA)	57	~227.3	24.7	218
RX (mA)	47	~227.3	27	215
Bit rate (Mb/s)	0.72	114	0.25	54

Table taken from Jin-Shyan Lee, Yu-Wei Su, and Chung-Chou Shen, "A Comparative Study of Wireless Protocols: Bluetooth, UWB, ZigBee, and Wi-Fi," IECON, 2007

A mobile device with a battery capacity 9.62Wh transmits continuously on:
- WiFi:
  - Considering only network energy consumption
    - power consumption: 3.3V*219mA = 722.7 mW
  - Duration = 9.62/0.7227 = 13.3hrs
  - In practice, the duration is much shorter due to energy consumed by other components
- Bluetooth: (1.8V * 57mA = 102.6mW)
  - Duration = 9.62/(1.8*0.057) = 93.76hrs

→ Calculate power consumption using P = IV, duration by taking battery capacity over power consumption
→ For networks, devices **rarely** transmit continuously! In fact, we have different power states which allows power consumption to be reduced.

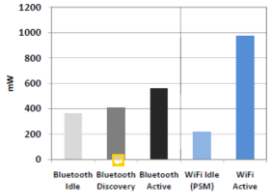
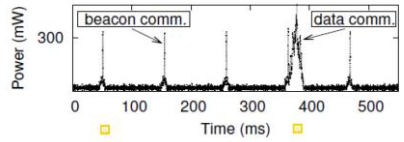


Figure 1: Power consumption of Bluetooth and WiFi radios in different states.

→ Observe wifi and Bluetooth active vs idle's current consumption.
→ Note: Bluetooth discovery mode is a low power, neighbor discovery mode

Wifi power consumption (Using Galaxy S4 as study)



→ Beacon timings are fixed no matter whether there is data transmission or not
→ Beacon – send small packet to AP, go back to sleep (to see if there are any messages for me). Sending packet is quite expensive (in terms of power), sent once in the order of 20 to 30 ms.
→ Wifi state transition for beacon communication (done by firmware) is relatively fast.

→ State transition for data communication is relatively slow due to driver/software overhead.

Classic Bluetooth

→ Always on, short range radio
→ It was initially designed as a way to let laptop make calls over a mobile phone
→ Data Rate: 1mbps – 3mbps (high data rate because it was initially meant for streaming)
→ 3 bluetooth classes and current consumptions:

- Three classes
  - Class 1: 100mW
  - Class 2: 2.5mW
  - Class 3: 1mW
- ISM band 2.4GHz (2.402GHz – 2.480GHz)

→ Same ISM band as wifi, can interfere
→ Time division duplex mac protocol(?), master-slave polling. (slaves talk to master, the access point)
→ Can support synchronous and async traffic
→ Point to Point (Only talk to master. It's a link layer jargon, not same as peer to peer)
→ Master is like phone, our speaker/earphones etc is like slave. Many slaves talk to master

Timing Considerations for Classic Bluetooth

- Time for slave and master to pair: 3 – 20s (slow due to handshake)
- Sleeping slave → active = 3s typically
- Active slave channel access time = 2ms typically

Bluetooth Low Energy (BLE)

- Data Rate: 125kbps – 2Mbps
- Similar specs to classic Bluetooth

- Classes
  - Class 1,1.5: 10-100mW
  - Class 2: 2.5mW
  - Class 3: 1mW
- ISM band 2.4GHz (2.402GHz – 2.480GHz)
- Point-to-point, broadcast, mesh

→ Typical Bluetooth now with reduced data rate, lower energy and supports mesh routing.
→ Behaviour similar to ZigBee

Bluetooth Low Energy (CC2640)

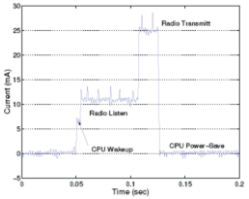
- Tx Current: ~ 6.1mA (0dBm)
- Tx Current: ~ 15mA
- Latency: 3 ms
- Sleep current ~ 1μA
- Designed for sending small chunks of data

→ Can just wake up and send, master no need look for slave so latency much lower, lower overhead on pairing

Why do we need different radio states?

→ Transmit power is high and listening is not as high
→ These consumptions are determined by mac protocol
→ After transmission can go back to sleep almost immediately

Current Consumption (MICA2)

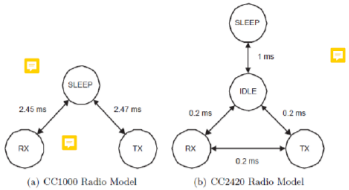


What are the desirable properties for radio to operate in low power?

→ Observe the different current consumption at different states, it is significant savings
→ We only send for a few ms, don't need to be awake for so long
→ state transition is fast

Radio States

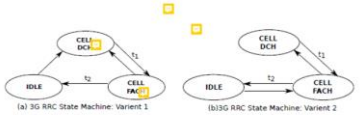
- Energy can be saved by putting network interface into low power states when not active.
→ Possible states: Sleeping, idle, transmitting, receiving.



Difference between the 2 radio models:
→ Left one has much slower transition speeds, overhead of waking up is much higher. This overhead can be considered very high if the data rate of the sensor is high. However, if the data rate is low, then a few ms is just a few kb so not as costly
→ Sleep vs idle: idle still use more power than sleep. Tradeoff of power consumption vs latency. Idle uses more power than sleep but can state transition faster.

3G/4G radio states

Cellular Network: 3G/4G Radio State



- State transition in cellular network incurs high overhead
- Radio stays in high power states in anticipation of data transfer

→ CELL FACH – Ready state, CELL DCH – active state.
→ idle to active → there is a control packet exchange using wifi protocol?
→ power consumption high in general for cell networks

Cellular "Tail" Behavior

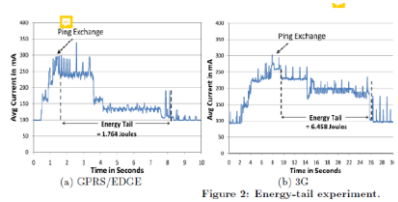


Figure 2: Energy-tail experiment.

→ As we can see, for both GPRS and 3G, there is a tail behaviour where current consumption is quite high, and takes about 20 seconds after ping exchange to go to sleep. We cannot go to sleep faster because waking already has high overhead and takes quite a while, so if we keep waking up it is very expensive
→ This tail behaviour causes cellular radio to be very inefficient for infrequent transfer of small amounts of data.

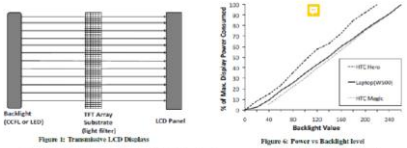
Duty Cycling Examples

- Compare WiFi (100Mbps) and ZigBee/802.15.4 (250kbps), 1s cycle, assume same voltage
- WiFi, three states:
  - Transmit: 280mA
  - Receive: 180mA
  - Idle: 10mA
- 802.15.4, three states:
  - Transmit: 30mA
  - Receive: 18.mA
  - Idle: 0.426mA
- 100% utilization (50% transmit, 50% receive)
  - WiFi: (180+280)/2 = 230mA
  - 802.15.4: (18.8+30)/2 = 24.4mA (10.6% of WiFi)
- 10% utilization (5% transmit, 5% receive, 90% idle)
  - WiFi: 230*0.1 + 10*0.9 = 32mA
  - 802.15.4: 24.4*0.1 + 0.426*0.9 = 2.823mA (8.8% of WiFi)
- 1% utilization (0.5% transmit, 0.5% receive, 99% idle)
  - WiFi: 230*0.01 + 10*0.99 = 12.2mA
  - 802.15.4: 24.4*0.01 + 0.426*0.99 = 0.665mA (5.46% of WiFi)

→ Transmission for example below is 100Mbps (wifi) and 250kbps (Zigbee)

- Wakeup 10s, sends 10,000 bytes of data
- WiFi (no overhead)
  - Transmit time = 10*8/100000 = 0.8ms
  - Average current drawn
    - (280*0.0008 + 10*9.9992)/10 = 10.0216mA
- 802.15.4 (no overhead)
  - Transmit time = 10*8/250 = 320ms
  - Average current drawn
    - (30*0.32 + 0.426*9.68)/10 = 1.372mA

Power Consumed by LCD Display

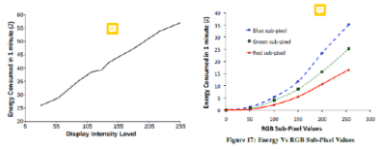


Most of the power in an LCD display is consumed by the backlight, the light filter requires comparatively less power.

→ Generally a linear trend for brightness vs power consumption

Power Consumed by OLED Display

→ Display without backlight (black consumes no energy)
→ OLED consumes less power than LCD display when image is dark but more when bright
→ Different color consumes different amt of energy



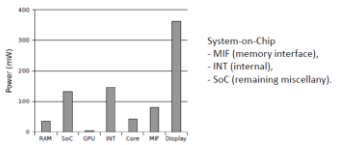
Energy Characteristics of Different Sensors

Sensor	Galaxy S3	Galaxy Nexus
Accelerometer	0.23mA	0.139mA
Light	0.2mA	0.75mA
Pressure	0.045mA	0.67mA
Proximity	1.3mA	0.75mA
Magnetic field	6.8mA	4.0mA
Gyroscope	6.1mA	6.1mA
Orientation	7.8mA	10.2mA

→ Some ways to save energy:
1. Reduce sampling frequency (it saves energy not just from sampling but also from processing)
2. Use low power sensors whenever possible (e.g. detect motion using accelerometer vs gyroscope, assuming similar accuracy)
3. Offload to hardware (sensor batching)

Experiment Results for Galaxy S3 on power consumption

Airplane Mode (805mW)



→ Most used on display.

Gaming (No network traffic)

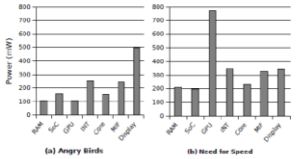
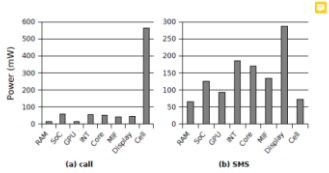


Figure 4: Gaming power consumption for: (a) Angry Birds (2D), $P_1 = 1516$ mW; and (b) Need for Speed (3D) $P_1 = 2425$ mW.

→ WIFI turned off (I think).

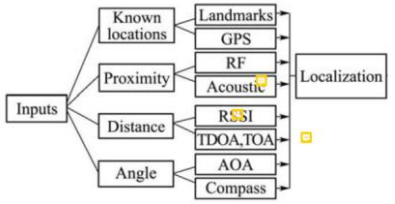
Voice Call vs. SMS



→ SMS actually uses more power than phone call, because display will wake up
→ For web browsing, display is also the dominant power drain. Moral of the story: Display uses a lot of energy
→ Drained battery can also affect packet reception rate between 2 battery powered devices communicating over wireless links

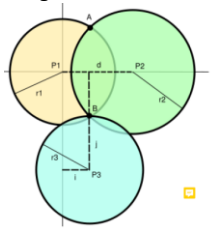
Localization

→ Determine the physical coordinates of a group of sensor nodes.
→ Coordinates can be global (e.g. using GPS) or relative (some rotation/reflection/translation) away from global coordinate systems
Overview:



→ TDOA = time difference of arrival, TOA = time of arrival.
Advantage of tdoa: Can just take the difference to estimate distance. For toa, there must be good synchronisation between the 2 devices, so time must be accurately recorded, more work needed.

Range based localization



Basic Idea: Trilateration

Beacon/Anchor/Landmark
• Ranging

How many beacons do you need?

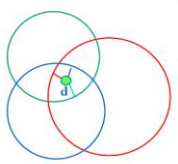
→ Each beacon gives information: 'I am r1/r2/r3 distance away from you'. We estimate location of person based on overlap of the 3 distances and the beacons' locations.
→ Beacons needed: 1 + dimensions

Anchor (Beacon) Nodes

- Necessary to localize a network in global coordinate system or relative coordinate system
- 3 (non-collinear) beacons to define coordinates in 2 dimensions, 4 for 3-d coordinates
- Beacon placement has significant impact on localization. Accuracy is improved if placed in a convex hull around the network with additional beacon in the center

Problem: What if there is no point (x,y) that satisfies all constraints?

→ There might be some estimation error in the measurements of each beacon.



There is no point (x,y) that satisfies all constraints.

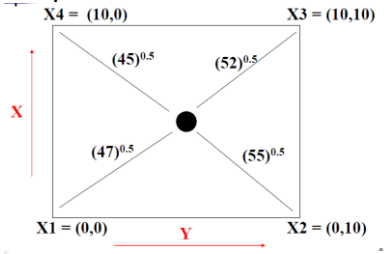
Let d_{ij} denotes the distance between anchor i and node j

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

Estimation Error:

$$E_j = \sum_{i=1}^{m} (d_{ij} - \hat{d}_{ij})^2,$$

Example Calculation:



Write down the constraints:

- $x^2 + y^2 = 47$
- $x^2 + (10 - y)^2 = 55$
- $(10 - x)^2 + (10 - y)^2 = 52$
- $(10 - x)^2 + y^2 = 45$

No value of x and y meet all 4 constraints.

Rewrite the equations (x_1=0, y_1=0, x_2=0, y_2=10, ….)

$(x - x_i)^2 + (y - y_i)^2 = d_i^2$

$-2xx_i - 2yy_i + x^2 + y^2 = d_i^2 - (x_i^2 + y_i^2)$

Let $R = x^2 + y^2$

$$-2x_i x - 2y_i y + R = d_i^2 - (x_i^2 + y_i^2)$$

→ x and y – location of the object, Xi and yi – location of beacon

→ Expand out equation to get the final equation,

Recall:

When $Ax = b$ has no solution, multiply by A^T and solve $A^T A\hat{x} = A^T b$.

In this case,

$$A = \begin{bmatrix} 0 & 0 & 1 \\ 0 & -20 & 1 \\ -20 & -20 & 1 \\ -20 & 0 & 1 \end{bmatrix} \quad x = \begin{bmatrix} x \\ y \\ R \end{bmatrix} \quad b = \begin{bmatrix} 47 - 0 \\ 55 - 100 \\ 52 - 200 \\ 45 - 100 \end{bmatrix}$$

x = 5.125, y = 4.625, Ej = 0.004578664

Note, R = 47.25, $x^2 + y^2$ = 47.65625

→ The matrix is following the blue equation, and each row is each 1 of the 4 constraints ($x^2 + y^2$ = 47…). For example, b has 55 – 100 because of $d_i^2 - (x_i^2 + y_i^2)$.

Ranging – TOA Time of Arrival

- Depends on speed of signal propagation, the distance to be measured and how synchronised are the clocks (clock accuracy)

Example:

- **Speed of light (c) ~ $3x10^8$ m/s**
- **Distance travel in 1μs = 300m**
- **Clock accuracy needed for meter accuracy (1/c) ~ 3ns**

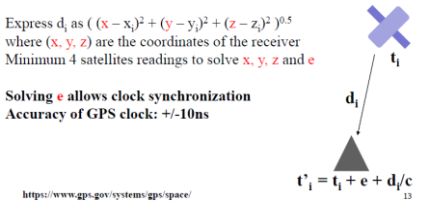
- **Speed of sound (v) ~ 330 m/s**
- **Distance travel in 1ms = 0.33m**
- **Clock accuracy needed for meter accuracy = 1/v ~ 3ms**

Global Positioning System (GPS)

- A set of >= 24 satellites, orbiting at altitude of 20,200km. GPS have very precise clocks.

Time of message reception = timestamp of satellite message + clock drift + distance/speed (time travelled).

→ Clock drift, e, is the same for any satellite, because all satellites are in sync with each other.

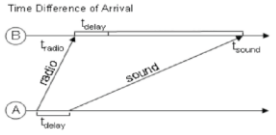


→ Need 4 satellite readings since there are 3 dimensions of the receiver's coordinates.

→ accuracy +- 10ns means your mobile phone's sync is +- 10ns of actual time

TDoA Time Difference of Arrival

- Exploit differences in speed of 2 frequency signals, such as Radio and audio.
- Send a radio message, wait some time t_delay, then send audio "chirp" sound.
- Listener hears radio signal, record time t_radio, hear audio signal, record time t_sound.



- **Equation 1.2 in reference does not work**
- **Should be time diff = $d/S_{Sound} - d/S_{Radio} - t_{delay}$**
- **Works well in LOS environment**
  - **Cricket (uses ultra sound) can obtain close to cm accuracy**
- **Requires additional hardware and works better with calibration**

→ We can measure the time difference, and since we know what is the time delay, the speed of sound and light, we can estimate the distance of the transmitting node from the receiving node

RSSI

→ RSSI – received signal-strength indication

→ Varies with $(1/d)^{\alpha}$

→ We assume that nodes that are closer have higher rssi than nodes that are further apart.

→ In practice, there is noise, so signal strength is non-uniform. Can differ over different surface (water, grass etc), obstacles such as walls etc

Model Based Distance Estimation – RSSI

- **Based on free space propagation model with additional noise model**
- **Estimate model parameters to improve accuracy**

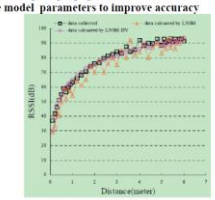
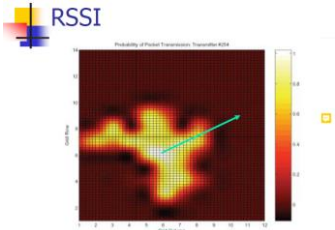


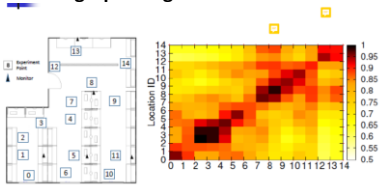
Figure 3. The curve of RSSI's movement tendency

- RSSI is on an absolute scale – that's why it increases here as distance increases.



→ RSSI isn't very good for measuring distance, but if you are nearby or both nodes are receiving high rssi (white part), can confidently say that they are close together. The yellow parts may not be so clear.

WiFi Fingerprinting



→ Determine location of node based on rssi.

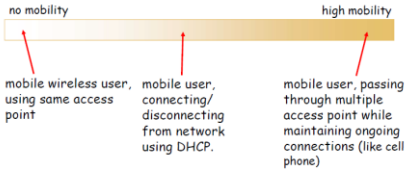
→ Middle values is about how rssi value change over time for 2 nodes that are at the same position (1-1,2-2…), don't change that much

→ Conclusion from this experiment: Same position has high similarity, different positions are quite different, so it can kind of do fingerprinting

→ Rssi values might differ for the same point on different days due to environment factors, number of access points, power of access points

Mobility

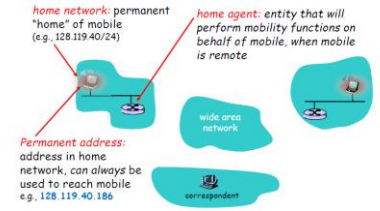
- In networking, mobility is a spectrum that refers to how mobile the nodes are in terms of its relative location



- Middle one would be something like a user who is intermittently moving (e.g. in school, from one tutorial room to another, to canteen etc)
- We want to know how to find a node/device that is mobile, and how to route packets to such a mobile node (internet or cellular network)

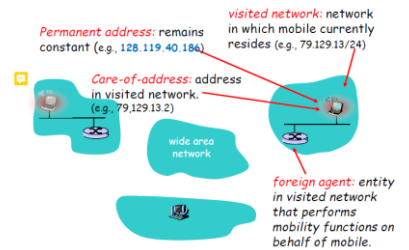
Some Mobility Jargon / Vocab

- Home network: Permanent "home" of a mobile device
- Permanent address: Address in home network which can always be used to reach mobile
- Home agent: Entity in home network that performs mobility functions on behalf of mobile when mobile is remote. It becomes a proxy and does routing for the mobile node



- Visited Network: Network in which mobile currently resides (if it is remote I think)
- Foreign agent: Entity in visited network that performs mobility functions (proxying) on behalf of mobile.
- Care-of-address: Address of mobile node in visited network
- Correspondent: Node that wants to communicate with mobile node.

→ Permanent address is the same in foreign network!



How to contact a mobile friend?

- Consider a user who frequently changes address, we need some way to know the most updated location of the user. Simply having some global directory/table is NOT scalable.

→ We let the end-systems handle it: perform indirect routing by going through home agent, or direct routing where correspondent can directly interact with mobile

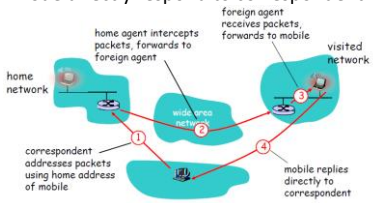
Step 1 to Mobility: Registration

- When mobile node visits foreign network, it contacts foreign agent on entering visited network.
- Foreign agent then tells home agent: "this mobile is resident in my network."

→ Now, foreign agent will know about mobile and home agent knows location of mobile (home agent will know where to forward packets to if correspondent reaches mobile!)

Mobility via Indirect Routing

- Correspondent sends packets to home address of mobile, which the home agent intercepts. It knows that the mobile node is not in the home network.
- It sends the packets to the foreign agent in which the mobile node is residing in, who forwards the packets to the mobile node.
- Node directly respond to correspondent.



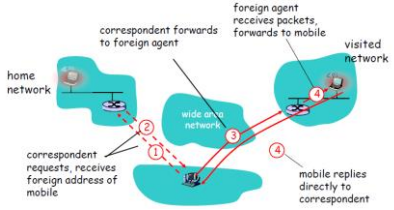
→ Why cant correspondent just directly send message to mobile node after? Because in indirect routing, the motivation is that the correspondent does not need to know of the mobile ip protocol. It just knows it is using ip protocol, and everything else is handled for him.

Indirect Routing: Pros and Cons
- Mobile uses 2 addresses
→ Permanent address: Correspondent sends packets to this address (and hence, the location of the mobile node is **transparent** to correspondent)
→ Care-of-address: This is used by home agent to forward datagrams to mobile
→ Functions of the foreign agent might be done by the mobile itself (foreign agent just routes packets only)
→ This is called **triangle routing:** correspondent-home-network-mobile. Quite inefficient! It is especially so if correspondent → home network is a long hop, yet correspondent → mobile node is quite a short hop

Indirect Routing: What if mobile node now moves to another network?
- Simply register with new foreign agent, who registers with home agent.
- Home agent can update care-of-address with mobile. Packets can then continue to be forwarded to mobile with new care-of-address.
- What if we want to keep an ongoing TCP connection? We can tell foreign address to forward packets to the new foreign address. Even more overhead

Mobile IP: Direct Routing
- In direct routing, the correspondent requests for the care-of-address from home agent. Home agent replies with foreign address of mobile.
- The correspondent can now directly communicate with the mobile (sends packets to mobile's ip, foreign agent receives the packets and forwards to mobile, who replies to correspondent directly)

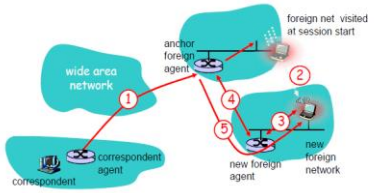
Mobility via Direct Routing



→ This is more efficient, but not preferred. The use case is not very strong, since most servers don't move anyway. Rather, we prefer for a correspondent not to have to learn about mobile ip and just perform its ip protocol as normal

Direct Routing: Pros and Cons
- It overcomes the triangle routing problem, saving a lot of overhead.
- However, the process is non transparent to the correspondent, so correspondent must get care-of-address from home agent.
- If mobile changes visited network, we can also maintain an ongoing tcp connection if we want

Accommodating Mobility with Direct Routing
- If the correspondent already has an ongoing connection with the mobile node in a foreign network, when the mobile node moves to a new foreign network, what happens?
- Mobile node now informs new foreign agent of its presence. The new foreign agent then tells the anchor foreign agent of the new location of the mobile node. Now, when the anchor foreign agent receives packet from correspondent, it relays the packet to the mobile node via the new foreign agent.
→ New foreign agent arranges to have data forwarded from old foreign agent (data chaining)

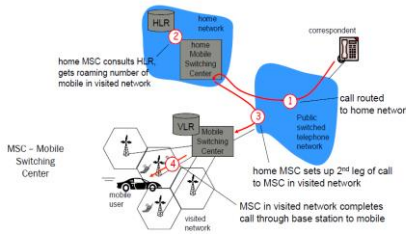


Mobile IP Specs
- RFC 3344, has many features like home agents, foreign agents, foreign-agent registration, care-of-addresses, encapsulation (packet within packet)
- Indirect routing of datagrams, agent discovery, registration with home agent standards

Mobility in Cellular Networks
- Home network – the cellular provider you subscribe to (Singtel/M1).
→ Home Location Register (HLR): A database in home network containing permanent cell phone #, profile info, info of current location
- Visited Network is the network mobile currently resides in
→ Visitor location register (VLR): Database with an entry for each user currently in the network.
→ Can also be home network

GSM (Global System for Mobile Communication): Indirect Routing to Mobile
- When correspondent calls home network, call is first routed to the home's mobile switching center (MSC).
- It consults the HLR and gets roaming number of mobile in visited network.
- The home MSC sets up call to MSC of visited network. MSC in visited network then completes the call through the base station to mobile

GSM: indirect routing to mobile



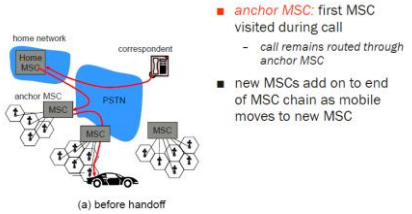
Design Decisions for Mobile IP in cellular network
- Device is mobile and can be associated to different base stations over time, regardless whether it is communicating or not.
→ How does the network know which base station a device is connected to when some device wants to communicate with me? **I communicate with nearest base station to inform my location.**
→ If I am stationary for a long time, frequent communication is wasteful. **Only communicate when I move**
→ What if I move frequently but do not communicate enough? **I inform base station only when I move "far enough"**
→ However, MSC has to search all possible base stations that a mobile can be associated with that is "close enough"**. We define "paging" areas, and if a mobile moves out of paging area, it informs the network.** The system only has to search all base stations in a paging area. Paging areas may or may not overlap (most likely not)!

MSC Handoff from 1 base station to another
- Goal: We want to route the call via new base station without any interruption to the current connection
- Why do we handoff?
1. Might have stronger signal from new BSS (continue connectivity and drain less battery)
2. Load balance: Free up channel in current BSS
Note: Only mechanism on how to perform handoff is mandated, why we perform handoff is not
- Handoff is initiated by old BSS

Handoff Algorithm between BSS
1. Old BSS first informs MSC of impending handoff and provides a list of 1 or more new BSSes (based on distance/where mobile is moving towards etc.)
2. MSC sets up path (allocate resources) to new BSS
3. new BSS allocate radio channel for use by mobile
4. New BSS signals MSC, old BSS that it is ready
5. Old BSS tells mobile to perform handoff to new BSS
6. Mobile and new BSS signal to activate new channel
7. Mobile signals via new BSS to MSC that handoff is complete. MSC reroutes the call
8. Old BSS resources released on old BSS and MSC

How do we deal with potential packet losses when switching?
- MSC buffer packets, so that packets that were not sent from old bss can be sent again in new bss to mobile
- Send packets to both old and new bss, so that while switch is happening, both bss has the packets. However, there might be duplicate packets for mobile
- Packet switching must be done with care, and is mainly only done in cellular. For wifi, we drop packets

Handoff Between MSCs
- We have an anchor MSC, which is the first MSC visited during call. Call will be routed through anchor MSC
- The new MSC add on to end of chain of MSCs as mobile moves to new MSC



Neighbor Discovery Protocols
Possible scenarios:
- You are in a shopping mall, how does your mobile phone discover presence of WiFi networks? (AP does not really need to sleep and can stay awake)
- Low power network interfaces are deployed in surrounding and carried by humans, how can these devices discover other devices in range in an energy efficient way? Take note every party wants to sleep, we need something energy efficient and can discover people fast

Overview of Neighbor Discovery
- Neighbor discovery can be bi-directional (a knows b and b knows a). Wifi is not bi-directional. Only mobile phone needs to know about wifi and only when we are connected, wifi knows about mobile.
- Challenge is in designing an energy efficient protocol.
- Async or sync

Synchronous Neighbor Discovery
- All nodes are synced to same clock/time
- Nodes agree to wakeup periodically at same time to transmit/receive packets
- Node A discovers node B when it hears transmission from B

Example:
- Assume time is divided into slots of 100ms and there is a global starting reference time t_0 sec
- With a 10% duty cycle, period is 1s,
  - *each node wakes up at t_0 + k sec for 100ms, where k = 0,1,2,3,4....*
- With a 1% duty cycle, period is 10s,
  - *each node wakes up every t_0 + 10k sec for 100ms, where k = 0,1,2,3,4....*

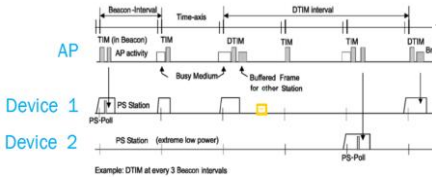
→ In the example above, a window of 100ms is quite big. What if we want a smaller period like 10ms, so with a 10% duty cycle, each node is awake for 1ms slot to discover each other? We can reduce the latency since we discover a neighbor within 10ms!
→ However, keep in mind the drawbacks: because this is a synchronous protocol, if there is clock drift, such a tight window means that some nodes might completely miss the window, and to prevent this from happening we need to allocate resources to sync clocks which is overhead
→ Other things to note: Even if we can sync for free, 1ms might not be enough to transmit data. If everyone transmits, no one has time to receive. Too short window – no time to do work
→ If wake up transition is slow: Wake up for 5ms to do 1ms of work – Very high overhead

Synced Neighbor Discovery – Drawbacks
- Sync can be expensive, and periodically, re-sync is needed to correct for clock drift over time
- As all nodes wakeup at the same time to transmit/receive, collision may increase if too many nodes transmit simultaneously

Synced Neighbor Discover – Asymmetric Nodes
→ Some nodes are more powerful, always "ON" E.g. WiFi APs (always on), vs mobile device (stays in OFF state as much as possible)
→ Since APs are always awake, when mobile device wants to discover nearby APs, mobile broadcasts a probe beacon and listens for probe replies. If wifi sends beacons instead, then mobile phone needs to stay awake and listen, so not as efficient though it might work

Power Save Mode
- A mobile node associated to an AP might want to conserve energy by going into sleep state

when there is no data communication. However, when will it know if there is data from the AP?
- When mobile wants to go into Power Save Mode (PSM), it informs the AP and goes into sleep state, switching off its radio
- Any packets arriving at the AP for the mobile device in PSM are stored in the AP's buffer
- The AP will send a beacon frame every fixed time interval. It has a field called traffic indication map (TIM – stores table of who has buffered packets) which informs the device in PSM about incoming packets.
- Device in PSM wakes up periodically to listen to beacon frames and switch to active state if there are incoming packets. Node in PSM and AP are synced, that's why node will know when to wake up and be able to hear beacon from wifi



→ AP sends beacons in intervals. Devices periodically wakes up and send beacon probe to check for buffered packets. When it sees packets buffered for it, it wakes up a little longer to receive the packet.
→ DTIM – Beacon interval for broadcast packets, TIM – beacon interval for unicast packets

Asynchronous Protocols
- Challenges: No time synchronization, all nodes are resource constrained (small batteries, cannot on gps, bluetooth, wifi continuously)
- Possible Assumptions: We divide into time slots of e.g. 100ms, each device can device randomly which time slot to wakeup to transmit/listen.

Birthday Protocol

- In a group of N people, what is the probability that there is at least one pair of peop who have the same birthday? (assume 365 days in a year)
- In general, Prob = 1 – P(none of the N people have the same birthday)
  - N = 2, Prob = 1 – (364/365)
  - N = 3, Prob = 1 – (364/365)(363/365)
  - N = 4, Prob = 1 – ((364/365)(363/365)(362/365)
  - N = n, Prob = 1 – 365!/(365^n(365-n)!)
- Example:
  - N = 2, Prob = 0.0027
  - N = 23, Prob = 0.5073
  - N = 50, Prob = 0.9704

→ Wake up for 50 slots out of 365 slots = 97% of seeing each other

→ Easy to implement, good average discovery latency but unbounded worst case

Application to Asynchronous Discovery

Michael J McGlynn and Steven A Borbash. Birthday protocols for low energy deployment and flexible neighbor discovery in ad hoc wireless networks. In MobiHoc, 2001

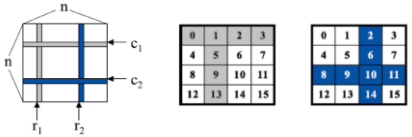
- Two nodes randomly choose to wakeup k time slots out of a total n slots. What is the probability that they wakeup at the same time slot at least once?
- Probability (Q) can be written down as: $Q(n,k) = 1 - \frac{\binom{n-k}{k}}{\binom{n}{k}}$
- (1%) n = 100, k = 1, Q(100,1) = 0.01
- (5%) n = 100, k = 5, Q(100,5) = 0.23
- (1%) n = 1000, k = 10, Q(1000,10) = 0.096
- (5%) n = 1000, k = 50, Q(1000, 50) = 0.928

Drawbacks – probabilistic bound for discovery time, can take very long sometimes

Can we do better in terms of the worst case?

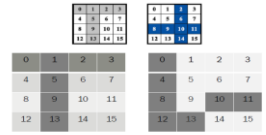
- If we want to know how long 2 nodes have been together, take the time gap of when they were discovered 2 separate times. That's the minimum time they were together
- Can we bound the worst case?

Quorum Neighbor Discovery Protocol
- Period: n^2 slots. We have 2n-1 active slots among these n^2 slots. Duty cycle: (2n-1)/ n^2
  - n = 4, period = 16 slots, duty cycle = 43.75%
  - n = 10, period = 100 slots, duty cycle = 19%
  - n = 100, period = 10,000 slots, duty cycle ~ 2%



- Protocol guarantees that 2 nodes will see each other within some time.
→ Organize grid into squares, where each square represents 1 time slot. Every user will randomly choose one row and one column. We claim that no matter what row and column is chosen, as long as a user picks one of a row and column, they will definitely see another user.
→ If we have more slots but each slot is same size – then duty cycle decreases, but take longer to discover
→ if we have more slots but each slot is smaller size – possible to achieve same latency and lower duty cycle, but can send less data since window is much smaller.

Quorum Protocol – What if there is clock drift?
→ Even if there is clock drift, no matter what there will be intersection in some square. Because there each row and each column will at least have 1 square that wakes up.



Starting point shifts by N slots
- time slots change from i to (i+N) mod 16

Let N=2 (shift schedule of right table to the reference of the left table)
- {2,6,8,9,10,11,14} becomes {4,8,10,11,12,13,0}

Quorum Limitations
- Nodes must choose same duty cycle, else the latency guarantee is gone.
- Duty cycle is around 1/n (to be exact (2n-1)/ n^2)
- Worse case latency is around n^2, if one node chooses first column last row and the other chooses first row, last column

Disco Neighbor Discovery Protocol
- Radio is scheduled to wake up at multiples of prime numbers to ensure deterministic discovery. Nodes can have different duty cycles.
- Version 0: Each node only picks one prime number p and wakes up every p slots. The duty cycle is 1/p. In the worse case, 2 nodes p and p' will meet after (p * p') timeslots
→ What if they pick the same prime number? If there is clock drift they will never meet!

Version 1: Nodes pick prime number, p1 and p2, where p1 != p2 and wakes up every p1 and p2 timeslots. Duty cycle: (1/p1 + 1/p2). If the nodes pick the same pair of prime numbers, they can still meet.
Duty Cycle: (p1 + p2)/p1p2. Worst Case latency is p1p2.

Asynchronous Protocols Analysis
→ In general, there is a general upper bound of the tradeoff between latency with duty cycle and time slot size

Category		Representative Protocols	Delay Bound
Asynchronous protocols	Asynchronous probabilistic	Birthday Protocol [2001]	None
	Asynchronous deterministic	Quorum-based [2002,2005], Disco [2008], U-connect [2010], SearchLight [2012]	$\theta(\frac{T}{f^2})$

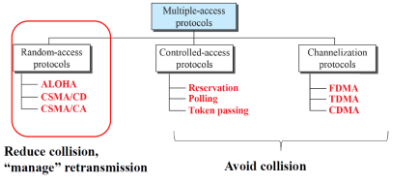
- f is the duty cycle (e.g. 1%)
- T is the time slot length (e.g. 20ms)

→ The bigger the slot size, the higher the latency. Smaller time slot reduces delay bound because we change states faster with same duty cycle. Linear relationship (T is slot size)
→ Increase duty cycle, shorter delay bound. Improving delay bound is quite expensive. If we reduce duty cycle, bound increases by squared factor. E.g. Reduce duty cycle from 2% to 1% will increase delay bound by 4x

Wireless MAC Protocols
Wired vs Wireless – Wireless need to support multiple access, broadcast in nature.
→ How to coordinate access? We need to coordinate because when multiple people speak at the same time, interference increases, and reliability drops significantly

Multiple Access
Mechanisms:
- **Avoid or reduce collisions**. Avoiding is not enough, because even if we design to avoid, it might still happen so we must also learn how to detect.
→ How to detect? We detect by waiting to receive ACK from receiver.
- **Detect Collisions.**
- **Deciding when to retransmit. (BEB)**

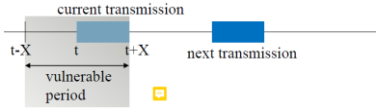
Multiple-Access Protocols



Random Access MAC Protocols
→ No coordination
→ Simplest? ALOHA
ALOHA
→ Node sends when it needs to send. If it hears an ACK within a given duration, transmission is successful. Otherwise, it resends after waiting a random amount of time.
→ Why wait a random amount of time? Because sending might have failed due to interference, if we resend after a fixed time or immediately, there is permanent interference and protocol breaks down

ALOHA Strengths
- Minimum overhead. All is needed is a random timer and ACK
- Low latency if load is light and network is not congested.
- Works quite well when not many nodes sending, very lightweight protocol as well

ALOHA analysis



Tprop = Max 1 way propagation delay between any 2 stations
X = packet transmission time, and **vulnerable period is 2X.**
→ Vulnerable period: Any node that transmits within this time period will interfere with this transmission
G: average number of new and retransmitted packets in X seconds

By poisson distribution, probability of successful transmission in k tries:

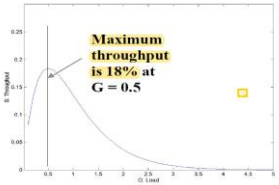
$$P(Y=k) = \frac{(2G)^k}{k!}e^{-2G}, k = 0,1,2,\cdots$$

ALOHA Throughput
- The throughput S is the total rate of arrival G * probability of no collision, i.e. k = 0

Transmission Rate

$$S = GP[k=0] = G\frac{(2G)^0}{0!}e^{-2G} = Ge^{-2G}$$

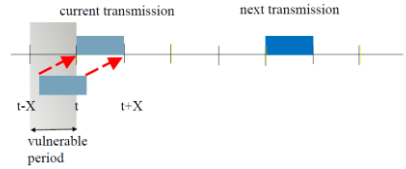
Probability of successful transmission



- **Maximum throughput = 1/(2e)**
- Average number of transmission attempts per packet $\frac{G}{S} = e^{2G}$

→ Max throughput is when G = 0.5, S = 18%

Slotted ALOHA



→ Nodes can only transmit at the start of some slot
→ Vulnerable period is now X
→ Model the number of total arrivals as a Poisson process again. Probability of successful transmission = probability that there is no other transmission in the vulnerable period of X:

$$P(Y=k) = \frac{(G)^k}{k!}e^{-G}, k = 0,1,2,\cdots$$

→ Performance of ALOHA can be improved by allowing transmission to start at fixed intervals. Now, there is only competition to send at the start of each slot (of length X)
→ Slotted aloha throughput:
- Throughput

$$S = GP[k=0] = G\frac{(G)^0}{0!}e^{-G} = Ge^{-G}$$

- Maximum throughput = 1/e = 0.368, occurs at G=1

Limitations of Slotted Aloha
→ Transmission is independent of channel state (nodes will send whether channel is congested or not)

→ Intuitively we want transmissions to only be performed when there is no other transmissions

ALOHA Summary
→ Slotted aloha/aloha is still used very commonly. Aloha especially so in low power networks because nodes don't use much of the channel's anyway.
→ Slotted aloha is actually used commonly in control channel. Our phones have a control channel running to coordinate phone to BSS or 5g/4g network, because it is very simple to sync and time sync isn't very expensive

CSMA (Carrier Sensing Multiple Access)
- Sense carrier, if idle, we send, wait for ack, and if there is no ack, we assuming there is a collision and retransmit.
- Even if a sense channel is idle, I insert random delay to reduce chance of collision

CSMA/CA
- Collision Avoidance (CA)
- Idea: If channel is clear, defer transmission for a random interval.
- Choose an interval from a uniform distribution of values between 0 and max windows size Wmax. This reduces the probability of collisions on the channel.
- How large should Wmax be? If too long, time is wasted, if too short, increase collision. So, we pick a wmax based on my notion of how busy the channel is.

Choosing Wmax
- Wmax should not be constant. Intuitively, it should be small when there is a low chance of collision, vice versa for Wmax to be high
- Wmax varies according to Binary Exponential Backoff (BEB) Algorithm.
- E.g. Set slot time to 2 times the max propagation delay on the channel. We send immediately for first transmission. If there is a collision, after the ith collision, pick Wmax randomly between 0 and 2^i-1 slots (0 – 1, 0 – 3…)
→ Usually capped at some max value e.g 2^{10} -1 (1023)

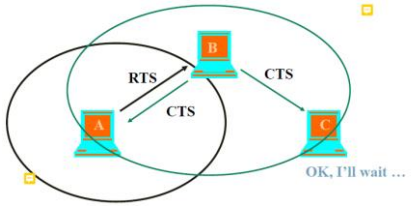
Exponential Backoff Algorithm
- Rationale – we want our nodes to all send same number of packets over the long term. Because of uniform distribution, BEB is fair.
- What happens when we have too many users? With more users, we have more collisions, increasing the back off time to be bigger and bigger – nodes end up waiting for each other, when throughput is already reduced from collisions

Tradeoff for Wmax
- Smaller Wmax (more aggressive) – more likely to achieve higher channel utilization but more likely to collide
- Less aggressive – lower channel utilization but less likely to collide
→ Utilization vs Delay – we need to strike a balance, since overly aggressive/conservative can increase delay.
→ Optimal point depends on number of station/nodes, traffic load

Window Size Tuning
- Wmax window can be varied to change the share of the bandwidth allocated to various mobile nodes
→ If we want to give some node higher share of bandwidth, give it a smaller Wmax, allowing it to send packets more often than other nodes.

Some Sharing/Allocation Problems:
1. Hidden/Expose terminal problem
2. Rate asymmetry
3. Capture effect

Hidden Terminal Problem

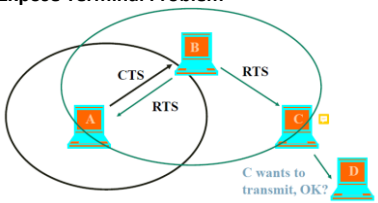


→ In this example, A and C wants to talk to B but cannot hear each other, because they are too far from each other. This is a problem because for protocols such as CSMA, which tries to sense the network before sending its packets, it does not realise that the network might actually be busy, and the CSMA degrades to ALOHA.
→ Solution? Ask B for help

RTS/CTS (Request to Send / Confirm to send)
- Idea: B conveys the state of the system to both A and C. Requires overhead and is not ideal but no choice
- When a station wants to send a packet, it first sends an RTS, and the receiving station responds with a CTS. Stations that can hear the RTS or CTS will mark the channel / medium to be busy for the duration of the request (which is indicated by Duration ID in the RTS and CTS)
- Nodes can go to sleep in this period to save power
- Stations will adjust their network allocation vector (NAV): time that must elapse before it samples the channel for idle status
- The RTS/CTS mechanism is called virtual carrier sensing.
- RTS/CTS are smaller than long packets that collide. They are also sent as a high priority packet. While there is chance of interference for rts/cts, it is low
- Solves problem when A is sending to B, because C will hear B's CTS

→ Issue: Nodes that are not concerned with the exchange (like C in this case) also cannot send to some other node like D due to RTS/CTS issue.
→ Might not be advisable to use all the time since control packet exchange incurs overhead and lowers throughput as well

Expose Terminal Problem



→ Caused by RTS/CTS mechanism. Now, since C received the CTS from B, it cannot talk to D even though it is not concerned with A/B's communication.
→ But if C sends to D it might affect communication between A and B because B can hear C.
→ We still say that RTS/CTS is overly restrictive because inefficiencies are introduced in the expose terminal problem.

Rate Asymmetry
- Wireless transmission rate achievable depends on channel quality (we get higher rate on good channels) and technology used.
- APs usually support different transmission rates and wifi versions of each node.
  - 802.11b (1 – 11Mbps)
  - 802.11a/g (6 – 54Mbps)
  - 802.11n (7.2 – 150Mbps)
- Common for an AP to support devices with different channel qualities and 802.11 protocols

→ Due to rate asymmetry, nodes that are sending on low transmission rates might hog up the channel.
- **Average Transmission Rate =** Number of packets sent in X seconds / X seconds.
- In standard CSMA/CA, nodes send almost same number of packets on average.
- We illustrate rate asymmetry problem with an example

- Configuration
  - *number of nodes: X (>1)*
  - *two rates: 10Mbps or 150Mbps*
- X-1 nodes at 150Mbps, 1 node at 10Mbps
- Each node sends same number of packets
- Let time unit be duration to transmit one packet at 10Mbps

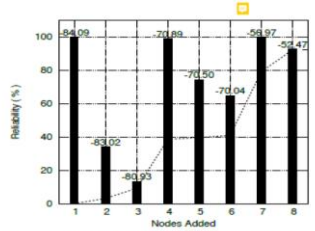
- Number of packets sent (by X nodes) = X
- Duration to send X packets = (X-1)/15 + 1
- Average rate = 10 * X / ((X-1)/15 + 1) = 150X/(X+14)
  - X=2, rate = 300/16 = 18.75
  - X=5, rate = 750/19 = 39.47
  - X=10, rate = 1500/24 = 62.5

→ For the nodes that send at 150Mbps, they take 1/15 of the time to send 1 packet. Time they take to send X-1 packets is (x-1)/15
→ Average transmission rate is given by number of packets sent / time taken, once we get the ratios, we can get the average rate in mbps.
→ When X=2 (1fast 1 slow), the average transmission drops a lot, because csma tries to be fair, when the slow guy is scheduled, it takes up a lot of time to transmit. If we schedule every person equally, the slow guy will take 15 times longer.
→ With current CSMA/CA protocol, our average rate will drop significantly when devices use very different rates. So, system is fair, but efficiency is very low. We can achieve better balance of fairness if we use a different notion of fairness: **each node is allocated same transmission duration**
→ works well in an environment with asymmetric data rate, so that the overall throughput is not bottlenecked by the slow node

- Configuration
  - *number of nodes: X (>1)*
  - *two rates: 10Mbps or 150Mbps*
- X-1 nodes at 150Mbps, 1 node at 10Mbps
- Duration = X
- Number of packets sent = 15(X – 1) + 1
- Average rate = 10 * (15(X – 1) + 1) / X = 150 – 140/X
  - X=2, rate = 150 - 70 = 80
  - X=5, rate = 150 – 28 = 122
  - X=10, rate = 150 – 14 = 136

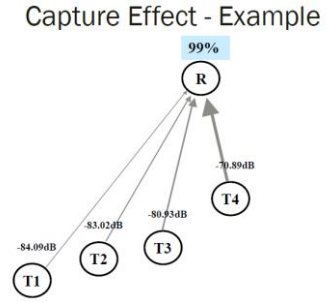
→ The difference is just now you send X packets, in time 1 + (x-1)/15. Now you have X time, send 15(x-1) + 1 packets. The transmission rate is now like a weighted average of the number of nodes

Capture Effect
- Collision does not always lead to packet loss!
- if receiver can hear from 2 or more stations, but the signal from 1 station is much stronger than the rest, the receiver can receive signal correctly from the station.



One receiver, multiple transmitters (transmit same packet) With capture effect, adding new transmitters may improve reliability!

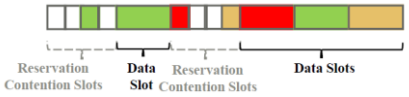
→ In this graph: As we add more and more nodes from left to right, the reliability actually increases, and the rssi even decreases



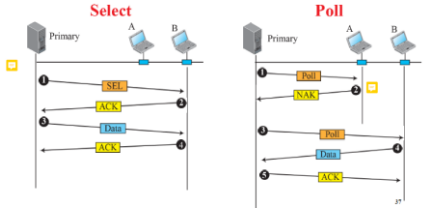
→ Captures the packets from T4 (lowest RSSI) with high reliability

Controlled Access Protocols
- Stations consult one another to find which stations has the right to send. Station can only send if it has been authorized by other stations

Reservation access method
- Station needs to make a reservation before sending data (by sending control packets)
- Example: Time is divided into reservation and data slots. Reservation frame precedes the data frames sent.
→ If there are many data frames sent after one round of reservation, system is more efficient.
→ **How it works:** station who wants to send sends a request, and coordinator gives allocated data slots to the station who made reservations.
→ For some systems, since we don't want to have a handshake to release data because it incurs overhead, one possible way is that if the coordinator sees that for 3 slots there is no data received, he releases that slot.



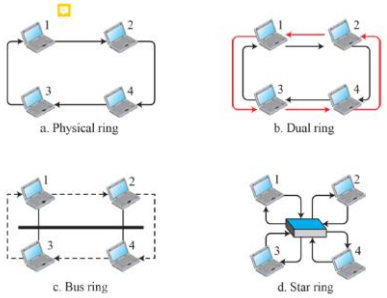
Polling
- One primary and many secondary stations.
- **Select:** Primary station sends to a secondary station, i.e. data from primary node downstream to end node
- **Poll:** primary station ask if a secondary station has data to send, i.e. data from end node upstream to primary node
→ Primary station is like a BSS. In polling, it is BSS driven (primary station asks if end-devices has packets to send), but reservation is end-device driven (the nodes make the reservations)



→ Polling: Overhead for primary to do everything and send a lot of control packets, but he has a lot of control and can assign priorities, can decide who goes first and how much time they have to send. Easier to manage resources.
→ Might not work well if there are a lot of nodes in the network because polling process will be very expensive.

More about Polling
→ It is defined in the IEEE 802.11 standard as the point coordination function (PCE)
→ AP polls the wireless stations to check for transmission requests, designed to support interactive traffic like VoIP
→ **No collision** since channel access is controlled by a central controller.
→ So for polling, the overhead is not in backoff or carrier sensing but for waiting to be polled!

Token Passing



→ Some token is passed around, where the holder of the token is allowed to send packets.
→ Physical ring, the first one is not too good, because it needs a lot of hops to get from 1 to 3 for example, which makes dual ring much better. Dual ring is also more fault tolerant since if 1 link is cut, we have 1 more ring as backup.
→ With token passing, only token holder can talk

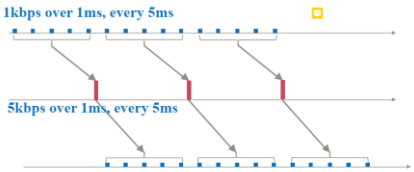
Channelization Protocols

FDMA (Frequency Division Multiple Access)
→ Similar to broadcast radio and tv where we assign different carrier frequency to different stations
- Modulation technique determines the required carrier spacing
  - *Radio: AM – 10KHz, FM 200KHz*
  - *Analog TV: PAL – 5MHz, NTSC – 6MHz*

- Each communicating wireless user gets his/her own carrier frequency on to send data on. (e.g. transmission of FM stations 98.7 (carrier freq) occupies the channel 98.6MHz to 98.8 MHz).
- **Carrier Frequency:** center of the entire frequency band

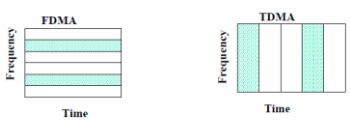
TDMA (Time Division Multiple Access)
- Each user transmit data on some time slot on multiple frequency bands
- In a sense, a time slot is a channel
- User sends data at accelerated rate when its time slot begins
- Data is stored at receiver and played back at original slow rate

Example: TDMA
- Consider a 1Mbps channel, divide the channel into 1000 equal time slots. Each slot is 1ms, and can transmit 1000 bits.
- Assign slots to 50 users, each user can transmit every 50 slots(or every 50ms) and can transmit 20 times (or 20 slots) per second
- So, on average, each user transmits 20kbps
- Realize that sender is sending "at a very slow rate" compared to the channel speed.
- Technically, what it is doing is that it is sending at the channel speed of 20kbps in 1ms. So, it needs to buffer information in the time it is not sending, and when its turn comes, sends what is in its buffer. "Buffer and burst method".
  - **Buffering is needed on both sender and receiver to match the differences in data rates**
    - **sender-side: 20kbps -> 1 Mbps**
    - **Receiver-side: 1Mbps -> 20kbps**

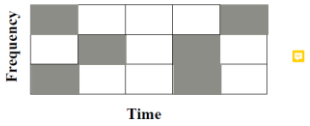


→ Sender buffers 1kbps over 1ms for 5ms to "get 5kbps" of data, which it sends over the channel in 1ms.

TDMA vs FDMA



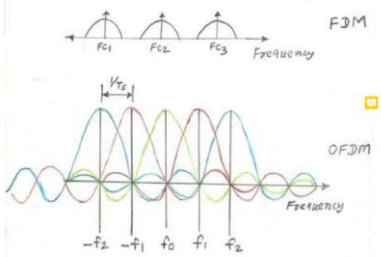
In practical systems, TDMA is often combined with FDMA



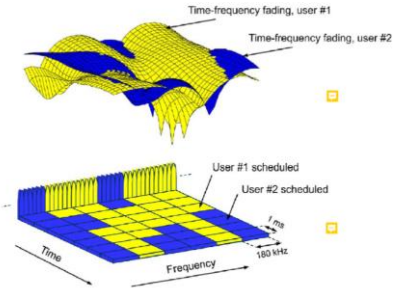
→ Why is TDMA combined with FDMA?
1. Flexible. Quantum of allocation is less chunky, unit of allocation more fine grained. Can improve utilisation and dynamic adjustment of allocation, since traffic demand comes and goes.
2. For fdma, channel allocated might be bad. If a node gets allocated to it, it might always have bad transmission due to interference, but combining solves that issue

Pros/Cons of "Static" Allocation
Pros: Resource is always available, performance guarantee. No interference from other users.
Cons: Inefficient if resource is not utilized. Resource cannot be used by other users, and is slow to adjust to traffic demand changes.

OFDM (Orthogonal Frequency Division Multiplexing)
- In FDM, the entire frequency spectrum is dedicated to carry data from a single source, but in OFDM, data is divided into many slower bit streams and multiple subcarriers is used.
→E.g. divide data source into 40 sources, each 1Mbps. So, you would divide maybe sending of one 1000 byte packet into 100 10 byte packets
→Orthogonal – peak of one carrier occurs at the null of another



→ Very bandwidth efficient – compared to FDM systems, carrier bandwidths are very far from each other
→ Can send much more bits given same spectrum



→ Top diagram refers to signal strength at the various channels, and bottom diagram refers to when users are scheduled to send, we can try to schedule in a way where when one channel has better signal strength, we prioritize that over the other user, so that the 2 users can get a better signal strength when they are transmitting.

Benefits of OFDM
- Robust to selective fading. Because we are sending on such small granularity of a channel width, if it is bad we can just avoid sending on that one without terribly affecting transmission
- Overcomes inter-symbol interference in a multi-path environment because data rate per subcarrier is lower
  - Applications:
    - *IEEE 802.11a, g, n and ac*
    - *LTE/5G*
    - *Digital radio and TV broadcast*

Energy Efficient MAC Protocols
→We want low latency, high utilization and fairness in traditional mac protocols. However, in sensor networks/energy efficient mac, we also need to consider power consumption! With low power, the sensor applications also typically have low channel utilization.

→ Consider traditional mac protocols where we do CSMA/CA where we do carrier sensing, rts/cts, data/ack, and repeat: nodes need to sense continuously to reduce latency and improve channel utilization
→ Tradeoff: We can save energy by increasing latency

Energy Consumption of MAC
→ The main sources of energy consumption in any contention-based MAC protocol is:
- Overhearing (hear and receive to check if the message is for you or not by inspecting header)
- Control packet overhead – data/ack
- Data transmission/reception
- Idle Listening

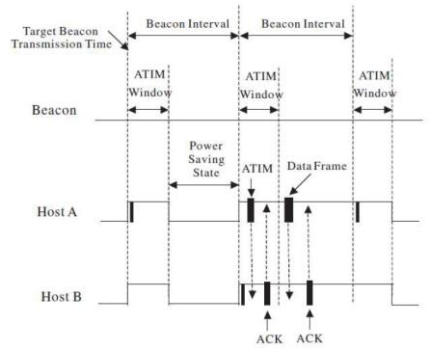
→ The first 3 occurs only when there is a transmission but the last one occurs even when there is no traffic

→ **Example of life cycle of MAC Protocols.** We increase idle listening and reduce real work done the less traffic we have

Idle Listening is expensive!
→ When the traffic load decreases, idle listening consumes most of the energy. Many measurements have shown that idle listening consumes 50-100% of energy required for receiving, but we need to idle listen to detect events!
Idea: Is there any way we can sleep as much as possible and wake up at the right time?

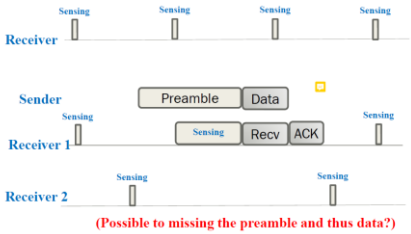
Duty Cycle MAC
→ Overhearing or passive listening is very expensive if utilization is low. **If we can find some way to sleep more and don't overhear,**
→ We consume very little energy when radio is idle/inactive!
→ However, when do we wake up? We need to coordinate wakeup schedule, and nodes involved should wake up together (i.e. receiver and transmitter)

Power Saving Mode in WIFI
→ In WiFi, we have an AP that is not concerned with ensuring that it has low duty cycle. So, nodes that need to duty cycle can be controlled by the AP. The AP can let each node know when they should wake up to receive messages.

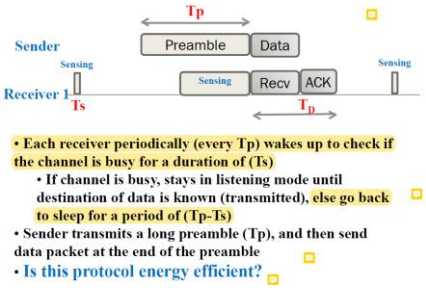


What if all nodes need to duty cycle?
→ Let's start with Aloha, where a node transmits whenever it needs to. Will this work? It might work when:
→ There is no contention of channel (usually true when utilization is low)
→ The receiver is listening (may not be true with duty cycle)

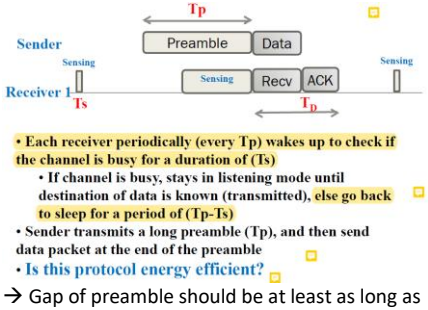
Aloha with Preamble Sampling



→ Sender who wants to send will first send a long preamble. The preamble will be as long as the sensing period of the receivers, ensuring that all receivers will be able to wake up and hear the preamble. Once the preamble is heard, the receivers sense until the sender sends the actual data, after which the receive acknowledges with an ACK.



→ TP: preamble length; TS = preamble sampling length; TD = (data + ack) length.
→ With no traffic: Duty Cycle = TS/TP
→ utilization of channel < TD/(TD+TP). **If Tp is long and Td is short,** then the max utilization is quite low because Tp is pure overhead (no utility at all), at utilization is at most Td/Td + Tp.



→ Gap of preamble should be at least as long as the receiver wake up gap

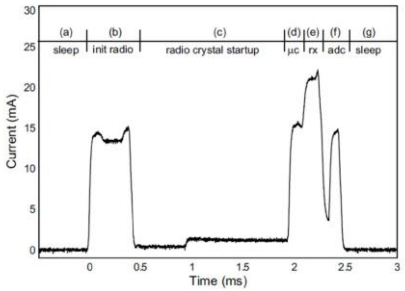
→ **ALL** receivers who were receiving the preamble will wait until data is sent before it checks whether it is intended for them
→ So, the sender will eventually wake everybody up!
→ Is this energy efficient? Not really, if we want to achieve energy efficiency, we need to increase the gap between sensing (i.e. sleep more). However, that will increase the preamble time, and transmitters who receive preamble will also stay awake for a longer time.
→ If sender sends infrequently, this protocol can still be energy efficient!
→ Overall, this protocol is not intended for high channel utilization, and works if nodes send data infrequently

Preamble Sampling Aloha Example
→ Assume we have very low channel utilization, say x%.
→ 1 transmitter and N receivers
→ Assume energy taken to transmit and receive are the same (e.g 20mA)
When there is transmission:
- Transmitter's energy consumption = (TP + TD)
- Receiver's energy consumption = 0.5TP + TD.
With N receivers, it is N(0.5TP + TD)
→ 0.5TP because on average, receiver will be awake for half the duration of preamble length, given uniform distribution of wakeup time.
When there is no transmission:
- All are receivers, energy = TS (just preamble sampling)
- With very small x% (channel utilization), total energy ~ (1+N)TS (i.e. energy of all 1+N nodes is approx. just sampling) (specifically, total energy = receivers' energy + transmitter's energy = x * (TP+TD) + (N(0.5TP+TD)) + (1-x)(TS)

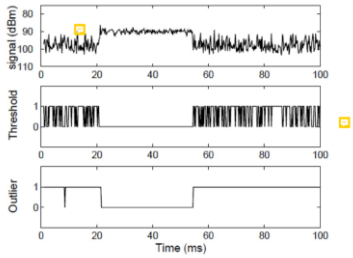
→ We still need to balance the conflict among reducing energy consumption, providing sufficient throughput, and lowering end-to-end delay (latency).
→ In aloha with preamble sampling, latency to send packet is TP+TD, max utilisation is TD/(TP+TD)

B-MAC
- An implementation of preamble sampling in TinyOS
- Each time a node wakes up, radio is on to check for activity (Basically the regular sampling of the receiver)
- Node returns to sleep after reception
- If no packet received after timeout period, a timeout brings the node back to sleep
- **Important:** Preamble length is at least as long as the interval of sampling
E.g. If the channel is checked every 100ms, preamble must be at least 100ms long
- Idle listening occurs when the node wakes up to sample the channel

Power Profile of B-MAC



→ Most energy is used in doing rx/tx and init of radio

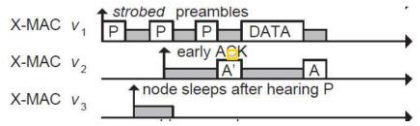


- If no outlier is found after a number of samples (e.g. 5), the channel is declared busy.
- 1 indicates the channel is clear, 0 indicates it is busy

→ It checks a few samples (e.g. 5) instead of just one because there might be outliers where node might receive one very high rssi value, and we don't want to incorrectly declare a channel as busy

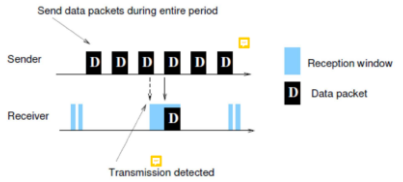
Improving BMAC / Aloha with Preamble Sampling
→ is suitable for low power/low duty operations, but its performance can be improved.
→ Consider on the transmitter: Is a long continuous preamble really necessary?
→ On the receiver: Is a long average waiting period really necessary?

XMAC
→ Improvement over BMAC
- Sender transmits many probes with gaps between the probes. Each probe will also contain address of the intended receiver (so that unintended receivers can go to sleep)
- Instead of receiver just waiting for probe to end, it can now check the address embedded in the prob and **send an ACK** if it is the destination during the gap period between probes
- Sender then transmits the data, receiver replies with ACK to complete data transfer

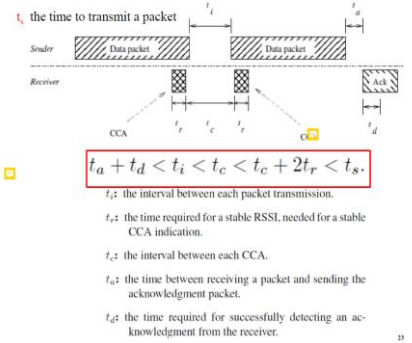


→ Receiver waits for the probe to finish, know that the message is intended for him, and sends ack. **Note:** The gap between 2 probes must be large enough for the transmitter to receive an ack, so that the transmitter knows that the receiver has received the message.
→ At worst, **a receiver will be awake for 2P + gap** if he wakes up just as the previous preamble started. He needs to receive the entire preamble to know the intended receiver, which is why the first preamble is simply wasted

ContikiMAC



→ In ContikiMAC, sender sends entire data packet as the preamble, since data packets wont be > 150 bytes in Contiki.
→ Receiver wakes up twice within a short span in contikimac.
→ Once receiver has received, it sends ack to sender, stopping the sender from sending anymore



- tr (time required before receiver can become stable to receive transmission) cannot be too short. But we don't want it to be too long because it determines the duty cycle of the receiver in idle state, so we perform it twice. Tc (interval between each tr) must be at least as long as ti (gap between 2 packets sent) so that tr will definitely overlap into a packet transmission.

Explanation for equalities in ContikiMAC
1. ta + td < ti (ta is time between receiving a packet and sending an acknowledgement packet, td is time to successfully detect ACK from receiver, ti is interval between each packet transmission)
Ti gap must be large enough so that the sender can receive the ack from the receiver. If its too short, even if receiver received and send ack, sender might miss it and send again

2. ti < tc
if tc is too small then the 2 trs can miss the packet transmission.

3. tc + 2tr < ts
Ts is the time to send the data packet. If tc + 2tr is bigger than ts, its possible to arrange the 2tr to be perfectly in the gap between data packet transmissions (i.e. the time period of ti)

Timing in Contiki MAC

- t_r is given by the data sheet of the CC2420 radio transceiver as 0.192 milliseconds.
- t_i = 0.4 milliseconds
- t_c = 0.5 milliseconds
- Minimum t_s = 0.884 milliseconds
  - *> 0.4 + 2*(0.192)*
  - *Determines the smallest packet size*
  - With a bitrate of 250kbps, and a 7 byte overhead per packet, *minimum packet size using Contiki MAC is 21 bytes*

ContikiMAC – very efficient for idle mode
Receiver wakeup frequency is lower, e.g. 8Hz

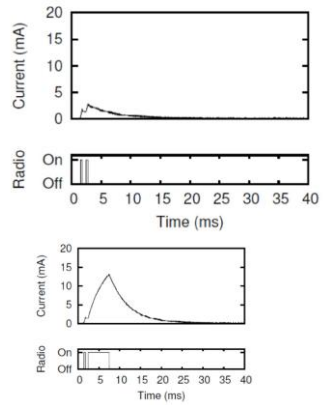


Figure 8: A ContikiMAC wake-up with radio activity detected and where the fast sleep optimization quickly turns the radio off.

Receiver-Based Preamble Sampling
→ Can we switch from sender-initiated to receiver initiated MAC?

Receiver Initiated MAC (RI-MAC)
- Change to receiver based (RI-MAC)
→ Transmitter wakes up and listens for a period of Tp (preamble time)
→Receiver will wake up every TP and sends a small packet to announce its presence.
→ Transmitter initiates transmission if the intended receiver wakes up
+ This allows for less transmission (no preamble sent), and hence, less collision.
+ May consume less energy
- During idle period, there might be more transmission because the receivers will send small packet to announce its presence. However, overall there is less transmission because the busy period is only triggered when the transmitter initiates transmission

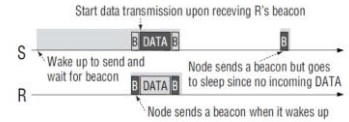


Figure 3: Overview of RI-MAC. Each node periodically wakes up and broadcasts a beacon. When node *S* wants to send a DATA frame to node *R*, it stays active silently and starts DATA transmission upon receiving a beacon from *R*. Node *S* later wakes up but goes to sleep after transmitting a beacon frame since there is no incoming DATA frame.

Lifetime calculation of BMAC Example

- Consider a node running B-MAC. It wakes up periodically every 200ms to sample the channel for 2ms. If there is no channel activity, it goes back to sleep. The transmission and reception/listening states draw 10mA.
- The sleeping state draws 0.1mA. If there is no transmission, what is the average current consumption of a node?
- Average = 0.99 x 0.1 + 0.01 x 10 = 0.099 + 0.1 = 0.199mA
- Consider a scenario with sensor node A transmitting a very small packet to node B every 2s and both nodes use B-MAC. Calculate the average current consumption of nodes A and B.
- Node A: 0.1 (10) + 0.9 * 0.199 = 1 + 0.1791 = 1.1791mA
- Node B: 0.1 (5.05) + 0.9 * 0.199 = 0.505 + 0.1791 = 0.6841mA

Lifetime calculation of RI-MAC Example

Lifetime calculation of RI-MAC

- Consider a node running R-MAC. In R-MAC, a node wakes up periodically every 200ms to transmit a small probe to check for active sender. If no sender is detected, it goes back to sleep. The entire period of probe transmission and sender's transmission detection is 2ms. The transmission, reception and listening states draw 10mA. The sleeping state draws 0.1mA.
- Consider the scenario whereby there is no transmission. What is the average current consumption of a node?
- Average: 0.99 x 0.1 + 0.01 x 10 = 0.099 + 0.1 = 0.199mA
- Consider a scenario with sensor node A transmitting a very small packet to node B every 2s and both nodes use R-MAC. Calculate the average current consumption of nodes A and B.
- Node A, 0.1 (5.05) + 0.9 * 0.199 = 0.505 + 0.1791 = 0.6841mA
- Node B, 0.99 x 0.1 + 0.01 x 10 = 0.099 + 0.1 = 0.199mA

<u>Synchronous MAC (SMAC)</u>
- Wake all nodes at the same time.
→ Solution? Some form of time division multiplexing, divide interval into listening and sleeping. Percentage of listening period is the duty cycle.

Fig. 1. Periodic listen and sleep.

- Example: 10% duty cycle
  - *Each node listens for 100ms and sleep for 0.9s*
  - *Each node listens for 1s and sleeps for 9s*

→ Both have same duty cycle, but first one has better latency

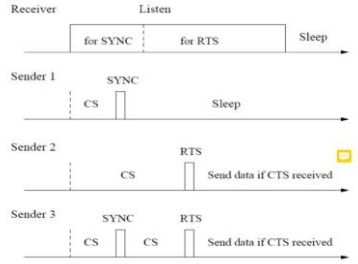
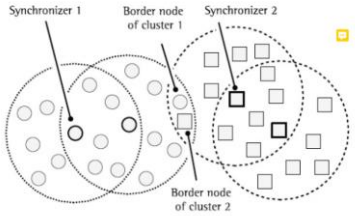


Fig. 3. Timing relationship between a receiver and different senders. CS stands for carrier sense.

→ Basically receiver wakes up and listens for rts. Sender wakes up and sends rts. Sends data if receives cts.
→ How do we synchronize the schedule of all nodes? Initially, nodes are deployed to start after a random time interval. So, we have a mechanism to pick a leader.
→ Nodes all wake up and randomly toss a coin (representing a delay). After delay, send a packet. If im the first to send the packet, im the leader.
Problem: We might have a multi hop network where a node has chosen itself as a leader because it could hear you and you couldn't hear it)
→ Boundary nodes end up adopting multiple schedules

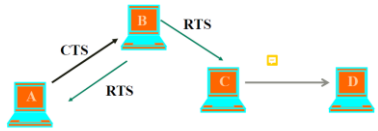
SMAC Schedule Coordination



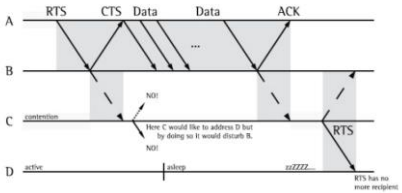
→ The border nodes of cluster 1 and cluster 2 have 2 leaders. So, they can wake up twice according to both schedules

Extension of SMAC

- Issue: in order to converse energy, a node wakes up and listens to the channel. If channel is idle, node goes to sleep
- Problem:
  - *many nodes may want to communicate*
  - *related to the exposed terminal problem*



Issue with RTS/CTS (with duty cycle)



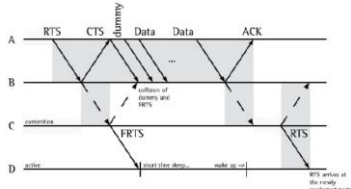
Does problem exist if D only sleeps when channel is idle?

→ C can potentially have very high latency (in terms of waiting before it can communicate to D). This might happen if A and B are communicating so C cannot send to D even though D is awake. Then, just as D goes to sleep, A and B finish communicating. Then C sends RTS but has no recipient, and needs to wait until D wakes again, incurring very long latency

Solution: TMAC

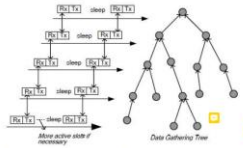
T-MAC

- Two proposals:
  - *Future Ready to Send (FRTS)*
  - *Full buffer priority - sends unsolicited RTS*



→ C sends a FRTS (Future ready to send) packet once for D. Now, D knows C wants to send to D and stays awake waiting.
→ FRTS might interrupt communication between A/B though. However, FRTS packet is a small control packet, and moreover, A is the one sending data, B is merely receiving, so C might be able to send to D without interrupting A and B's communication.
→ Use case of TMAC: Give transmitter (C) more priority to send (maybe because it wants to reduce latency, or because his buffer is almost full and needs to send soon, if not his buffer is going to be full and will need to drop packets)

DMAC
- Good for applications such as data gathering, as the topology falls in nicely with it



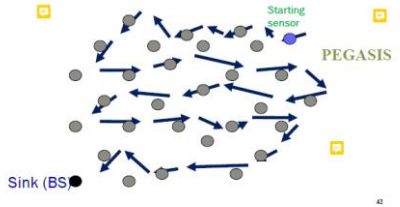
- For specific application such as data gathering, topology information can be exploited
- Synchronize schedule based on node's position from root
  - Nodes with the same depth will have same active periods
- Reduce latency and collision
  - If an event occurs too early, a node has to wait for the next cycle

→ All nodes of the same depth have the same active periods (e.g. the 3 leaf nodes at the bottom have the same period)
→ At any time, 1 node is sending, the level above will be receiving. This has a nice latency property because at each wake up time, from the perspective of the data, it does not have to wait to be received by the node at the level above.
→ When there is more than one child pointing towards the parent, we assume that the 2 nodes can finish transmitting in 1 cycle. If the fan in is very big, this might not be possible.
- **Cons of DMAC:**
1. Must handle edge case where a lot of children send to 1 parent.
2. If the sibling nodes are close to each other, might have interference issue
3. If some guy at the bottom wants to send right after its turn has past, it must wait for quite awhile before it can send

WiseMAC
- In WiseMAC, sender knows when the receiver wakes up to listen. So, it can start transmitting preamble just before node wakes up
How do you learn about the receiver's schedule: A node informs neighbors of its wakeup schedule by putting this info in the ACK packet. Nodes can then learn and update when they wake up.
→ We actually have something like this in Contiki, as it will try to predict when the transmitter wakes up (if the transmission is periodic). Then, it wakes up just before the transmitter sends, decreasing duty cycle since transmitter does not have to keep sending preamble (or data in case of contikimac), and receiver does not have to wait to receive full preamble. **Also decreases latency and even if you're wrong,** we are running the usual contikimac so we are still guaranteed a reception of data

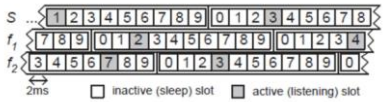
Issues with WiseMAC:
- Needs synchronisation
- Overhead in changing wakeup schedule
- Change in receivers (Not sure what this is about, maybe it has to relearn the sender's schedule?)

Collaborative Information Processing

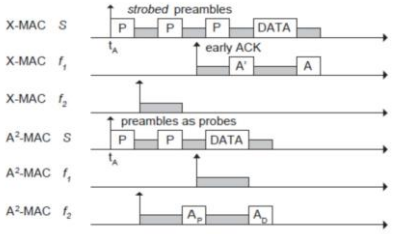


- Chain the sensors together so that each sensor receives only one message and transmits it till the message reaches the destination or base station.
- Designed for synchronous and periodic traffic from all nodes in a static topology
→ Power-Efficient Gathering in Sensor Information Systems (PEGASIS)
→ It is similar to TMAC in that the nodes will make an assumption of when the data is coming (in TMAC its because of the FRTS)
→ works when we have clock sync, static topology and highly regular and amount of data sent is very small
→ E.g. If we have a chain of 200 nodes where sending and receiving each takes 10ms, we can hear from every single node in 200*10ms = 2s, and duty cycle is only 20ms/2s = 1%. Each node only needs to wake up for 2 time slots to send and receive once, much faster than any other mac for 200 nodes.
→ **Improvement:** We can wake up 2 nodes instead of 1 for more receivers. Within one time slot, if the packet is small, we can send 2 packets in 1 slot too. Increase fault tolerance and reliability

Anycast MAC (Asynchronous, Slot based A^2-MAC)
- So far, it has been one sender transmitting one receiver. What if there are multiple possible receivers?



A²-MAC vs. X-MAC



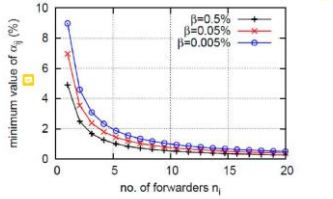
→ Similar to XMAC, but now, whoever receives the message can ack, and data will be sent by sender.

Minimum Duty Cycle Needed

$P_n = [\prod_{j \in F_i} (1 - \frac{\alpha_{ij}}{v})]^v \leq \beta$

P_n is the probability that none of the forwarders will wake up within a cycle of v (e.g. 100) slots

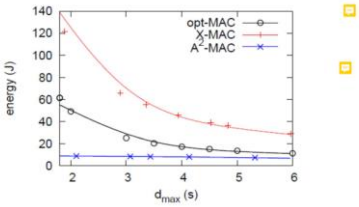
α_{ij} is the duty cycle
β Is the constraint



→ Minimum duty cycle needed, alpha a, such that the beta B, is less than some value like 0.05%.
→ Pn is the probability that none of the forwarders wake up within a cycle of v slots. So, we want high probability that some node is awake.
→ Graph represents the different duty cycles needed against the number of forwarders to achieve some duty cycle beta.

Energy vs Delay (dmax – delay constraint)

Energy vs. Delay (dmax - delay constraint)

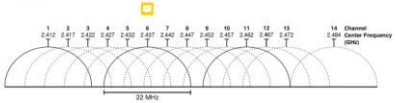


→ If we have some delay constraint, then will expend more energy
→ Anycast can bring down duty cycle a lot (because any node can receive I think)
→ Cons of anycast mac: If 2 nodes receive packet, it might cause duplicates, affecting throughput and wasting energy if there are a lot of duplicates.

Multi-Channel
- Have more than 1 channel so that transmissions can happen simultaneously orthogonally.
→ E.g. Zigbee has 4 orthogonal channels, i.e. 4 transmissions can happen simultaneously without interference.
→ Increase throughput via spatial diversity, increase reliability without redundancy. We can also choose channel with best rssi
Cons? Need to coordinate channels. Must know which channel receiver/transmitter are both on, else, it defeats the purpose of no interference.

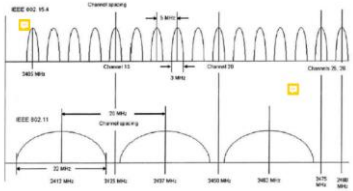
WiFi Channels

Wifi Channels



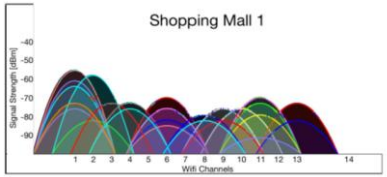
14 channels designated in the 2.4 GHz range spaced 5 MHz apart (with the exception of a 12 MHz spacing before channel 14)

→ 3 nonoverlapping channels, 14 channels
ZigBee Channels



→ Above graph is zigbee, wifi is below. Wifi/zigbee communication can interfere with each other.

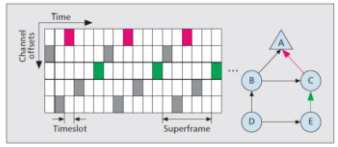
WiFi Channel Usage



Multi Channel MAC Protocols
→ Use control channel. Use a default channel and time sync for control message exchange. Then, nodes will switch to different channel for data communication.
→ Nodes can switch to another channel when link quality is bad.
→ **However, how do we find a good channel to switch to?** We find a good channel by switching channel and sampling. Channels with bad rssi are reported in control channel and avoided. Keep blacklist/whitelist of good/bad channels. Must be constantly/dynamically updated.
→ **What traffic pattern benefits from multichannel?** When there are a lot of things to be communicated between 2 nodes, because in multi channel communication, there is channel switching costs and cost of storing dynamic table, so these overhead are amortized with high data transfer

TSCH Network

TSCH Network



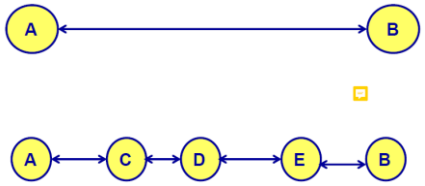
- **Time is divided into slots, multiple slots form a slotframe**
- **Sends to receiver on slots it is listening on (e.g. green to C, red to A)**
- **Frequency chosen for a timeslot is based on a pseudo-random pattern**
- **Scheduling policy is not standardized**

→ Each node assigned to some channel and duty cycles on that channel
→ Sender can switch to that channel to send to the receiver on that channel I think?

Wireless Network Routing
- If everyone is connected to each other, routing is easy and is all 1 hop
- Improvement: If we have 1 big switch with N links, routing is still simple but we reduce number of links. However, what if N is large and nodes are far apart like the internet?

Wireless Routing
- Routing Protocols are complex, hard to maintain/debug and incur significant overhead. Even more so for wireless networks.
- In practice, common wireless network deployments are single hop and stations/devices communicate directly with base stations
- E.g. Wifi, Cell network, LoRa
- Can one-hop communication be always sufficient? Can we do multi-hop routing such that it can potentially provide more routes and thus is more robust? Can it be better since it has more coverage/reach?

For same distance, is multi or single hop better?



→ Single hope typically better because use less resources, simpler, faster.
→ Multi-hop: Potentially higher data rate (send to nearer node first → send at higher bit rate with same power)

However, we also need to consider energy consumption.
→ Can multiple hop transmission be more energy efficient than a single-hop transmission?
- Recall the free space model,
  - $Pr = C P_t / d^\alpha$ (C = constant)
- Example: transmission over a distance d, let minimum received power needed be P_0
  1. Use 1 transmission to cover distance d, minimum transmission power (P_1) needed is
     - $P_0 = C P_1 / d^\alpha$ or $P_1 = P_0 / C * d^\alpha$
  2. Use N transmissions to cover distance d, minimum transmission power needed (P_N) per hop is
     - $P_0 = C P_N / (d/N)^\alpha$. or $P_N = P_0 / C * (d/N)^\alpha$
     - Total transmission power is $N P_N = N P_0 / C * (d/N)^\alpha$

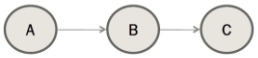
→ typically alpha is more than 2, so we can cancel the N on top and get a lower power than 1 big hop
→ However, there is a lot of overhead from having 1 more node (cpu, peripherals power consumed). Also more interference.

Wireless Mesh Routing
- Formed by a network of static wireless nodes
- Main challenges: Interference (so need routing/mac protocols), self-interference (because of rts/cts, 1 packet within a flow can interfere with its own connection. E.g. a talk to b, b talk to c using rts/cts, a cannot talk)
- **Goals:** Select routes that minimize interference and error. Methods? Pick better link (one that is stronger/higher signal strength/less interference/more reliable)

Routing
- Simplest routing protocol is a tree-based protocol (like a spanning tree)
- Tree suitable for data collection application
- What are some possible routing metric? **Hop Count, Delay, reliability, energy**

Hop Count: If all link quality is same, can route by hop count.
Reliability: Choose node that is more reliable. In general, wireless is quite unreliable
Energy: Energy to send a packet, which we can vary to send at higher rate/longer distance, or lower rate/shorter distance. Energy can also be interpreted as amount of packet loss
→ Retransmitting is expensive because it leads to higher latency and requires more energy

ETX (Expected No. of Transmission)
- Depends on packet reception ratio PRR
  - Packet reception ratio (PRR)
    - *A -> B is 0.8, ETX = 1/0.8 = 5/4*
    - *B -> C is 0.5, ETX = 1/0.5 = 2*
    - *Path ETX (A -> B -> C) is 5/4 + 2 = 13/4*



→ To get etx, you need to get prr. To get prr, you need to send packets

Shortest Path Routing
Many practical routing algorithms are based on the notion of shortest path between 2 nodes
- We assign a positive number to each link, call it length
- Route each packet along minimum path between source and destination
- If the length is always 1 then shortest path becomes minimum hop routing
- Cost, delay etc can also be used as "length"

Tree Based Routing
- Select root node, root node will broadcast. Neighboring nodes will obtain distance (hop count) from root, link quality from root.
- Each node repeats the process, selecting "preferred" node as parent. Selection can be based on minimizing hopcount of maximizing link quality (i.e. etx)
- At the end, all nodes should have path towards root

Ad-Hoc Routing
- Nodes are not static (i.e. they are mobile).
- A MANET (Mobile Ad hoc network) is an autonomous system of mobile nodes (MSs) connected by wireless links. No infrastructure exists in a MANET.
- **The network's wireless topology may change dynamically in an unpredictable manner since nodes are free to move and each node has limited transmitting power**
- **Assumption: Due to node mobility, routing path may be available but needs to be "discovered"**

→ No root node, network topology can change anytime.
→ While nodes are mobile, they cannot move too fast, as everyone still needs to at least be connected to somebody to be discovered

AdHoc Routing Classification
Existing Routing protocols can be classified as
1. Proactive: When a packet needs to be forwarded, the route is already known.
→ Constantly calculating routing table, no latency when doing routing. However, table might be outdated
2. Reactive: Determine a route only when there is data to send
→Packets come in then start to calculate next hop. Overhead only occurs when there is a need to send. But table will be more updated! If there is a lot of work to do in a short span of time, there might be congestion with more calculations needed.

Routing Protocols may also be categorized as:
1. Proactive – Table driven protocols
2. Reactive – Source initiated (on demand) protocols

Proactive Table Driven Routing Protocols
- Each node maintains routing information to all other nodes in the network.
- When topology changes, updates propagated throughout the network.
E.g. DSDV – Destination Sequenced Distance Vector Routing

DSDV
- Based on Bellman-Ford. Each mobile node maintains a routing table in terms of number of hops to each destination.

- Routing table updates are periodically transmitted
- Each entry in the table is marked by a sequence number which helps to distinguish stale routes from new ones, and thereby avoiding loops

Reactive - Source-Initiated On Demand
- Reactive – Adhoc On-Demand Distance Vector (AODV)
- Nodes that are not in a selected path do not maintain routing info or participate in routing table exchanges
- A source node initiates a path discovery process to locate the other intermediate nodes, by broadcasting a RREQ (Route Request) packet to its neighbors.

Delay Tolerant Network (DTN)
→ Nodes are mostly disconnected, range is relatively short, mobile nodes meet one another opportunistically. Nodes are mostly in "sleep" mode to conserve power.
→ To improve delivery ratio, messages are duplicated to many nodes. We can further improve performance by estimating contact probabilities. So, we choose who we send packets to based on this contact probability
→ Constraints: Number of messages buffered in each node, and duration of contact.

- Some Examples
  - *Terrestrial Mobile Networks: e.g. bus network*
  - *Exotic Media Networks, e.g. deep space network*
  - *Military Ad-Hoc Networks*
  - *Sensor/Actuator Networks*

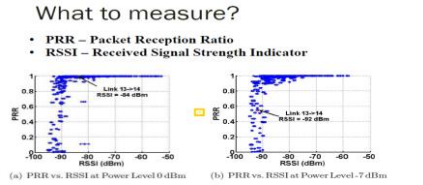
Simplest DTN Protocol: Epidemic
→ Send all packets to every node one encounters.
→ Claim: Epidemic has best performance but highest overhead. Largely true, if there are no constraints.
→ Constraints: How many messages can you buffer at contact time? If the other node is not able to buffer the packets, he starts throwing packets away, affecting performance.
→ Secondly, do we have enough time to send everything? Recall that contact duration is a function of/depends on duty cycle and mobility pattern, if the contact duration is insufficient we may not get good performance.
→ **Enhancement:** Only relay message up to x-hops from the source. (i.e. how many times should I send the message. Reduces the duplicated packets issue)
→ Another possible enhancement would be to use ACK message to remove delivered messages from buffer. If sent, and ack received, we can remove message from buffer.
Single Copy
→ There is only one "single copy" of any packet in the entire network. Transmit to destination if the two nodes are in direct contact.
→ We relay the message to another node with probability p. Intuition: If a message can keep jumping around, it covers more distance in the network, and a packet can eventually reach the destination, compared to waiting for a node to wait until it meets the destination before sending it

Other Possibilities in DTN

- Duplicate a few copies instead of using only one
  - *Problem: How many copies do we need?*
- How should contact probabilities be estimated?
  - *Example 1: Selects next hop based on time of last encounter, transmits to nodes that encounter destination nodes more recently than itself*
  - *Example 2: Records last contact information, e.g. location and time, duplicates messages to nodes which are closer to destination*

→ Still, it depends on the mobility pattern. For example, if I saw some node 2 hours ago, it might mean he sees him again soon, but if the pattern duration is 7 days it will be very long before they meet again

Sensor Network Routing
→ Similar to mesh routing, designed for static, connected networks. Main issues: Energy efficiency, duty cycling
→ In a wireless environment, each node needs to decide if a link to another node is available. How do we do that, we need to transmit to test reliability of the link (rssi).
→ A node sends a beacon and other nodes check if the beacon can be received. Do it broadcast-style so that in one beacon, all neighbors can measure rssi



→At higher RSSI values >= -70dbM, PRR is basically 1. Around -90 quite noisy, -95 cannot get much

Measuring Link Quality (PRR)

- To measure PRR,
  - *Receiver assumes that a transmitter sends S packets over a given period of time T sec*
  - *In T sec, record the number of packets received from transmitter, let this be R*
  - *PRR = R/S*
- Example:
  - *T = 1sec, S=100 (if R=95, PRR=0.95)*
  - *T= 10sec, S=100 (if R=55, PRR=0.55)*
  - *T= 100s, S=10 (if R=8, PRR=0.8)*

Measuring Link Quality
→ Tradeoff between communication cost and reaction time (i.e. updating of cost values).

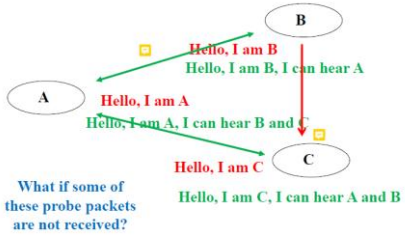
- Assume ZigBee radio is used (250Kbps), each transmission takes 1ms (~250bits, ~30bytes) and link measurement probes should not collide
- If T=1s, S=100
  - *Measurement takes up 10% of bandwidth 1 node*
  - *If there are 10 nodes, 100 probes need to be transmitted in 1 sec (100% of bandwidth used only for measurement!)*
- If T=100s, S=10
  - *Measurement takes up very little bandwidth*
  - *Is S too small?*
  - *If there is a change in link quality, how long does it take to notice?*
- What are "good" values for T and S?

→ S is the number of packets sent over time T seconds.

A Simple Beacon-Based Protocol

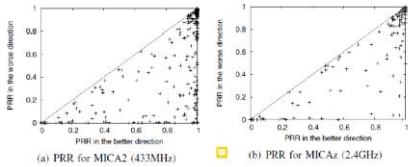
2. When a node receives a "HELLO" message from another node, it identifies the latter as one of its inbound neighbors, and it will include the latter in the "I-HEAR-YOU" message and broadcast the message back to the latter.
3. In response, when a node receives "I-HEAR-YOU" message from another node (i.e., its ID is included in this "I-HEAR-YOU" message), it identifies the latter as an outbound neighbor.
   1. *The two nodes will become both inbound and outbound neighbors for each other, if they can hear each others' messages from each other*
4. In practice, the "I-HEAR-YOU" message contains the list of inbound neighbors as well as their PRR values, and the "HELLO" message, are often combined into one packet.

→ Detection is one way: Receiver detects transmitter.
→ Beacon has 2 functions:
1. Send "hello" and attach address for someone to hear you
2. While listening, collect all "hellos" from other nodes, and when you send your own "hello" message, include in all previous "hello" messages from previously heard nodes
→At the same time, we can measure PRR by observing sequence numbers of packets received. E.g. If we receive 3 packets number 10, 11, 13 then PRR is ¾ = 0.75
→ We do not reply immediately that we can hear some node, because doing that can lead to explosion, causing a lot of interference. If we do carrier sense + backoff communication will become very expensive. So, we just add who we can hear in our beacon.



→ B didn't hear C, that's why the red arrow is unidirectional. The message can also include the PRR in which a node heard the other 2 nodes in

Link Asymmetry



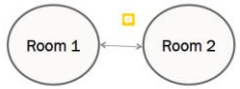
Usually only symmetry links are used
What is a (sufficiently) good link?

→ We don't really want links that have one side with 1.0PRR but other like 0.3, because most communication is 2 way, and most protocols **require ACK**, even if we can send one way without feedback, it serves as a problem for protocols
→ However, usually links are better from one way to another. In the graph, we put the better one in the x axis, which is why there are only points on the bottom.
→ Most links are asymmetrical! What links should we choose? Typically those where both are around 0.8 (good rule of thumb). If too restrictive, network might not be connected. Poor links might cause a lot of retransmission

Tracking Neighbors in Table
→ We usually only track around 6-10 neighbors to reduce memory and computation cost. We also only try to keep track of "good" neighbors.

Network Partition in Neighbor Tracking

- Possibility of network partition
  - *Links within each room is good*
  - *The one link between the 2 room is weak*
  - *If all nodes select the best links, there will be no connection between rooms 1 and 2*



How do you address this problem?
→ This problem arises because we only choose good neighbors, causing a network partition where nodes in room 1 wont choose those in room 2 and vice versa

Collection Tree Protocol (CTP)

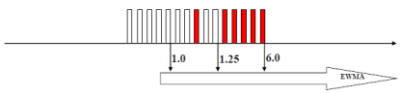
- Current default routing protocol in TinyOS
- Tested on 12 different testbeds (20 nodes – 310 nodes)
  - *> 90% delivery rate in all cases*
  - *> 99% in many cases*
- Some limitations in evaluation
  - *In testbed evaluation, there is no issue with battery drain*
  - *Power is often set to maximum value*
  - *Deployment tends to be very dense*

Major Subcomponents of CTP:
1. 4-bit link estimator
2. Datapath validation
3. Adaptive Beaconing
→ Adaptive: Addresses challenge of reducing overhead of neighbor discovery while having fast reaction to link failure
→ Link quality measurement is piggybacked on data traffic. We have static and dynamic beaconing as well

Data Driven Estimation
→ Estimation is agile and driven by ACK-bit

If a out of K_u packets are acknowledged then $ETX = K_u / a$

If $K_u = 5$



→ For each path, we hold an estimated etx value, calculated by using a sliding window. The exact numbers are not important, relative order of paths are more important
→ Piggyback these measurements on data transmission
→ With moving window of last few packets sent, PRR is a/Ku, etx = Ku/a. How do we know the PRR of what we sent? There will be ack, so we can know a
→ In a moving window, if the connection is very bad i.e. a = 0, etx = 5/0 = infinity, and we store etx as a placeholder (e.g. 6), implying the link is very bad, and we should switch to another shorter path

Data Path Validation
→ Data Packet carries estimated ETX value to root. This value keeps adding up across path from node to root.
→ Receiver will compare ETX value of packet received to its own estimated. The received value should ALWAYS be higher than its own value, because theoretically it is in between the path of child to root. If the ETX received is smaller, then the topology is stale and we need to update.
→ If etx at parent is higher than what was calculated by child, it tells child to consider changing to another path, because the etx on this path is no longer as low.
→ Allows detection on first data packet, rather than waiting for the routing protocol update (??)

Analysis of Data Path Validation
→ Compared to when etx is measured statically and only based on control packet management, now, it is measured whenever there is data transmission.
→ For nodes without data transmission, we do not really care about them anyway so it doesn't need an ETX

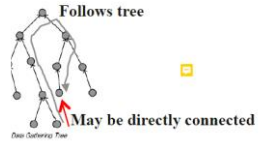
Adaptive Beaconing
→ We want to balance having to send as few beacons as possible (for energy efficiency), but react quickly to topological changes.

- Do not use periodic beaconing
- Starts from a fixed and small interval T_0 (minimum interval)
  - *If there is no "change", double interval till T_H (maximum interval)*
  - *If a change is detected, drops to T_0*
- Change is defined as:
  1. *Receiver sees larger ETX from transmitter*
  2. *ETX drops significantly (> 1.5)*
  3. *Neighbor ask for update (e.g. a new node joins)*

→ Change can be detected by parent telling child that there is a change in etx measurements and to consider another parent. This resets interval to T0
→ Another way is through sampling yourself, and you notice that the etx is terrible, so you reset interval to t0 and send beacon
→ With adaptive beaconing, if i am triggered to remeasure my etx, i tell my children, triggering more nodes to remeasure etx, propagating through the network.

Point-to-Point Routing
→ Can use tree routing.

- A tree can be used, just like spanning tree protocol in Ethernet by routing packet toward the root
  - *Inefficient in terms of path length*



→ Go all the way to root and come back down using tree routing.
→ Cons? Might be quite inefficient since physically these 2 nodes are actually closer

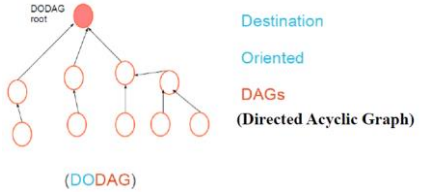
General Sensor Routing
→ Support point-to-point and point-to-multi-point

- Data centric: addressing a node based on its properties, e.g. availability of information
  - *Do not require unique address*
  - *Directed Diffusion (2000), Rumor Routing (2002)*
- Location based
  - *A more specialized class of data centric routing whereby location is one of the property utilized*
  - *Location information allows better route to be chosen*
  - *GPSR (2000), GHT (2002)*

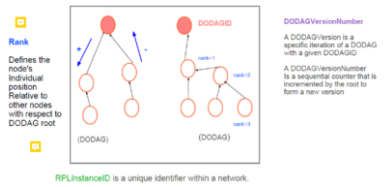
RPL (Routing Over Low Power and Lossy Networks)

- Distance Vector (DV) protocol
  - *Requires that a router to inform its neighbors of topology changes periodically*
  - *Each node maintains a vector (table) of minimum distance to every node*
- Source Routing Protocol
  - *Allows a sender of a packet to partially or completely specify the route the packet takes through the network.*

DODAG (Directed Acyclic Graph)
→ DAG that points towards some destination or root



→ Maintained by some sort of rank



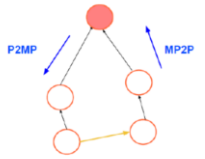
→ Rank defines the node's position relative to its parent. (i.e. Root is 0, child of root is 1..)

Control Messages

- DODAG Information Solicitation (DIS): used to solicit a DODAG Information Object from a RPL node.
- DODAG Information Object (DIO) - The DODAG Information Object carries information that allows a node to discover a RPL Instance, learn its configuration parameters, select a DODAG parent set, and maintain the DODAG
  - RPL nodes transmit DIOs using a Trickle Timer.
- Destination Advertisement Object (DAO): The Destination Advertisement Object (DAO) is used to propagate destination information Upward along the DODAG.
- DAO-ACK

→ Similar to adaptive beacon that adjusts to the environment.
→ Control messages to update frequently who are your neighbors

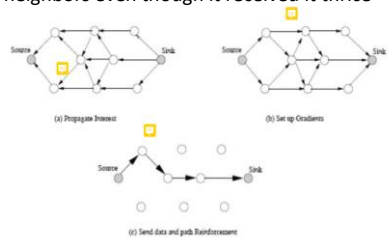
Traffic Flows Supported by RPL



- Rank-based data-path validation mechanisms
- In Storing mode, the DAO message is unicast by the child to the selected parent(s).
- In Non-Storing mode, the DAO message is unicast to the DODAG root.

→ Always going to root can be problematic, which is why point-to-multi-point, where one node talks to other nodes aside of root was implemented
→ MP2P – several people talk to 1 guy

Directed Diffusion
- Diffusion – broadcast. Directed – directed to shortest path when coming back
→ As seen in the last figure, the top node chooses to send to the middle node because he originally received from the middle one first, so he knows it is nearer (has lower latency).
→ It knows to only send the packet once to neighbors even though it received it thrice



- We have a node (sink) that asks for a particular kind of information but does not know if the information exists or not.
→Sends query to express an interest for specific information
→ Neighboring nodes act if they were the sink and forward queries to their neighbors.
→ Only 1 copy of interest/query is forwarded
→ Flooding/propagation stops when a source node can satisfy an interest/query.
→ Source node then floods answer to sink
- A lot of redundancy and quite expensive in flooding the network, wasteful but is needed since we don't know who has the answer to the query, or if there even is an answer or not
- Sink assumes that the node from which the first reply arrived is also the neighbor on the shortest path
→ We reinforce this path by sending more queries on it.

Path Reinforcement
→ Cost: Every node is informed about an event, even the nodes that are not interested.
→ Propagation of data depends on "reinforcement" from interested sources
→ Initial flooding expires after some time while reinforcement messages continue

Issues with Path Reinforcement
→ Store large amount of status info, which is not very bad.
→ Substantial communication overhead in discovery phase. Flooding is very expensive, since wireless uses shared medium, there will be a lot of collision, and so we need to do carrier sensing…
→ Usually finds a path as long as it exists. Works well with large amount of sensed events and a few queries.
→ Due to flooding, when reply is going back to sink, it has to compete for resources with those who are still sending query to everyone else because of flooding. "Large amount of interest for same data"

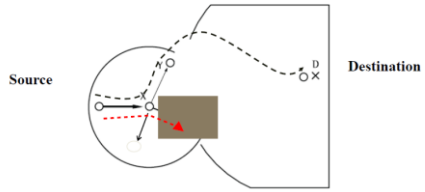
More about Directed Diffusion
→ if we want a very reliable band, we might choose some ghz that is below 1k like 900mhz, which is more reliable than 2.4ghz. however, this link has lower bandwidth and therefore bitrate
→ there is a lot of collision and transmission is slow due to low bit rate, so the query takes very long, and everyone is congesting the network. when the query is coming back to the sink, it is competing for resources with the packets that are still going towards the source node
→ therefore, we need sufficient bandwidth to have reliability, i.e. in this case we cannot go to a frequency band that is lower for reliability!

GPSR
- Greedy Perimeter Stateless Routing

- One of the earlier and very popular location based routing protocols
- Each node knows its location and the location of the destination node
- To reach destination, node chooses the node that is "closest" to the destination as the next hop

→ applications: placing sensors in water bodies where we dont want wire, the sensors almost dont move in the eyes of some gps sensor

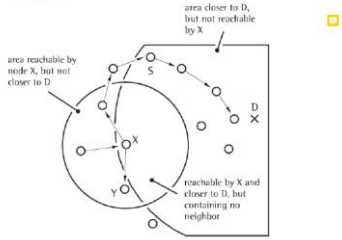
Example



→ just keep choosing the node that brings me closer to D (greedy)

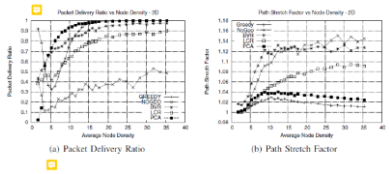
Pros and Cons of GPSR
→ Main advantage: minimum routing overhead. Each node only needs to keep list of neighbors and its locations.
Problem: Greedy routing does not always work, and we can get "trapped" in a local minima and cannot get out.
→ "Recovery" schemes proposed don't always work, i.e. "right hand" rule
→ Works with anycast mac

Example



→ Right hand rule: draw some invisible line, try to get around the obstacle by constantly going right, if goes past the line then go back to the normal routing protocol (i.e. find path that gives largest progress). Doesn't always work.

Simulation Results of Greedy



(a) Packet Delivery Ratio (b) Path Stretch Factor

→ works very well when the network is not very dense (because we are probably having a tree)

Hierarchical Routing in Sensor Networks

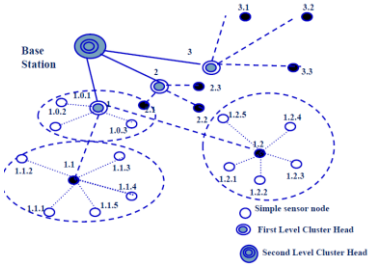
→ Network consists of a BSS, away from nodes, through which end user can access data from sensor network

→ BSS can transmit with high power

→ Sensors cannot reply directly to BSS due to low power constraints, resulting in asymmetric communication.

→ Nodes transmit only to their immediate cluster head, saving energy. Cluster head needs to perform additional computation on data (like aggregation).

→ Adjacent cluster members have similar data



Geographic Hash Table

- Problem: wants information generated by sensor nodes but do not know where to find them or who generate them
- Assumption: nodes know their locations
- Approach: combine location knowledge and storage capability
  - *PutData(data, coordinate)*
  - *GetData(coordinate)*

→ nodes can ask if there are information stored at that coordinate (it becomes a key in this hash table)

- Initialization
  - *Divide area into a 10x10 grid, total of 100 squares*
  - *Nodes know their locations*
- Insert data
  - *Input = (key:"apple", data)*
  - *Hash("apple") = coordinate(1.2, 4.5)*
  - *Store data in grid (1,4)*
- Lookup data with key "apple"
  - *Hash("apple") = coordinate(1.2,4.5)*
  - *Send query to grid (1,4)*
  - *Broadcast query to all nodes in grid (1,4)*
  - *Nodes with matching data reply*

→ We have information of different keys stored in different parts of network.

Issues

- How do you get, approximate or generate the coordinate?
- What if there is no node at the specified location/coordinate?
- What if over time, some nodes leave and others join?
- What is a node fails? How do you provide redundancy?

WLAN/WWAN

First generation analog system – there was demand for mobile transmission, a lot of people who needed to travel from different parts of a city etc

AMPS – Advanced Mobile Phone Service

Design Goals:

- High voice quality (near wire line)
- Small coverage area (cell radius: 1-16 miles)
- Frequency reuse planned in system design
- Low power mobile (handheld) transmitters (4 watts or less)
- Medium power base stations (10's of watts)
- Low blocking (2%) during busy hour
- System capacity for 100,000 or more customers per city

Characteristics of AMPS

- Frequency range (different frequencies for uplink/downlink), 25 MHz in each direction
  - *824 MHz ~ 849 MHz for mobile stations to transmit*
  - *869 MHz ~ 894 MHz for base station to transmit*
- Channel Bandwidth: 30kHz
  - *Max number of channel = 25,000/30 = 833*
- Data rate: 10kbps
- Spectral Efficiency: 10kbps / 30 kHz = 0.33 bps/Hz

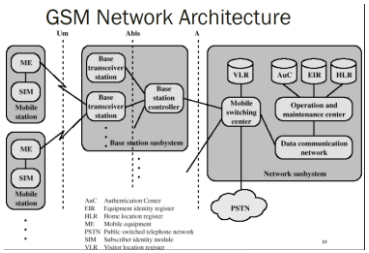
2G

- Introduced to early 1990s
- No more analog, use digital signals

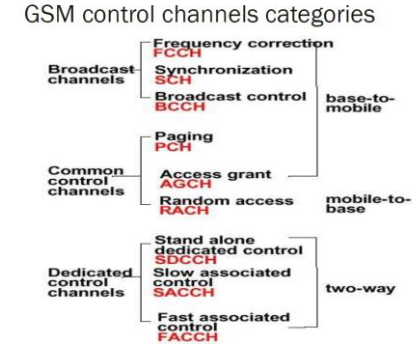
Digital traffic channels

- improve spectrum efficiency, increase capacity by operating with smaller cells
- Error detection and correction – digital transmission allows for the use of modern error detection and correction techniques
- first-generation systems are almost purely analog;

- Encryption – all second generation systems provide robust network security technology based on encryption and secure key distribution encryption to prevent eavesdropping
  - *AMPS authentication procedures are weak*
- Roaming
  - *Requires information transfer and business arrangement between systems, introduce IS-41*
- Support for non-voice services



- Base transceiver station → base station controller – converts analog signals to digital signals
- Only at data communication network then we have the IP packets



→ Mobile to base: RACH – mobile need to ask permission to base station to communicate to it. We use ALOHA here

→ Paging: I am looking for you

→ Very TDM based

→ While full rate is 22.8kbps, in practice we only send at 9600/4800/2400 because we need to do forward error correction, and sending at lower rate increases reliability

- Short Message Service (SMS) provides a connectionless transfer of messages with low capacity and long latency performance
- Each short message can contain up to 140 octets (or 160 7- bit characters)
- Messages are transported on the GSM SDCCH signaling channels
- Two types of SMS have been defined:
  - *Cell broadcast service*
  - *Point-to-point service*

→ SMS – done on the control channel that's why only can send short messages

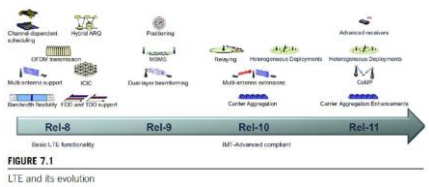
3G

- Provide high speed wireless communication for multimedia

→ "3.5G" – packet based connection, focused on sending data instead of voice

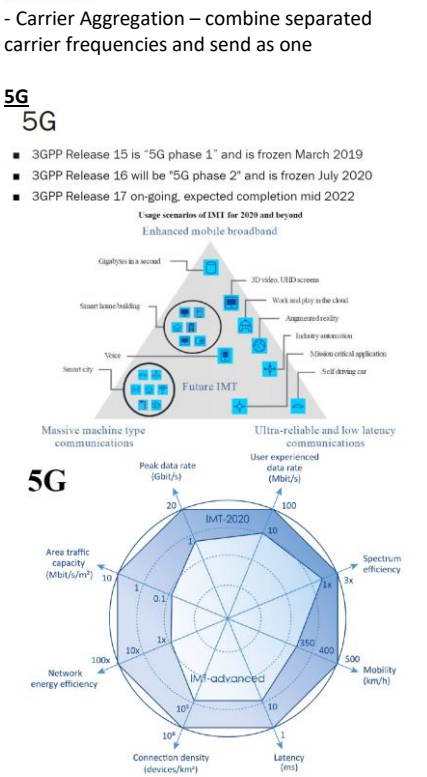
3GPP Long Term Evolution (LTE)

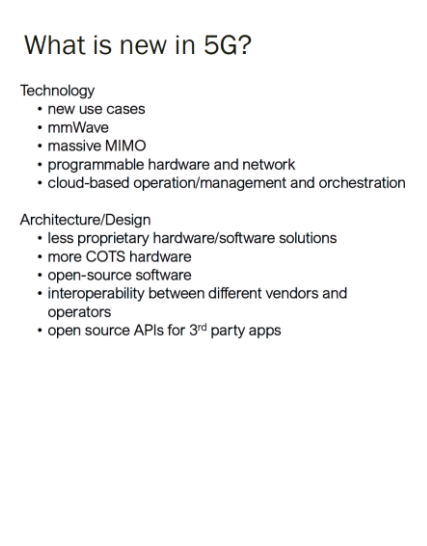
- Constantly add new stuff to it, technology is always improving



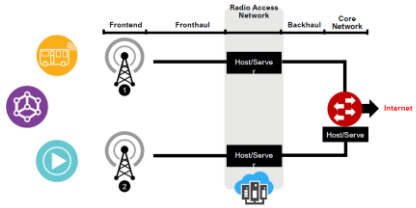
- Carrier Aggregation – combine separated carrier frequencies and send as one

5G





5G Architecture



C-RAN to vRAN

- Proprietary baseband hardware
- Proprietary baseband software
- Baseband HW/SW and radio should belong to the same vendor
- COTS baseband hardware
- Virtualized functions for baseband software
- Baseband SW and radio should belong to the same vendor

vRAN to O-RAN

Proprietary APIs for virtualized baseband functions	Open interface APIs for virtualized baseband functions
Proprietary radio and fronthaul	COTS radio hardware and open fronthaul interface
Baseband SW and radio should belong to the same vendor	Baseband HW/SW and radio can come from multiple vendor

Clock sync ppm: parts per million.
25 ppm = 25 * 10^{-6}

Question 4

The maximum channel utilizations using the ALOHA and slotted ALOHA protocols are 18% and 36% respectively. Network using the CSMA/CA protocol can achieve much higher channel utilization. Explain how CSMA/CA is able to achieve much higher utilization.

Two factors:

(1) With carrier sensing, a station should transmit only if it detects that channel is idle, reduces collision significantly.

(2) Collision detection overhead should be small compare to the amount of work that can be done per packet transmission.

Question 2

(a) Assume that time is divided into slots of duration 200ms. In each time slot, a node wakes up with a probability of 0.1. What is the average time taken for two such nodes to select a common slot to wake up?
(b) Consider an asynchronous quorum based neighbor discovery scheme [1], with time slots organized into a n by n grid. What is the smallest value of n needed to obtain a duty cycle of less than 5%?
(c) **Consider an asynchronous quorum based neighbor discovery scheme [1], with time slots of 200ms organized into a 10 by 10 grid. There are two users. User 1 chooses the 2nd row and the 8th column. The second user chooses the 5th row and 1st column. Assuming that the two users start at the same time (the same first row and first column), what is the shortest and longest duration between discovery?**

(a) P(both active) = 0.1 x 0.1 = 0.01
Average = 1/0.01 = 100 slots or 20 sec.
(b) D = 2n-1/n^2, want 0.05 > (2n-1)/n^2
n^2 > 40n – 20, n^2 – 40n + 20 > 0
n >= 40

(c) Meet at slots 11 and 48.

Shortest (11 to 48) = 37 slots, or 7.4s

Longest (48 to 11) = 63 slots or 12.6s

Q1 (4pt) Consider the RI-MAC protocol. A node wakes up periodically every 100ms and transmits for a total of 1ms to announce its presence and listen for response, if any. The radio bit rate is 200kbps. The transmission and reception/listening states draw 20mA. The sleeping state draws 0.1mA. You can assume that state transition is immediate.
(a) (1pt) Consider the scenario whereby there is no transmission. What is the average current consumption of a node?
(b) (3pt) Consider a scenario where sensor node A transmits a 100-byte packet to node B every 10s and B transmits a 100-byte packet to node A every 10s. You can assume that node A and node B will never transmit at the same time. Assume that ACK and other control messages incur no overhead. Calculate the average current consumption of node A.

a) Average current consumption = 0.99 * 0.1 + 0.01 * 20 = 0.099 + 0.2 = 0.299mA

b) Time to send 1 packet = 100*8/200 = 4ms
Transmit = 0.5 * 0.1 + 0.55 * 20 = 0.05 + 11 = 11.05mA or = 0.5*0.1 + 0.5*20 = 0.05+10 = 10.05mA
Receive = 0.95 * 0.1 + 0.05 * 20 = 0.095 + 1 = 1.095mA

Total = 0.98 * 0.299 + 0.01 * 11.05 + 0.01 * 1.095 = 0.29302 + 0.1105 + 0.01095 ~ 0.414mA
Or

Total = 0.98 * 0.299 + 0.01 * 10.05 + 0.01 * 1.095 = 0.29302 + 0.1005 + 0.01095 ~ 0.404mA

→ For RI-MAC, as transmitter, just take it as half of the time awake and half asleep. Don't need to include the time to transmit data

Link state based vs distance vector routing protocols

→ in distance vector, you only exchange routing table with your neighbours, but in link state you need to flood the whole network

Why CSMA/CA can achieve much higher channel utilisation compared to (slotted) ALOHA

Question 4

The maximum channel utilizations using the ALOHA and slotted ALOHA protocols are 18% and 36% respectively. Network using the CSMA/CA protocol can achieve much higher channel utilization. Explain how CSMA/CA is able to achieve much higher utilization.

Two factors:

(1) With carrier sensing, a station should transmit only if it detects that channel is idle, reduces collision significantly.

(2) Collision detection overhead should be small compare to the amount of work that can be done per packet transmission.

Rate Asymmetry

Question 1

Four 802.11 stations share a single channel. The four stations have different link rates, namely 2Mbps, 20Mbps, 50Mbps and 100Mbps. The standard 802.11 CSMA/CA MAC protocol with BEB is used.

(a) What is the average throughput if all four stations always have packets to transmit?
(b) What is the average throughput if the protocol used (not CSMA/CA) ensure that all stations are given same amount of time to transmit?

Let time to transmit 1 packet at 2Mbps be 1 time unit.

Time to transmit 4 packets = 1 + 1/10 + 1/25 + 1/50 = 1.16

Normalize rate (to 2Mbps) = 4/(1.16) = 3.448

Average throughput = 2 * 3.448 = 6.897Mbps

Try to work out the case for equal transmission time.

→ For part b, let y be the duration, packets sent = 1 + 10 + 25 + 50 = 86, throughput = 86/4 * 2 = 43Mbps.

Choosing TDMA for Wifi

2 factors influence MAC selection:
1. Dominant traffic type: Wifi's expected dominant traffic is data (prefers high throughput, not latency sensitive). There can also be many stations but only some stations will transmit at any one time
2. Expected QoS (Quality of Service): TDMA is a better choice if the traffic is continuous / constantly active, and the predictable latency is important

CSMA/CA – High throughput and good protocol if there are many users sharing a channel, good for short bursty traffic
Polling/TDM/Reservation – Low loss, short latency (good for VoIP)
Aloha/Slotted aloha – Good when there is low data rate on the channel, cheap to implement/low overhead, does not perform carrier sensing

Question 1

What are (1) wireless mesh network, (2) mobile ad hoc network and (3) delay/disruption tolerant network? What are the differences in the design of the respective routing protocols for each of these 3 different type of networks?

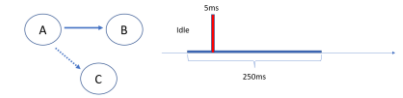
- Wireless mesh network - wireless network of static wireless nodes, highly interconnected.
- Mobile adhoc network - non static mobile network, where wireless topology may change dynamically in an unpredictable fashion. nodes are free to move and each node has limited transmitting power.
- DTN - nodes mostly disconnected, short range, we want to improve delivery ratio.

XMAC Average Current Consumption

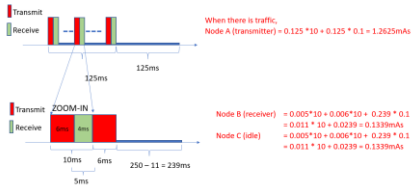
2020/2021 Written Assignment Q5b

There are 3 sensor nodes. Every 1s, Node A transmits a very small packet to node B. Node C does not send or receive any data. Assume that **X-MAC** is used. A receiver wakes up every 250ms to sample the channel for 5ms. The transmission/reception/listening and sleep states draw 10mA and 0.1mA respectively. **You can ignore the energy consumed by the transmission/reception of the data and ACK packets.**

In this X-MAC implementation, the duration of a probe packet send by the transmitter is 6ms and the interval between probe packets is 4ms. A node wakes up every 250ms to sample the channel for 5ms.



For the receiver to decode the probe packet correctly, it must wakeup before the start of the probe packet. Hence, if it detects channel activity but cannot decode the probe packet correctly, it has to stay awake to receive the next probe packet. Calculate the average current drawn (in mA) by nodes A, B and C.



→ For receiver: Keep sending probes until someone ACKs. On average, he will be sending for half of the entire time slot so half sleeping, half transmitting
→ For receiver and packet is for that node: Node needs to pick up a full packet. In this case, the probe is a 6ms transmission, 4ms gap, 6ms transmission. So on average, he will listen for 6 + 4 / 2 = 5ms, before receiving for 6ms. The rest of the time (250 – 6 – 5 = 239ms) he will be sleeping. Might need to include data packet afterwards too if there is.
→ For node that receivers but not intended for it, it is exactly the same, assuming the probe packet is the data packet itself.

3G vs 4G

	3G	4G
Switching Technique	packet switching	packet switching, message switching
Peak Download Rate	100 Mbps	1 Gbps
Network Architecture	Wide Area Cell Based	Integration of wireless LAN and Wide area.
Frequency Band	1.8 – 2.5 GHz	2 – 8 GHz
Services And Applications	CDMA 2000, UMTS, EDGE etc	Wimax2 and LTE-Advance
Forward error correction (FEC)	3G uses Turbo codes for error correction.	Concatenated codes are used for error corrections in 4G.

TDMA vs FDMA

Sr. No.	FDMA	TDMA
1.	FDMA stands for Frequency Division Multiple Access.	TDMA stands for Time Division Multiple Access.
2.	Overall bandwidth is shared among number of stations.	Time sharing of satellite transponder takes place.
3.	Guard bands between adjacent channels is necessary.	Guard time between adjacent slots is necessary.
4.	Synchronization is not required.	Synchronization is necessary.
5.	Power efficiency is less.	Power efficiency is high.
6.	It requires stability of high carrier efficiency.	It does not require stability of high carrier efficiency.
7.	It is basically used in GSM and PDC.	It is basically used in advanced mobile phone systems.

GSM vs LTE

GSM	LTE
GSM is short for Global System for Mobile Communications.	LTE stands for Long Term Evolution.
GSM is used for both voice calls and data.	LTE is used for high-speed data communication.
Uses two frequency bands 900 MHz and 1800 MHz as GSM-900 and DCS-1800 systems.	Frequency bands 1 to 25 are reserved for FDD and frequency bands 33 to 41 are for TDD.
Information is carried through channels which are separated into physical and logical channels.	LTE uses physical, logical, and transport channels for processing of information.
It uses FDMA and TDMA.	It uses OFDMA and SC-FDMA.

DB Difference Between.net

How does SMS work?

When you send an SMS message, the message gets transmitted from the sending device to the nearest cell tower. That cell tower passes the message to an SMS center (SMSC). Then the SMSC forwards the SMS message to a cell tower near the receiving device. Lastly, that tower sends the message to the recipient's device.

Poisson

Poisson Distribution Formula

$$P(X = x) = \frac{\lambda^x e^{-\lambda}}{x!}$$

where
x = 0, 1, 2, 3, …
λ = mean number of occurrences in the interval
e = Euler's constant ≈ 2.71828