Università degli Studi di Padova

Dipartimento di Matematica

Corso di Laurea Magistrale in Informatica

Wireless Networks

v0.1.1

A.A. 2017-2018

For new releases visit https://github.com/Polpetta/WirelessNetworkNotes/releases
Contributions and issues reporting are welcome.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

Wireless Networks popularity is growing over the years, with Internet and laptop use exploding. We are experiencing a demand for both low and high rate data, with smartphones having opened new wireless scenarios. We now talk of Web 2.0, where people can modify and upload new data and Web Squared, where sensors and machines generate and upload data. Think of Google Maps using your phone to detect traffic levels.

We envision a future with ubiquitous communication among people and devices, all thanks to WN. But, we can't forget that WN, while having a lot of potential and being used in a lot of different applications pose a series of challenges one cannot ignore:

- Wireless channels are a difficult and capacity-limited communication medium, and wired connections will always be better;
- Wireless Networks are very hard to plan: the position of the nodes changes and you often don't have patterns you can exploit to design them. Also, WN can be very heterogeneous;
- Energy and delay: having a shared channels with rules, usually hard to implement, to regulate traffic naturally leads to delays. As for energy, don't forget we are dealing with a mobile device with a limited amount of battery.

We can see an overview of multimedia different requirements in Table 1.1.

From this table it is clear that one-size-fits-all protocols and designed, used in wired networks, do not work well with WN. More in detail:

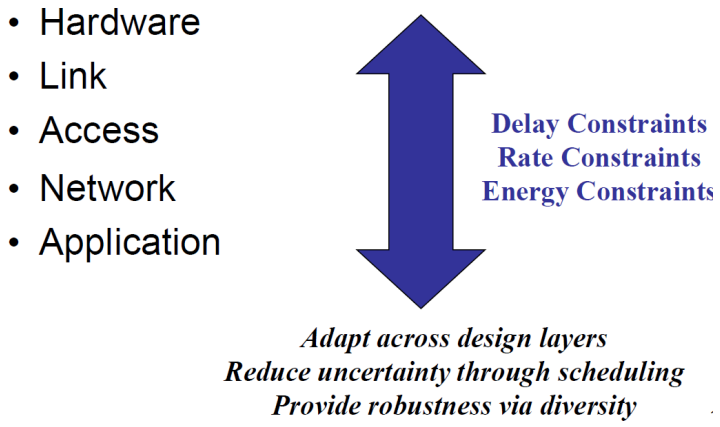
- Delay: for video, <100 ms if we have interactive videos, otherwise we can afford a bit more (voice is more important). In games, jitter* may not be a problem if I am delaying with constant jitter and have a specific type of game (for example: race games) because the player adapts to the jitter, reacting a bit earlier.
- Packet Loss: for all media types except data, we can in reality tolerate up to 5%. For examples, in games not all data is equally important, we can afford losing the less important ones and still be satisfied with the overall game experience. Also for voice, even with 5% loss I can still understand well.

	Voice	Data	Video	Game
Delay	<100ms		<100ms	<100ms
Packet Loss	<1%	0	<1%	<1%
BER	10-3	10-6	10-6	10-3
Data Rate	8-32Kbps	1-100Mbps	1-20Mbps	32-100Kbps
Traffic	Continuous	Bursty	Continuous	Continuous

Table 1.1: Multimedia Requirements

- Bit Error Rate: we assume that, when we lose a bit, we lose the whole packet.Voice packets are sent very frequently and are very small, that's why we have a high BER, while for video the packets are bigger. Games are similar to VoIP.
- Data Rate: indicates how much bandwidth we are consuming.
- Type of Traffic: for voice, video and games, we have a device transmitting an uninterrupted stream (even though video only transmits in 1 direction). Data is different, we don-t care for every single packet we are downloading, we are interested in the total time of the complete thing. That's also why we require 0 packet loss.

Crosslayer Design When dealing with the different network layers, we can decide to keep them strictly separated, with no communication and information exchanged between them. That means, we have different layers with different functionalities that work together without overlapping. The Crosslayer Design blurs the separation between layers, allowing information in one layer to become available to other layers too. This means we adapt across deign layers, reducing uncertainty through scheduling and providing robustness via diversity. On one hand, one could argue that this messes up and complicates the code but, on the other, better communication can lead to higher performances. In WN, Crosslayer Design is not seen badly and therefore quite commonly used.



CHAPTER 2

Current Wireless Systems

We now present a brief review of some of the most commonly used wireless systems.

2.1 Cellular Systems

Cellular Systems are based on cells. Each cell has a different signal from the ones in its immediate vicinity to allow for seamless transitions (see the different colors in Figure 2.1). Frequencies, time slots and codes are reused at specially-separated locations. At the center of each cell there is an antenna, certain antennas are more powerful than others and can manage hand-offs and control functions. These antennas are called Base Stations. Hand-offs can be can be horizontal if they occur between the same technology (4G→4G) or vertical otherwise (3G↔4G). Having larger cells means having lesser transitions but also more people accessing the same cell. If there are too many people and there isn't enough bandwidth you won't be able to use your device. Shrinking cell size increases capacity, as well as networking burden.

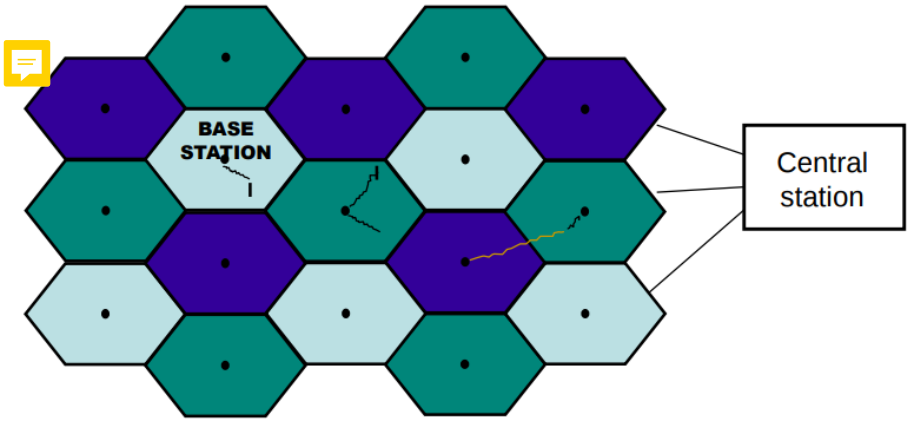


Figure 2.1: Cellular Systems

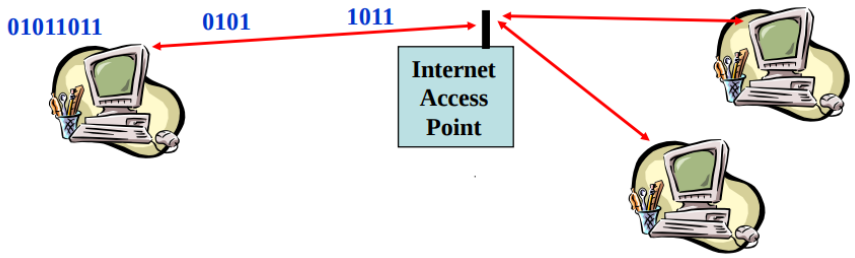


Figure 2.2: Wireless Local Area Networks (WLAN)

2.2 Wireless Local Area Networks (WLAN)

WLAN is used to connect local computers (within 100m range). We have an access point, connected to the Internet through a wire and using protocol 802.11. The access point not only forwards data, it also coordinates users and breaks data into packets. In fact, even though data arrives already broken, access points break it further, because the smaller the packet transmitted, the better. Obviously, this becomes a problem when we relate it to overhead : since overhead has a fixed size, independent from the packet's size, we may find ourselves in the situation where, faced with 1000 *data* + 40 *overhead* we end up with 500 *data* + 40 *overhead* plus 500 *data* + 40 *overhead*. We still prefer splitting because, when facing a big error rate, we don't want to lose all the packets. The WLAN channel access is shared using random access and this system forms the backbone of current Internet, providing a best-effort service.

* *overhead*

splitting

random access

Wireless LAN Standards See overview in Table 2.1.
Note that:

- 802.11b is a free frequency and therefore crowded
- 802.11g is quite popular, using a higher bandwidth. It brought changes in the physical layer.
- 802.11n MIMO (Multiple-Input and Multiple-Output). A device using this standard can be easily recognized because it uses multiple antennas and multiple channels.

	802.11b	**802.11g**	**802.11n**	**802.11ac**
Frequency	2.4GHz	2.4GHz/5GHz	2.4GHz/5GHz	2.4GHz/5GHz
Bandwidth	1-11Mbps	54Mbps	300Mbps	500Mbps
Add. features		OFDM	MIMO	MIMO

Table 2.1: Wireless LAN Standards

2.3 Satellite Systems

They are used to cover very large areas. There exist two types of satellites that have different orbit heights: GEO satellites (geostationary) stay at about 39000 km while LEO satellites (Low Earth Orbit) stay lower, at 2000 km. GEO are so high that remain stationary and can cover the whole surface while LEO revolve continuously in order to not fall. Satellites are usually a two way system, optimize for one-way transmission: usually satellite $\Rightarrow$ earth is faster than the other way.

2.4 Bluetooth

Bluetooth was invented as a cable replacement, and this is why it is short-range (about 10 m). The band used (2.3 GHz) is crowded but cheap, and Bluetooth originally has one data and three voice channels. One other reason why the range is so short is to avoid consuming too much battery. Also, note that the signal strength decreases exponentially and not linearly with the distance.

CHAPTER 3

Emerging Wireless Networks

We now further our system overview into some new, emerging systems.

3.1 Ad-hoc Networks

Ad-hoc wireless networks are an emerging wireless system, initially developed for military purposes. Ad-hoc networks are networks without a fixed infrastructure, where you don't have an access point, nor a backbone infrastructure to your network. The network is made up of equal nodes and based on peer-to-peer communications. To reach nodes that are too far away from the sender, we exploit multihopping. We could memorizing the network's routing scheme, but note that we have a dynamic network topology! So, since you don't usually know where you're going, you can't use the more common routing protocols.

Ad-hoc networks

Ad-hoc Networks are very flexible, useful for many emerging applications. The capacity of such networks is generally unknown because depending on which node is transmitting, the nearby nodes will be blocked to avoid interference. Consider this structure (capital letters are nodes, arrows represent links):

$$B \leftrightarrow A \leftrightarrow C \leftrightarrow D$$

If D wishes to transmit to C while A transmits to B, it wouldn't be able to do it because C is already occupied with A's messages (interference!). Furthermore, we never really know the size and routing of our network, both can change very frequently. In this kind of networks **crosslayer design** is critical for efficiency and very challenging. Finally, energy constraint impose interesting design tradeoffs for communication and networking: for example, choosing to increase transmission power to avoid multihop leads to problems both in energy consumption (remember, if the distance doubles, the energy required is four time greater) and interference. I may occupy the whole network for just one transmission.

crosslayer design

3.1.1 Sensor Networks

Sensor Networks are a specific type of ad-hoc networks. You have different sensors, each used to monitor a specific thing. The driving constraint is energy, since sensors

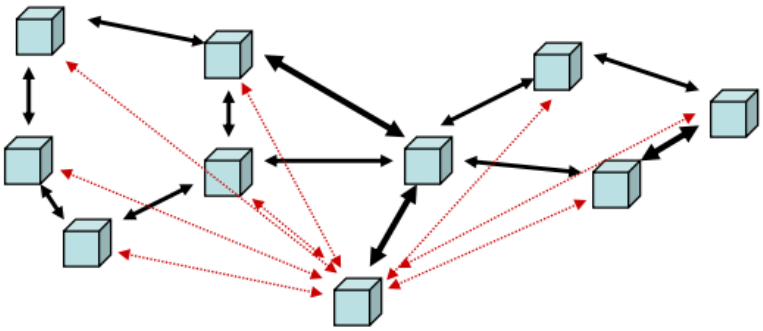


Figure 3.1: Ad-Hoc Network

usually have a limited amount of energy and are powered by non-rechargeable batteries. To spare energy, they are usually turned off and periodically turn on to monitor the environment. When they are on, they may help other sensors to send messages, even if a very good synchronization is needed. Usually these networks are very large (up to 100,000 nodes) but, unless they are sending pictures of the environment, nodes have a low per-node rate, sending very simple data: temperature values, pollution levels, water levels... The data flows to a centralized location that then analyzes, processes or sends it farther away. It is important not to lose packets, so aggregating should be mitigated to avoid losing big portions of data. Also, avoid big delays, since data is highly correlated in time and space.

3.1.1.1 Underwater Sensor Networks

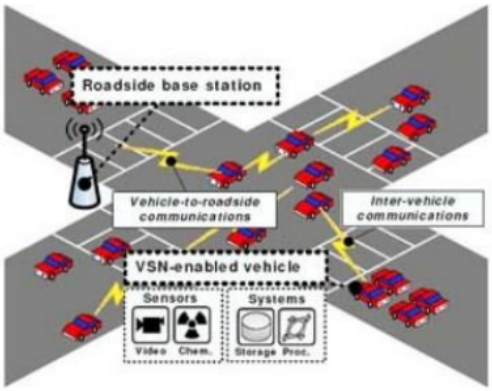
Main challenge: electromagnetic waves don't propagate well in water, so one has to resolve other forms of communication, for example sound waves. Using sound waves means that the connection will be much slower (sound speed instead of light speed). On the other hand, two waves transiting at the same time but far enough from each other will not collide. Remember: there is a collision if the receiver receives at the same time two different signals, not when two messages travel together. As for battery consumption, they usually rely on battery.

3.1.2 Mobile Ad-Hoc Networks (MANET)

These are ad-hoc networks with a focus on mobility. They were initially used for military purposes and then extended to civilian tasks. The topology of the network continuously changes and the protocol used has to adapt consequently, keeping into account that every information received may already be obsolete. In some cases, one may need to resolve to flooding. Like ad-hoc networks, they are deployable, re-configurable and portable. They may be created to satisfy a temporary need.

3.1.2.1 Vehicular ad-hoc Networks (VANET) ==> the nodes are the vehicles

Note that, when considering vehicles, since they have an engine, reducing energy consumption isn't a priority. Even further, a group of vehicles on a road moving



DSRC CHARACTERISTICS

Parameter	Value
Range	1000m
Frequency Band	5.9Ghz
Speed	$\leq$ 85mph
Data Rates	6-27Mbps(depending on Range)

Figure 3.2: Vehicular ad-hoc Networks (VANET)

in the same direction move together in respect to each-other. The technology for automated vehicles is already present, the two main concerns are:

1. Legal problems: in case of accidents, whose fault is it?
2. Moral decisions: if a car has to steer to avoid an accident but in doing so it hits a passer-by, what is the best decision?

Accidents are estimated to occur mainly when having automated cars mixed with regular cars (SW malfunctioning or hijack may still occur but are the secondary concern now). In a controlled environment we can monitor, while people are unpredictable. Finally, there is already a frequency reserved for vehicles communication (Standard IEEE 802,11p).

3.1.3 Opportunistic ad-hoc Networks

Opportunistic Nets were driven by commercial application needs and designed for when there may be an available Internet connection, but we try to replace it opportunistically with an ad-hoc network. Basically, they don't use homogeneous nodes (like in traditional ad-hoc) but, instead, always try to use the best available. Their main feature is flexibility.

3.2 Mesh Networks

Mesh Networks are something between ad-hoc networks and wireless ones. They are different because the can connect to both regular and completely wireless access points. In this sense, they are ad-hoc opportunistic extensions of a fixed urban infrastructure. Their main purpose is to create a low-cost, easily deployable, high performance wireless coverage. When designing a Mesh Network, one must carefully choose the routing protocol, considering all the types of connection you may need. Routing protocols have to achieve fairness and local balancing, also remembering that in multihop, having two nodes communicate may prevent others in range. Finally, it is difficult to determine a quality of service, that is stating a quality standard and being able to assure it.

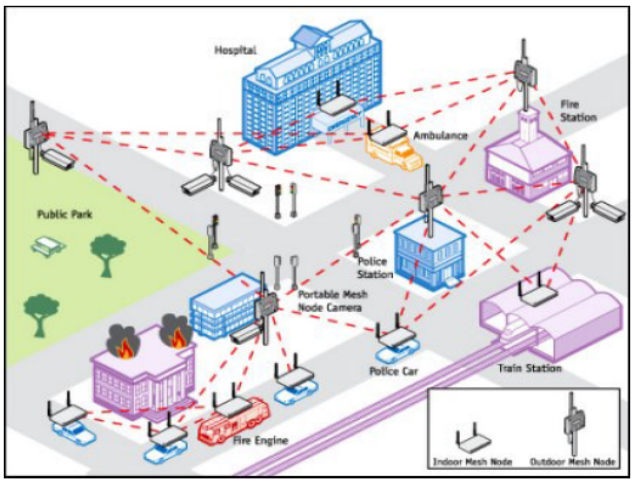


Figure 3.3: Mesh Network (the green ones are regular access points)

3.3 Distributed Control over Wireless Links

These networks deal with automated vehicles: cars, Unmanned Airborne Vehicles (UAV), insect fliers and so on. In these kind of networks it is important to have a good control of the environment, so it is important to avoid packet loss and/or delays and also assuring a good bandwidth and speed when delivering messages. Avoid queues since they cause delays. The controller design should be especially robust to network faults.

3.4 Radio Frequency Identification (RFID)

RFIDS are based on magnetic fields and are usually tags without a battery (even though they can have it). They store more info than a simple barcode and are therefore used in supply chains instead of them. They don't need a direct optical reading but, instead, require an emitter of electro-magnetic waves that charge the tag. The charged tag sends a message containing all the information back to the server, that can then check it. Even though it contains more information that a simple barcode, it is more expensive. Their popularity is increasing, and standards are currently under development.

3.5 Nano-Networks

Networks made up of very tiny objects and very small distances. You can't just replicate in scale what you would normally do but instead you have to rely on different means of communication, very popular are calcium signals.

Figure 3.4: Radio Frequency Identification (RFID)

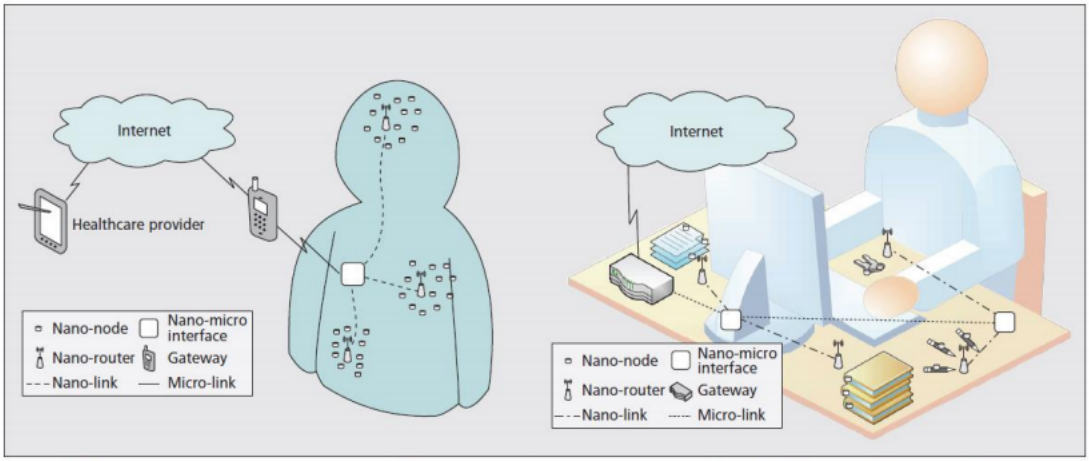


Figure 3.5: An example of nano-network

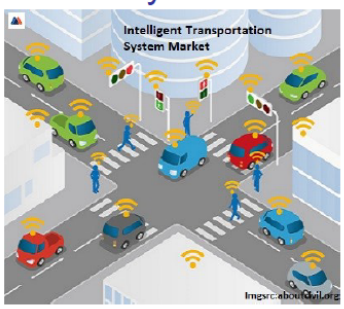
Main Points

- The wireless vision encompasses many exciting systems and applications
- Technical challenges transcend across all layers of the system design
- Wireless systems have limited performance and interoperability
- Standards and spectral allocation heavily impact the evolution of wireless technology
- Huge potential for future applications and systems

Case Studies and Project Topics

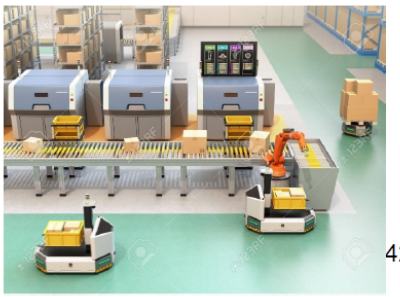
Internet of Space Things (Cubesat)

Intelligent Transportation System

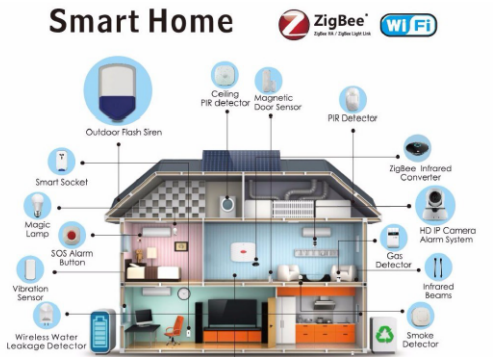


AR and Interactive Games

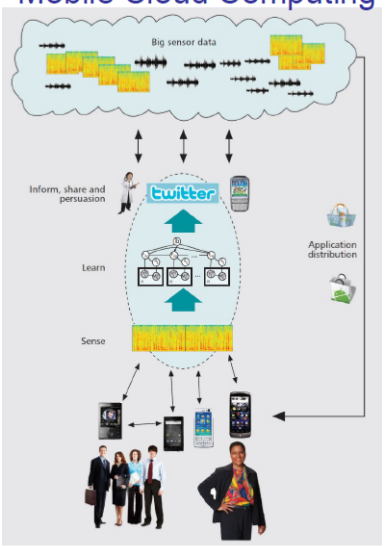
Industry 4.0



Side Channel Attacks



Mobile Sensing & Mobile Cloud Computing



And more…

CHAPTER 4

Physical Layer

4.1 Radio Signals

Most of the wireless technology used today is based on radio frequency signals. Radio signals are electromagnetic energy generated by high frequency alternate current in antennas. They can have different frequencies and lengths and propagate differently depending on the medium they need to cross.

Radio frequencies have three main properties:

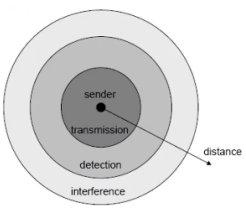
- **Amplitude**: determines how far a signal can travel (remember that reaching 2x distance requires 4x power). It is the difference between the highest and lowest wave peak.
- **Frequency**: the number of oscillations in 1 second
- **Wavelength**: the distance between two high (or low) peaks. The wavelength is $\frac{c}{frequency}$ and this means that, in practice, antennas work better with sizes 1, $\frac{1}{2}$, $\frac{1}{4}$ of wavelength.

Propagation From its origin, after a certain point the signal is no longer detectable. If there is an obstacle the signal loses power more quickly because obstacles can reflect or absorb waves depending on materials and waves frequencies. When dealing with RF propagation we can rely on a rule of thumb: high frequencies are good for short distances and are affected by obstacles, while low frequencies are good for long distances and are less affected by obstacles.

RF have other properties that describe their behavior: they can phase, that is done shifting the wave (in degrees or radians) and are affected by the physical orientation of the antenna (polarization). Waves propagate in a toroid form (donut form).

The propagation range depends on power, obstacles, the receiver's sensitivity and many other factors. We have different ranges:

- **Transmission Range**: marks how far can the communication reach;
- **Detection Range**: marks how far the detection of the signal is possible;



- **Interference Range**: marks the distance at which the signal is too far away from the sender to be detected and so it just adds to the background noise.

physical effects

Also, a lot of different physical effects can affect propagation, in particular: shadowing, reflection, refraction (happens when passing to a different medium), scattering (at a corner the power is sent in different directions), diffraction (splitting the signal in 2). The final signal received could therefore be better or worse than the one originally sent, with signals taking different paths towards their destination. See Figure 4.1 to see the different effects of propagation.

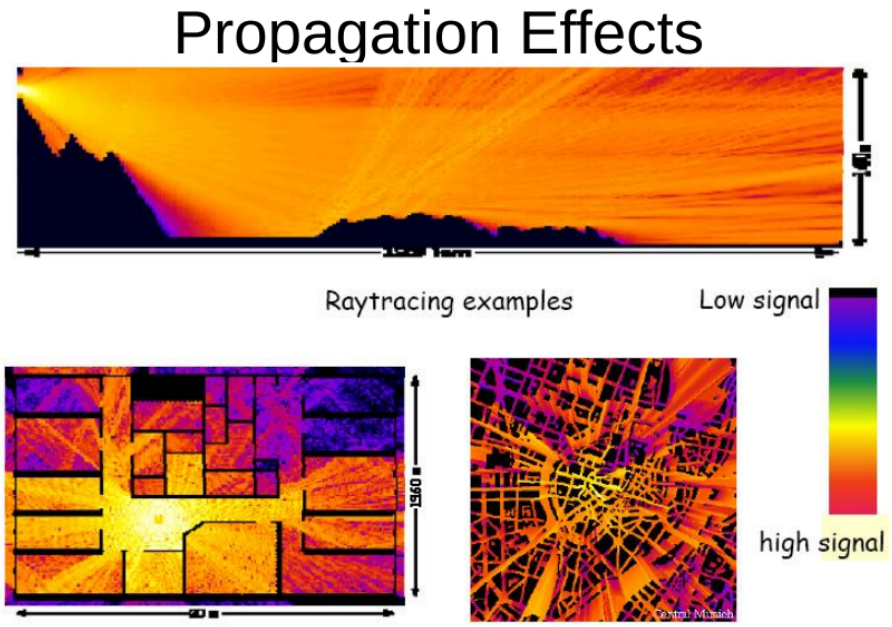


Figure 4.1: The different effects of propagation

4.2 Antennas

Antennas convert energy in RF waves (transmission) and vice-versa (reception). As explained in the previous section, the size of antennas is related to the RF frequency of transmission and reception. Starting from the simple dipole antenna, we can have different types of antennas.

4.2.1 Omnidirectional Antennas

They radiate RF power equally in all directions around the vertical axis, the most common example is the dipole antenna. They can cover both short and long range. Near and below the dipole the signal is weak, a better radiation is obtained in sub-areas around the dipole. An omnidirectional antenna is better suited when there is a need for uniform radio coverage around a central point. Outdoors, it can be used for point-to-multipoint connections (star topology). Note that the tilt of the antenna changes the position of the y axis and can be exploited to reach computers on different floors.

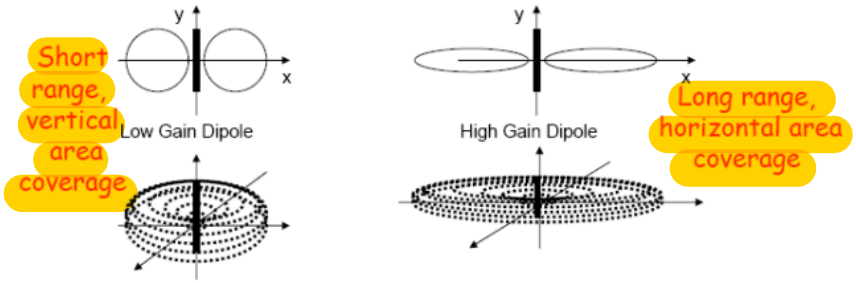
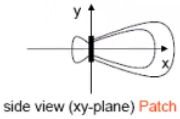


Figure 4.2: Omnidirectional Antennas

4.2.2 Semi-directional Antennas



They don't transmit in all directions and can be positioned against walls without wasting too much signal. Some types are Patch/panel and Yagi.

4.2.3 Highly-directional Antennas

They are used by parabolic dishes and grids. They reach very far, covering a distance of up to 60 km assuming no obstacles in-between (LOS=Line Of Sight). Putting the sender and the receiver too far away causes the signal to hump against the curve of the Earth. Highly-directional Antennas need to be placed carefully, for the signal to be received they have to be perfectly aligned and prevent wind or bad weather from tilting them. Another important point to consider is the **Fresnel Zone** : when transmitting, the power isn't all concentrated in the center and it doesn't actually proceed in a straight line.

Fresnel Zone

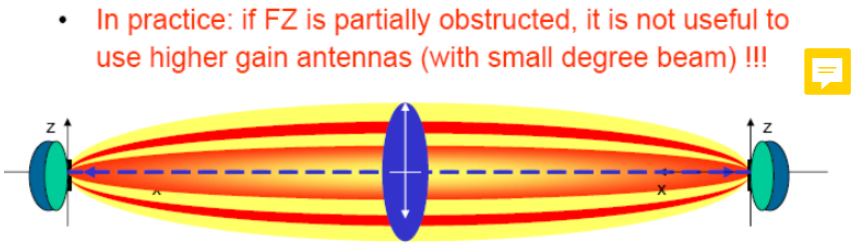


Figure 4.3: The Fresnel Zone: the red areas represent high-energy areas of the signal

We have to be aware of this fact and check that there are no obstacles blocking the external high-energy areas (that end up carrying most of the signal!). The diameter of the Fresnel Zone depends on 2 parameters:

- Distance (between two antennas)
- Frequency (of the signal emitted)

So it is independent from the energy used.

4.3 Wireless Spectrum

Wireless uses a dedicated area of the spectrum. See Figure 4.4 for a breakdown of the wireless spectrum.

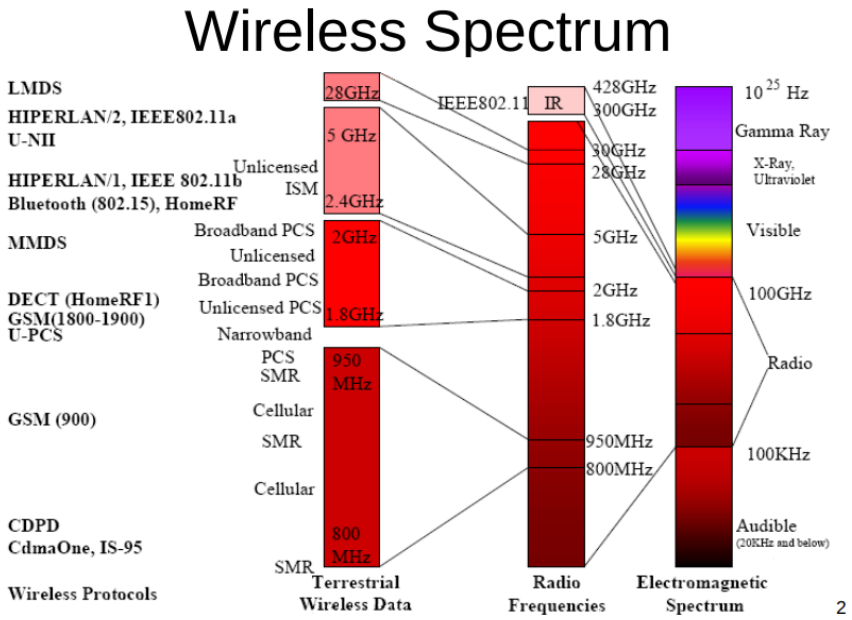
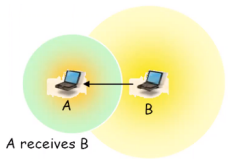
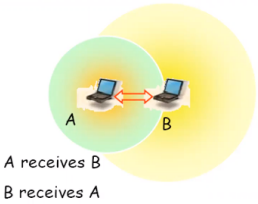


Figure 4.4: The wireless spectrum

How can wireless channels have different bandwidth (BW)? We can't make bits move faster, they already travel at the speed of light. But, instead, we can think of sending them faster (more frequently, they travel at the same speed but are more packed). Two nodes that need to exchange information need to be in each other's radio transmission coverage radius. Depending on the transmission power of the nodes we can have different situations:



- Unidirectional Link: sometimes improperly referred as "asymmetric link" (we have an asymmetric link if A transmits at 10Mb/s while B transmits at 2Mb/s), we have a unidirectional link if, given A and B, A receives B but B cannot receive A. This is the case, for example, when B is just broadcasting or flooding the channel. Note that with a unidirectional link we can't share a file between A and B: B will be waiting for an ACK that never comes.



- Bidirectional Link: A receives B and B receives A. It can be both:
  - symmetric: all devices send at the same speed;
  - asymmetric: different speed of transmission between devices in the network.

4.4 Examples of wireless network technology

4.4.1 Frequency Hopping Spread Spectrum

To make a better use of the whole band, we can divide it in smaller frequencies. The sender and the receiver will know the jump sequence and transmit in a "secret" pattern, to others this will appear as impulse noise and so the communication will be more difficult to detect. Because of the continuous hopping, it has a low throughput.

4.4.2 Infrared Technology (IR)

Requires short rage and line of sight (LOS). It uses frequencies just below the visible light and cannot penetrate opaque objects. It has a high bandwidth.

4.5 Wireless Network Coverage

We have different names for wireless technology depending on their coverage:

- Wireless Wide Area Network (WWAN): it covers large geographical areas, like countries or regions;
- Wireless Metropolitan Area Network (WMAN): metropolitan coverage;
- Wireless Local Area Network (WLAN): used for local area coverage, like campuses, and buildings (even the one commonly used at home);
- Wireless Personal Areal Network (WPAN): reduced area coverage, for example Bluetooth, houses are already too much;
- Wireless Indoor Area Network (indoor): really short coverage.

Note that, even in wireless connections, access points need to have both the wired and wireless protocols because they don't just relay the information, they have to translate it.

4.6 Multiplexing

The goal of multiplexing is to have different channel that can share the same medium using different dimensions. These dimensions are:

- space;
- time: user uses one channel for a certain amount of time. This technique requires synchronization;
- frequency: spectrum is divided into smaller frequency bands. A certain band is given all the time to one single channel. May be a waste of bandwidth if use of channel are unbalanced;

- code: all channels use the same spectrum but speak a different language. Data of channels is blob together but can be decode thank to the different code assigned to channels.

CHAPTER 5

TCP

TCP (*Transmission Control Protocol*) is the most widely used Internet protocol. It's located in the middle between the Application layer and the IP layer and it's a two way, reliable, byte stream oriented end-to-end protocol which includes both flow and congestion control.

Main features TCP has the following characteristics:

- it's an end-to-end protocol (software only on the end hosts) *end-to-end*
- it establishes a connection...
  - reliable. A checksum is used to detect bit level errors and sequence numbers to detect sequencing errors. This means that, if something arrives out of order, TCP reorders it before forwarding it to the application level.
  - correct
  - ordered (thanks to sequencing numbers)
- full duplex *full duplex*
- it is not so appropriate for current wireless scenarios (but there are many variants of TCP)
- byte-stream oriented. TCP sends segments (packets) through a physical layer (IP, MAC, whatever) and then waits for an ACK. A missing ACK means that the packet was lost.
- has congestion control: it keeps the sender for overrunning the network

TCP does 2 types of control, the **flow control** in order to avoid overwhelming the receiver and the **congestion control** where the flow slows down when there is a loss of packets. ***flow control*** ***congestion control***

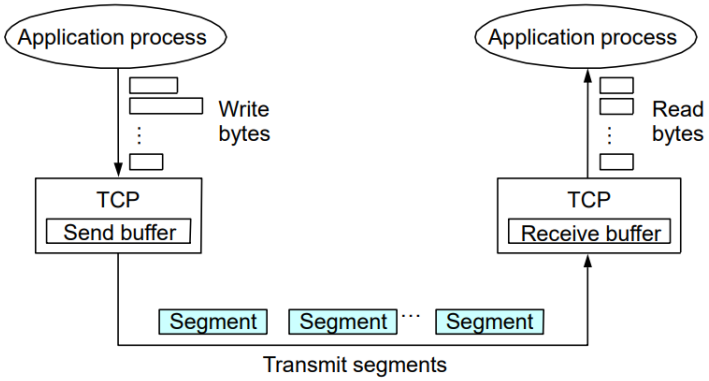


Figure 5.1: A image representing how TCP works

5.1 Reliability in TCP

RTO Timeouts are used to detect lost packets: this requires setting an RTO (*Retransmission Timeout*). If the ACK is lost, after waiting the RTO, we retransmit that packet. There are obviously some issues: how much should the RTO be? Are we sure that, in wireless networks, a missing ACK equals a missing packet? As for the optimal setting of the RTO, we usually want to wait at least one RTT (*Round Trip Time*) before retransmitting. This values have to be calculated accurately, because a low RTO means unneeded retransmissions, while having a high RTO leads to a poor throughput. Basically, the RTO estimator must adapt to change in RTT, usually having $RTO = 4 \cdot RTT$.

Sequencing numbers are used to detect sequencing errors, meanwhile timeouts are used to detect lost packets. RTT determines the speed of TCP connection.

5.2 Fast Re-transmission

Consider duplicate ACKs, repeated ACKs for the same segment. An ACK is generated every time we receive a packet, but it doesn't state the number of the last packet received, instead it returns the number of the last packet of the longest complete sequence received. For example: if A sends packets 1-6 without errors, it receives ACK(6). Then 7 goes missing and 8 is received. The ACK will be ACK (6). So, we have a *dupack* duplicate ACK (dupack). Duplicate ACKs occur in case of packet loss, packet re-ordering or window update. Generally, we can assume that the receipt of 3 or more dupacks indicates a packet loss and so we don't need to wait for the timeout to retransmit. This is a faster way to detect packet loss, since the name **Fast Retransmit**. Instead of waiting $4 \cdot RTT$ we resend after $1 \cdot RTT$, which is after three or more dupacks. Duplicate ACKs can happen due to loss of packets, packets re-ordering or flow control window update.

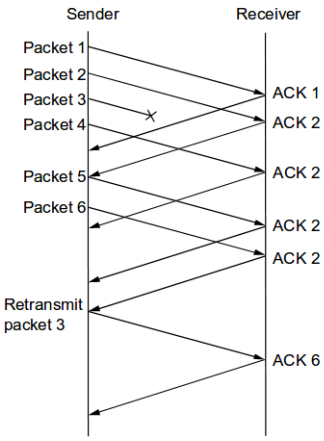


Figure 5.2

5.3 Fast Recovery

TCP normally increases the number of packets sent by 1 every RTT and, every time a packet is lost, it restarts form 1. In Fast Recovery, instead of going back to 1, we start at SSTHRESH (*Slow start threshold*), a threshold usually put at about $\frac{1}{2}$ of where we were. Note that even using Fast Recovery we don't have an optimal situation for wireless networks, having lost a packet due to network errors doesn't mean I have bandwidth problems and so it doesn't make any sense to slow down restarting from $\frac{1}{2}$, we should just resend the packet.

SSTHRESH

5.3.1 Congestion Window

A Congestion Window (*cw*) is used to keep the sender for overrunning the network sending too many packets. A *cw* sets the number of packets without ACK that can be currently traveling in the network. The window starts small and, every time a whole window has been received (ACKs have been sent back), the window increases by 1. Otherwise, if some packets are lost, the window either return to 1 or goes back to the slow start threshold (depending on the TCP version in use).

Congestion Window

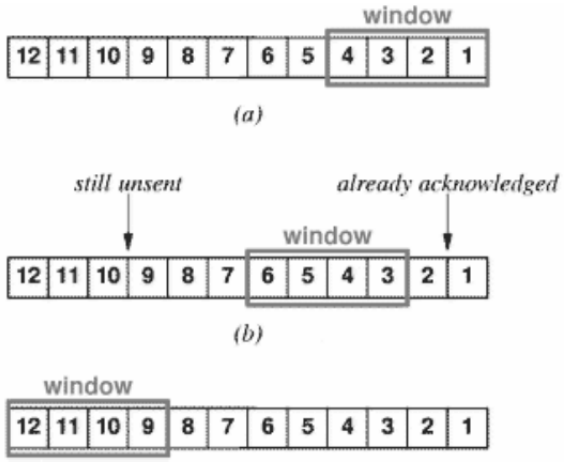


Figure 5.3: Congestion Window schema

5.4 Flow control

The receiver uses an **Advertise Window** to manage flow control[1]. The adWindow is the max number of bytes that it can receive (basically how much space is left in the buffer). It is communicated back to the sender in the ACK. On the other hand, the sender's actual window is the **Sending Window** and it represents the actual bytes sent out. It corresponds to the minimum between its congestion window and the receiver's advertised window ($min(congestion\ window,\ advertise\ window)$). We can think of the network channel as of a tube and our objective is to keep

[1] As said by the professor during lesson: "The bandwidth/capacity depends on the smallest part of the pipe" $pipeSize = Delay \cdot Bandwidth$

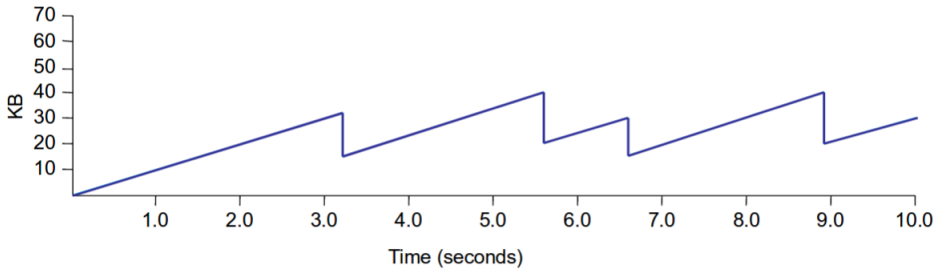


Figure 5.4: Saw tooth shape behavior. Timeouts cause **multiplicative decrease:** cwnd/2

it full at any time (otherwise we would be wasting bandwidth). By multiplying the bandwidth by the delay ($delay = \frac{RTT}{2}$) we get how much data we can have in a connection without clogging it. Incidentally, this is also the max size of the congestion window. When using TCP, to get the size of the congestion window we calculate $Bandwidth \cdot RTT$, because we also have to consider ACKs, which are very small and can be ignored.

5.4.1 Additive Increase/Multiplicative Decrease

The purpose is to adjust changes in the available capacity. We actually have an additive increase because we increment the *cw* by one packet per cwnd sent completely, which is +1 every time an ACK is received and a Multiplicative Decrease because, every time a drop is detected via triple-dupacks we set the $cw = \frac{cw}{2}$. If we plot the cwnd size (or the KB sent) over time we get a typical saw tooth behavior (Figure 5.4). We can see how the cwnd grows linearly in time and that TCP sends a cwnd's worth of bytes each RTT.

In practice we have:

- $increment = \frac{1}{cwnd}$
- $cwnd += increment$
- if a timeout or a three dupAcks happens the *cwnd* is a value based on the TCP version

5.4.1.1 Slow Start Phase SS

Purpose: quickly determine the available capacity.
This phase happens before addictive increase. It starts when first starting connection or when connection goes dead waiting for timeout.

- begin with $cwnd = 1$ packet
- $increment = 1$ (increment by 1 packet for each ACK)

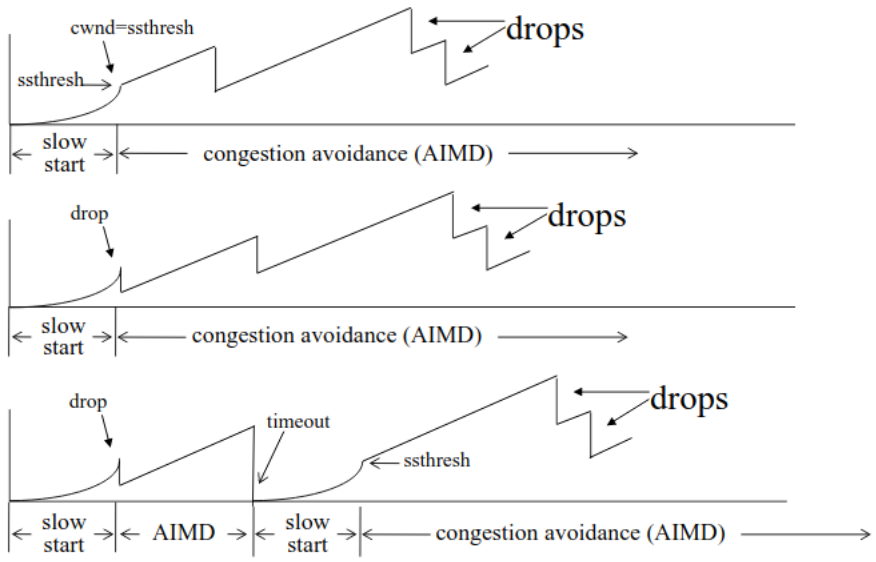


Figure 5.5: TCP behavior using AIMD and Slow Start

- $cwnd += increment$

This exponential increase is to check for available bandwidth (up to half of cwnd may get lost). "Slow" in this context means that the initial start, without this method, would be slow. **SSTHRESH** (slow start threshold) indicates when to begin additive increase phase and it's set to one half of *cwnd* when a packet loss happens. So, SSTHRESH goes through multiplicative decrease for each packet loss (because also the *cwnd* does so). If we have $cwnd \geq SSTHRESH$ we use congestion avoidance.

SSTHRESH is very large on connection setup. Since detecting losses with timeout is considered to be an indication of severe congestion, when a timeout occurs:

- $SSTHRESH = \frac{cwnd}{2}$ (multiplicative decrease)
- $cwnd = 1$
- $RTO = RTO \cdot 2$
- Enter slow start

Note that *cwnd* and SSTHRESH always ≥ 1 MSS[2].

Summary Summarizing, the complete congestion control functionality works in 2 phases:

- **Slow Start Phase**: until $cwnd \leq SSTHRESH$. Exponential growth, at each returning ACK a new packet is transmitted ($cwnd = cwnd + 1$) having, when every packet has received its ACK, a $cwnd = 2 \cdot cwnd$;

[2] MSS stands for *Maximum Segment Size*, and it's a parameter of the options field of the TCP header that specifies the largest amount of data, specified in bytes, that a computer or communications device can receive in a single TCP segment.

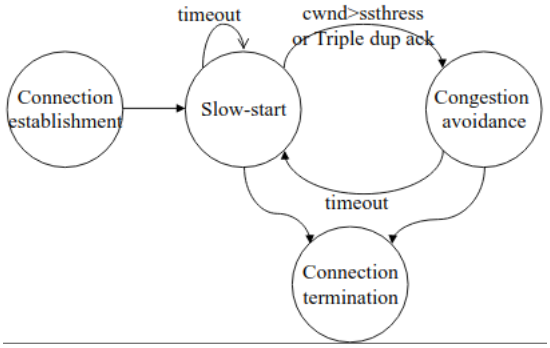


Figure 5.6: TCP Congestion Control Summary

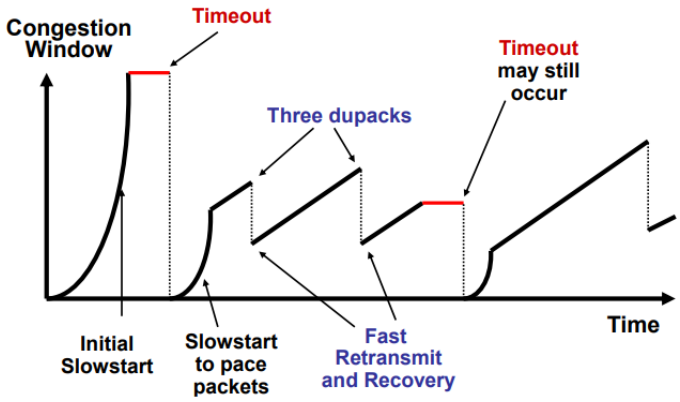


Figure 5.7: TCP Saw Tooth Explained

- **Congestion Avoidance**: when $cwnd > SSTHRESH$. We switch to linear growth, at each returning ACK a new packet is transmitted ($cwnd = \frac{cwnd+1}{cwnd}$) and at every RTT $cwnd = cwnd + 1$.

We have two ways to detect losses:

- **Timeouts**: we set $SSTHRESH = \frac{cwnd}{2}$ and $cwnd = 1$ (or, restart in Slow Start Phase);

- **Three Dupacks**: $SSTHRESH = \frac{cwnd}{2}$, $cwnd = \frac{cwnd}{2}$ (and restart from congestion avoidance phase).

Finally, see Figure 5.6 and 5.7 for details.

5.5 TCP versions

5.5.1 TCP Tahoe

One of the first protocols, very basic. It uses the Slow Start Phase and AIMD, but it only relies on timeouts to detect losses, so no fast retransmit nor fast recovery (it doesn't check for dupacks). At every loss, it always restart from 1.

Characteristics

- Standard TCP functions
- "Slow" start
- Congestion control: AIMD, only timeouts to detect losses (no dupacks)

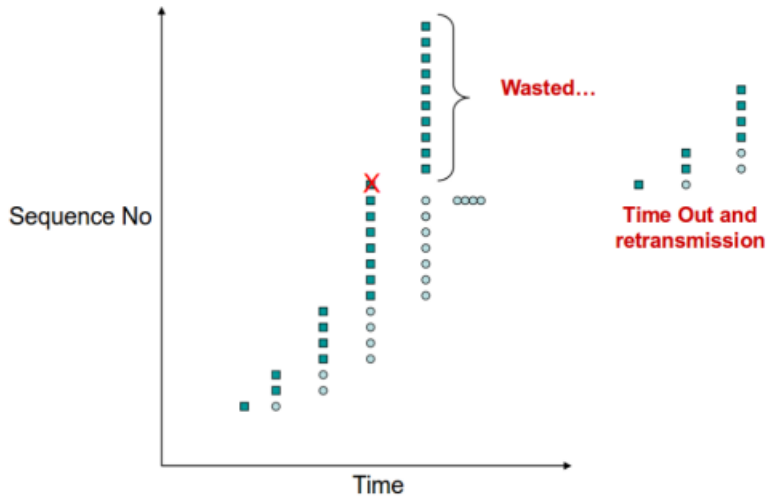


Figure 5.8: TCP Tahoe uses only timeouts to detect losses

5.5.2 TCP Reno

Characteristics

- Fast Retransmit/Fast Recovery (3 dupAcks to recover one packet loss)
- TCP Reno can recover from one packet loss without having a time out

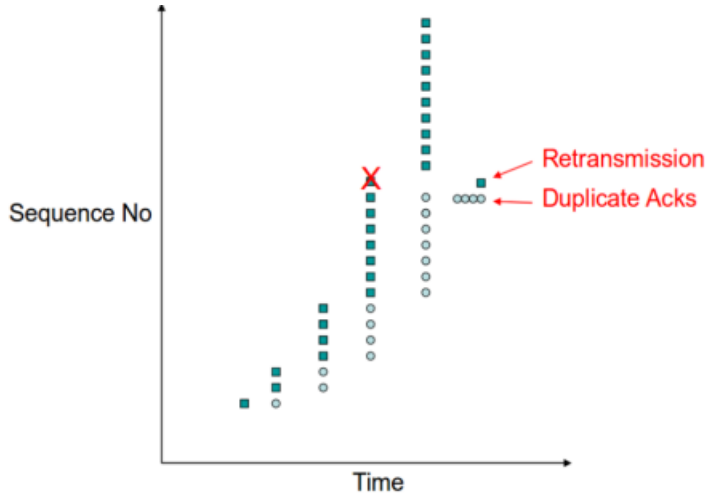


Figure 5.9: TCP Reno - Fast Retransmission

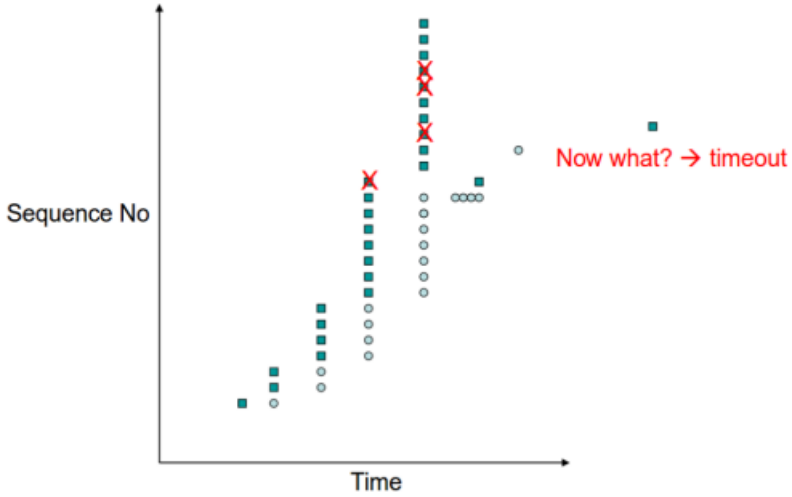


Figure 5.10: TCP Reno - more than one packet loss

5.5.2.1 TCP New Reno

TCP New Reno introduces partial ACKs to recover more packets without the use of timeouts.

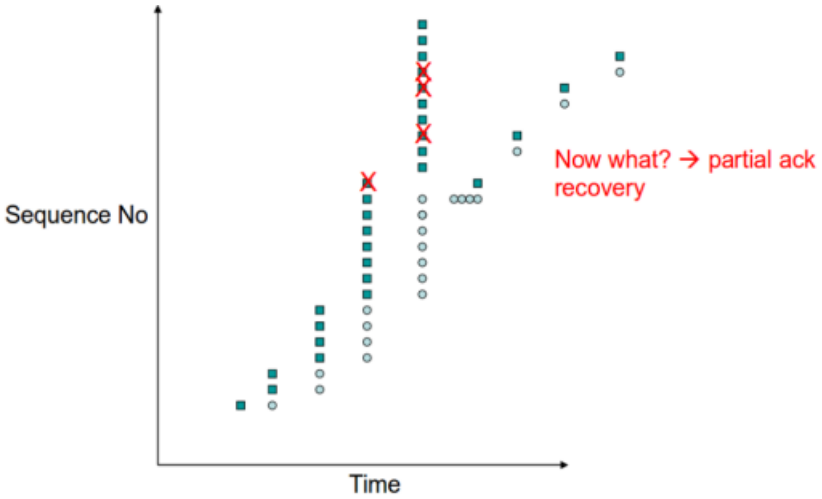


Figure 5.11: TCP New Reno - partial ACKs

5.5.3 TCP SACK

SACK means *Selective acknowledgment*. This means that all ACKs are a little bigger but carry more information, they can give the sender the complete picture.

Returning ACKs declares which packets (even non contiguous) were received. All non-received packets can be retransmitted and so it recovers from multiple losses in just one RTT.

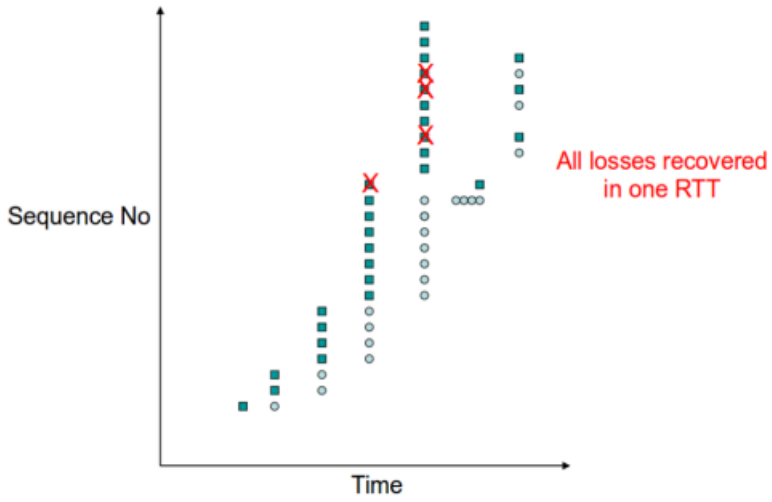


Figure 5.12: TCP SACK - Recovery from multiple losses in just one RTT

5.5.3.1 Differences between TCP New Reno and TCP SACK

TCP New Reno works by assuming that the packet that immediately follows the partial ACK received at fast recovery is lost, and retransmit the packet. However, this might not be true and it affects the performance of TCP. SACK TCP adds a number of SACK blocks in TCP packet, where each SACK block acknowledges a non-contiguous set of data has been received. The main difference between SACK TCP and Reno TCP implementations is in the behavior when multiple packets are dropped from one window of data. SACK sender maintains the information which packets is missed at receiver and only retransmits these packets. When all the outstanding packets at the start of fast recovery are acknowledged, SACK exits fast recovery and enters congestion avoidance.[3]

5.5.4 TCP Vegas

5.5.4.1 Congestion avoidance

Reaction to congestion episode and not to packets loss. Modifications:

Reaction to congestion episode

- Modified Congestion Avoidance
- Aggressive Retransmission
- Aggressive Window Adaptation
- Modified SS phase

Congestion Avoidance It's the reaction per congestion episode not per loss (congestion control). It uses change in observed end to end delay to detect onset of congestion, comparing the expected throughput...

$$Expected\ throughput = \frac{window\ size}{RTT} \tag{5.1}$$

[3] http://www.roman10.net/2011/11/10/tcp-tahoe-reno-newreno-and-sacka-brief-comparison/

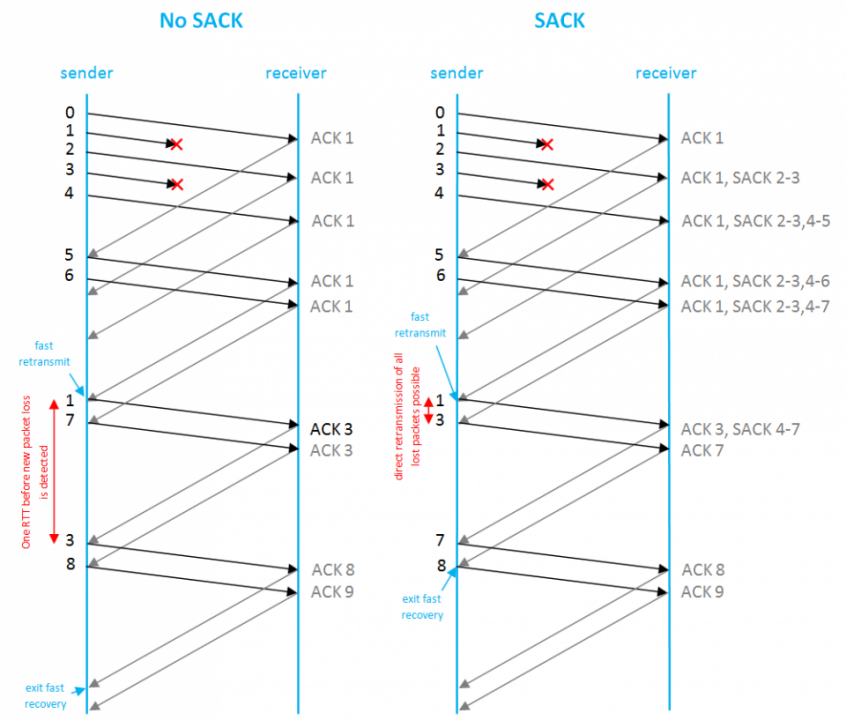


Figure 5.13: A comparison between TCP SACK versus one without SACK

. . . and with the actual one:

$$Actual\ throughput : \frac{actual\ transmitted\ amount}{RTT} \tag{5.2}$$

If there is no congestion, the Actual Throughput will be very close to the (optimal) Expected Throughput. The difference between Actual Throughput and Expected Throughput will be greater when there is congestion.[4] In this protocol it's important to note that actual throughput can only be equal or less than expected. The monitor transmission rate is:

monitor transmission rate

- If $expected - \alpha < actual < expected$: Queues decreasing $\rightarrow$ increase transmission rate. $cwnd = \frac{cwnd+1}{cwnd}$
- If $expected - \alpha < actual < expected - \beta$: Don't do anything
- If $actual < expected - \beta$: Queues increasing $\rightarrow$ decrease transmission rate before packet drop (maybe there is a congestion in network or maybe the connection is shared among other services). $cwnd = \frac{cwnd-1}{cwnd}$

Thresholds of $\alpha = \frac{1pkts}{RTT}$ and $\beta = \frac{3pkts}{RTT}$ correspond to how many packets Vegas is willing to have in queues.

throughput *goodput*

There are some definitions to keep in mind: the throughput that is the speed of transmission, and goodput that is the speed of reception

[4] http://www.mathcs.emory.edu/~cheung/Courses/558/Syllabus/02-transport/vegas.html

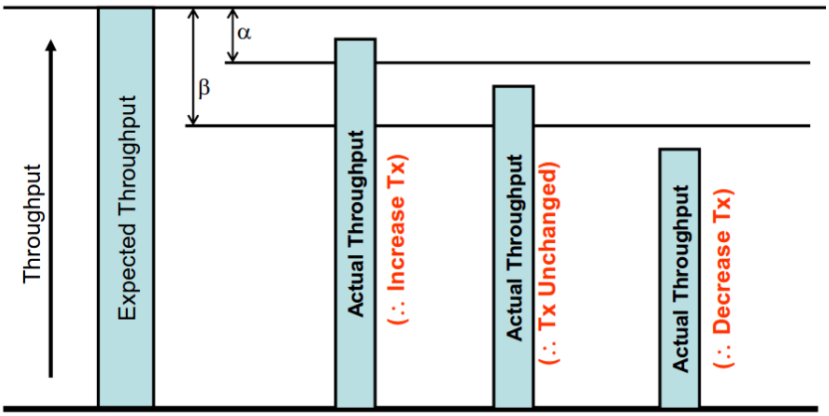


Figure 5.14: Vegas - Modified Congestion Avoidance

Aggressive Retransmission With dupacks, when Vegas receives the first dupack or the second dupacks, it checks the fine grained timer expiry. This helps in wireless environments with non congestion loss. The retransmission activation happens when

$$(currentTime - sendingTime) + RTT_{min} > TimeoutValue$$

Aggressive Window Adaptation With recovery, the cwnd is divided by four instead of two when it enters into recovery. With multiple loss, in case of multiple segment loss from a single window, it reduces the cwnd only once. The cwnd is initially set to 2 instead of 1.

Modified SS phase The formulas for throughput are the same of 5.1 and 5.2, but they're calculated in every alternate RTT instead of once every RTT. There is a new parameter, γ, that's calculated as $\gamma = \frac{1pkts}{RTT}$, having that if:

- $Expected - Actual < \gamma \rightarrow$ continue using slow start
- $Expected - Actual > \gamma \rightarrow$ switch to congestion avoidance

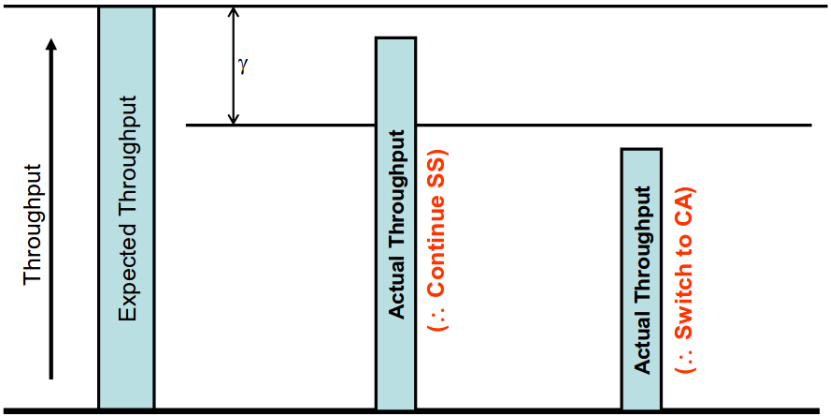


Figure 5.15: Vegas - Modifies Slow Start

5.5.4.2 Final considerations

Unfortunately, TCP Vegas is not actually used: it's sensitive to delay variation and cannot coexist with legacy TCP versions. At the end of the day, thanks to its ideas, parts of this protocol have been implemented in other TCP versions.

5.6 TCP and Wireless

Wireless is less reliable than cable, and losses are common. In TCP though, every loss is treated as a congestion. This causes:

- Multiple cwnd[5] reductions
- Disconnections due to timeouts
- Bandwidth waste

This has also an impact on multi-hop paths: it takes twice the time to transmit data with two hops, because is not possible to forward the message at the same time, but you have to wait, otherwise you'll end up with a collision. Increasing number of hops beyond three allows simultaneous transmission on more than one link, however, degradation continues due to contentions between TCP data and ACKs traveling in opposite directions.

With movement, throughput degrades, because there is a possibility of link breakage and route failure.

To solve these issues, we could adopt a different way of seeing the network: *traditional TCP*, *connection split* and *pure end-to-end* connection.

5.6.1 Traditional TCP

The protocol is not aware of wireless link, and possible timeouts are treated like congestions. Implementations:

- TCP Reno
- TCP New Reno
- TCP Vegas
- TCP SACK

5.6.2 Connection split

The network is split, having local retransmission and quick actions on the wireless link. Examples:

- I-TCP
- M-TCP

[5] Congestion window

- Proxy
- Snoop Protocol

5.6.2.1 Snoop Protocol

In this protocol, there is a tower with a big buffer keeping a list of ACKs: it handles dupAcks/lost packages, without stopping the sender from sending new data to the tower. Considerations:

- Pro:
  - Local (and timely) loss recovery
  - End-to-end semantics preservation (almost)
- Cons:
  - Requires a little RTTs on the wireless link
  - No guarantee against long disconnections

5.6.3 Pure end-to-end

In this protocol, the sender is aware of the wireless link. There are endless number of implementations of end-to-end protocols:

- TCP WestWood/WestWood+
- TCP Cubic
- TCP Hybla
- TCP High speed
- ...

5.6.3.1 TCP WestWood

This TCP version involves changes only to the sender side of the connection, and not to everyone. It's useful for satellite communications.

Flow control is based on an estimation of the eligible bandwidth (BWE). TCP WestWood can perform an estimation determined from ACK internal-arrival times and info in ACKs regarding amounts of bytes delivered. The rate estimation enhance congestion control, and is used by the sender to properly set *cwnd* and $SSTHRESH$ after a packet loss (either a three dupAcks or a timeout). Rate estimation is based on three formulas.

rate estimation

$$RE_k = a_k \cdot RE_{k-1} + (1 - a_k) \cdot \left(\frac{b_k + b_{k-1}}{2} \right) \tag{5.3}$$

This formula calculate the filter. In particular, this formula is quick to compute, and allows to set proportions like: 75% RE_{k-1} (the portion of the previous volume)

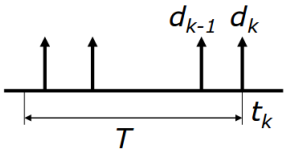


Figure 5.16: Representative schema of how b_k is calculated

and 25% of samples. Obviously, it takes some time to have a good estimation of the bandwidth. To calculate the samples, the formula is:

$$b_k = \frac{\sum_{t_j > t_k - T} d_j}{T} = \frac{ACK\ seen}{Time\ Passed} \tag{5.4}$$

Note that this formula is based on T (indicating typically the RTT): on one hand having a small T it means being more aggressive at the rate estimation, on the other hand having a big T it means being more conservative.

At the end, the filter is calculated another time:

$$a_k = \frac{2t - \Delta t_k}{2t + \Delta t_k} \tag{5.5}$$

it's important to consider that his formula is difficult to compute and slows down the whole formula, 5.3.

With the way 5.3 is calculated, the value is fine-tuned over time.

Handling losses in TCP WestWood When three dupAcks occurs we have that $SSTHRESH = BWE \cdot RTT_{min}$ and if the cwnd is greater than the SSTHRESH we set $cwnd = SSTHRESH$. Otherwise if a timeout occurs we have $SSTHRESH = BWE \cdot RTT_{min}$ but the cwnd becomes equals to 1.

Final considerations

- Pro:
  - bandwidth estimation done at the sender side
  - code modifications only at the sender side
- Cons:
  - wrong bandwidth estimation over asymmetric links
  - no specific mechanism to handle disconnections
  - problems with fairness & friendliness[6]

5.6.3.2 TCP Cubic

In this TCP version the congestion window (cwnd) is a cubic function of time since the last congestion event. The inflection point (of a cubic function) is set to the window prior to the last congestion event.

[6]**Fairness**: One flow with the **same** protocol → same bandwidth usage.
Friendliness: One flow with **different** protocols → same bandwidth usage.

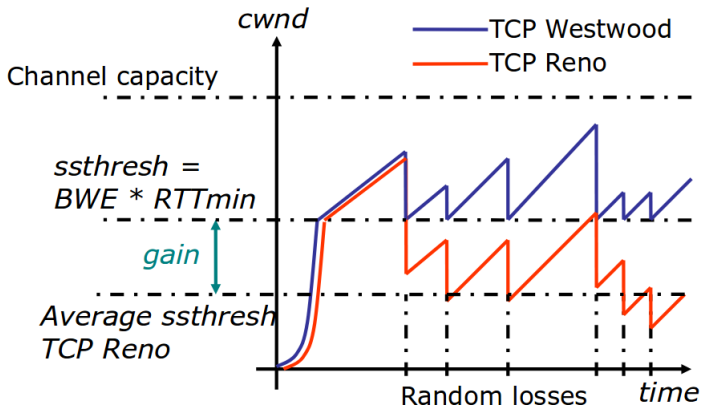


Figure 5.17: A comparison between TCP WestWood and TCP Reno

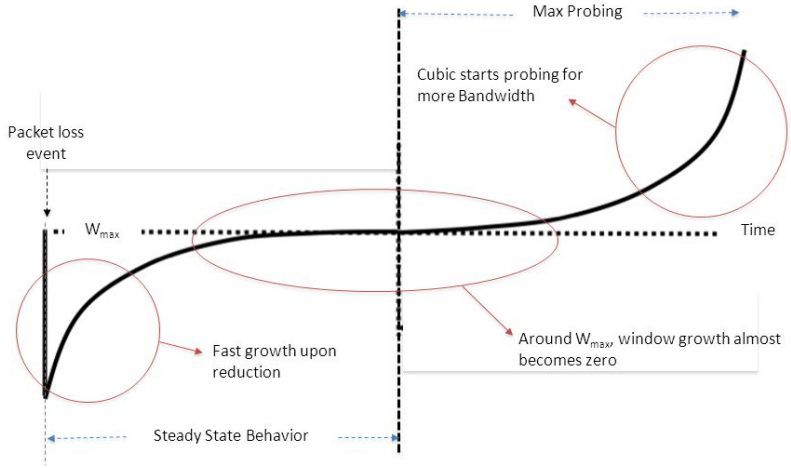


Figure 5.18: Image explaining how TCP Cubic works.

TCP Cubic does not rely on the receipt of ACKs to increase the cwnd, it depends only on the last congestion event. This is good, but leads to a less RTT-unfairness.

This version of TCP is currently used in Linux OSes, thus there is the possibility to change with another one.

5.7 Challenged Networks

Challenged networks are systems that deals with links that have characteristics very different from the standard ones:

- satellites links → they are sensitive to meteorological conditions and they have a long RTT
- 3G, 4G cellular systems
- WiFi (802.11), WiMax (802.16)
- fiber Optic Links
- car-to-car

- interplanetary net

Due to packet losses, dupAcks, timeouts etc. . . the link is never used to his full potential.

Disruptive/intermitted links: with this kind of links there is no fixed infrastructure, no antenna and no cable. In these cases it is not possible to use the usual protocols. This leads to the following opportunities:

- ubiquitous computing
- back-up networks (after-disaster recovery, fast network deployment)
- always-on connection
- very fast connection
- traffic control
- space communication

. . . but with many issues:

- great increase of RTT
- there is a PER (Packet Error Rate) not reducible (average: 0-10%)
- redundancy is needed
- the acceleration speed is slow

There is an interest in exploiting disruptive links (no end-to-end connectivity), as for example opportunistic communication.

TCP is not able to distinguish between losses due to errors in links or due to congestion and it has several timers that would get crazy in case of intermitted/disruptive links. Different algorithms are used to address different problems:

- large RTT
- high PER
- high speed
- disruptive links
- . . .

We'll analyze for every kind of link possible solutions.

5.7.1 Satellite communications

In today's communications there are two types of satellites used:

- GEO - orbiting at 3600km from earth
- LEO - orbiting at 100-150km from earth

One of the problems of satellite communications is the great RTT[7]. On top of it, there is a non-negligible Packet Error Rate (influenced by the weather conditions and constellations positioning), that force the adoption of some CRC[8] system. To solve these problems, a new protocol needs to be used, and it's TCP Hybla, covered in 5.7.3.1.

5.7.2 Terrestrial communications

Terrestrial communications are:

- 3-4G
- WiFi (802.11)
- WiMAX (802.16)

Characteristics There are some characteristics in terrestrial communications: first of all, there is a variable RTT, and, as for satellite communications there is a non-negligible packet error rate. Finally, there are performances issues for cabled links (e.g. fiber communications), because internet protocols aren't able to exploit high-speed networks.

5.7.3 TCP solutions

As we can see, traditional TCP protocols don't work very well with these links. High-speed networks requires very high volumes of W[9]: $B(t) = \frac{W(t)}{RTT}$, and too little retransmission timers lead to frequent retransmissions. Thus, new TCP protocols were made.

5.7.3.1 TCP Hybla

This protocol equalizes the transmission rate against the RTT, and it's good for satellite links. Since the TCP window-based transmission algorithm depends on network delays, TCP Hybla proposed to obtain for long RTT connections the same instantaneous transmission rate. This can be achieved with two steps:

- by making *cwnd* independent of RTT;
- by compensation the effect of the division by RTT.

[7] RTT means *Round Trip Time*
[8] CRC: cyclic redundancy check
[9] congestion window

In order to do that Hybla sets a new parameter, ρ, setting it to $\rho = \frac{RTT}{RTT_0}$, where RTT_0 is a reference RTT (usually equals to 25ms). Differently to the "classic" TCP formula for $W(t)$ that is:

$$W(t) = \begin{cases} \frac{t}{2 \cdot RTT}, & \text{with } 0 \leq t < t_y \text{ for slow start phase (SS)} \\ \frac{t - t_y}{RTT} + y, & \text{with } t \geq t_y \text{ for collision avoidance phase (CA)} \end{cases}$$

the Hybla formula for $W(t)$ is:

$$W(t) = \begin{cases} \rho 2^{\rho \frac{t}{RTT}} & \text{with } 0 \leq t < t_{y,0} \text{ for slow start phase (SS)} \\ \rho \cdot \left[\rho \frac{t - t_{y,0}}{RTT} + \gamma \right] & \text{with } t \geq t_{y,0} \text{ for collision avoidance phase (CA)} \end{cases} \tag{5.6}$$

Hybla problems Hybla has some flaws: it's a bit aggressive as protocol, but it's ok cause it'd be too slow otherwise. Keep in mind that being aggressive means that you risk to lose many packets all together. In this case, aggressiveness could cause friendliness problems too, such as disruption of other protocols. When the RTT is big it'll cause even more aggressiveness, and the solution to this is to provide bigger buffers, that's bad for real time communications.

5.7.3.2 TCP High-speed

The aim of this protocol is to improve the exploitation of high-speed network bandwidth (e.g. fiber links). Similar to Hybla, when the cwnd is high, it's more aggressive.

5.7.3.3 Disruption Tolerant Networks - DTN

DTN are designed to work whenever end-to-end connectivity may be not always available. DTN adds a new layer (bundle layer) under the application one, that sends the messages to the network as soon as it's available, creating a TCP connection.

- Pro:
  - no end-to-end TCP semantics violation
- Cons:
  - the new bundle creates a little overhead

5.7.3.4 Performance Enhancing Proxies

This proxy isolates the challenge link with the rest of the network, through an intermediate agent before it.

- Pro:
  - performance boost
  - can support every TCP enhancement

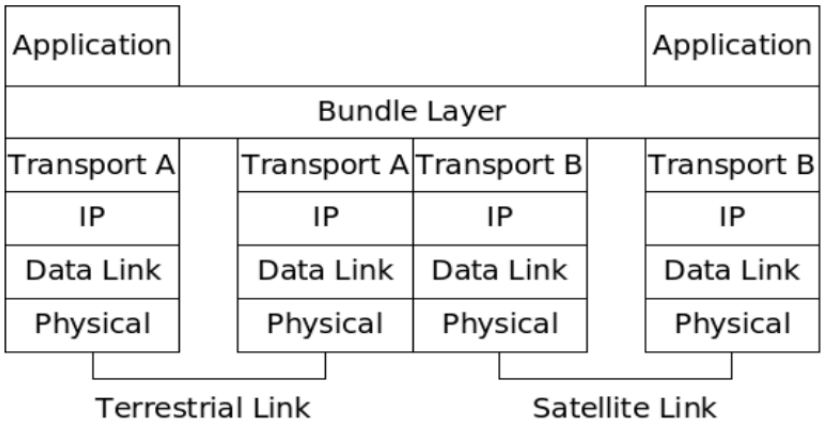


Figure 5.19: DTN Stack

- Cons:
  - violation of the TCP end-to-end semantics
  - no IPSec support

5.7.3.5 TCP adaptive - selection

Proposed from the DTN group, the right TCP protocol is selected based on the connection. This selection is performed using the following characteristics:

- from the agent that perform TCP selection
- exploiting cross-layer possibilities
- changing TCP protocol on-the-fly, even if it's not easy

5.7.4 Summary

None of this solutions solve all the problems. At the end, the server always needs to change the TCP implementation.

CHAPTER 6

The MAC layer

6.1 Introduction

The MAC layer handles access to the same medium, avoiding collisions that cause waste of time. It has the following goals:

- efficiency in bandwidth use
- avoid collisions (resilience)
- fairness
- robustness: the protocol should be decentralized
- protocols easy to implement

Typologies The MAC layer has different typologies. As a matter of fact, it offers different ways to share the channel: **TDMA** (*time division*), **FDMA** (*frequency division*), **CDMA** (*code division*).

Random access MAC provides two way to randomly access the channel, with **CSMA** (*carrier sense multiple access*) and **MACA** (*multiple access with collision avoidance*).

Other protocols There are also "taking turns" MAC protocols, that basically carries out a polling scheme.

6.1.1 Wired vs Shared Channel

When the channel is a cable it's easier, because you just have to listen to it before transmitting any data (this technique is called CSMA, *Carrier sense multiple access*[1]): for example with Ethernet you can check what you've sent and check if a collision happened. In a wireless environment this is not possible, because when

[1] As an additional note, with CSMA you can even listen when a collision happens.

you're transmitting you can't receive, otherwise you'll end up having a collision, but you can check if a package has arrived to the destination with the ACKs[2].

6.2 MAC Layer approaches

There are different ways the MAC layer can approach the access to the medium, and these are:

- Random
  - without carrier sensing (such as Aloha, Slotted Aloha)
  - with carrier sensing (CSMA, CSMA/CD, MACAW)
- Controlled
  - centralized (FDMA, TDMA, CDMA)
  - distributed

6.2.1 Aloha

This protocols was used to communicate between Hawaii islands. It doesn't have synchronization, and the total throughput of this protocols is $\frac{1}{2e}$

6.2.1.1 Slotted Aloha

This version of Aloha has time divided slots: every transmission has to stay in a defined time slot, that is not easy to do but it's possible. In case of a collision, a re-transmission occurs with a probability P.

Throughput The throughput of this Aloha version with synchronization is $\frac{1}{e}$

6.2.1.2 Considerations about Aloha protocols

In general, Aloha protocols have these common characteristics:

- not efficient
- unfair
- robust & easy to implement

[2] In wireless communications TCP packets are split into smaller packages of 100 bytes each one when they arrive to the MAC layer: the ACKs are generated during transmission for the MAC layer and when a complete TCP packet is sent the receiver will generate a TCP response. Any TCP packet can generate one o more MAC packet, allowing the MAC layer to hide some losses to the top protocol. N.B.: this is not true for ad-hoc communications though.

6.2.2 CSMA

With this approach, you first listen to the channel and then the protocol algorithm tries to avoid transmission at the same time. CSMA comes with different versions, based on the probability p of transmission after the channel becomes free.

1-persistent CSMA When the channel is free → transmit, otherwise, wait until the channel becomes free. At this point, immediately transmit. If a collision occurs, wait a random amount of time before re-transmission.

It's easy to see that if two nodes are waiting the channel to become free they will probably cause a collision because they will start transmitting at the same time.

non-persistent CSMA Similar to 1-persistent CSMA, but if the channel is busy then wait a random amount of time before checking it again.

p-persistent CSMA This version is slot based. If the channel is free then the node transmits with a probability p, otherwise it waits a random amount of time.

6.2.2.1 CSMA/CD

The CD version has a collision detection system, that allows to abort the transmission when a collision is detected, reducing time wastage. This, unfortunately, is not possible in wireless communications, but only in the wired one.

SINR SINR (*signal-to-interference-plus-noise ratio*) is commonly used in wireless communication as a way to measure the quality of wireless connections. Typically, the energy of a signal fades with distance, which is referred to as a path loss in wireless networks. Conversely, in wired networks the existence of a wired path between the sender or transmitter and the receiver determines the correct reception of data. In a wireless network ones has to take other factors into account (e.g. the background noise, interfering strength of other simultaneous transmission). The concept of SINR attempts to create a representation of this aspect[3].

The collision detection is based on the SINR and the formula to calculate it is:

$$SINR = \frac{Signal\ of\ Interest\ (SoI)}{Interference(I) + Noise(N)} \tag{6.1}$$

The formula to calculate the SINR between two nodes (given interference between C and B) is:

$$SINR_B^A = \frac{\frac{P^A_{transmit}}{d^\alpha_{AB}}}{N + \frac{P^C_{transmit}}{d^\alpha_{CB}}} \tag{6.2}$$

In this formula $P^X_{transmit}$ represents the power transmission of X, and d^α_{XY} is the SoI between X and Y. The noise (N) is somehow a fixed value.

The nodes position and the signal interference cause communication problems in wireless networks, e.g. the **hidden terminal problem** and the **exposed terminal problem**, that will be explained in the next part.

[3] https://en.wikipedia.org/wiki/Signal-to-interference-plus-noise_ratio

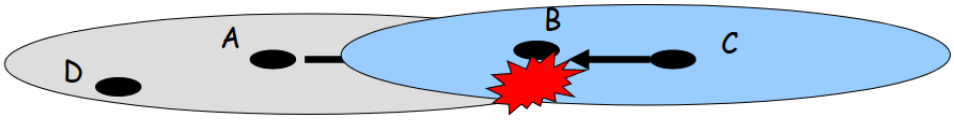


Figure 6.1: An example of a collision detection

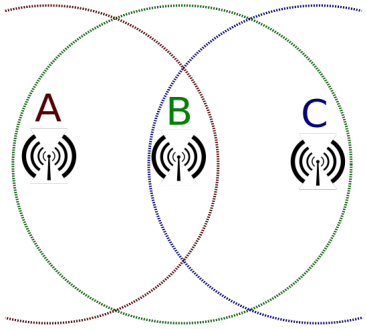


Figure 6.2: Hidden terminal problem. Station A can communicate with Station B. Station C can also communicate with Station B. However, Stations A and C cannot communicate with each other since they cannot sense each other on the network, because they are out of range of each other.

6.3 Communication problems with wireless networks

As already mentioned, there are problems with wireless networks related to the nodes' position and the strength of the signal that will be explained here.

6.3.1 Hidden terminal problem

The hidden node problem occurs when a node is visible from a wireless access point (AP) but not from other nodes communicating with that AP, due to difficulties in the media access control sub-layer.

6.3.2 Exposed terminal problem

Exposed node problem occurs when a node is prevented from sending packets to other nodes because of a neighboring transmitter.

6.3.3 MACAW

This part is very confused. Please check it out very closely and critically because I'm not sure about it.

Given that wireless MAC proved to be non-trivial, in 1994 a research lead by Bhargavan made MACAW (*Multiple Access with Collision Avoidance for Wireless*) that now is used for congestion avoidance. It has the same operating principle of 802.11, and it's widely used in ad-hoc networks. MACAW helps solving the hidden terminal problem, but not the exposed terminal problem.

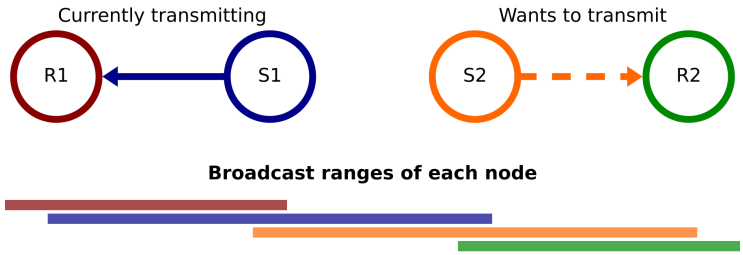


Figure 6.3: Exposed terminal problem. Station S2 wants to communicate with R2, but has sensed S1 that's currently transmitting to R1 so it doesn't transmit for not causing a collision. In this case, though, S2 should not defer transmission to R2 because this won't create any interference with S1's transmission.

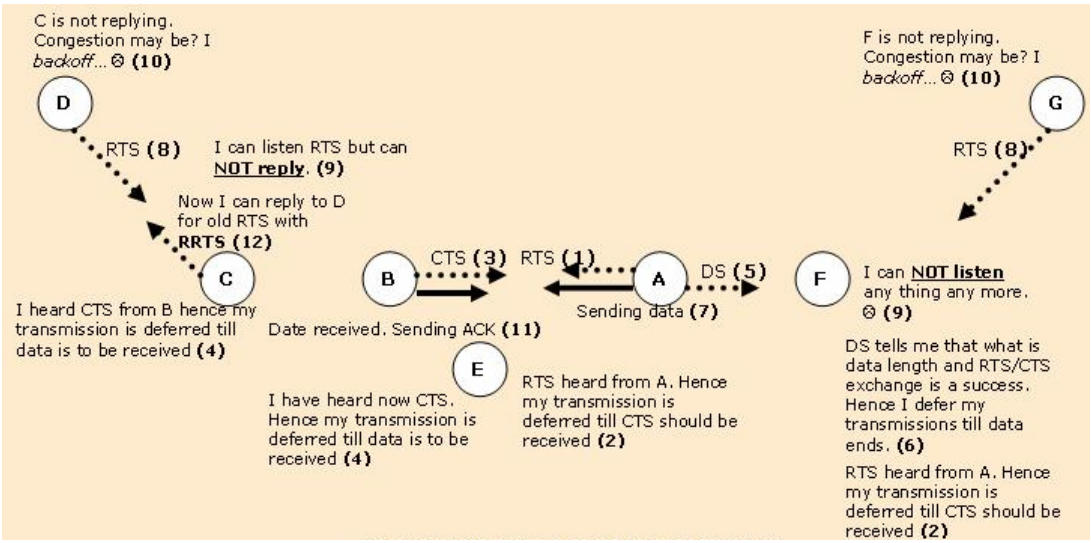


Figure 6.4: A schema describing an example of how MACAW works

Channel access method MACAW uses CTS and RTS packets to knowing if the channel is free to be used to transmit new data. These two types of packets are very small, so it's difficult to have a collision transmitting them.

RTS *Request to Send* is a frame sent by the sender to the receiver before starting a transmission that asks if there are no current transmission that could interfere with the new one

CTS *Clear To Send* is a frame sent by the receiver that informs the sender that there is a possibility to use the channel to start a new transmission.

How it works From Figure 6.4, assume that node A has data to transfer to node B. Node A initiates the process by sending a Request to Send frame (RTS) to node B. The destination node (node B) replies with a Clear To Send frame (CTS). After receiving CTS, node A sends data. After successful reception, node B replies with an acknowledgment frame (ACK). If node A has to send more than one data fragment, it has to wait a random time after each successful data transfer and

compete with adjacent nodes for the medium using the RTS/CTS mechanism.

Any node overhearing an RTS frame (for example node F or node E in the illustration) refrains from sending anything until a CTS is received, or after waiting a certain time. If the captured RTS is not followed by a CTS, the maximum waiting time is the RTS propagation time and the destination node turnaround time.

Any node (node C and node E) overhearing a CTS frame refrains from sending anything for the time until the data frame and ACK should have been received (solving the hidden terminal problem), plus a random time. Both the RTS and CTS frames contain information about the length of the DATA frame. Hence a node uses that information to estimate the time for the data transmission completion.

Before sending a long DATA frame, node A sends a short Data-Sending frame (DS), which provides information about the length of the DATA frame. Every station that overhears this frame knows that the RTS/CTS exchange was successful. An overhearing station (node F), which might have received RTS and DS but not CTS, defers its transmissions until after the ACK frame should have been received plus a random time.

To sum up, a successful data transfer (A to B) consists of the following sequence of frames:

1. "Request To Send" frame (RTS) from A to B
2. "Clear To Send" frame (CTS) from B to A
3. "Clear To Send" frame (CTS) from B to A
4. "Data Sending" frame (DS) from A to B
5. DATA fragment frame from A to B, and
6. Acknowledgment frame (ACK) from B to A.[4]

MILD[5] Algorithm MACAW uses contention windows (cw) to determine the packet transmission rate. When a node fails to receive CTS in response to its RTS it multiplies his contention window by 1.5. This approach is less aggressive than 802.11, which multiplies by 2. When a node successfully completes a transfer, it reduces the contention window by 1[6]. This avoids wild oscillation of the contention window when congestion is high. The congestion window helps with the fairness: when a node transmits the packet it appends its current congestion windows, that is used by the others for future transmission output.

Fairness There are two main ways to establish fairness in MACAW, via *Weighted fair queuing* and via *Distributed fair scheduling (DFS)*.

Weighted fair queuing Each node has a weight in order to have a bandwidth usage proportional to it

[4] Taken from https://en.wikipedia.org/wiki/Multiple_Access_with_Collision_Avoidance_for_Wireless

[5] That means *Multiple Increase Linear Decrease*

[6] That reduction could be a little too conservative.

DFS The *Distributed fair scheduling*:

- chooses back-off intervals proportional to packet size/weight
- works well on LAN
- always include some randomness

Scanning Scanning it obviously join, find and initialize a network. It can be of 2 typologies:

- active: the client actively search for beacons
- passive: the AP sends beacons continuously

6.4 MAC for 802.11

802.11 The 802.11 protocol is used mostly indoor, it uses three bands (900 MHz, 2.4GHz, 5GHz) and its applications are related to nomadic internet access, portable computing and ad-hoc networking. 802.11 has two types of configurations:

- with a base station (control station, access point)
- ad-hoc networking (point-to-point)

Steps for establishing a connection The steps for establishing a connection through the MAC protocol in 802.11 are:

1 DIFS (*Distributed Inter Frame Space*): time where the channel has to be free. If a node gains access to the channel goes to step 2a, otherwise to step 2b.

2a Data transmission

2b NAV (*Network Allocation Vector*): time where the others have to wait before starting another transmission

3 SIFS (*Short Inter Frame Space*): time where the channel has to be free

4 MAC ACK

6.4.1 Channel connection & priorities

Channel contentions can be of the following typologies:

- SIFS
- PIFS (only with polling schema)
- DIFS

The contention time starts before the next frame and after SIFS/PIFS/DIFS occurs. The packages transmission order is the following:

1. DIFS
2. RTS
3. SIFS
4. CTS
5. SIFS
6. Data
7. SIFS
8. ACK

6.4.2 Access methods

802.11 has different access methods:

DCF
- MAC-DCF [7] CSMA/CA (mandatory)
- MAC-DCF with RTS/CTS (optional)

PCF
- MAC-PCF [8] (polling the AP) (optional)

CSMA CSMA doesn't entirely solve the hidden/exposed terminal problem. The CSMA doesn't work well with the hidden terminal, so a CA (collision avoidance) solution is used. This adds two packets before data transmission, RTS and CTS. These packets are very little, thus fast to transmit.

Congestion avoidance The congestion avoidance works in this way: there is a countdown determined by a random back off (that's the contention window), that tells the node to wait to send. This countdown is suspended if the medium is busy (pay attention that doesn't mean that the countdown is reset, because it's not!). The congestion avoidance phase starts after the DIFS phase, and it is larger[9] when there are many nodes, smaller if fewer. When a node fails to receive CTS in response to its RTS, it can act in different ways. In 802.11, it usually doubles it's contention windows, meanwhile in MACAW uses MILD.

Binary exponential back off This technique doubles the *cw*, up to a maximum of five times. When the first time the node complete a transmission the *cw* is reset to its original CW_{min}.

MILD See 6.3.3

PCF With this access method the AP announces it in the beacons. After associations frames the node announces to the AP whether it's pollable and capable to transmit during the contention free period (CFP).

[7] DCF means *distributed coordination function*
[8] PCF means *point coordination function*
[9] Large *cw* cause large overhead

Power Management Stations are allowed to go to sleep and this state is saved in the AP memory. Pay attentions to the fact the the energy a node consumes to wake up is more that the one it uses to stay asleep, if it wakes up too often. Sleep is intended for long times.

When a node sleeps, the packets for it are buffered in the AP. When it wakes up, it downloads all the packages not yet received. TSF (*time synchronization function)* assures AP and power save stations are synchronized.

CHAPTER 7

Routing

Routing determine the "good"[1] path (sequence of routers) thru network from source to destination. Routing protocols use graphs as a mathematical abstraction.

7.1 Typologies of routing

Routing can be **global**, **decentralized**, **static** (where the routes change slowly over time) and **dynamic** (where the routes change quickly over time).

7.1.1 Global routing

In this kind of routing routers have the complete topology of the network, with information about links cost. Characteristics:

- "link state" algorithm;
- efficient;
- not feasible to implement.

An example of a global routing algorithm is the one made by Dijkstra.

7.1.1.1 Dijkstra's algorithm

In this iterative algorithm all the nodes know link cost between each other. Basically, it computes the path with the lowest cost from one node to all the others.

[1] The minimum path typically

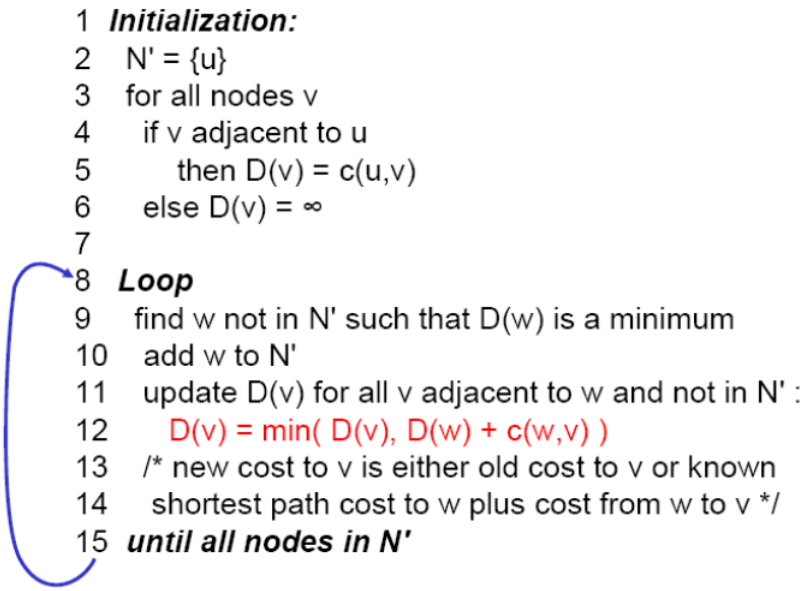


Figure 7.1: Dijkstra's algorithm. Notation:

- $c(x, y)$: link cost from node x to y. It's ∞ if x has not direct neighbors;
- $D(v)$: current value of cost of path from source to destination v;
- $p(v)$: predecessor node along path from source to v;
- N': set of nodes whose least cost path is definitively known.

7.1.2 Decentralized routing

In this type of routing routers know **physically-connected** neighbors, and link cost to them. There is an iterative process of computation and exchange of information with neighbors. These algorithms are defined as **"distance vector"** algorithms.

7.1.3 Routing based on network typology

There are different types of routing based on different types of networks.

Infrastructure based In this case routing is relatively simple. There is just a single hop from the access point (AP) to the wireless node.

MANET [2] It's also know as ad-hoc, and has the following characteristics:

- the **network constantly changes**,
- wireless nodes are **not** necessarily all adjacent, so data forwarding could be necessary.

Additionally, there are other difficulties, such as **energy consumption** and **limited bandwidth** (due to exchange of routing information) on mobile systems.

[2] MANET stands for *Metropolitan Area Network*

In wireline networks we have symmetric links, limited redundancy and fixed topology, instead wireless has asymmetric links and random redundancy with unplanned, dynamic links. With all these peculiarities, traditional routing algorithms for MANETS are inefficient, and the new one must rely on data link information, not just network layer updates[3], without using centralized approaches because they just don't work. In MANETS nodes can be routers, and they can send different type of messages between them.

Messages classification There are different type of messages:

- Unicast: transmission 1 to 1. In order to achieve a good unicast routing protocol there are some goals:
  - minimal control overhead
  - multi-hop path routing capability
  - no loops
- Multicast: transmission 1 to N: more than one receiver of the same message, that it can follow different paths.
- Broadcast: transmission 1 to N: everyone receives the message

7.2 Protocols classification

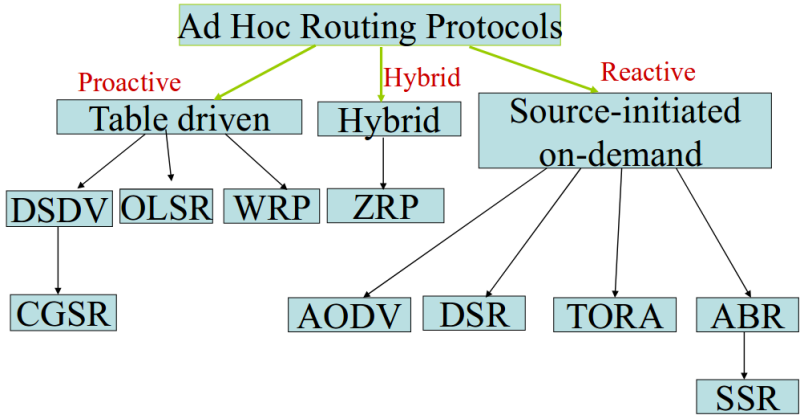


Figure 7.2: Tree listing all the ad-hoc routing protocols

As shown in Figure 7.2 protocols can be proactive or reactive, and are classified as:

- table-driven (proactive) → up-to-date routing information are maintained
- source-initiated (on demand/demand driven) → create routes only when needed and maintain them until they are no more useful to the source.
- hybrid protocols → combination of proactive and reactive protocols

[3]Layer updates only determine connectivity

7.2.1 Proactive protocols

Proactive protocols are based on traditional distance-vector and link-state protocols, where each node maintains a route to the other edge of the network. This cause a high overhead in most scenarios, but since each node maintains and updates its routing table, there is a low packet forwarding latency.

Due the proactive nature, changes to the network topology are immediately propagated to all the nodes.

There are different typologies, that will be described below.

7.2.1.1 DSDV (Destination Sequenced Distance Vector)

Based on Bellman-Ford algorithm it has characteristics similar to Dijkstra's algorithm, but with **slower performance and a better versatility** (it supports negative weights).

With DSDV, every route has a sequence number and packets are transmitted according to the routing table: each node has a routing table with all the nodes listed, and each node has its own sequence number. Each node periodically transmits updates to keep table consistency. When two routes to a destination are received from two different neighbours the one with the greatest destination sequence number is chosen, otherwise the smallest hop-count is kept. Continuously updating the routing table can generate a lot of overhead, and there are two ways to carry updates:

- performing a full dump
- having incremental updates

7.2.1.2 OLSR (optimized link state routing)

This is based on the link state algorithm. It minimizes flooding of broadcasting packets by using only the selected *Multipoint Hop (MH)* called *Multipoint relays (MR)*. All links with neighboring MHs are declared and are flooded in the entire network. MR minimize the flooding by reducing duplicate retransmission.

In general, OLSR is good for large and dense networks.

7.2.1.3 CGSR

In this typology, nodes are organized into a hierarchy of clusters: every cluster has a *cluster master (clusterhead)* that forward packets from the other sets node using DSDV.

7.2.2 Reactive protocols

The source builds routing on demand with a "flooding" method, but maintains only active routes. Advantages:

- less overhead
- better scaling properties

- route acquisition latency

Source-initiated on-demand routing protocols have the following characteristics:

- route created only when needed
- flooding is used in order to find routes
- router maintenance procedure is used to repair broken routes

7.2.2.1 AODV (Ad hoc On-Demand Distance Vector Routing)

This routing protocol, based on DSDV, exchange the routing table only along a given route. In order to discover routes, there is a specific package, RREQ (*Route Request Packet*). The packet is sent from the sender to the receiver, and when it's received, a RREP(*Route Replay Packet*) is sent back to the source, creating a path.

Routes are timed: after a while they expire: unicast and multicast routes are kept until a timeout happens.

Properties AODV's properties:

- discovers routes as and where necessary
- routes are maintained just as long as necessary
- sequence number increases every time the node notices changes in the neighbourhood topology

How a package is sent AODV use the following procedure to send a packet:

1. check routing table to determine if the current route to destination is present. If so, forward the packet to the next hop, otherwise, go to step 2.
2. begin route discovery process with an RREQ [4]
3. once an intermediate node receives a RREQ, the node sets up a reverse route entry for the source node in its route table.
4. when RREQ reaches destination in order to respond to RREQ a node should have it in its route table (the route has to be unexpired). If conditions are true the node responds to RREQ by sending a RREP back using unicasting and not flooding to the source (using reverse path), otherwise the hop count is incremented and the packet broadcast to its neighbors. Eventually, the

[4] This packet is initially sent with a small time-to-live (TTL), to limit its propagation and avoid an excessive flooding. The packet contains:

- source node's IP address
- source node's current sequence number
- destination IP address
- destination sequence number

RREQ will be able to reach the destination. Intermediate nodes can send an RREQ if they know a more recent path of the one previously known to senders.

Broadcast The broadcast is performed by flooding the network, and a sequence number on the packets is used to stop loops from happening, meaning that a package could expire without have reached the destination. To solve this problem, the life of a packet can be augmented, changing its time-to-live (TTL). Using the flooding to broadcast packets has the following pros:

- simplicity
- efficient when overhead for route discovery is higher than the data transferred
- potential higher reliability of data delivery

...but it has some cons too:

- very high overhead potentially (in the worst case, all nodes reachable from sender may receive the packet)
- possibility of collision during packet propagation (despite the flooding, the destination may not receive the packet at all, if collision happens!)

7.2.2.2 DSR (Dynamic Source Routing)

With this ad-hoc routing protocol nodes maintains route caches, and the full source-route is aggregated in RREQ and sent back in RREP. **Each data packet has full source route**, causing a little overhead increment. Route discovery starts when the sender initiates requests and intermediate nodes add their address onto request so that when the packet reaches destination it contains all the full path.

Differences with AODV Being similar to AODV, it has the following differences:

- DSR includes source routes, having large headers (and suffering degraded performances)
- AODV keeps routing tables temporarily
- AODV keeps routes only between nodes which need to communicate with
- DSR route cached entries don't expire

7.3 Other routing & metrics

7.3.1 Routing protocols

Signal stability routing Here, the signal strength is used as metric, in combination with a DSR-like routing. RREQ is forwarded only if the packet is received over a link with good signal strength.

Geographic routing protocols These protocols use location. In particular, GPSR (greedy perimeter stateless routing) has the following properties:

- location of the destination node is assumed to be known
- each node knows location of its neighbours → packets are forwarded to the closest one
- if there is a "hole" in the graph, use perimeter routing (right-hand rule)

Maybe this role could be explained a little bit.

7.3.2 Metrics

Associativity-based Defines metric with a "degree of association stability" (with how many frequency do I move from my neighbours?). Nodes with less mobility or better links have higher stability volume.

This part is confused, check this out!

Expected Transmission Time (ETT) This metric is easier to compute than the signal strength, and it can be more useful.

Weighted cumulative expected transmission time This metric is better for multi-radio and asymmetric route links.

CHAPTER 8

Wireless MAC and Transport Protocols Interference

Usually in a home there are a lot of wireless devices: there are different requirements, from low delay to high bandwidth availability and high reliability.

Home routers can be a bottleneck[1] of the network traffic, acting as a Gateway between in-home devices and the outside world. Most network protocols developed assuming mostly TCP-based traffic. This assumption needs a radical reconsideration when UDP-based services for entertainment come into picture (because it's extremely delay sensitive).

8.1 Issues

One of the issue of wireless MAC is the packet retransmission, that cause delays that can be extremely dangerous for UDP-based services. This delay is increased with long TCP flows (e.g. FTP).

8.1.1 Solutions

In order to solve this problem several solutions were implemented. The first solution is to set the MAC packet retransmission to 3. This will ensure the maximum throughput[2] possible. This number of retransmission will ensure less jitter, so like 1-2 packets lost every 900 (that's 0.2%).

Jitter is the deviation from true periodicity of a presumably periodic signal, often in relation to reference clock signal. Jitter is a significant, and usually undesired,

[1] Typically the bottlenecks are at the edges of the network:

- Server bandwidth
- Last mile to router
- In home routers (wireless bandwidth)

[2] The throughput is the maximum rate of production or the maximum rate at which something can be processed in networking: is the rate of successful message delivery over a communication channel.

TCP SACK	TCP Vegas	SAP-LAW
Queuing	No queuing	No queuing
Unfair bandwidth usage	Unfair bandwidth usage	Fair bandwidth usage
	No compatibility versus other legacy TC	TCP compatibility (with old TCP Protocols) as is used on top of legacy transport protocols

Table 8.1: Comparison between different TCP version

factor in the design of almost all communications links.

8.1.1.1 SAP-LAW

SAP-LAW means Smart Access Point with Limited Advertisement Window.

Usually, TCP behavior is too aggressive, because it continuously probes the link for more bandwidth, and call fill up the AP (Access Point) buffer with its packets. A solution can be change TCP behavior to allow UDP based applications to be not jeopardized, granting high goodputs for downloading applications and avoiding queues, that are bad for real-time applications.

Smart Access Point This technique exploit the advertised window to limit the bandwidth utilized by TCP flows, limiting the speed of the sender. The AP can even avoid using buffer, having all the information about UDP bandwidth usage and can change the advertise window of TCP flows. The *maxTCPRate* is calculated with this formula:

$$maxTCPRate(t) = \frac{(C - UDPTraffic(t))}{\sharp TCPFlows(t)} \tag{8.1}$$

Where C is the total capacity of the channel.
The AP can change on-the-fly the advertise windows, that's:

$$min\{clientAdvertiseWindow,\ maxTCPRate\}$$

This make SAP-LAW easy to deploy as it requires modifications only at the AP.

About TCP unfairness A TCP throughput is inversely proportional to the RTT length, causing different throughput based on the RTT (even with the same windows). This leads to a new *maxTCPRate* formula, that includes RTT:

$$maxTCPRate_i(t) = \frac{(C - UDPTraffic(t)) \cdot RTT_i)}{\sum \frac{RTT_i}{avg-RTT_{min}}} \tag{8.2}$$

With this new formula, congestion windows are now based on the RRT[3].

[3] The previous formula was the "easier" version, and it had the flaw of sharing the same bottleneck with different RTTs, leading to RTT-unfairness.

CHAPTER 9

802.11 Wireless Standards

Currently there are a lot of 802.11 standards. The important ones are listed in the table 9.1

9.1 802.11e

802.11e as already said brings QoS (quality of service) extension. This doesn't mean you'll always have a good quality of service. 802.11 used initially two type of ways to challenge data: **DCF** (distributed coordination function) and **PCF** (point coordination function). From these two methods, **HCCA** was created.

DCF This is the basic access method for 802.11, and offers CSMA/CA for transmitting in the channel.

PCF It's a priority that is centrally controlled, the PC (point coordinator) is usually also the AP (access point). After each beacon there is a CP (content period) and a CFP (contention free period). PCF has several problems that lead this channel access method to not being used in practice, also because there is no mechanism to preserve bandwidth or characterize traffic in any way.

This paragraph is very confused, please check it out carefully!

HCCA It's a DCF inspired system, created as an extension of PCF, and it uses contention free periods. In contrast to PCF, in which the interval between two beacon frames is divided into ten periods of CFP and CP, the HCCA allows for CFPs being initiated at almost anytime during a CP. In this case, CFP is called CAP. During the CP, all stations function in EDCA mode, instead during CAP, the hybrid coordinator (the AP) controls the access to the medium. HCCA has the following characteristics:

- Efficient use of bandwidth
- Guarantees latency and bandwidth
- Has a complex scheduler and added complexity

802.11 Typology	Description
802.11a	54Mb/s, 5GHz. Launched in 2001, took more to develop than “b”
802.11b	5.5/11Mb/s. Launched in 1999, it was the first 802.11 protocol ever made
802.11e	It brings QoS (Quality of Service) extension
802.11g	54Mb/s, 2.4GHz, compatible with "b"
802.11n	High throughput technology - introduces MIMO
802.11p	It brings communication between vehicles but also for payments
802.11s	It brings extension for mesh networks

Table 9.1: List of most important 802.11 wireless standards

This way to challenge data is optional, and it’s not implemented to any significant level.

Additionally, 802.11 provides two types of contention window: **normal** and **adaptive**.

I don’t understand from where this *cw* came out :/

Normal contention window This type of contention window select a random number from a range that goes from 0 to *cw*. The size of *cw* is small, ensuring less wastage of idle slots time but causing a large number of collisions with multiple senders (two or more station can reach zero at once).

Adaptive contention window The adaptive contention window starts with $cw = 31$ and when no CTS or ACK are received from a communication it then sets the cw to $2 \cdot cw + 1$[1]. Finally, when the transmission succeed, the *cw* will be reset to 31. The adaptive scheme is unfair, and under contention, unlucky nodes will use larger *cw* than luckier one (due straight reset after a success). Having larger *cw* means that while unlucky nodes are counting down for access to the channel lucky nodes may be able to transmit several packets. 802.11 adaptive contention window doesn’t provide QoS.

9.1.1 EDCA

access categories

EDCA is a supported QoS mechanism in 802.11e. It has four access categories :

1. AC_V0 (for voice) → when you want to be fast but you leave the channel soon. it’s privileged to the others, especially during congestion. It has a small cw_{max} (7) because otherwise you’ll end up with collisions.
2. AC_V1 (for video)
3. AC_BE (best effort) → the difference is present only on the initiation phase

[1] So it’ll become $63, 127, 255$.

4. AC_BK (background) → with this access category, you don't care to be fast

EDCA has 8 traffic classes (TC) too. Each AC starts a back off after detecting the channel being idle for AIFS[2], and after waiting for it, each back off sets counters choosing between 1 and $cw + 1$. The back off formula is:

$$newCW[AC] \geq ((oldCW[TC] + 1) \cdot PF) - 1 \tag{9.1}$$

The MSDU (Max Service Data Unit) are delivered through multiple back offs within one station using AC specific parameters.

With EDCA video streams capacity drops. With more collisions I have larger contention windows, and with multiple streams I lose some overall bandwidth.

Prioritized channels Similar to DCF, but with four priorities, EDCA offers prioritized channel, based on the QoS parameters per traffic classes, which includes:

- AIFS[AC]
- CWmin[AC]
- PF[AC][3]

9.1.1.1 Pro and Cons

Using EDCA has the following advantages:

- Voice and video have priority over data
- It works well with a slightly loaded network

but has the following disadvantages:

- Streams of the same priority competes: making difficult for ECDA to be able to guarantee a good access, latency, bandwidth or jitter to every connection. A solution to this problem is using **admission control** .

admission control

Admission Control This solution allows only a limited number of peers to access the network, limiting (but not eliminating) the stream contention and reducing the latency of QoS streams. In this way the best-effort approach is no more necessary.

How Admission Control works The AP advertises ACM[4] in the beacon to indicate if admission control is mandatory for any Access Category. In this way, to use an AC that sends AddTS (*Add Traffic Specification*) Request Action Frame to AP that includes a TSPEC. When a new node wants to connect with the AP, the AP runs the admission control algorithm and replies with an AddTS Response Action Frame.

[2]AIFS means *Arbitration Inter Frame Space*
[3]PF means *Persistence Factor*
[4]This is a bit that says if the AP supports Admission Control.

9.2 802.11n

This version supports legacy mode (a/b/g), and offers more bandwidth for QoS applications, a greater wireless range and throughput. The high data rates sometimes can cause interference with radar signals, typically with 2 transmitters MIMO the data rate can go up 300Mbps (on a 40MHz channel).

9.2.1 802.11n enhancements

This 802.11 version brings different enhancements.

MAC layer at the MAC layer there are the following benefits:

- Smaller packets
- Combine ACK for more packets
- Packets aggregation

Physical layer At the physical layer, both for 2.4GHz and 5GHz we have:

- Smaller slots
- Better modulation (OFDM)
- MIMO (*Multiple Input Multiple Output*) radio technology with spatial multiplexing → transmit and receive with multiple radios simultaneously in some spectrum. MIMO is useful because one antenna can break but the communication would still be possible, thing that would not be possible with SISO (*Single Input Single Output*), that, on the other hand, has a lower-cost and has a simpler implementation than MIMO[5]. At the end of the day, MIMO presents the following advantages:
  - the signal has more power
  - transmits twice the amount of data

9.3 802.11p

This protocol is designed to provide connectivity to vehicles. Since it's still under heavy development it's not freely available at the moment. Governments have reserved in USA and EU a particular channel, 5.9GHz. 802.11p allows up to 300m communication range with 6Mbps. Transfers are possible even at 200km/h.

This wireless version uses DCF schema with a different period. A lot of research is going on in particular about routing schemes and MAC layers.

[5] SISO has this additional disadvantages though:

- the energy is wasted sending in all directions
- it's sensitive to interference from all directions

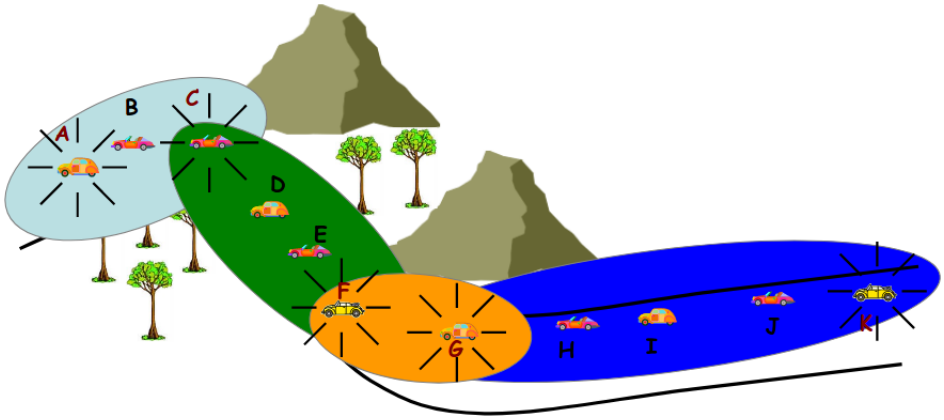


Figure 9.1: An example of an optimal broadcast: A, C, F, G and F broadcast the alert message, meanwhile the other cars receive the message but don't send it again, thus reducing the overall interference.

9.3.1 802.11p applications

The mayor application for this wireless standard is **safe driving**. When an incident is detected, alert messages are sent from the Abnormal Vehicle[6] to the following vehicles, via broadcast. This messages are faster than the normal human reaction, and can prevent chained car accidents. How much time does it take to the following vehicles to receive the message to start breaking? The latency it's an important aspect. Broadcasting messages could lead to an explosion when we have multi-hop path. To solve this, different solutions have been proposed. First of all, all the cars needs to have locations, so they can include details when they send the "emergency message", such as:

This footnote needs a rewrite

- geographical location
- acceleration
- speed
- moving direction

The car movement needs to be kept in consideration: cars can move very fast, causing variable transmission range. On top of that, a car cannot be sure to be the farthest car receiving that broadcast message, as explained in Figure 9.1. So who has to forward the alert message? It's important to pick up the right vehicle that has to propagate it. If the message has already been rebroadcast by a following vehicle do not forward it, because it could be redundant. A way to reduce the message retransmission is to use a sort of back-off mechanism, or otherwise set vehicles contention window with an inverse proportion of the distance from the sender, but it has the unrealistic assumption that there is an unique, constant, and known-a-priori transmission range.

[6] AV is a vehicle that its behavior is not conform to some threshold

9.3.1.1 Using minimum connect dominating set

The MCDS identify the minimum number of sets that allows you to cover the whole set[7]. Although it seems a good solution for this problem, it has the following issues:

- only the one in the MCDS can propagate the message, making the solution optimal, but not feasible: I would need a central station then knows all the other nodes position
- this solution is not considering the number of hop a message has to traverse

9.3.1.2 Urban multi-hop broadcasting protocols

This broadcasting protocols uses **jamming** signals to determine the next forwarder: vehicles receiving an alert message emit a jamming signal for a time that's proportional to the distance from the sender. The last one stops the jamming signal and forward the message.

The problem of this solution is that is slow, and for this is not suitable for alert messages (jamming signal phase delays the transmission of the message).

9.3.1.3 Fast Broadcast

Fast broadcast[8] is designed to have Alert Messages covering the area of interest in less time possible. It uses two type of packets: *Hello messages* and *Alert messages*. Hello messages contain the sender's position and the maximum forward distance from which another vehicle has been heard transmitting another hello message. In particular it contains:

- CMBR (*Current-turn Maximum Back Range*): the estimation of the maximum frontward distance from which another car along the strip-shaped area-of-interest can be heard by the considered one;
- CMFR (*Current-turn Maximum Front Range*): the estimation of the maximum backward distance at which the considered car can be heard.

Fast Broadcast has two phases:

1. estimation of transmission range between two vehicles (called *hello messages*). This phase has little overhead and it's divided in rounds: at every round hello messages are randomly sent
2. transmission is used during broadcasting to reduce message number of hops (*alert messages*[9])

[7] A minimum connected dominating set of a graph G is a connected dominating set with the smallest possible cardinality among all connected dominating sets of G. The connected domination number of G is the number of vertices in the minimum connected dominating set. See more at https://en.wikipedia.org/wiki/Connected_dominating_set.

[8] http://www.math.unipd.it/~cpalazzi/lucidi/Read-FastBroadcast.pdf

[9] This message includes the estimated transmission range for that hop.

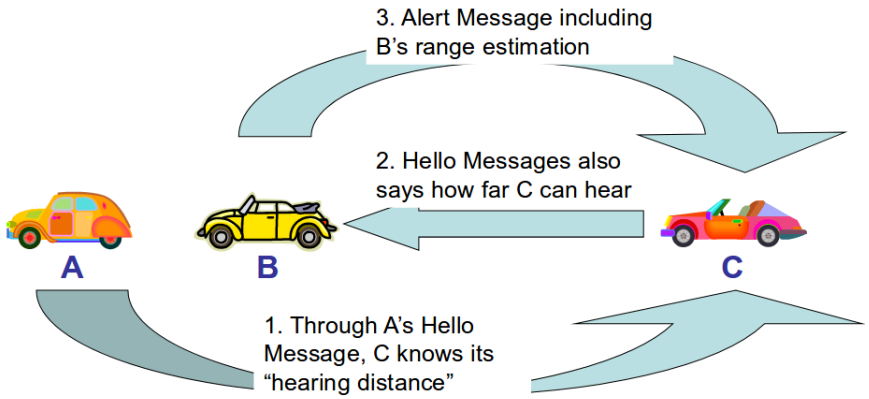


Figure 9.2: A little illustration of how hello messages are delivered through the cars.

When a car receives a message, as illustrated in Figure 9.2, it can be from two directions: in front of it (and in that case it's an update), or behind it (and in that case you update your transmission range).

The characteristics of this multi-hop broadcast protocol for vehicular networks are:

- estimation of the maximum transmission range with few hello messages
- minimization of the number of hops to be traversed
- reduction of the delivery time

Handling an AV When an Abnormal Vehicle (AV) occurs, an alert message is generated, and received by the nearby cars. The cars that received the message wait a time that is proportional to the node's position with respect to the estimated maximum transmission range. In particular, farthest nodes transmit before the others. If, while waiting, some of the following cars already transmitted the message, preceding ones abort their forwarding countdowns as the message has already been propagated. The contention window during this procedure is calculated in this way:

$$\left[\left(\frac{MaxRange - Distance}{MaxRange} \cdot (CW_{max} - CW_{min})\right) + CW_{min}\right] \tag{9.2}$$

Comparing Fast Broadcast Let's compare Fast Broadcast with Static 300 and Static 100. Static 300 has a lot of collisions for 1 km of transmission range, determining Fast Broadcast as the best solution. Static 100 performance are in general very poor.

These results are due the fixed nature of the contention window of Static 300 and Static 100, that after a certain distance start to have the same contention window for farther cars. With small contention windows there is a surge of collisions.

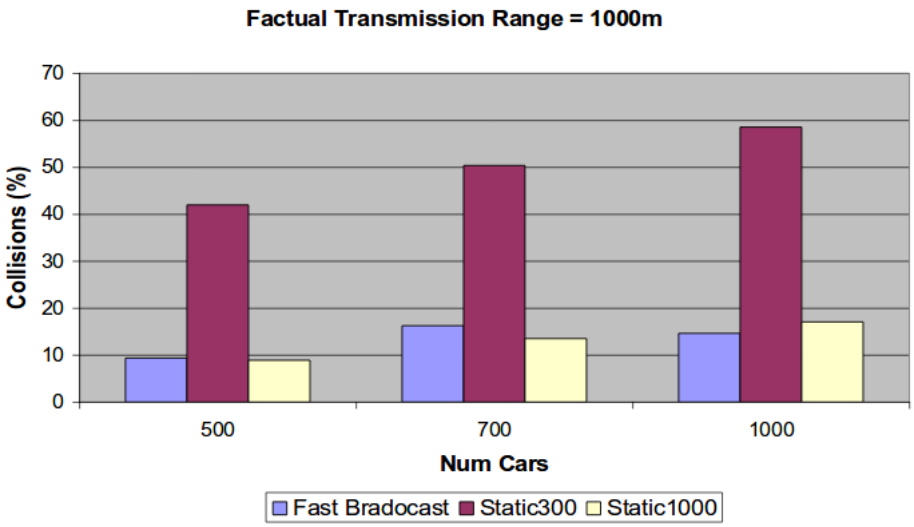


Figure 9.3: This histogram illustrates a comparison between Fast Broadcast, Static 300 and Static 100 over 1000m of actual transmission range: how is possible to see, the number of collision is very high for Static 300 at this distance, while Fast Broadcast and Static 100 have similar results.

CHAPTER 10

WPAN Protocols

10.1 Bluetooth

Bluetooth, known as 802.15, is a communication protocol designed in order to create personal area networks (PAN). This opens up to new possibilities, such as automatic synchronization of calendars between devices, or to transmission of audio between smartphones and cars.

As you can see in Figure 10.1 the Bluetooth stack is modular. You don't have to use all of it, you can use/implement only a subset. There are different parts of the protocol:

- core protocol
- cable replacement and telephony control protocol (RFCOMM and TCS BIN)
- adopted protocol (use already existing protocols):
  - PPP
  - TCP/UDP/IP
  - OBEX
  - WAE/WAP
- radio
- baseband
- link manager protocol (LMP)
- logical link control and adaptation protocol (L2CAP)
- service discovery protocol (SDP)

This modularity allows only some parts of the Bluetooth to be used, based on the necessity. There are predefined configurations, called **profiles**, that are:

- file transfer
- internet bridge
- LAN access
- synchronization

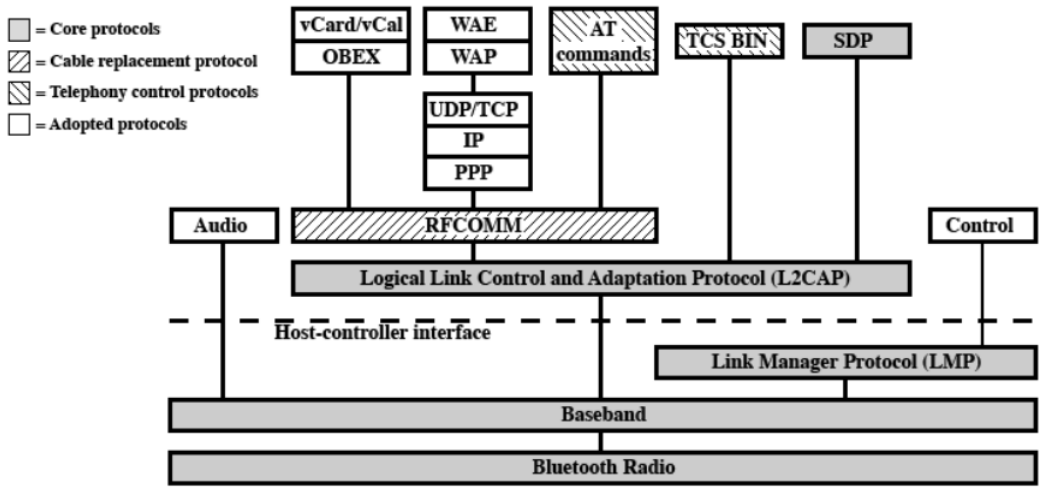


Figure 10.1: A representation of the Bluetooth stack

- three-in-one phone
- headset

10.1.1 Bluetooth specifications

Bluetooth topology can support up to seven simultaneous links in a logical star. The peak data rate is of 1Mbps, using a band of 2.4GHz. There is the capability to create a **piconet** or a **scatternet**.

Piconet A piconet is a collection of devices connected in an ad-hoc fashion. One node will act as a master and the others as slaves for the duration of the piconet connection: each master can connect up to 7 simultaneous devices or 255 inactive slaves per piconet: all the slaves communicate through the master, that doesn't polling to regulate communications, even the hopping pattern.

Scatternet Piconets can be interconnected, with one master per piconet, few devices are shared between the network, and it can be a combination of Mater/Slave or Slave/Slave. This is still an ad-hoc network, without a central structure. This mode allows many devices to share the same area, and makes efficient use of bandwidth.

Connection Setup There are two scan typologies: *Inquiry-scan protocol* and *Page-scan protocol*. The first one learns about about the clock offset and device address of other nodes in proximity, instead the second one establish links with nodes in proximity.

10.2 ZigBee

ZigBee is defined as 802.15.4 and its focus is mainly about power usage (over 100 times less energy required for connection operations). It uses mesh networking, although it can use different topologies, as star and cluster tree.

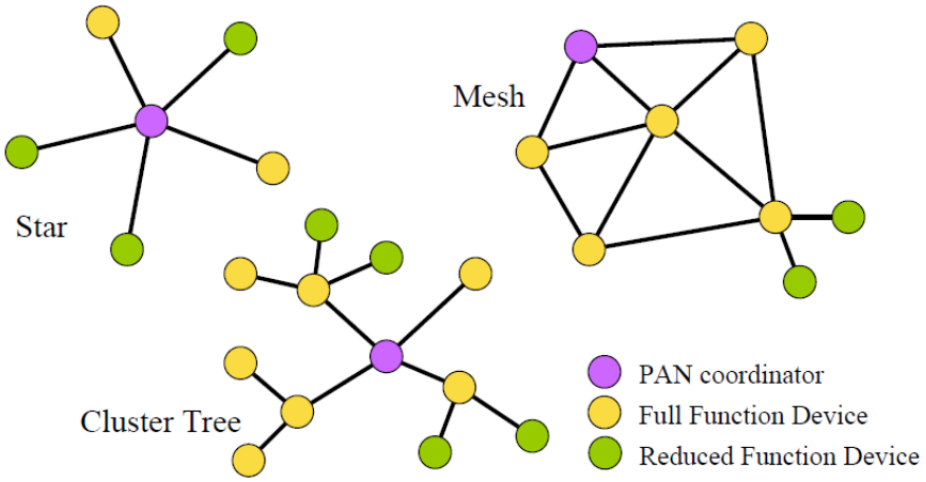


Figure 10.2: ZigBee Typology. As you can see, there are different kind of devices:

- ZigBee coordinator (ZC): there is only one per network, it initiates the network formation and it may act as router once the network is formed
- ZigBee router (ZR): it's an optional network component, it may associate with a ZC or with another ZR and it can participate in multihop routing of messages
- ZigBee end device (ZED): it's an optional network component, that hasn't association and routing.

ZigBee supports up to 65k of nodes, and it's optimized for timing-critical applications and power management. It only takes 15ms for a node to access the channel, and 30ms to join the network. With WiFi you need seconds to perform this actions. These sorts times make the system very reactive.

Wibree Wibree is a Bluetooth version that tries to reach ZigBee low-energy and performance. It hasn't Bluetooth frequency hopping and uses the same Bluetooth hardware.

10.2.1 ZigBee vs Bluetooth

There are main differences with Bluetooth. BT is designed to be a cable replacement for this sort of things:

- synchronization of call phone to PDA
- hands-free audio

ZigBee instead works better where there are:

- Controls
- Sensors
- Small data packets
- Long battery is critical

ZigBee is more suited for medic transfer or data transfer.

CHAPTER 11

Molecular Communication

Molecular communication is performed through nano machines: a nano-scale device able to perform a specific task at nano-level. It can be a natural device or an artificial one. Each one has specific characteristics, but both work in a similar way.

The development of this devices can have different approaches:

- Top-down → down-scaling existing components
- Bottom-up → develop using individual molecules (they grow developing them self)
- Bio-hybrid

Molecular communications found applications in different environments, such as:

- Oil spilling containment
- Health monitoring
- etc...

All of these applications should be highly reliable and secure.

11.1 Nano-network

A nano network is composed by a set of components on a nano-scale and their interconnection. Usually, these nano networks are more biological than electronic. There are different ways communications are carried on:

- Standard communication
- Nano-mechanical communication
- Molecular communication

11.1.1 Standard communication

There are two types of standard communications:

1. Electromagnetic waves
    - wiring a large quantity of nano-machines is unfeasible
    - wireless solution could be used but antennas are hard to be integrated
2. Acoustic waves

11.1.2 Nano-mechanical communication

With this kind of communication you need to have junctions between the nano-machines, to perform actions like DNA transfer. Having physical contacts between two nano-machines isn't always easy.

11.1.3 Molecular communication

Molecular communication is the more feasible solution for the knowledge humanity has today. The information and the transmission are encoded in molecules, and it's the most promising approach for nano networking.

There are two types of transmission: **short range** (within mm) **long range** (up to km).

11.1.3.1 Short range transmission

Short range transmission can happen through molecular motors or calcium signalling. With molecular motors proteins can transform chemical energy into mechanical work. They can travel along molecular rails, and data need to be encoded, transmitted and decoded somehow. On the other hand, calcium is a well-known molecular for communication, as it's responsible for many coordinated cellular tasks (like messages that triggers an action). This is more flexible than railways communication, and different kind of signals can be specified using different levels of calcium.

11.1.3.2 Long range transmission

Long range communications can use pheromones or bacteria: in the first case the pheromones are molecular compounds containing information that can only be decoded by specific receivers, and messages consist of molecules. Bacteria have interesting characteristics: conjugation (bacteria interconnection that allows plasmoids transmission), chemotaxis (movement induced into bacteria by chemical stimuli), and can develop antibiotics resistance, that can be useful to get rid of interference when there are too many bacteria.