Lezione 1

Qual è lo scopo di una **rete**? La rete ci deve dare accesso ai dati, alle informazioni ma ci deve permettere anche di condividere risorse e una comunicazione efficiente e facilitata. I componenti di un'infrastruttura di rete sono di due macro categorie:

- **Hardware**: sono gli apparati che fanno funzionare la rete. Quindi sono dispositivi di interconnessine che realizzano la rete stessa. (apparati di interconnessione e apparati per il controllo della trasmissione);

- **Software**: sono tutte le implementazioni dei protocolli che vengono usati ai vari livelli di interesse della rete. I device driver che a livello di sistema operativo, ci permetteranno di interfacciare i dispositivi che poi realizzano i meccanismi di interconnessione alla rete. (protocolli e drivers);

Da una rete noi chiediamo la massima affidabilità, un adeguato grado di efficienza, la scalabilità e l'abilità di interconnettere dispositivi eterogenei. Parlando di affidabilità, i mezzi trasmissivi in realtà non sono affidabili perché analogici, noi dobbiamo predisporci al fatto che questi mezzi possono generare dei problemi (quando inviamo informazioni possono arrivare corrotte, si può essere la perdita di dati, duplicazione di dati, distribuzione di pacchetti in ordine diverso) e una rete deve essere in grado di risolvere tutti questi problemi.

Un'infrastruttura di rete deve essere in grado di assolvere un'altra funzione molto importante chiamata **instradamento**: cioè individuare i cammini ottimali da una sorgente a una specifica destinazione. Quindi deve essere in grado di indirizzare i nodi della rete e deve essere in grado di trovare un percorso fra ciascuna coppia di nodi.

I problemi di base di quando si parla di infrastrutture di comunicazione sono: il caso più semplice è quando abbiamo solamente due dispositivi che devono comunicare attraverso un mezzo trasmissivo che è un **canale fisico**. Gli unici problemi, in questo caso, sono solo di genere fisico: elettrico, ottico, onde elettromagnetiche.

Quando, invece, molti più nodi si affacciano sullo stesso canale fisico cominciano a nascere nuovi tipi di problemi:

- Innanzitutto la rete deve **indirizzare** le informazioni ad uno specifico nodo;
- Si potrebbe avere un'infrastruttura complessa costituita da tanti nodi e alcuni dei quali svolgono la funzione di **relay intermedi**, cioè favoriscono il passaggio di informazione da un nodo all'altro. In questo caso si parla di **commutazione**;
- Il problema dell'**instradamento**, cioè dell'individuazione dei percorsi;
- Il problema dell'**affidabilità**: quando si hanno molti nodi e molti circuiti potremmo avere una serie di bisogni differenti di individuazione di soluzioni ad alta affidabilità (es. per far parlare due nodi potremmo necessitare di più percorsi in modo che se la comunicazione fallisce su percorso ne avremo uno di riserva).

Definizione funzionale di rete

Un possibile modello fisico che implementa la definizione di rete di telecomunicazione deve prevedere la presenza di:

- **Hosts** (o stazioni): dispositivi connessi ad una rete come *terminali*, cioè punti di terminazione di una comunicazione;
- **Links**: mettono in comunicazione i nodi della rete;
- **Nodi intermedi di commutazione** (o Network Switch): il compito è quello di riconoscere le richieste per l'apertura di una connessione e fare in modo che i dati, relativi a tale connessione, arrivano al nodo di destinazione.

Una rete si può rappresentare sotto forma di grafo in cui gli host sono le foglie, i commutatori sono i nodi interni e le linee tra host e commutatori sono i link.

Tipi di rete

Le infrastrutture di rete possono essere di vari tipi:

- Una rete caratterizzata solamente da un canale fisico tra due nodi è detta **punto-a-punto**. Potremmo però avere anche una rete **multipunto**, cioè un canale fisico condiviso da più stazioni. Questo tipo di rete però può generare un *problema di contesa*, quando due stazioni possono richiedere l'utilizzo dello stesso canale fisico causando una *collisione*. (unicast)
- Una rete **broadcast**, di fatto è una specializzazione di una rete multipunto, la differenza è che tutti sentono tutto cioè tutte le stazioni connesse alla rete sono in grado di percepire la comunicazione. A volte però si vuole inviare una risorsa solo ad una specifica stazione e qui nasce il concetto di "indirizzamento sulle reti broadcast" cioè invio un'informazione specificando l'indirizzo della stazione che dovrà riceverla gli altri la scarteranno. Se invece voglio che l'informazione arrivi a tutti, allora userò un indirizzo riservato detto *indirizzo di broadcast*;
- Una rete **multicast** è una specializzazione di una rete broadcast. È in grado di inviare un'informazione ad un gruppo di stazioni e non a tutti tramite un indirizzo multicast assegnato a quel gruppo. In genere l'appartenenza ai gruppi multicast avviene sottoscrizione, cioè è il dispositivo stesso che si aggancia al gruppo.

Flussi trasmissivi (come si può trasmettere)

Il caso più semplice è quello della trasmissione **simplex** ovvero i dati viaggiano in una sola direzione.

Se vogliano una doppia comunicazione abbiamo una prima opzione, lavorare in logica **half-duplex** facendo viaggiare i dati in due direzioni ma mai nello stesso tempo. Quando si deve cambiare direzione le due stazioni si devono mettere d'accordo, in questo caso si parla di **line turn around**.

Un'altra possibilità di trasmissione, e anche più complessa perché richiede mezzi trasmissivi in cui abbiamo componenti per trasmissione e componenti per la ricezione, è la trasmissione **full-duplex** in cui le informazioni possono viaggiare contemporaneamente in entrambe le direzioni.

La commutazione

La commutazione è quell'operazione che predispone il percorso che le informazioni emesse dal mittente devono seguire per raggiungere il destinatario attraversando tutta una serie di dispositivi intermedi detti **commutatori** o **switch**. Ogni switch intermedio assolve la funzione di **multi-hop relay**, cioè riceve l'informazione dal precedente trasmettitore e la spedisce su un'altra interfaccia che può essere collegata o alla destinazione oppure ad uno switch intermedio che farà da inoltro intermedio.

Esistono due tipi di commutazione: **commutazione di circuito** e **commutazione di pacchetto**. La funzionalità è la stessa, è sempre una funzione di *cross-connessione*, cioè avere una matrice di commutazione che mi collega dinamicamente un'interfaccia all'altra. Questa connessione può essere temporanea, dura solo per la trasmissione del dato, o semipermanente, l'impegno del collegamento sulla matrice di switch dura un lasso di tempo desiderato.

Quindi i commutatori, programmati in maniera opportuna, sono i dispositivi che costituiscono la scelta del percorso di rete.

Commutazione di circuito: una rete di commutazione di circuito è costituita da un insieme di switch connessi da collegamenti fisici partizionati in più canali logici di comunicazione.

Prima di avviare la trasmissione di dati c'è una richiesta di stabilire un circuito logico che parte dalla sorgente e arriva alla destinazione. Questo circuito prevede, per l'intera durata della trasmissione fino a che non faccio una chiusura del circuito, un impegno statico delle risorse sugli switch intermedi. Quindi i canali di comunicazione, all'atto del setup/dell'apertura della connessione vengono riservati dalla sorgente fino a destinazione; quando ho finito la trasmissione delle informazioni, parte una procedura di chiusura del circuito, rilasciando tutte le risorse precedentemente riservate.

Quindi quando lavoro a commutazione di circuito, assegno tutte le risorse di quella connessione in maniera permanente finché non viene buttata giù la connessione, il che significa che sto utilizzando le risorse in maniera inefficiente perché se in quel momento la connessione è su ma

nessuno trasmette le risorse sono bloccate lo stesso. Il vantaggio è che io ho sempre e comunque la garanzia d'impegno di quelle risorse.

Nella commutazione di circuito, temporalmente, abbiamo 3 fasi:

1. **Connection setup** in cui c'è una comunicazione completa fra A e B;
2. **Trasferimento dei dati;**
3. **Chiusura di connessione**.

Commutazione di pacchetto: la commutazione di pacchetto si basa sulla suddivisione del messaggio in più unità autonome, dette **datagram**, ciascuna corredata delle opportune informazioni di controllo:

- Non ci sono canali riservati;
- Datagrammi gestiti generalmente in logica FIFO;
- Ogni datagram è instradato indipendentemente (e su percorsi differenti);
- La rete non ne garantisce l'inoltro e la ricezione nel giusto ordine;
- Utilizzo ottimale delle risorse in ragione del principio di multiplazione statistica.

Pro e contro commutazione a circuito e di pacchetto:

- La commutazione di pacchetto è in media più scalabile e ottimizza la gestione delle risorse;
- È estremamente efficiente per il trasporto di pacchetti di piccole dimensioni che comportano trasferimenti di alcune centinaia di KB;
- Però è evidente l'inadeguatezza della commutazione di pacchetto per il trasporto di grandi quantità di informazioni sulla rete (si parla di diversi Tera o Petabytes);
- Gestire decisioni di instradamento ogni 1500 bytes per trasferire ad es. 1,5TB di dati, richiede di reiterare le stesse decisioni circa un bilione di volte su tutti i routers coinvolti;
- È inoltre praticamente impossibile riservare risorse in anticipo o fare ingegneria del traffico scegliendo i percorsi;
- La logica del multiplexing statistico non scala su grandi volumi.

Modalità di colloquio

Una caratteristica di una rete riguarda il modo in cui può avvenire il colloquio tra due nodi. La **modalità di colloquio** è una proprietà nella comunicazione end-to-end, da nodo terminale a nodo terminale, cioè da host a host. Ci sono due modi di colloquio: **connection oriented** e **connectionless** (la prima è tipica della commutazione di circuito e la seconda di quella di pacchetto).

Quando devo stabilire una comunicazione *connection oriented* fra due nodi devo:

- prima di tutto stabilire una connessione fra due nodi (connection setup), ciò non significa che bisogna lavorare a circuito bensì devo fare in modo che fra due nodi ci sia un mutuo accordo per essere pronti a trasmettere: in questi casi si parla di una "negoziazione"

iniziale della connessione (three hand shake). Tutto ciò è gestito in software (invece nella commutazione a circuito viene gestito in hardware;
- fatta questa verifica, viene instaurata la connessione (o colloquio o sessione), la quale dura per tutto il tempo necessario allo scambio di dati;
- non appena tale scambio è terminato, anche la connessione viene abbandonata.

Lavorare in modalità connection oriented mi dà una serie di funzionalità evolute che sono:

- il **canale perfetto**: una serie di meccanismi che mi permettono di gestire e controllare gli errori;
- **gestione del ritmo di interscambio**: un controllo di flusso, cioè il controllo della velocità dello scambio di informazioni. Sincronizzazione della velocità fra trasmettitore e ricevitore.

Nella modalità *connectionless*, invece, non si stabilisce una connessione: non mi interessa un canale permetto, mi importa uno smistamento veloce. Non mi interessa di una negoziazione tra nodi, nemmeno che i pacchetti arrivino in ordine o che si perdano, non ho bisogno di servizi di controllo o di servizi di supporto.

- Scarica la rete in reti efficienti ed affidabili;
- Può essere svantaggioso su reti con problemi di scarsa affidabilità.

Pro e contro connection oriented e connectionless:

- Il problema principale del connectionless mode riguarda il controllo degli errori che, sia pure raramente, possono verificarsi: infatti non essendoci controlli immediati durante la trasmissione, il nodo sorgente non può sapere come è andata la trasmissione;
- D'altra parte, l'onere dei controlli ripetitivi spesso diventa inutile sulle reti ad alta affidabilità, dove gli errori sono decisamente pochi;
- La soluzione cui si può pensare è allora quella di affidare il controllo degli errori direttamente alle applicazioni, il che alleggerisce i protocolli di linea, che possono occuparsi solo del trasporto dei dati, nonché anche i nodi intermedi, che devono occuparsi solo di instradare i dati sui percorsi desiderati.

Topologie di rete

Una topologia di rete è la configurazione geometrica dei collegamenti tra i vari componenti della rete. Ovviamente la topologia caratterizza anche gli obiettivi della rete stessa:

- ad esempio, in una rete strutturata ad albero avrò un'affidabilità limitata perché se mi salta un circuito si partiziona la rete; in una rete strutturata a grafo potrei avere dei percorsi alternativi per cui la rete è più affidabile.
- la topologia caratterizza anche la scalabilità dell'infrastruttura;
- anche il rendimento in termini prestazionali, quindi avendo a disposizione certi tipi di topologie, si è in grado di erogare più banda;
- un altro aspetto è quello economico: una topologia di rete di un certo tipo potrebbe costare molto meno di un'altra.

Esempi:

- una **topologia a stella**:
  - bastano n-1 collegamenti per connettere n nodi;
  - controllo centralizzato del traffico;
  - problemi di robustezza: se muore un nodo che collega molti nodi, muoiono tutti gli altri;
  - non scala col numero di nodi;

- una **topologia ad anello**:
  - bastano n collegamenti per connettere n nodi;
  - doppio percorso fra coppie di nodi;
  - non scala col numero di nodi: percorsi infinitamente lunghi;
  - problemi di congestione sui percorsi;
  - problemi di capacità dei nodi;

- una **topologia full mesh** (tutti connessi con tutti):
  - servono n(n-1) / 2 collegamenti per connettere n nodi;
  - massima affidabilità e percorsi dedicati;
  - non scala col numero di nodi: costo inaccettabile;

- rete **ad albero** (o rete gerarchica):
  - è un'evoluzione della stella; possiamo vederlo come un insieme di stelle connesse fra loro;
  - ha gli stessi problemi della rete a stella ma è ottima per gestire piccole infrastrutture di rete;
  - il concetto di "gerarchia" fa in modo che il traffico venga distribuito in modo che vada dai nodi di livello più basso (nodi di accesso) verso i nodi intermedi (nodi di distribuzione) fino ai nodi di livello più alto (radice dell'albero) che centralizza il controllo della rete;
  - la criticità sta nei nodi intermedi perché in caso di fallimento salta tutta la rete però posso risolvere questo problema raddoppiando i componenti critici, ovvero i nodi intermedi e il nodo centrale. A questo inconveniente si può però ovviare adottando una **politica di back-up**: bisogna cioè mettere in grado uno o più altri nodi della rete di svolgere le stesse funzioni del nodo principale nel momento in cui questo dovesse venire a mancare;
  - un inconveniente di questa topologia è che se il nodo principale è sovraccarico di lavoro, può diventare un collo di bottiglia per l'intera rete, il che comporta un rallentamento dei servizi per tutti gli utenti;

- reti a **bus condiviso** (o a dorsale):
  - è la configurazione adottata dalle reti locali di tipo ethernet;
  - la caratteristica è che c'è un unico cavo che collega tutte le stazioni;
  - il problema principale è che è una infrastruttura **BMA** (broadcast multi-access): avendo un bus comune avrò problemi di indirizzamento e di **arbitraggio** (capire chi è che deve trasmettere in un determinato momento);
  - il vantaggio fondamentale è la condivisione del *portante* comune, che è molto economico da implementare. L'interruzione di servizio del portante comporta un'interruzione di servizio dell'intera rete;

- rete ad **anello** (ring):
    - questa configurazione è stata resa popolare dalle prime LAN di tipo *Token-Ring* o *FDDI*. Oggi è utilizzata nelle infrastrutture medio-grandi (reti metropolitane) per via del costo contenuto di implementazione;
    - si può implementare ad **anello singolo** oppure ad **anello doppio controrotante**, abbiamo un anello che va in una direzione e uno che va nell'altra. Vengono implementate reti anche a più livelli per avere un percorso alternativo in caso fallimento sul singolo collegamento;
    - Anche qui è necessario un meccanismo di arbitraggio (spesso basato sul possesso di token, che abilita alla trasmissione);

- rete a **maglia**:
    - consiste nel collegare le varie stazioni con diversi circuiti;
    - è una topologia *non strutturata* in accordo a una logica di tipo opportunistico: non stabilisco i collegamenti in accordo a una struttura geometrica ben delineata, ma semplicemente stabilisco i collegamenti per come mi viene più utile, in ragione di criteri economici, in ragione di criteri di opportunità;
    - prevede dei link ridondanti che in qualche modo sono strutturati sulla base di criteri di tipo economico;
    - è la topologia più comune su internet perché assicura buone prestazioni in quanto il traffico viene ripartito sui vari percorsi;
    - inoltre, essa conferisce una elevata affidabilità all'intera struttura, proprio grazie alla presenza di *percorsi multipli*;
    - allo stesso tempo, però, i costi dei collegamenti possono anche essere elevati ed inoltre la gestione della struttura è chiaramente più complessa rispetto agli altri casi esaminati.

Qualunque sia la topologia che usiamo, si parla sempre di **grafi di rete**; potremmo avere grafi casuali o meno casuali: consideriamo una situazione in cui abbiamo una geometria di tipo ordinato (ad esempio le topologie sopraelencate) oppure potremmo avere una geometria completamente casuale. La rete mesh è il compromesso tra l'ordine assoluto e il disordine assoluto: la rete mesh, infatti, si chiamerà *organizzazione di tipo caotico*, dove caos non significa confusione bensì organizzazione difficile da percepire (il caos è un punto di equilibrio tra l'ordine e il disordine).

Protocolli

Un **protocollo** è una serie di norme, convenzioni e tecniche per lo scambio di dati, comandi e informazioni di controllo tra due elementi. Sono le regole del gioco secondo cui vengono stabilite le modalità di colloquio tra dispositivi eterogenei.

Esistono molti **livelli** di protocolli: si va dal livello più basso, che regola il modo di trasmettere i segnali sulla linea (**protocollo di connessione**), al livello più alto, che indica come interpretare dati e comandi a livello applicativo, passando per una serie variabile di ulteriori livelli.

I protocolli più importanti sono i **protocolli di routing** (o di instradamento), utilizzati dai commutatori per scambiare le informazioni che servono ad individuare i percorsi.

Funzionamento:

- una volta individuata la stazione di destinazione, bisogna stabilire quale strada usare per connetterla alla stazione sorgente;
- questa scelta compete al protocollo di routing che quindi si aggiunge al *protocollo di linea* necessario al passaggio di dati su ciascuna linea. In altre parole, solo dopo la scelta del percorso interviene il protocollo di linea per la gestione dei singoli collegamenti;
- tale protocollo viene usato tante volte quante sono le linee che costituiscono il percorso fissato.

Altri tipi di protocolli sono i **protocolli di trasporto** che servono a stabilire le connessioni end-to-end. Ad esempio, quando due nodi devono stabilire una negoziazione di tipo connection oriented dovranno rispettare un protocollo comune per stabilire questa negoziazione.

I protocolli si traducono in **standard** che riguardano sia implementazioni hardware che implementazioni software. Differenziamo due famiglie di standard: **De Jure** e **De Facto**. Poi ci sono i cosiddetti standard **Proprietary** che non sono standard veri e propri: una scelta fatta da un singolo costruttore vale solo per gli apparati costruiti dal medesimo costruttore (si parla di tecnologie legacy).

Questi standard vengono promulgati da *organismi di standardizzazione* ognuno specializzato per certi tipi di problemi (IEEE, OSI, ANSI, ATM, FORUM, …).

Gli standard **De facto** sono tipicamente standard flessibili che si implementano a livello di sistema operativo o di applicazioni. Se voglio cambiarli possono farlo facilmente senza creare danni particolari. Viceversa, quando avrò una gestione di tipo industriale (ad esempio, cambio il processo industriale di come si produce la fibra ottica), tutto ciò che prevede un'implementazione in hardware deve essere progettato attraverso standard molto più rigidi. In questo caso si parla di standard **De jure**.

Le principali autorità nel mondo degli standard sono:

- **PTT** (**Post, Telephone and Telegraph**): amministrazione statale che gestisce i servizi trasmissivi;
- **CCITT** (**Consultative Committee for International Telegraph and Telephone**): organismo internazionale che emette le specifiche tecniche che devono essere adottate dalle PTT. È entrato da poco a far parte dell'**ITU** (**International Telecommunication Union**);
- **ISO** (**International Standard Organization**): il principale ente di standardizzazione internazionale, che si occupa fra l'altro anche di reti;
- **ANSI** (**American National Standards Institution**): rappresentante USA nell'ISO;
- **UNINFO**: rappresentante italiano, per le reti, nell'ISO;

- **IEEE** (**Institute of Electrical and Electronic Engineers**): organizzazione professionale mondiale degli ingegneri elettrici ed elettronici; ha gruppi di standardizzazione sulle reti;
- **IRTF** (**Institute Research Task Force**): comitato rivolto agli aspetti di ricerca a lungo termine in merito alla rete internet;
- **IETF** (**Institute Engineering Task Force**): comitato rivolto agli aspetti di ingegnerizzazione a breve termine della rete Internet;
- **IAF** (**Internet Architecture Board**): comitato che prende le decisioni finali su nuovi standard da adottare per Internet, di solito proposti da IETF o IRTF.

Copertura delle reti

Piccole distanze velocità maggiori.

Una WAN viene utilizzata per realizzare l'interconnessione fra LAN o fra MAN.

Una **internetwork** è un aggregato di reti locali attraverso LAN o MAN. (non globale)

I dispositivi che si usano per interconnettere reti diverse prendono il nome di **gateway**. Non sono altro che commutatori o router che operano in logica multiprotocollo, cioè possono supportare interfacce eterogenee (collegamenti di mezzi trasmissivi molto differenti tra loro, ad esempio, rame e fibra ottica, wireless e rame.

Subnet: sia nel contesto di una rete geografica sia nel contesto di una LAN, è un sottoinsieme di macchine che vengono raggruppate per costituire una sottorete isolabile.

(vedere slide e libro)

Lezione 2

Il modello **ISO-OSI** è un modello a stack (a livelli) in cui a ogni livello: confiniamo un certo grado di astrazione e corrispondono funzioni specifiche che vanno a definire quelli che sono gli standard operativi a quel livello. Questi livelli sono progettati in maniera tale che i confini fra livelli devono minimizzare il flusso delle informazioni che fluisce tra un livello e l'altro. Il numero di livelli deve essere scelto in maniera opportuna da essere ottimale per rappresentare il tipo di problema a cui è interessato.

Il modello OSI ha sette livelli suddivisi in due insiemi: un insieme di livelli più vicini all'utente che sono i livelli di **applicazione**, **presentazione** e **sessione** e sono chiamati "*livelli di processo*", invece, l'altro insieme di livelli più vicini alla modalità di comunicazione fra host sono i livelli di **trasporto**, **network**, **data link** e **fisico** e sono detti "*livelli data flow*".

Il primo livello è il livello fisico, esso si occupa di trasportare bit di informazione fra dispositivi utilizzando mezzi fisici, cioè facendo variare una grandezza fisica (un segnale) che può essere un voltaggio, un segnale luminoso o un'onda elettromagnetica. A questo livello cominciamo a definire le regole fisiche per trasmettere il segnale: come codifichiamo il segnale come variazione di una grandezza fisica su un messo fisico, la velocità a cui trasmettiamo, come è fatto il mezzo fisico e le sue interfacce.

In questo livello troviamo una serie di standard che sono associati con processi industriali di produzione di interfacce e di mezzi trasmissivi; quindi, sono standard di diritto (de jure).

Il secondo livello è il livello di data link, esso non vede più le informazioni come segnali che trasportano bit bensì come aggregati di bit detti "trame". Questi bit cominciano a fluire da un lato all'altro della comunicazione, in questa fase si cominciano a verificare i primi problemi: bisogna gestire eventuali errori e correggerli (questi errori possiamo anche gestirli in livelli più alti, tutto dipende dal tipo di stack di protocolli che andiamo ad utilizzare); il problema dell'arbitrato dell'accesso al mezzo, nel caso in cui abbiamo un mezzo BMA.

Gli standard di questo livello definiscono lo scambio di bit strutturati fra un'entità e un'altra.

(cominciamo a vedere già una certa astrazione)

Il terzo livello, quello più importante di tutti, è il livello di network esso assolve le funzioni di una rete. Il suo compito è quello di definire i due meccanismi fondamentali per la funzionalità di una rete che sono *indirizzamento universale* (locale al data link) e *instradamento*. L'indirizzamento universale assegna un indirizzo univoco alle stazioni che permette di identificarle sulla global internet ovunque esse siano. In questo livello, inoltre, si gestisce l'*accounting*, ovvero il tracciamento delle quantità di traffico che vengono gestite dalla rete stessa.

Questo livello può interfacciare reti completamente distinte fra loro.

I protocolli di questo livello definiscono la logica di networking che si utilizza(IP).

Il quarto livello è il livello di trasporto, dove troviamo diversi servizi di trasporto: connection oriented e connectionless. A questo livello possiamo trovare meccanismi di correzione degli errori di trasmissione.

I protocolli di questo livello sono protocolli di trasporto: TCP, implementa servizi di trasporto connection oriented; UDP, implementa servizi di trasporto connectionless.

Finché noi siamo fermi tra livello fisico, data link e network, si parla di livelli "hop-to-hop": mano mano che passiamo per i nodi intermedi c'è una negoziazione fra questi tre livelli, cioè ci sono una serie di salti intermedi dove l'inoltro si ferma al livello di network senza arrivare ai fabbisogni di trasporto end-to-end della comunicazione. Quando poi si arriva alla destinazione, all'host, ritorniamo al livello end-to-end, significa che troveremo un servizio di trasporto di livello end-to-end che mi gestirà la correzione degli errori.

Gli standard che troviamo ai livelli hop-to-hop, soprattutto nei livelli fisico e data link, sono sempre implementati in hardware. A partire dal livello di trasporto in poi, gli standard sono implementati a livello software (de facto).

Il livello di applicazione è quello più vicino all'utente e implementa i servizi applicativi che si interfacciano direttamente con i fabbisogni dell'utente (DNS, http, WWW, FTP, streaming, …).

Il livello di presentazione è un livello di mediazione: si occupa della codifica dei dati che viaggiano sulla rete in formati astratti. Queste informazioni vengono poi riconvertite nel formato proprietario della macchina destinataria. Inoltre, questo livello può gestire anche operazioni di compressione o cifratura di flusso per crittografare l'informazione.

Il livello di sessione è quello che si interfaccia con il livello di trasporto. In esso si stabiliscono le sessioni di dialogo end-to-end fra le macchine: ad esempio, la sincronizzazione di un trasferimento dati in accordo ad uno specifico protocollo, il login di una sessione su un database remoto, l'establishment di una sessione end-to-end verso applicazioni particolari.

Incapsulamento

In ogni livello troviamo la **PDU** (Protocol Data Unit), l'unità di interscambio dati che si utilizza a quel livello. (screen slide 14)

Se devo trasportare i dati dal livello più alto a quello più basso, i passaggi sono i seguenti:

- Si prendono i dati che vengono dai livelli di processo e li passo al livello di trasporto, il quale aggiunge le sue informazioni di controllo (una testata di trasporto). Nel caso in cui utilizzo un servizio TCP, aggiunge una testata TCP che non è altro che un insieme di dati che contiene le informazioni di controllo tipiche del protocollo di quel livello che saranno necessarie, quando arriveranno i dati a destinazione, per l'altro lato end-to-end della comunicazione, al pari livello che dovrà gestire queste informazioni;

- Dal livello di trasporto, questo unico aggregato costituito dai dati più la testata di controllo passano al livello di network. Qui, il livello network aggiunge il suo header di protocollo (informazioni di controllo IP);
- Si passa poi al livello data link, dove avremo un doppio incapsulamento: nella parte alta di questo livello (LLC) aggiungo una serie di informazioni di controllo per la parte della gestione logica del link. Questa volta avrò un header (prefisso) e una coda (suffisso) che è un insieme di bit, denominato FCS (Frame Checking Sequence), che è un codice di controllo per gli errori: facendo un controllo su questa coda posso sapere se la trama mi arriva in maniera corretta o meno;
- Tutto questo blocco aggregato viene incapsulato al livello MAC, livello fisico, dove ci aggiungo le informazioni di controllo di livello MAC e un FCS;
- Tutto questo blocco viene tradotto in bit, codificato in un segnale e trasmesso a livello fisico.

Dall'altro lato end-to-end avviene il **deincapsulamento**: avviene esattamente il processo inverso, ogni livello usa l'header messo dallo stesso livello di prima per assolvere le proprie funzioni.

Questo modello è didattico, anche se è uno standard internazionale, è molto sofisticato. Nella realtà delle reti moderne si fa riferimento ad un modello semplificato che è quello che si usa comunemente, detto **modello ARPANET** (basato sulla suit di protocolli TCP/IP). È molto simile al modello didattico ma c'è una semplificazione nei livelli di processo: vengono compressi in un unico livello detto *livello di processo/applicazione*. I livelli data flow sono gli stessi, cambiano solo alcuni nomi (fisico, net interface, internet layer, trasporto).

Il vantaggio di TCP/IP rispetto a OSI è che è molto più semplice nello stack, cioè ci sono meno livelli.

Il livello fisico

È un livello che realizza l'astrazione del mezzo trasmissivo fisico che utilizziamo per far passare l'informazione; facciamo passare un'informazione trasformando una grandezza fisica in accordo a un *segnale*.

Un **segnale** è la variazione di grandezze fisiche che trasportano informazioni, possono essere di vari tipi: elettrico, acustico, luminoso, elettromagnetico. I segnali elettrici che noi possiamo trasmettere possono essere di due tipi:

- Analogici: è un segnale continuo che varia nel tempo, e nel tempo può assumere tutti i valori possibili fra un massimo e un minimo che sono i limiti del canale di comunicazione;
- Digitali: è un segnale che può assumere solamente due valori (0 e 1), oppure potremmo avere una trasmissione digitale multilivello, cioè un certo numero discreti di valori e non tutti i possibili valori di un range (solo una parte).

I segnali possono essere **periodici** e **aperiodici**: il primo si ripete, uguale a sé stesso, nel tempo in ragione di un determinato intervallo di ripetizione che si chiama *periodo*; il secondo, invece, non ha nessuna regolarità nel tempo, cioè non si ripete.

Alcuni segnali periodici hanno delle caratteristiche particolari, sono *sinusoidali*, evolvono in ragione di un andamento caratterizzato da un trend geometrico che si chiama "sinusoide". Tipicamente questi segnali oscillanti hanno la caratteristica di viaggiare più a lungo di una corrente diretta, quindi è comodo utilizzarli

nelle telecomunicazioni. Questi segnali oscillanti si differenziano dal periodo di oscillazione e dall'ampiezza di picco.

Il **periodo** è il tempo necessario al segnale affinché si ripeta, mentre la **frequenza** (inverso del periodo) è il numero di ripetizioni nell'unità di tempo. (f = 1/T, T = 1/f)

La frequenza è una proprietà direttamente collegata alla velocità di trasmissione, infatti, rappresenta la velocità con cui un segnale cambia nel tempo. Cambiamenti frequenti implicano una frequenza alta, cambiamenti lenti implicano una frequenza bassa.

Un'altra proprietà dei segnali sinusoidali è la **fase angolare**: rappresenta la posizione angolare del segnale rispetto al tempo 0.

Altra proprietà dei segnali sinusoidali è la **lunghezza d'onda**: è una relazione fra il periodo (o la frequenza) con la velocità di propagazione (c) del segnale sul mezzo. Quindi è funzione del mezzo oltre che del segnale. La lunghezza d'onda rappresenta la distanza che un ciclo del segnale occupa sul mezzo trasmissivo (è la distanza fra due creste). $\boldsymbol{\lambda} = \mathrm{c} * \mathrm{T} = \mathrm{c} / \mathrm{f}$

I segnali digitali assumono valori discreti (es. -5V, +5V) ed è facile rappresentare dei bit con essi: in generale, se il segnale ha L livelli si possono facilmente rappresentare $\log_2 L$ bit. Questi valori discreti si ottengono facendo variare, nel modo quanto più brusco possibile, il valore del segnale da un livello all'altro. (nella realtà non esiste questo cambiamento così brusco, infatti si approssima il cambiamento analogico per avere un calo o una salita drastica)

Trasmissione di segnali

Il termine **trasmissione**, nel campo delle telecomunicazioni e dell'informatica, indica il processo e le modalità/tecniche finalizzate all'invio di informazione, tramite impulsi elettrici e segnali codificati, su un canale fisico di comunicazione da un mittente ad uno o più destinatari.

Quando si invia un segnale, ad esempio elettrico, lungo un canale di comunicazione, esso viene ricevuto diverso dalla sua configurazione di partenza, infatti l'energia elettrica tende a dissiparsi con la distanza. Ciò può comportare un'interpretazione sbagliata o una perdita del segnale. I principali fenomeni che caratterizzano il deterioramento del segnale in trasmissione sono: **attenuazione** e **distorsione**.

Quando si parla di fenomeni attenuativi significa che il segnale parte con una certa potenza e man mano che viaggia, il segnale si attenua conservando lo stesso andamento ma diminuendo le ampiezze d'onda. Il fenomeno di attenuazione si misura in decibel: $10*\log_{10}$(potenza del segnale al punto d'arrivo) / potenza del segnale al punto di partenza.

Il fenomeno di attenuazione si può risolvere utilizzando dei dispositivi di amplificazione del segnale che si chiamano appunto *amplificatori*. Il problema è che introduce, ad ogni passo di amplificazione, degli errori; quindi, più amplificazioni faccio e più il segnale si distorce. Questo problema lo si può risolvere con la *rigenerazione*.

Quando si parla di fenomeni distorsivi significa che il segnale, non solo perde potenza, ma potrebbe anche cambiare forma, ma anche sfasare le componenti sinusoidali.

Serie di Fourier

Jean-Baptiste Fourier ha dimostrato che un segnale periodico si può sviluppare in una serie, che si chiama **serie di Fourier**, che è costituita in un termine costante (c) e da una somma di infinite sinusoidi caratterizzate da ampiezze particolari e da frequenze che sono multiple di una frequenza fondamentale. (vedere slide 16)

Questa serie mi permette di rappresentare il segnale in base alle sue componenti in frequenza: proietto il segnale dal dominio del tempo al dominio della frequenza. Dunque, un segnale variabile nel tempo è di fatto equivalente ad una somma di funzioni sinusoidali aventi ciascuna una propria ampiezza e frequenza. Si può quindi rappresentare un segnale g(T) di durata T in un modo diverso, e cioè attraverso il suo "spettro di frequenze", ossia attraverso la sua scomposizione in sinusoidi. Qualunque segnale è dunque caratterizzato da un intervallo di frequenze nel quale sono comprese le frequenze delle sinusoidi che lo descrivono. Esso va sotto il nome di **banda di frequenza** (frequency band) del segnale.

Tutto questo ci serve per definire le componenti in frequenza del segnale. Queste componenti si chiamano *armoniche* e caratterizzano il comportamento, in termini di variazioni, del segnale in frequenza.

Questa cosa ci interessa molto perché i quadrati delle ampiezze sono proporzionali all'energia che trasmettiamo ad una specifica frequenza. Dato che nel mezzo trasmissivo noi perdiamo parte delle energie, se tutte le componenti si attenuassero nello stesso modo allora avremmo solo una riduzione in ampiezza ma in realtà il segnale viene distorto.

I mezzi fisici sono caratterizzati da una proprietà particolare che si chiama **banda passante**: il mezzo fisico si comporta come un filtro passa-banda che taglia tutte le frequenze fuori da un certo intervallo (delimitato da "frequenze di taglio"). È definita pertanto *larghezza di banda* (B) l'insieme delle frequenze che un canale di telecomunicazioni fa passare.

Perché un segnale sia ricevuto come è stato trasmesso, è necessario che la banda passante sia uguale o più ampia della banda di frequenza del segnale stesso; altrimenti, il segnale viene privato di alcune delle sue armoniche, e viene quindi distorto, cioè alterato. Se un numero sufficiente di armoniche arriva a destinazione, il segnale è comunque utilizzabile. (più armoniche perde il segnale, più il segnale viene alterato)

Ricostruzione dei segnali – segnali campionati e quantizzati

Per utilizzare informazioni digitali noi abbiamo bisogno di appunti segnali digitali che, nella realtà, non sono altro che approssimazioni di improvvisi cambi di valori di una funzione continua.

Per trasformare un segnale continuo in uno "discreto", lo devo **campionare**, cioè ne faccio una lettura ogni specifico intervallo di tempo: il campionamento deve avvenire in maniera uniforme, cioè utilizzando intervalli di tempo regolari.

Prendendo un qualsiasi segnale, lo leggo ogni Δt e ne ottengo un certo numero di campioni che lo rappresentano. Il valore che leggo di questo segnale sarà un numero reale definito con continuità in un certo intervallo di valori, ma dato che questo valore lo devo rappresentare su uno strumento digitale, in dipendenza dalla rappresentazione che vado ad usare, devo rendere discreti anche i valori: quindi oltre al campionamento devo fare anche una **quantizzazione**, cioè trasformare i valori reali in valori discreti (ad esempio con un filtro a soglia).

Teorema del campionamento

Conoscendo il mezzo trasmissivo, a che frequenza lo devo campionare per poterlo ricostruire a partire dal segnale campionato?

Nel 1924 H. Nyquist dimostrò che un **segnale a larghezza di banda B può essere ricostruito perfettamente campionando lo stesso al doppio della larghezza di banda del segnale stesso.**

In generale la frequenza di campionamento dovrà essere almeno leggermente superiore a 2B, per disporre di un intervallo utile ("banda di guardia") al fine di prevenire tagli del segnale alle estremità dell'intervallo di banda.

Quindi il segnale s(t) può essere ricostruito a partire dai suoi campioni presi a frequenza $f_s = 1/\Delta t$.

Con l'ausilio di questa relazione, Nyquist, riuscì a stabilire che considerando di usare un numero V di livelli trasmissivi equiprobabili, dato che la quantità di informazione associata è esprimibile come $Q = \log_2 V$, allora la massima quantità di informazione trasmessa in un canale non rumoroso, dato un segnale costituito da V livelli, è di $2BQ$, cioè: $I[bit/s] = 2B \log_2 V$.

Dove 2B è la banda del canale e non del segnale.

Quindi si potrebbe pensare di aumentare sempre di più i livelli trasmissivi così da avere più velocità di banda da parte del mezzo trasmissivo, in realtà non è così. Mano mano che aumento il numero di livelli trasmissivi diventa sempre più difficile riconoscere un livello rispetto a un altro in fase di quantizzazione. L'altro problema è che il canale, oltre all'attenuazione e alla distorsione, presenta anche un altro problema che è chiamato **rumore** che rende piuttosto difficile l'interpretazione del valore che io vado a leggere in fase di quantizzazione ad un certo livello. Aumentando il numero dei livelli di tensione, è possibile aumentare la quantità di informazione che va a destinazione nello stesso tempo. Ma aumentare il numero di livelli, a parità di tensione massima, comporta che il singolo livello diventa sempre più piccolo, finché in ricezione non sia più distinguibile dal rumore. Quindi significa che la massima capacità del canale trasmissivo non può dipendere dal numero di livelli. Esiste comunque un limite massimo all'aumento dei livelli, definito analiticamente da una formula determinata da **C. Shannon**.

N1l 1948 C. Shannon estese il lavoro di Nyquist a canali soggetti a rumore casuale (termico). Se indichiamo con S la potenza del segnale e con N la potenza del rumore, la massima informazione trasmessa (massima capacità del canale di banda B) è: $I[bit/s] = B \log_2(1 + S/N)$.

Nella formula di Shannon non ci sono più i livelli trasmissivi, ciò vuol dire che indipendentemente dal numero di livelli trasmissivi, non si può andare oltre a livello di capacità trasmissiva.

Secondo la relazione vista, sembrerebbe possibile aumentare il tasso di trasferimento dati aumentando il livello del segnale. Questo è vero, ma l'aumento del livello del segnale comporta l'aumento di effetti indesiderati, come la non linearità, che vanno ad accrescere il tasso di errore di ricezione. Quindi effettivamente la limitazione di banda costituisce un limite alla velocità di trasferimento dei bit.

A parità di mezzo utilizzato, tanto più è corto il canale di trasmissione tanto più è alto il numero di bit/s raggiungibili.

Il rumore

Il rumore è una forma di energia indesiderata che si somma al segnale utile degradandone il contenuto informativo, ed impedendo così di rilevare, in ricezione, tutto l'insieme delle informazioni trasmesse. Il rumore si misura in **SNR**: rapporto segnale-rumore.

$$SNR = (\text{potenza segnale})/(\text{potenza rumore})$$

$$SNR_{dB} = 10\log_{10} SNR$$

Esistono vari tipi di rumore:

- **Rumore bianco**: forma di rumore il cui spettro comprende energia a tutte le frequenze dello spettro elettromagnetico ed equamente distribuita;
- **Rumore di intermodulazione**: rumore prodotto dalla non linearità dei dispositivi elettronici e che consiste nella presenza, nel segnale in uscita dal dispositivo, di armoniche indesiderate non presenti nel segnale in ingresso;
- **Rumore di modo comune** (o di modo normale): rumore presente in ingresso ad uno strumento di misura insieme al segnale da misurare e non separabile da questo;
- **Rumore di quantizzazione**: perdita di informazione che ha luogo durante la trasformazione di un segnale analogico in digitale, ad esempio nel P.C.M.;
- **Rumore termico**: rumore dovuto all'agitazione termica degli elettroni presenti in una resistenza. È funzione della temperatura ma è anche un rumore bianco.

Prestazioni di una rete

Le prestazioni di un canale di comunicazione di una rete si misurano partendo da delle grandezze caratterizzanti esso stessi che sono:

- la **larghezza di banda** è il range di frequenze che il mezzo trasmissivo fa passare. La posso misurare in Hz (per misurare la proprietà trasmissiva del mezzo) oppure in bps (per misurare la larghezza di banda effettiva dipendente dalla tecnologia di trasmissione, velocità);
- il **throughput** misura quanto velocemente possiamo spedire i dati su una rete. È diverso dalla larghezza di banda perché il throughput è una misura effettiva;

- la **latenza** misura quanto tempo occorre a trasferire un intero messaggio dall'origine alla destinazione. Il tempo viene misurato da quando parte il primo bit fino all'ultimo bit arrivato. La latenza non è un tempo solo, è la composizione di tanti tempi:
    - *tempo di trasmissione*: tempo per immettere i dati sul mezzo. Dipende dal tipo di trasmissione che si utilizza;

(T_t = dimensione dati/larghezza di banda)

- *tempo di propagazione*: tempo necessario al segnale per propagarsi sul mezzo trasmissivo;
  (T_p = distanza/velocità di propagazione del segnale)

- *tempo di attesa* e *tempo di inoltro*: dipendono dalla caratteristica dei nodi intermedi. È il tempo necessario al nodo intermedio per smistare il messaggio.

- il **prodotto banda-ritardo** è il prodotto della larghezza di banda per la latenza. Questo prodotto ci fa capire come il canale si comporta dopo un certo tempo. Ci serve a capire qual è il numero di bit da trasmettere per saturare la capacità del canale;

- il **jitter** è una varianza del ritardo. È un parametro importante perché mi condiziona molto negativamente l'effetto che ottengo nelle trasmissioni multimediali.

Caratteristiche dei sistemi di trasmissione dei segnali sui mezzi trasmissivi

La trasmissione dei segnali può avvenire, ovviamente, in maniera analogica o digitale: la trasmissione analogica non tiene cura del significato del segnale trasmesso, in questo caso la trasmissione si limita a recapitare il segnale, eventualmente amplificandolo in intensità quando necessario; invece, la trasmissione digitale tiene conto del contenuto dei dati se si deve intervenire per amplificare il segnale: il segnale non viene semplicemente amplificato, ma viene interpretato, si estrae il contenuto informativo e si rigenera il segnale tramite apparati detti "ripetitori". I vantaggi della trasmissione digitale sono: l'**immunità maggiore** all'alterazione dei dati verso lunghe distanze; l'**omogeneizzazione** della trasmissione per **diverse tipologie** di dato; **sicurezza** e riservatezza. Gli svantaggi, invece, sono: costi elevati; maggiore complessità dell'elettronica; richiede il rinnovo di infrastrutture già esistenti.

Si può trasmettere in due modi:

- **trasmissione in parallelo**: ho più canali che uso simultaneamente;
- **trasmissione in serie**: ho una sola via di trasmissione dove devo leggere i dati con un certo ordine e due convertitori uno parallelo/seriale e l'altro seriale/parallelo.

La trasmissione, inoltre, può avvenire in **banda base** o in **banda modulata**: la prima permette di utilizzare tutta la capacità che il canale mi offre per trasmettere il segnale (digitale) dopo averlo codificato; invece, la seconda permette di trasformare il segnale traslandolo di banda utilizzando un segnale sinusoidale ausiliario, cioè sposto il segnale da trasmettere su una banda differente.

Codifica dei dati numerici

La rappresentazione di dati numerici con segnali numerici è normalmente fatta tramite sequenze di impulsi discreti di tensione di una certa durata temporale. Il dato binario è **codificato** in modo d far corrispondere al valore di un bit un determinato livello del segnale. Il ricevitore deve sapere quando inizia e finisce il bit,

leggere il valore del segnale al momento giusto e determinare il valore del bit in base alla codifica utilizzata; quindi, trasmettitore e ricevitore si devono sincronizzare. La migliore valutazione si ottiene campionando il segnale al tempo corrispondente a metà bit.

Una **codifica di linea** è un insieme di regole che ci permette di associare i valori dei bit agli elementi del segnale. L'"elemento dei dati" è un bit che può valere 0 o 1; l'"elemento del segnale" è un valore che posso utilizzare per rappresentare il segnale. Il rapporto tra elementi di segnale ed elementi di dati costituisce il numero di bit che un elemento di segnale può rappresentare.

La "velocità dei dati" (bps) sarà il numero di bit che posso spedire in 1 s; la "velocità del segnale" (baud) è il numero di elementi di segnale che posso spedire in 1 s.

Sono possibili diverse scelte di codifica, con caratteristiche differenti che possono migliorare le prestazioni della trasmissione. Le caratteristiche determinanti sono:

- **spettro del segnale**:
    - componenti ad alta frequenza richiedono una banda maggiore;
    - l'assenza di componente continua è preferibile;
    - spettro concentrato nel centro della banda;

- **sincronizzazione temporale**: il ricevitore deve essere sincronizzato con il trasmettitore per identificare i bit (devono capire quando inizia e quando finisce un bit);

- **rilevazione di errori**;
- **solidità del segnale** rispetto ad interferenza o rumore;
- **costo** e **complessità** di realizzazione.

La **codifica unipolare RZ** (Return to Zero) prevede la trasmissione di un segnale di lunghezza T per ogni bit. Il segnale è nullo in corrispondenza del bit 0, mentre è un impulso di tensione di durata T/2 per il bit 1. Una variante di questa codifica è la cosiddetta **codifica unipolare NRZ** (Non Return to Zero) differisce dalla RZ perché il livello di tensione per il bit 1 rimane alto per tutta la durata del bit. Quest'ultima ha come pregi: una facile progettazione e realizzazione, ed un utilizzo efficiente della larghezza di banda. I difetti, invece, sono: che se trasmetto tutti 0 o tutti 1 avrò una componente continua, se ho tutti 0 è peggio perché non saprò se c'è una trasmissione o non c'è nulla; le lunghe sequenze di bit di uguale valore producono un segnale continuo senza transizioni e questo causa una perdita di sincronia tra il trasmettitore e il ricevitore.

Un'alternativa è la **codifica NRZ-L** che prevede un segnale a +V per il bit 0, ed a -V per il bit 1 o viceversa. Questo riduce l'impatto della componente continua, ma non la annulla.

Altra tecnica è la **codifica differenziale NRZI**: il segnale rimane costante quando sto trasmettendo bit 0 e cambia quando trasmetto bit 1.

Si può avere anche una codifica a più livelli come la **codifica AMI** (Alternate Mark Inversion) che utilizza tre livelli: lo zero indica il bit 0, mentre il bit 1 è identificato con segnali a +V e -V alternati. Potrei utilizzare anche una **codifica pseudoternaria** che è la stessa, con 1 e 0 invertiti.

La codifica AMI ha i seguenti vantaggi rispetto alla NRZ:

- risolve il problema della sequenza di bit 1, che presentano sempre una transizione utilizzabile in ricezione per sincronizzare (ma resta il problema per sequenze di 0);
- la componente continua è di fatto azzerata;
- utilizza, a parità di transmission rate, una larghezza di banda inferiore;
- errori isolati possono essere evidenziati come violazione del codice (nel caso in cui nella trasmissione di più 1 di seguito, se non si alternano +V e -V posso accorgermi di una violazione del codice e quindi di un errore di trasmissione.

Vi sono anche svantaggi:

- utilizza 3 livelli, quindi ogni simbolo potrebbe trasportare più informazione ($\log_2 3 = 1.58$) sprecandola;
- a parità di bit rate richiede circa 3dB in più rispetto alla NRZ.

Esiste un altro tipo di codifica, la **codifica Manchester** che è molto costosa in termini di frequenza, perché per ogni bit utilizza due livelli di tensione: il bit 1 è rappresentato da un segnale -V per mezzo periodo, +V per il seguente mezzo periodo; il bit 0 è rappresentato in modo opposto. La versione **differenziale** della codifica Manchester funziona con la stessa logica però rappresenta il bit 1 come variazione di codifica rispetto alla codifica del bit precedente. I vantaggi della codifica Manchester sono:

- **auto-sincronizzazione**: ogni bit ha una transizione in mezzo, che può essere utilizzata per la sincronizzazione dal ricevitore;
- totale assenza della **componente continua**;
- **rivelazione di errori**: ogni bit deve prevedere una transizione nel mezzo quindi posso accorgermi facilmente si un eventuale errore di codifica.

Visto che c'è una transizione per ogni bit, quindi due elementi di segnale per un elemento di dato, è più costosa come codifica perché richiedere un segnale a frequenza doppia rispetto al bit rate: 1 bit richiede 2 baud, quindi richiede una banda doppia.

La modulazione

La **modulazione** è un processo secondo il quale il segnale da trasmettere (segnale **modulante**) viene utilizzato per far variare nel tempo le caratteristiche di un segnale ausiliario sinusoidale caratterizzato da un range di frequenze (segnale **portante**). Questa operazione trasforma il segnale modulante in un segnale sinusoidale incentrato su un determinato range di frequenze. Quindi mi permette, sulla base della scelta del segnale ausiliario che faccio, di spostare la trasmissione su un range di frequenze che voglio. Non utilizzo più l'intero canale ma un range di frequenze che mi mette a disposizione il segnale portante. Quindi avrò un segnale cha ha un'occupazione di banda che ha lo stesso ordine di grandezza del segnale modulante, ma è centrato intorno alla frequenza del segnale portante.

Con una trasmissione a banda modulata potrei trasmettere più segnali simultaneamente se uso portanti che sono definiti in range di frequenze che non si sovrappongono fra loro; oppure posso spostare il mio segnale al centro del range di frequenze che il mezzo trasmissivo mi fa passare (avendo meno problemi di trasmissione dovuti alle caratteristiche del mezzo).

Tecniche di modulazione

A priori c'è sempre una conversione del segnale digitale/analogico per poi modulare il segnale: devo modulare la portante facendola variare in ragione del comportamento del segnale modulante (che è quello che voglio trasmettere). Posso far variare il segnale in vari modi:

- *ampiezza* (Modulazione ASK, Amplitude Shift Keying): il segnale viene utilizzato per modificare il valore dell'ampiezza della portante;
- *frequenza* (Modulazione FSK, Frequency Shift Keying): il segnale modulante modifica istante per istante la frequenza della portante;
- *fase* (Modulazione PSK, Phase Shift Keying): il segnale modulante cambia la fase della portante.
- posso creare anche una modulazione ibrida che mette insieme la modulazione PSK e la modulazione ASK, creando la *modulazione QAM* (Quadrature Amplitude Modulation).

Nella modulazione PSK bifase (BPSK) si ha una sola portante e quindi i due valori numerici 1 e 0 sono fatti corrispondere a due fasi diverse della stessa frequenza: 0° e 180°. Possiamo fare certamente meglio utilizzando **modulazioni polifase**:

- si ottiene una migliore **efficienza** del **canale** perché ad ogni elemento di segnale (quindi una fase) faccio corrispondere due elementi di dati (quindi trasporto più bit);
- la modulazione polifase si realizza effettuando una codifica preliminare dei bit provenienti dal terminale, raggruppandoli in parole di *n* bit e facendo corrispondere a ciascuna delle 2^n parole possibili, una determinata fase della frequenza portante.

Nella modulazione QPSK (Quadrature PSK) si utilizzano quattro angoli di fase per trasmettere due bit per simbolo. Ovviamente non si può aumentare indefinitamente il numero di bit per fase perché entrano in gioco fattori di distorsione o rotazione del segnale, detto anche **phase jitter**. Ovviamente piuttosto che lavorare in maniera assoluta posso lavorare anche in logica differenziale, quindi modulare la fase in ragione delle variazioni dei bit.

QAM

Visto che non si può aumentare troppo il numero di fasi, posso combinare la modulazione PSK con la modulazione ASK. Per aumentare la velocità di trasmissione, mantenendo costante la velocità di modulazione, invece di trasmettere solo due valori angolari, 0° e 180°, si trasmette un maggior numero di angoli diversi fra loro, e per consentire una più facile demodulazione in ricezione, si fa variare anche l'ampiezza del segnale modulato. Queste tecniche di combinazione di ASK e PSK si chiamano **costellazioni**.

Lezione 4

Multiplexing

Supponiamo di avere un canale a 10 Mbps e dobbiamo spedire dati a 3 Mbps, se dedichiamo l'intero canale ad una singola comunicazione stiamo sprecando 7 Mbps. Allora potrei pensare di trasportare insieme 3 comunicazioni da 3 Mbps: questa logica di condividere il mezzo trasmissivo per farci viaggiare più flussi insieme si chiama **multiplexing**.

Al canale fisico si interfaccia un multiplexer da un lato e un demultiplexer dall'altro: al multiplexer entrano *n* canali logici che vengono messi insieme su un unico canale fisico; dal demultiplexer entra un unico canale fisico che poi viene spacchettato in *n* canali logici.

Esistono diverse **tecniche di multiplazione**:

- a **divisione di tempo** (TDM): possiamo lavorare sia in maniera deterministica che in maniera statistica;
- a **divisione di frequenza** (FDM e WDM): usa differenti frequenze o lunghezze d'onda per differenziare i dati trasmessi;
- per **codifica** (CDM): la differenziazione dei dati trasportati è ottenuta utilizzando diversi tipi di codifica.

TDM

Il multiplexing a divisione di tempo è utilizzato quando si dispone di un canale digitale capace di un elevato tasso di trasmissione dati in cui poter trasmettere contemporaneamente un insieme di comunicazioni a tasso inferiore. Quindi significa che mettiamo insieme le varie comunicazioni e procedono simultaneamente.

Di fatto, quando lavoro in logica TDM, partiziono la capacità del canale in intervalli temporali chiamati **time slot**, e questi intervalli temporali costituiranno dei canali logici. Il mezzo per quell'intervallo temporale è di proprietà del canale che si deve trasmettere (?). In poche parole, c'è proprietà assoluta per il time slot assegnato, sia che venga utilizzato che non. Nel time slot posso trasmettere tipicamente un insieme strutturato di bit (trama).

Esistono essenzialmente due metodi di multiplazione:

- **deterministico**: ogni canale di comunicazione è identificato dalla sua posizione in termini di slot temporali all'interno della trama. Questa correlazione fissa fra il canale di comunicazione e il relativo time slot è il principale svantaggio del TDM deterministico: se il canale non è usato, comunque occupa il time slot.
  Le trame hanno la stessa taglia e ciò implica che trasmettitore e ricevitore devono essere perfettamente sincronizzati per sapere dove comincia e dove finisce una trama. D'altra parte, non sono necessari schemi di indirizzamento né di bufferizzazione.

- **statistico**: non esiste una correlazione fra canale di comunicazione e relativo time slot. La capacità del mezzo è distribuita statisticamente fra gli utenti che ne concorrono all'uso. È necessario uno schema separato di tramatura e indirizzamento per garantire le associazioni

dinamiche: se un canale non è usato, gli altri canali possono disporre della sua capacità trasmissiva.
Le trame possono essere di taglia differente e quindi avrò bisogno di un meccanismo di indirizzamento e uno di bufferizzazione. La TDM statistica è fortemente dipendente dal tipo di protocollo che si utilizza. Però, a differenza del TDM deterministico utilizzerò in modo ottimale il mezzo trasmissivo.

FDM

Il mezzo trasmissivo si comporta come un filtro passa-banda, dove al centro di esso non ci saranno errori di trasmissione, a differenza alle estremità di esso potremmo avere dei tagli di frequenze.

Posso utilizzare il range di frequenze che il mezzo trasmissivo mi mette a disposizione, dividendo la sezione spettrale in tante bande separate, ognuna dedicata a ciascun canale; l'importante p che nessuna coppia di canali può condividere la stessa porzione di spettro. (potrò spostare le frequenze del segnale nel range del mezzo grazie alla modulazione). In questo modo potrò trasmettere tanti segnali, sul canale trasmissivo, contemporaneamente.

Per ovviare ai fenomeni di interferenza fra canali si tende a dislocare i range di frequenze, assegnati ai diversi canali, in modo da riservare uno spazio fra essi.

Si parla di **WDM** quando lavoriamo su fibre ottiche e destiniamo ai canali lunghezze d'onda differenti.

Multi-Carrier Modulation

Nella trasmissione parallela di dati, la banda di frequenza disponibile è generalmente suddivisa in molti canali. Allo scopo di eliminare l'interferenza intercanale, gli spettri dei sottocanali non devono sovrapporsi; questo però non consente un utilizzo efficiente della banda disponibile. La sovrapposizione spettrale può essere permessa a patto di sfruttare relazioni di ortogonalità tra i canali.

Nella modulazione multi-portante (MCM) il flusso di dati è diviso in più sottoflussi (substream), ognuno dei quali ha un bit rate molto più basso e ogni substream è usato per modulare una diversa portante (carrier). Utilizzando portanti *ortogonali* (**OFDM**) non si ha l'interferenza intercanale.

Nelle tecniche di modulazione multiportante:

- la trasmissione delle informazioni non avviene più attraverso un unico flusso supportato da una sola portante, bensì suddividendo il flusso dati in appositi sottoflussi tutti paralleli tra loro, detti anche sottocanali, ciascuno dei quali con una propria specifica sottoportante.
- La velocità di trasmissione dei dati di un singolo sottocanale risulta inferiore, rispetto al caso monoportante, e la banda necessaria è minore rispetto alla banda di coerenza del canale.

- Quando una portante è al suo massimo e le altre no, allora le posso overlappare così da avere molti più canali trasmessi nello stesso spazio. (velocità inferiore ma multi trasmissione)

Code Division Multiplexing

La logica di questa multiplazione è multiplare più trasmissioni sullo stesso portante (mezzo trasmissivo).

Il segnale associato ad ogni canale, lo codifico con una particolare parola di codice costituita da un insieme di bit/di valori, rappresentati come "alto" e "basso". Ogni bit verrà moltiplicato per la parola di codice ottenendo un effetto di **allargamento** di quello che trasmetto. L'output della moltiplicazione lo vado a modulare e lo trasmetto sul canale: questo lo faccio per tutti i canali simultaneamente.

In ricezione il segnale ottenuto è la somma vettoriale (comprensiva di modulo e di fase) di tutti i segnali trasmessi dalle singole sorgenti di informazione, con in più un eventuale termine dovuto al rumore del canale.

Il codificatore CDM è costituito principalmente da due parti, la prima che divide la sequenza di bit generata da un codificatore in *k* repliche, e la seconda parte che moltiplica ogni replica generata per il chipping code di lunghezza *k* anch'esso. Ogni bit del chipping code (es -1 e +1) viene moltiplicato con le repliche della sequenza iniziale in modo da generare un codice in base alla sequenza di partenza. In ricezione, quindi, il segnale potrà essere decodificato soltanto da chi avrà il codice di canalizzazione esatto.

Mezzi trasmissivi

Un mezzo trasmissivo è un mezzo fisico che trasporta i segnali. Li possiamo distinguere in due categorie: **mezzi guidati** (elettrici, ottici) e **mezzi non guidati** (onde radio, laser via etere, suono, raggi X). I mezzi guidati sono quelli in cui il mezzo si comporta come una guida d'onda (ogni segnale è un'onda, quindi il segnale segue il mezzo trasmissivo); invece, dove si lavora in assenza di una guida d'onda si parla di mezzi trasmissivi non guidati.

In ogni caso, ogni mezzo trasmissivo è caratterizzato da: larghezza di banda, delay, costo e facilità di installazione e manutenzione.

(mezzi trasmissivi guidati)

I mezzi trasmissivi più comuni sono quelli **elettrici** che non sono altro che conduttori che devono trasportare un segnale in forma di energia elettrica. È necessario che le caratteristiche del mezzo siano tali da rendere massima la trasmissione dell'energia da un estremo all'altro e minima la dissipazione in altre forme (ad esempio calore, irradiazione elettromagnetica).

Il mezzo elettrico è formato da: un *generatore* che deve mantenere la differenza di potenziale fra i due poli; un *carico* che assorbe potenza; due *conduttori* caratterizzati da una serie di proprietà: resistenza, tendenza di un corpo ad opporsi al passaggio di corrente elettrica; capacità, attitudine

di un corpo conduttore ad accumulare carica elettrica; induttanza, proprietà dei circuiti elettrici tali per cui la corrente che li attraversa induce una forza elettromotrice; conduttanza, attitudine di un conduttore ad essere percorso da corrente elettrica.

Un mezzo trasmissivo elettrico si dice **ideale** quando trasporta tutta l'energia del segnale trasmesso senza attenuazione né distorsione: però un mezzo del genere non esiste. Però possiamo avere un mezzo trasmissivo elettrico **ottimale** che è caratterizzato da bassa resistenza, bassa capacità e bassa induttanza, cioè un mezzo poco dispersivo e poco dissipativo. In tale mezzo quasi tutta la potenza inviata sul canale dal trasmettitore arriva al ricevitore ed il segnale non viene distorto.

In più, i mezzi trasmissivi elettrici si comportano come "antenne", cioè un conduttore quando viene attraversato dall'energia elettrica, non solo dissipa calore, ma dissipa una radiazione esterna/disturbo elettromagnetico che va ad influenzare l'ambiente che lo circonda. Quindi un mezzo trasmissivo elettrico non solo dissipa ma viene anche influenzato dagli altri mezzi trasmissivi elettrici che ci viaggiano vicino, il che significa che questi mezzi saranno caratterizzati anche da fenomeni di interferenza. Per evitare l'interferenza si ricorre a meccanismi di ***schermatura***: i mezzi sono protetti da uno schermo realizzato in maniera tale da ridurre la sensibilità del mezzo ai disturbi elettromagnetici migliorando anche le caratteristiche trasmissive del cavo stesso.

I sistemi di schermatura possono essere:

- Un **foglio** di alluminio che circonda il singolo conduttore oppure tutta la coppia di conduttori. Poiché l'alluminio presenta elevata resistenza elettrica rispetto al rame, e, a spessori così ridotti, una notevole fragilità, lungo il foglio scorre un filo di rame nudo, detto *drain*, che garantisce continuità elettrica anche in caso di eventuali crepe; tale filo è utilizzato per il collegamento di terra;
- Una **calza**: treccia di fili di rame che avvolgono il cavo in due direzioni opposte. Presenta una conducibilità molto migliore del foglio di alluminio, ma la copertura non è completa, in quanto in corrispondenza degli intrecci rimangono inevitabilmente dei fori nello schermo.

I risultati migliori si ottengono dalla combinazione di più schermi diversi.

Tipi di mezzi trasmissivi

Il mezzo più comune, con un costo molto limitato, che si utilizza per la trasmissione elettrica è il **doppino**, costituito da una coppia di fili di rame schermati e "binati", attorcigliati a treccia in accordo a un processo che si chiama *binatura*. Inizialmente è nato per la telefonia, quindi doveva trasmettere solo la voce, attualmente i doppini possono raggiungere velocità tra i 10 e 1000 Mb/s, sempre in base ai materiali utilizzati per costruirlo.

La **binatura** serve a ridurre i disturbi elettromagnetici. Quando abbiamo cavi che ospitano più doppini, ogni doppino è binato a un passo di binatura diversi rispetto agli altri. Lo scopo di questo è quello di ridurre un fenomeno di interferenza, fra un doppino e l'altro, che si chiama **diafonia**. La diafonia è un fenomeno di accoppiamento elettrico tra mezzi trasmissivi vicini non isolati adeguatamente ("effetto antenna" dei cavi).

I punti di forza del doppino sono: il costo molto limitato e ha buone caratteristiche trasmissive, ma soprattutto è molto facile da installare e da manutenere.

A livello0 industriale i doppini vengono prodotti in vari formati, dipendenti dall'uso: il più comune è quello a 4 coppie che può essere **STP**, schermato con il foglio; **FTP**, schermato con la calza/treccia; **UTP**, non schermato.

Sulla base dei parametri operativi del cavo che vado ad usare per realizzare i doppini, potrò cambiare le caratteristiche costruttive e le capacità trasmissive del mezzo che sto usando. P stata creata una classificazione che prevede 7 categorie, in base alle applicazioni per le quali i cavi sono idonei. (ogni categoria è idonea a fornire tutti i servizi offerti da quelle inferiori).

- La **categoria 1** (Telecommunication) comprende i cavi adatti unicamente alla telefonia analogica.
- La **categoria 2** (Low Speed Data) comprende i cavi per la telefonia analogica e digitale (ISDN) e trasmissione dati a bassa velocità. (non più in uso)
- La **categoria 3** (High Speed Data) è la prima categoria di cavi adatti a realizzare reti locali fino a 10 Mb/s. (non più in uso)
- La **categoria 4** (Low Loss, High Performance Data) comprende i cavi per LAN Token-Ring fino a 16 Mb/s. (non più in uso)
- La **categoria 5** (Low Loss, Extended Frequency, High Performance Data) comprende cavi per applicazioni fino a 100 Mb/s, su distanze di 100 metri;
- La **categoria 6** (Low Loss, High Frequency, High Performance Data) comprende I migliori cavi disponibili per applicazioni fino a 1000 Mb/s, su distanze di 100 metri;
- La **categoria 7** (ISO/IEC 11801 Class F), nome informale. Lo standard specifica 4 STP all'interno di un unico cavo, per velocità fino a 10 Gb/s.

Se si vuole fare una trasmissione di maggiore qualità si deve ricorrere al **cavo coassiale**. Il coassiale a banda base consiste in un'anima trasmissiva che è un filo di rame, un isolante, uno schermo di metallo e un altro strato isolante. A differenza del doppino, è un cavo utilizzato per trasmettere a velocità molto elevate (300 MHz fino a lunghezze di 100km) e, grazie alla schermatura, si presta ad un collegamento tra dispositivi dove si necessita grande sensibilità (es. antenne). Esistono anche i cavi coassiali binati (twinax).

È un cavo molto costoso ma è estremamente migliore del doppino in termini di capacità trasmissive, sia per la distanza che può percorrere che per l'immunità a fenomeni interferenziali.

Un altro modo di trasmettere è la **trasmissione power line** (o a onde convogliate): è una tecnologia per la trasmissione dati che utilizza la rete di alimentazione elettrica come mezzo trasmissivo. Se realizza sovrapponendo al trasporto di corrente elettrica, continua o alternata a bassa frequenza (50/60 Hz) e un segnale a frequenza più elevata che è modulato dall'informazione da trasmettere. La separazione dei due tipi di correnti si effettua grazie al filtraggio e la separazione degli intervalli di frequenze utilizzate.

La **fibra ottica** ha un'anima trasparente di silicio (*core*) avvolto da un rivestimento di vetro (*cladding*) con indice di rifrazione diverso. Tutta la parte in vetro è ricoperta da una guaina di plastica nera (*isolante ottico*) che non fa passare segnali luminosi. Sulla fibra ottica viaggiano dei fasci di fotoni, fasci di luce che non sono altro che lunghezze d'onda che rappresentano colori in base alla frequenza che ha. Le fibre sono normalmente raggruppate insieme e schermate sia individualmente che globalmente.

Le caratteristiche principali della fibra ottica sono:

- Garantisce meno latenza possibile;
- Ha una banda passante molto grande (alcune decine di THz);
- È immune ai disturbi elettromagnetici;
- È leggera e flessibile;
- Ha un costo inferiore rispetto ai mezzi metallici;
- È molto più sicura rispetto ai mezzi metallici. Visto che il rame produce una radiazione esterna, questa radiazione può essere utilizzata per intercettare parte del segnale; invece con la fibra, per intercettare il segnale, devo aprire lo schermo e utilizzare un rilevatore ottico;
- È molto difficile da produrre perché richiede una catena di lavorazione molto sofisticata;
- È difficile da interfacciare perché i connettori sono molto sofisticati;
- Ha dei fenomeni di dispersione e da effetti non lineari.

Legge di Snell

La fibra ottica funziona in base ad un meccanismo fisico derivante dalle **leggi di Snell**: un raggio luminoso che incide su una superficie che interfaccia due mezzi con densità diverse ($n_1 > n_2$), viene in parte riflesso e in parte rifratto.

$$n_1 \sin\alpha_1 = n_2 \sin\alpha_2$$

- α_1 è l'angolo di incidenza del raggio rispetto alla normale della superficie di incidenza;
- α_2 è l'angolo che il raggio rifratto forma con la stessa normale nel secondo mezzo;
- Se $n_2 < n_1$, allora α_2 tende ad aumentare al crescere di α_1 sino a quando si arriva alla condizione per cui si ha $\alpha_2 = 90°$ ovvero assenza di raggio rifratto.

Quando faccio viaggiare il segnale all'interno del core, gradirei massimizzare la riflessione perché l'energia rifratta è energia che si perde. Quindi devo calcolare l'angolo di incidenza in modo che tutta l'energia rimbalza per riflessione tra core e cladding. Quello che si cerca è un fenomeno di **riflessione totale** in cui ho assenza di rifrazione. L'unico caso in cui c'è bisogno di energia rifratta è quando il segnale deve entrare nella fibra ottica: quando il segnale entra nella fibra, c'è un emettitore di luce (led o laser), parte dell'energia viene rifratta nella fibra e parte viene riflessa perdendosi. L'angolo di incidenza che permette l'ingresso nella fibra viene scelto in base all'**apertura numerica** (NA) della fibra che non è altro che la quantità di luce che è possibile lanciare all'interno della fibra senza che questa venga perduta. Questa NA è caratterizzata un angolo limite che garantisce di massimizzare la riflessione all'interno della fibra: più è grande questa tolleranza maggiore sarà il cono di accettazione della fibra.

Modi di propagazione

Le leggi di Maxwell stabiliscono come si muovono le onde, e visto che nella fibra ottica viaggia un insieme di fotoni, queste leggi stabiliscono quindi i modi di propagazione della luce nella fibra ottica.

Il segnale che invio nella fibra si propagherà secondo una serie di **modi**, definiti dalle equazioni di Maxwell, che sono paralleli all'asse della fibra che passa per il core. Questi modi saranno caratterizzati da diversi raggi dipendenti dall'angolo di accettazione. Esistono tanti modi in base all'angolo di accettazione della fibra in base al NA.

Ogni singolo modo ha sue caratteristiche di propagazione:

- Il tempo di propagazione varia da modo a modo con l'angolo di incidenza;
- Si possono ricevere varie copie ritardate dello stesso segnale;
- Il modo assiale ha il ritardo minore in quanto compie il tragitto più breve;
- Il modo con un angolo di ingresso prossimo all'angolo di accettazione ha il ritardo maggiore in quanto compie un percorso più lungo a causa del numero maggiore di riflessioni che subisce.

Quando immetto un segnale, esso si propaga in diversi modi e quindi possono arrivare più copie dello stesso segnale ritardate: questo può causare una sorta di dispersione perché l'energia associata ai vari modi arriva a destinazione in tempi diversi in dipendenza all'angolo di incidenza che ha il modo stesso. Quindi potrò avere una sorta di sovrapposizione di segnali in uscita che porta un fenomeno chiamato **dispersione modale**.

Possiamo distinguere le fibre ottiche in due diversi tipi:

- **Fibre ottiche multimodali**: hanno un core con diametro molto ampio e quindi sono in grado di far passare molti modi. Il segnale si propaga in ragione di diversi modi e più copie del segnale stesso arrivano simultaneamente a destinazione. Si possono dividere in: **step-index**, in cui l'indice di rifrazione dei due mezzi varia in maniera netta, come un gradino; viceversa, sono chiamate **graded-index**;

- **Fibre ottiche monomodali**: hanno un core più sottile e sono in grado di far propagare un unico modo (quello che va dritto). La fibra si comporta come una guida d'onda perfetta, riesce a coprire distanze molto lunghe e senza nessuna dispersione modale.

Lunghezza d'onda

Le frequenze che usiamo per trasmettere sulla fibra ottica viaggiano appena fuori rispetto allo spettro del visibile.

Finestra operativa

Per trasmettere si possono utilizzare, sulla base del range di frequenze che uso (quindi sulla base delle lunghezze d'onda che uso), tre possibili intervalli di lunghezze d'onda che sono chiamati **finestre operative** per le quali risultano tecnologicamente ottimizzate sia le fibre che i dispositivi trasmettitori e ricevitori di luce. Le finestre operative per trasmettere sono state scelte studiando il grafico dell'attenuazione: dove l'attenuazione raggiunge un minimo locale, lì avrò una finestra operativa. (la terza finestra è la migliore ma richiede apparati troppo sofisticati e quindi costosi)

Attenuazione ottica

Parte dell'energia luminosa che si propaga lungo la fibra viene assorbita dal materiale o si diffonde in esso, costituendo quindi una perdita ai fini del segnale trasmesso. Il rapporto tra la potenza ottica trasmessa e quella ricevuta, dopo aver percorso una lunghezza di fibra di riferimento, definisce l'attenuazione della fibra stessa, in funzione della lunghezza d'onda e del tipo di fibra.

Multiplazione WDM

Consente di veicolare più lunghezze d'onda λ (oggi fino a 320) all'interno del medesimo portante fisico, ciascuna con capacità trasmissiva fino a 100 Gbps, dipendente dalla qualità della fibra e degli apparati di trasmissione.

Questa logica WDM è la logica naturale di trasmissione nel dominio fotonico. Ovviamente avremo sempre il problema dell'attenuazione che risolveremo o con un amplificatore ottico che usa l'effetto di *interferenza costruttiva*: il segnale viene fatto passare in un avvolgimento di fibra dogata, è una fibra che viene trattata con l'erbio, dove il segnale originario viene influenzato da un segnale interferente esterno (segnale di pompa) il quale cede parte della sua potenza al segnale originario introducendo rumore.

Però oltre un certo numero di amplificazioni si deve per forza rigenerare il segnale.

(vedere architettura WDM)

Trasmissione sulla fibra

La trasmissione attraverso la fibra ottica può essere effettuata con tre diverse modalità:

- Con **led** sulle fibre multimodali;
- Con **laser** sulle fibre monomodali;
- Con **VCSEL**, laser a semiconduttore che è un ibrido tra laser e led.

La potenza totale emessa da un trasmettitore è distribuita su un range di lunghezze d'onda diffuse intorno al centro d'onda. La larghezza d'onda dello spettro dipende dal tipo di sorgente utilizzata: il laser ha un'ampiezza d'onda molto piccola, viceversa, il led ha un'ampiezza d'onda molto larga. Larghe ampiezze di spettro portano a incrementare la dispersione.

La differenza fra led, laser e VCSEL è nella sofisticazione dell'apparato.

I ricevitori

Per ricevere il segnale della fibra ottica ho bisogno di un fotorilevatore che trasforma il segnale luminoso in segnale elettrico, e sono chiamati **fotodiodi**. Esistono due tipi di fotodiodi: **PIN** che genera corrente proporzionale alla potenza luminosa generata; **ADP** genera più corrente ma è più sensibile al rumore.

(mezzi trasmissivi non guidati)

Trasmissione wireless

Con la trasmissione wireless si utilizzano mezzo trasmissivi non guidati e il mezzo per eccellenza è l'**aria**, in ragione della frequenza uso sezioni diverse dello spettro quindi onde elettromagnetiche diverse che si propagano con caratteristiche diverse.

Il **sistema di trasmissione** avrà due componenti: un *trasmettitore* che emette un'onda elettromagnetica la quale si propaga per una certa distanza in dipendenza dalla potenza con cui la trasmetto; e un *ricevitore* che intercetta l'onda che diventa un segnale. Dietro il concetto di trasmissione delle onde c'è sempre l'equazione di Maxwell.

Nella trasmissione stimoliamo la coppia di conduttori con una tensione generando un'onda elettromagnetica. Se una coppia di conduttori riceve un'onda, incamerano parte di questa energia e la trasformano in un segnale elettrico. L'oggetto che trasmette e riceve è l'**antenna**.

Sistema di antenna

Dato che trasmetto e ricevo simultaneamente sulla stessa stazione, avrò sia un trasmettitore che un ricevitore che sono collegati all'antenna da un **circolatore** che impiega l'antenna sia per trasmettere che per ricevere. Poi c'è un **feeder di antenna** che è un cavo coassiale che collega il circolatore con l'antenna.

Assegnazione delle frequenze

Le frequenze non sono libere, ma assegnate in termini di legge: l'uso va autorizzato perché sono una risorsa pregiata. L'ente che provvede ad assegnare le frequenze è il MISE e l'uso appropriato o non è vigilato dalla polizia postale. Una volta che le frequenze sono state acquisite e l'uso ne è stato autorizzato, esiste un piano nazionale per la ripartizione delle frequenze che comunica chi può trasmettere su una determinata frequenza (PNRF).

Ciò non vale per tutte le frequenze, infatti esiste un range di frequenze, detto **banda ISM** (Industriale Scientifico Medico), riservato per applicazioni industriali, scientifiche e mediche. Tutto ciò che non richiede una licenza deve prevedere queste bande.

Tipologia di trasmissione radio

Esistono delle tecniche di **spreading** dello spettro utilizzate per irrobustire il segnale, allargandolo, ed evitare che venga perso a causa di interferenze. Ci sono due tipi di tecniche:

- **Trasmissione a sequenza diretta**: ogni bit viene trasmesso come una sequenza ridondante di valori (chip) influenzati dal codice di chipping;
- **Trasmissione a spettro diffuso**: si prende una banda più larga ma trasmetto sempre in un intervallo di banda che serve per il segnale, però saltello continuamente da un intervallo di

banda all'altro. Non avrò un codice che individua la trasmissione bensì una sequenza di salto.

Radiodiffusione a grande distanza

Usando le tecniche di trasmissione sopra elencate, sulla base delle frequenze che uso il segnale si comporta in maniera diversa:

- quando si lavora con segnali a bassa frequenza (fino al MHz), il segnale si propaga seguendo la curvatura terrestre ed attraversa bene gli ostacoli: una stazione trasmittente può essere ricevuta fino a 1000km di distanza. (sono segnali a bassa velocità ma raggiungono distanze molto elevate);
- quando si lavora, invece, con segnali ad alta frequenza (dal MHz al GHz), il segnale si propaga seguendo una logica rettilinea, però viene riflesso molto bene dalla ionosfera.

Quando si va oltre il GHz, da 1 a 40 GHz, si hanno dei segnali fortemente collimati ed estremamente direzionali. Dato che si propagano in linea retta, per comunicare a grandi distanze, richiedono una forte collimazione, un buon puntamento fra trasmettitore e ricevitore: significa che richiedono tipicamente quella che si chiama **LOS** (Line Of Sight), cioè trasmettitore e ricevitore si devono vedere.

Si può quindi realizzare una comunicazione punto-punto, che si chiama **ponte radio**, tra sorgente e destinazione con allineamento ottico delle antenne. Questi ponti radio diventano alternativi alla fibra ottica dove è difficile installarla. Utilizzando diverse stazioni ripetitrici si riescono a coprire distanze elevate.

Data la dipendenza dell'attenuazione dalla distanza, per tratte lunghe si utilizzano generalmente due bande di frequenza: 2-6 GHz e 10-14 GHz. Le connessioni a breve distanza possono utilizzare le frequenze più alte (fino a 40 GHz) per le quali si hanno i seguenti vantaggi:

- antenne più piccole;
- fascio più collimato (quindi minore necessità di potenza);
- minori problemi di interferenza per lo scarso utilizzo di trasmissioni in quella regione di frequenza. Per avere problemi o si deve interporre qualcuno con un disturbatore oppure si deve interporre un oggetto fisicamente; si possono avere problemi di interferenza anche con condizioni metereologiche avverse.

Generalmente, i ponti radio vengono utilizzati per trasmissioni analogiche (fonia, televisione) o digitali (reti private o utilizzate dalle compagnie telefoniche fornitrici di servizi). Le diverse bande di frequenza sono suddivise in canali di diversa larghezza, con canali tra i 7 MHz (a 2 GHz) ed i 220 MHz (a 18 GHz), e tassi trasmissivi che vanno dai 12 ai 274 Mbps (in funzione della banda disponibile e del livello di modulazione utilizzato, solitamente QAM-x).

Trasmissione radio – microonde

Le microonde non sono altro che onde radio ad alta frequenza, nell'ordine dei GHz. Sono poco costose e non richiedono il diritto di passaggio, però si deve comunque comprare la frequenza a meno che non si lavora su banda ISM. Più si sale di frequenza e più le onde si propagano in linea retta. C'è sempre un problema di rifrazione da parte dell'atmosfera: l'aria attraversata comunque danneggia il segnale inducendo fenomeni di rifrazione. Oltre i 8 GHz si ha assorbimento da parte dell'acqua.

Trasmissione a infrarossi

Salendo di frequenza, oltre le microonde, troviamo gli **infrarossi**, onde di lunghezza millimetrica; sono relativamente direzionabili e non passano attraverso i solidi (questo è uno svantaggio ma anche un vantaggio perché non interferiscono con sistemi vicini). Esistono due tecnologie:

- **infrarossi diretti**: trasmettitore e ricevitore devono essere perfettamente allineati per potersi illuminare reciprocamente con un fascio di luce; (trasmissione punto a punto)
- **infrarossi a diffusione**: la radiazione luminosa è emessa da una stazione, viene diffusa in tutte le direzioni e rimbalza su soffitto e pavimenti, venendo riflessa verso tutte le altre stazioni; (trasmissione di tipo broadcast)

La trasmissione a infrarossi è molto utile per comunicazioni a brevissime distanze; sono più sicure delle onde radio; non richiedono nessuna licenza; vengono usate per LAN interne con infrastruttura fissa (beacon); non sono utilizzabili all'esterno per la presenza di emissioni solari; c'è una modalità piuttosto primitivi che permette di lavorare a 115 Kbps e una modalità più sofisticata che arriva a 4 Mbps.

Trasmissione Lightwave

Salendo ancora di frequenza, oltre i raggi infrarossi, c'è la **luce visibile** o leggermente oltre il visibile. È una trasmissione molto più costosa di quella a infrarossi, richiede una potenza molto più elevata e disperde molto più calore ma mi permette di direzionare la luce ad altissime frequenze su un fascio molto stretto, permettendo la trasmissione a velocità molto elevate. In gergo questi sistemi si chiamano **ponti laser**: hanno un laser come trasmettitore e un fotorilevatore come ricevitore.

Le caratteristiche dei ponti laser sono:

- come gli infrarossi, non possono attraversare ostacoli;
- richiedono un grande larghezza di banda;
- non richiedono nessuna licenza;
- necessitano di grande precisione nel puntamento, alcuni sono autopuntanti;
- costi bassi;
- c'è un enorme sensibilità alle condizioni atmosferiche e viene disturbato anche da turbolenze nell'aria dovuti a convezioni.

Trasmissione satellitare

Il satellite si comporta come una stazione ripetitrice (dato che è in LOS con tutto ciò che vede) del segnale di un ponte radio. Il segnale viene inviato dalla stazione terrestre al satellite (**uplink**), che lo rimanda a terra verso la stazione o le stazioni riceventi (**downlink**), generalmente utilizzando frequente differenti. Le frequenze usate devono essere molto elevate perché devono muoversi in logica rettilinea senza assorbimenti o riflessioni da parte della ionosfera.

Un satellite è dotato di un dispositivo di trasmissione-ricezione molto sofisticato che si chiama **trasponder** che permette di usare un range piuttosto elevato di bande (tra 15 e 500 MHz) tramite una tecnologia FDM; inoltre, i satelliti multiplani i canali con TDM per servire ancora più stazioni contemporaneamente. Quindi la trasmissione satellitare, aldilà del problema della latenza, è piuttosto efficiente in termini di capacità.

Le bande utilizzate sono quelle comprese tra 1 e 10 GHz, solo che tipicamente questa zona è affollata di trasmissioni, perciò a volte si assume l'onere di arrivare fino a 40 GHz.

Esistono tre tipi di satelliti in dipendenza dalla loro distanza orbitale:

- **GEO** (Geostationary Earth Orbit): stanno a circa 36000 km di quota e ruotano alla stessa velocità della Terra, per cui per noi risultano fissi. Sono molto adatti alla trasmissione dati perché non dobbiamo cambiare puntamento delle antenne; sono usati anche per la trasmissione televisiva; per motivi di interferenza i satelliti vengono distanziati di due gradi, quindi si possono avere al massimo 180 satelliti (anche se ne bastano solo 3); il problema è la latenza pari a 0.25 secondi però sono ottimi per la trasmissione dati;

- **MEO** (Medium Earth Orbit): stanno a circa 18000 km di quota e ruotano con un periodo orbitale di 6 ore. Sono inadatti per la trasmissione dati, però, visto che ne possiamo mettere tanti, sono ottimi per i GPS. Ha latenza medio bassa;

- **LEO** (Low Earth Orbit): stanno tra i 750 e i 1500 km di quota. Sono molto veloci nel transito, ma vicini, quindi si ha poco ritardo e si richiede poca potenza in trasmissione. Possono essere utilizzati per la comunicazione cellulare.

Comunicazione con satelliti

Quando comunico con i satelliti tipicamente si ha che il segnale parte da una stazione di terra, si avrà un salto sul satellite, la comunicazione potrebbe tornare a terra in un HUB che smista il segnale alla stazione ricevente. (**VSAT**)

Relaying in space con satelliti LEO e **relaying on the ground**.

Datalink

Questo livello di trasmissione riceve pacchetti dati dal livello di rete e forma i frame che vengono passati al sottostante livello fisico con l'obiettivo di permettere il trasferimento affidabile dei dati attraverso il sottostante canale.

Il livello **datalink** deve svolgere più funzioni specifiche:

- In trasmissione dovrà raggruppare i bit provenienti dallo strato superiore e dividerli (non sempre) in piccoli **frame** (framing);
- In trasmissione dovrà operare una qualche forma di accesso multiplo/multiplazione per l'accesso condiviso tra più utenti al canale fisico che eviti collisioni tra pacchetti e interferenze in ricezione o sul canale;
- In ricezione riceverà i bit dallo strato fisico, toglierà gli header per interpretare l'informazione per ricomporre i frame e cercare di controllare il flusso, di comprendere se c'è stato un errore, se c'è stata una perdita di dati. Poi dovrà ricostruire tutto il messaggio e inviarlo allo strato superiore.

I servizi che dovrà, quindi, offrire il datalink sono:

- **Framing**:
  - i protocolli incapsulano i datagrammi del livello di rete all'interno di un frame a livello di link;
  - i frame vengono passati al livello fisico con l'obiettivo di permettere il trasferimento affidabile dei dati;
- **Accesso al mezzo**:
  - Uso di protocolli per disciplinare l'accesso multiplo dei nodi ad un unico canale di comunicazione evitando o gestendo le collisioni;
- **Consegna affidabile:**
  - È considerata non necessaria nei collegamenti che presentano un basso numero di errori sui bit (fibra ottica, cavo coassiale e doppino intrecciato);
  - È spesso utilizzata nei collegamenti soggetti a elevati tassi di errori (es. collegamenti wireless);
- **Controllo di flusso**:
  - Evita che il nodo trasmettente saturi quello ricevente;
- **Rilevazione degli errori**:
  - Gli errori sono causati dall'attenuazione del segnale e da rumore elettromagnetico;
  - Il nodo ricevente individua la presenza di errori. Ciò è possibile grazie all'inserimento, da parte del nodo trasmittente, di un bit di controllo di errori all'interno del frame;
- **Correzione degli errori**:
  - Il nodo ricevente determina anche il punto in cui si è verificato l'errore, e lo corregge;
- **Half-duplex e full-duplex**:

- Nella trasmissione full-duplex gli estremi di un collegamento possono trasmettere contemporaneamente: non in quella half-duplex.

Tutte queste azioni richiedono tempo, soprattutto il controllo dei bit in ricezione. Se, mentre il datalink sta lavorando, arriva un'altra informazione dal livello Network il livello datalink non può riceverlo perché occupato allora questo messaggio verrà inserito in un buffer. La stessa cosa accade in ricezione: i dati inviati dal livello fisico al livello datalink verranno inseriti in un buffer anch'essi. Una volta che il buffer è saturo, gli altri dati che non sono stati inseriti andranno persi, allora si potrebbe pensare di ritrasmetterli ma così facendo invierò ancora più dati al buffer che è già saturo.

Connection oriented

Un'altra serie di servizi, che deve offrire il DLL (datalink layer), che si dicono **con connessione** o **senza connessione**: gli interlocutori prima di iniziare la comunicazione si accertano, reciprocamente, di essere sulla stessa rete. È una trasmissione affidabile o meno (senza riscontro), dipende se riesce a trasmettere i dati privi di errori e senza perdite. Rendere il servizio affidabile richiedere la costruzione e l'implementazione di un protocollo di comunicazione che richiederà del tempo per poter essere eseguito ma anche degli spazi fisici (quindi è una questione di risorse fisiche e temporali). Se non si hanno a disposizione queste risorse, non si può utilizzare una trasmissione affidabile, però se voglio comunque farla allora potrò utilizzare quella senza riscontro.

La classe di servizio non affidabile è adatta sulle linee di elevata qualità (es. fibra ottica):

- Il controllo sugli errori e la ritrasmissione di frame errati comporta una inefficienza in termini di numero di bit trasmessi rispetto ai dati, con riduzione del tasso utile ed aumento della probabilità di errore;
- Il controllo può essere demandato ai livelli superiori a vantaggio della efficienza del livello di datalink;
- Generalmente questi servizi sono utilizzati su rete locale e anche per il traffico voce e video.

(non lavora solo la scheda di rete ma anche la CPU, per verificare tutti questi controlli)

La classe di servizio affidabile è adatta, invece, su linee più frequentemente soggette ad errori:

- Demandare il controllo e la ritrasmissione ai livelli superiori (che generalmente trasmettono pacchetti costituiti da più frame) in caso di elevata probabilità di errore potrebbe causare la ritrasmissione di molti pacchetti, mentre al DDL può essere sufficiente la ritrasmissione del singolo frame;
- Implementa meccanismi di riscontro per verificare la necessità di ritrasmissioni;
- Tipicamente utilizzata su linee a grande distanza (WAN), anche se la fibra ottica riduce notevolmente questo problema.

Il DDL deve quindi poter offrire le diverse classi di servizio, per soddisfare le diverse esigenze conseguenti alle diverse circostanze.

Funzionamento DLL

In trasmissione:

1. Riceve i pacchetti dal livello Network;
2. Li organizza in trame (frame) eventualmente spezzando in più frame il blocco di dati ricevuto;
3. Aggiunge ad ogni frame un header e un trailer e passa il tutto allo strato fisico per la trasmissione.

In ricezione:

1. Riceve i dati dal livello fisico;
2. Effettua i controlli necessari, elimina header e trailer, ricombina i frame e passa i dati ricevuti al livello Network.

Adattatori

Il lavoro in trasmissione e quello in ricezione lo svolge un hardware esterno chiamato **NIC** (la classica scheda di rete). Dal lato trasmittente, la NIC, riceve messaggi dal software e svolge tutte le operazioni dette prima. In ricezione, la NIC, fa esattamente il contrario. Il NIC è un'unità semiautonoma: non fa tutto per conto suo perché parte degli stack protocollari vengono eseguiti in hardware all'interno della NIC, ma altre parti (soprattutto quelle di livelli superiori) vengono eseguite a livello software che girano sulla CPU causando un overhead.

Problematiche del livello datalink

Per poter svolgere le sue funzioni, il datalink layer dovrà curare i seguenti aspetti:

- **Organizzazione** del flusso di bit in **frame**, con controllo per la **sincronizzazione**; inserimento e rimozione di header e trailer, riordinamento dei frame in ricezione;
- Organizzare il trasferimento dei dati in modo da **gestire** eventuali **errori di trasmissione**, utilizzando codici di **correzione** degli errori o codici di **identificazione** degli errori e gestendo la **ritrasmissione** dei frame errati;
- Realizzare il **controllo di flusso**, per utilizzare in modo efficiente il canale trasmissivo impedendo al contempo ad un **trasmettitore veloce** di sovraccaricare un **ricevitore lento**.

Framing e sincronizzazione (risoluzione primo problema)

Per trasportare i bit, il DLL utilizza i servizi dello strato fisico, il quale non può garantire il trasferimento privo di errori, che dovranno essere gestiti dal DLL (organizzandoli in frame ed effettuando controlli per ognuno trama). La gestione del frame deve prevedere in primo luogo la possibilità del ricevente di identificare il frame; quindi, si devono adottare regole per delimitarlo e poterne identificare i limiti in ricezione.

Esistono diverse tecniche per capire quando inizia e quando finisce un frame:

- **Conteggio dei caratteri**: viene messo nell'header un numero che indica il numero di caratteri del frame (compreso sé stesso). Se si perde il sincronismo non si riesce a trovare l'inizio del pacchetto successivo;

- **Caratteri di inizio e di fine**: vengono inseriti nell'header dei caratteri ASCII DLE (Datalink Escape – 0x10) e STX (Start of Text – 0x02), e nel trailer dei caratteri DLE ETX (End of Text – 0x03). La risincronizzazione è più semplice, basta vedere ETX. Questi caratteri ASCII DLE però li possiamo trovare anche all'interno del messaggio (nel payload) e per evitare che venga confuso con un trailer, si aggiunge un altro DLE, che verrà poi eliminato quando verrà consegnato il messaggio al livello Network;

- **Indicatori di inizio e di fine**: viene aggiunto un byte (01111110) che rappresenta la separazione tra un messaggio e l'altro. Nel caso in cui nel payload ci sia la sequenza di flag, si utilizza la tecnica del bit stuffing, cioè inserisco uno 0 ogni 5 bit 1 consecutivi nel payload. Quando poi il messaggio verrà consegnato al livello Network, il DLL toglierà tutti i bit di stuffing;

- **Violazione di codifica**: è possibile segnalare l'inizio o la fine di una trama con una deliberata violazione delle regole di codifica del segnale. Ad esempio, usando la codifica Manchester differenziale è possibile ottenere una violazione omettendo la transizione da 1 a 0 o da 0 a 1 nel mezzo di un impulso per indicare rispettivamente la fine o l'inizio di una trama.

Rilevazione dell'errore (risoluzione secondo problema)

Dobbiamo capire se il mezzo fisico, nel trasportare i dati, ha commesso un errore: errori sul singolo bit, replicazione di bit, perdita di bit.

Una tecnica per rilevare la presenza di errori è quella che consiste nell'inserire nel header un campo denominato **checksum**: una funzione matematica che, dati i dati, restituisce un valore che viene inserito nell'header e il messaggio viene inviato al livello fisico. In ricezione, il DLL va a guardare sia il checksum che il payload; calcola il checksum dato il payload con la stessa funzione utilizzata in trasmissione, e va a controllare se i due checksum coincidono: se sono uguali non ci sono errori, altrimenti c'è un errore ma non sappiamo se sta nel payload o nel CK.

Modi per calcolare il checksum

Una prima tecnica è il cosiddetto **bit di parità**, un codice di controllo utilizzato per prevenire errori nella trasmissione o nella memorizzazione dei dati. Prevede l'aggiunta di un bit ridondante ai dati, calcolato a seconda se il numero di bit che valgono 1 nel payload, sia pari o dispari. Ci sono due varianti del bit di parità:

- **Bit di parità pari**: si pone tale bit uguale a 1 se il numero di "1" nel payload è dispari;
- **Bit di parità dispari**: si pone tale bit uguale a 1 se il numero di "1" nel payload è pari.

Se si usa il bit di parità pari non sarà in grado di rilevare gli errori se essi sono un numero pari, stessa cosa vale per il bit di parità dispari.

Una possibile soluzione è la seguente: spezzettiamo l'intero messaggio in altri messaggi di lunghezza *k* e li metto in colonna; vado a vedere le righe e aggiungo il bit di parità, e faccio la stessa cosa per ogni colonna. Dopo aver inviato questa tabella al ricevitore esso sarà in grado di trovare il bit corrotto semplicemente controllando i bit di parità sia delle righe che delle colonne. Purtroppo, questa soluzione non funziona sempre può capitare che rileva gli errori ma non è in grado di individuarli, cioè non sa quale bit deve correggere, oppure non è in grado nemmeno di rilevarlo. Questa cosa succede soprattutto quando ci sono 2 o più errori sulla stessa riga e/o sulla stessa colonna.

Un'altra tecnica utilizzata soprattutto a livello di internet è il **checksum di internet**: i dati vengono trattati come se fossero degli interi a 16 bit e vengono sommati, il risultato ottenuto gli applico il complemento a 1 e diventa il checksum inserito nell'header. Il ricevitore riceve il checksum calcolato in trasmissione, lo ricalcola guardando il payload, se sono uguali è tutto in ordine altrimenti si è verificato un errore.

I campi di Galois (risoluzione secondo problema)

Questa tecnica non garantisce di individuare l'errore bensì di rilevarlo con una probabilità molto alta.

Un campo finito con *q* elementi su cui sono definite due operazioni aritmetiche (addizione e moltiplicazione) che godono della proprietà commutativa ed associativa, viene chiamato **campo di Galois** ed indicato con **GF(q)**: è chiuso rispetto all'addizione e moltiplicazione e in generale *q* deve essere sempre primo o potenza di numeri primi. Le operazioni di somma e moltiplicazione vengono calcolate utilizzando i concetti aritmetici tradizionali con l'applicazione di un ulteriore operazione di **mod q**.

Un esempio di applicazione è il **CRC** (*Cyclic Redundance Check-sum*) su GF(5): si fa la somma di tutti i numeri del payload ed infine modulando il risultato per 5.

Rappresentazione di sequenze di bit tramite polinomi (Sempre Galois)

Una sequenza di N bit può essere rappresentata tramite un polinomio a coefficienti binari, di grado pari a N-1, tale che i suoi coefficienti siano uguali ai valori dei bit della sequenza. Il grado del polinomio è determinato dal primo bit a sinistra di valore 1 presente nella sequenza.

La tecnica consiste nel considerare i dati (m bit) da inviare come un polinomio di grado m-1. Trasmettitore e ricevitore si accordano sull'utilizzo di un **polinomio generatore** G(x) di grado *r*. a questo punto, il trasmettitore aggiunge in coda al messaggio una sequenza di bit di controllo (CRC) in modo che il polinomio associato ai bit del frame trasmesso, costituito dall'insieme di dati e CRC, sia divisibile per G(x). In ricezione si divide il polinomio associato ai dati ricevuti per G(x):

- Se la divisione ha resto 0, allora non ci sono stati errori nella trasmissione;
- Se la divisione ha resto diverso da 0, allora ci sono stati errori in trasmissione.

Come calcolo il CRC da aggiungere in coda al messaggio da trasmettere?

m = 10100011 ⇒ associamo un polinomio ad esso => $\mathrm{P(x)} = \mathrm{x}^7 + \mathrm{x}^5 + \mathrm{x} + 1 \Rightarrow$

⇒ a questo punto scelgo il polinomio generatore di grado *r <= p – 1* (dove *p* è il grado di P(x))
⇒ $\mathrm{G(x)} = \mathrm{x}^3 + 1 \Rightarrow$ ora moltiplico il primo termine di G(x) per tutto P(x) ⇒

$\mathrm{x}^3 P(x) = \mathrm{x}^{10} + \mathrm{x}^8 + \mathrm{x}^4 + \mathrm{x}^3$, cioè P = 10100011000 (il messaggio originario avrà grado aumentato di 3 e aggiungerà tre zeri a destra che dovranno <u>rappresentare il checksum</u>).

Farò la divisione $\frac{x^r P(x)}{G(x)} \rightarrow Q(x)G(x) + R(x) = x^r P(x)$ cioè $\mathrm{x^r\ P(X) - R(x) = Q(x)G(x)}$. Poiché operiamo nel caso dei codici binari, il campo di Galois utilizzato è GF(2). Quindi $-\mathrm{R(x)} = +\mathrm{R(x)}$, e la formula precedente diventa $\mathrm{x^r\ P(X) + R(x) = Q(x)G(x)}$. Quindi il messaggio che si trasmette è il messaggio originale aggiunta una quantità di bit a destra pari a *r*, e al posto di questi bit dovrò aggiungere il CRC che non è altro che il resto della divisione polinomiale tra il polinomio originale per $\mathrm{x^r}$ diviso il polinomio generatore.

<u>In ricezione</u>: prendo il polinomio e lo divido per G(x).

CRC standard

Esistono dei CRC standard perché hanno una bassissima probabilità di commettere errori e sono **CRC-12**, **CRC-16**, **CRC-CCITT**, **CRC-32**. Il 16 e il CCITT riconoscono errori singoli e doppi. Ci dicono che c'è l'errore ma non lo correggono. Il calcolo del CRC può essere fatto in hardware.

Controllo di flusso (risoluzione terzo problema)

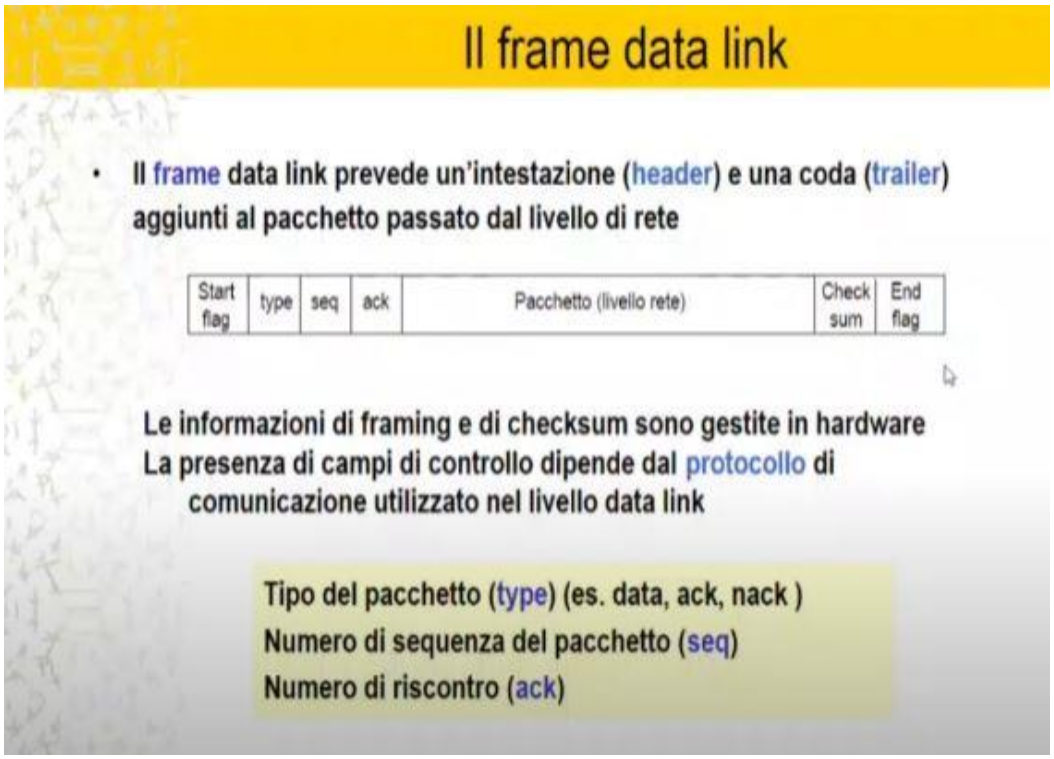
Il problema del controllo di flusso nasce perché la velocità di trasmissione è maggiore rispetto alla velocità che il DLL ha per elaborare i dati. Ad un certo punto non sarà in grado di elaborare i dati che gli arrivano e comincerà a metterli in un buffer, ma quando si riempirà anche il buffer i dati verranno rigettati, andranno persi.

<u>Perché si crea il problema del flusso?</u>

L'implementazione del DLL prevedrà la realizzazione di interfacce con i livelli adiacenti, ad esempio due procedure *from-network-layer()* e *to-network-layer()* per scambiare dati con il livello superiore, e due procedure analoghe per scambiare dati con lo strato fisico. In aggiunta sarà prevista una procedura *wait-for-event()* che metterà il DLL in attesa di un evento, che sarà una segnalazione, da parte di uno dei due layer adiacenti, che sono disponibili dei dati. Quando il DLL si metterà al lavoro non potrà ricevere altri dati che andranno nel buffer. Il problema è che il tempo di esecuzione del DLL una volta che è stato svegliato, il suo lavoro lo deve fare in un tempo finito in cui non potrà fare altro se non mettere i dati nel buffer o rigettarli se esso è pieno. Se i dati si sono persi il trasmettitore li invierà di nuovo ma il buffer se prima era bloccato continuerà ad esserlo se non si è svuotato.

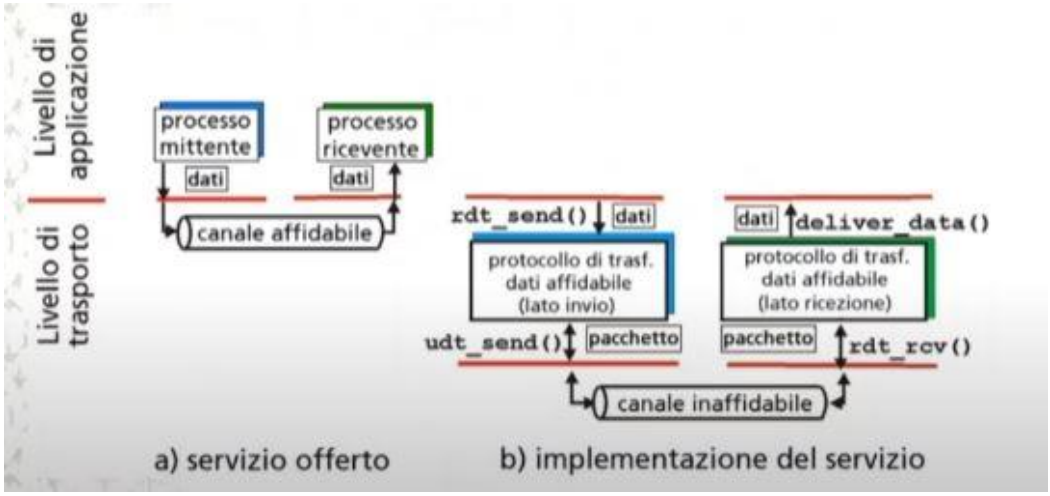
Esempi di soluzioni:

Un semplice meccanismo può essere quello di valutare i tempi di risposta del ricevente, ed inserire dei ritardi nel processo di trasmissione per adattarlo alla capacità di ricezione. Il problema è che il tempo di processamento in ricezione non è una costane e può dipendere dal numero di linee che il nodo ricevitore deve gestire. Quindi ci si dovrà basare sul caso peggiore che però comporta un grosso limite di efficienza.



Principi del trasferimento dati affidabile

Durante il trasferimento di dati non si devono perdere dati e nemmeno corrompere. Se il canale di trasmissione è affidabile non c'è bisogno di implementare numerosi protocolli. Ma visto che i canali trasmissivi che si usano sono inaffidabili quindi c'è bisogno di implementare tutti i protocolli del caso per risolvere i problemi che il canale dà. In trasmissione è stato aggiunto un **protocollo di trasferimento dati affidabili** (Reliable Data Transfer, **RDT**).



- **rdt_send()**: sarà chiamato dal network layer quando questo ha necessità di mandare dati sul mezzo (rdt_send() = from_network());
- **udt_send()**: invia il pacchetto di dati al mezzo fisico (udt_send() = to_physical());
- **rdt_receive()**: il pacchetto è arrivato in ricezione e deve essere preso dal DLL (rdt_receive() = from_physical());
- **deliver_data()**: consegna il pacchetto di dati al network layer (deliver_data() = to_network()).

Per comprendere al meglio l'RDT si fa uso dei **diagrammi di stato**: un grafo in cui ogni nodo rappresenta uno stato in cui si trova il protocollo, la transizione da uno stato all'altro è rappresentata dalla linea di collegamento tra due nodi; per passare da uno stato all'altro si è dovuto verificare un evento, e per svolgere una transizione si compie un'azione.

RDT 1.0

Il canale di trasferimento è affidabile. Il mittente ha bisogno di un solo stato, quello di attesa di una chiamata dal network layer: la chiamata arriva attraverso rdt_send() e va nello stesso stato creando il pacchetto attraverso make_pkt(data) e la invia con udt_send().

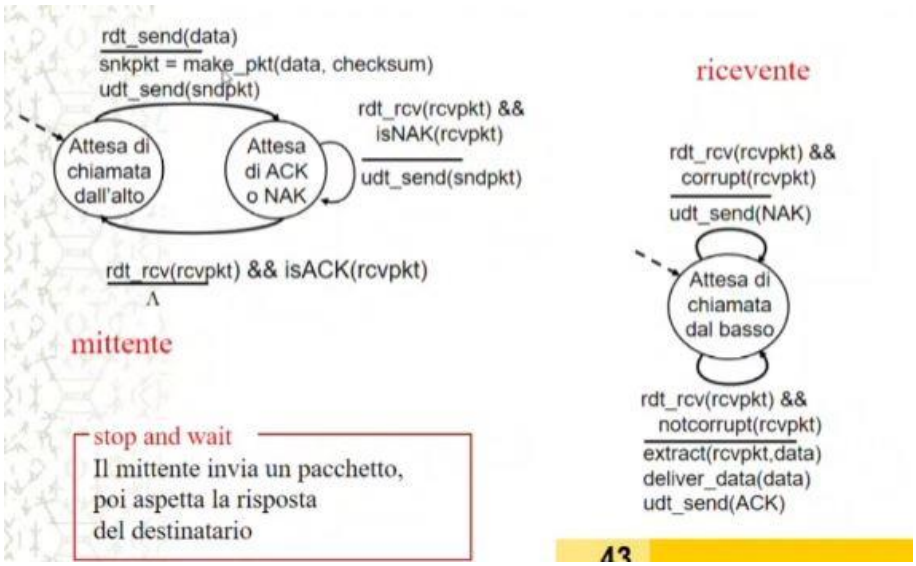
Il ricevente avrà bisogno sempre di un solo stato, quello di attesa di una chiamata dal livello fisico: arriverà un rdt_receive(packet), lo estrarrà e lo consegnerà al network layer con deliver_data().

RDT 2.0

Il canale di trasferimento potrebbe confondere i bit nei pacchetti, i dati arrivano corrotti ma non persi: riuscire a rilevare la presenza di errori tramite il checksum. Per notificare lo stato della trasmissione si usa:

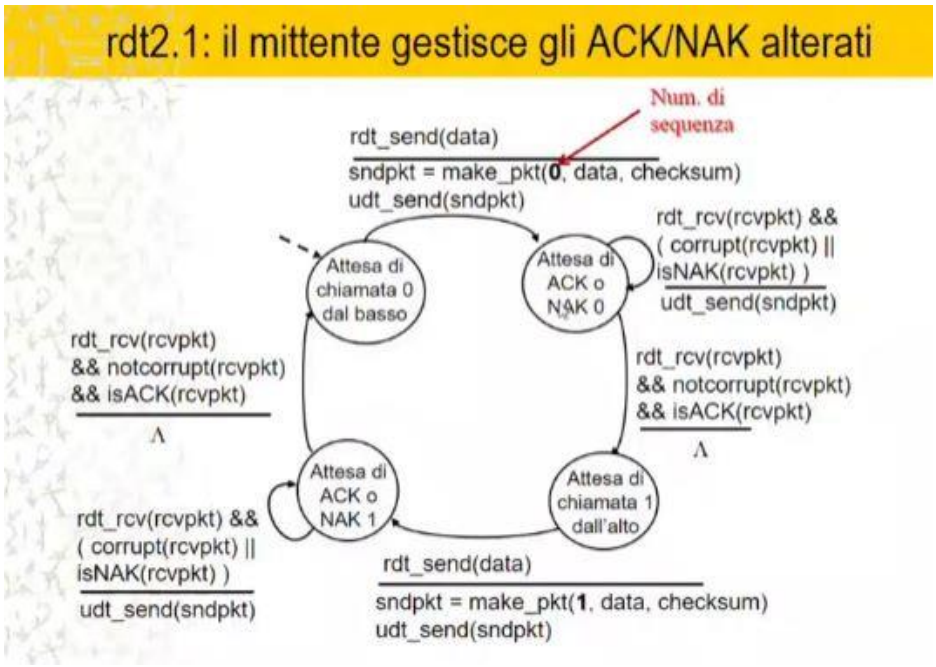
- **notifica positiva (ACK)**: il ricevente comunica espressamente al mittente che il pacchetto ricevuto è corretto;
- **notifica negativa (NAK)**: il ricevente comunica espressamente al mittente che il pacchetto contiene errori. Il mittente ritrasmette il pacchetto se riceve un NAK.

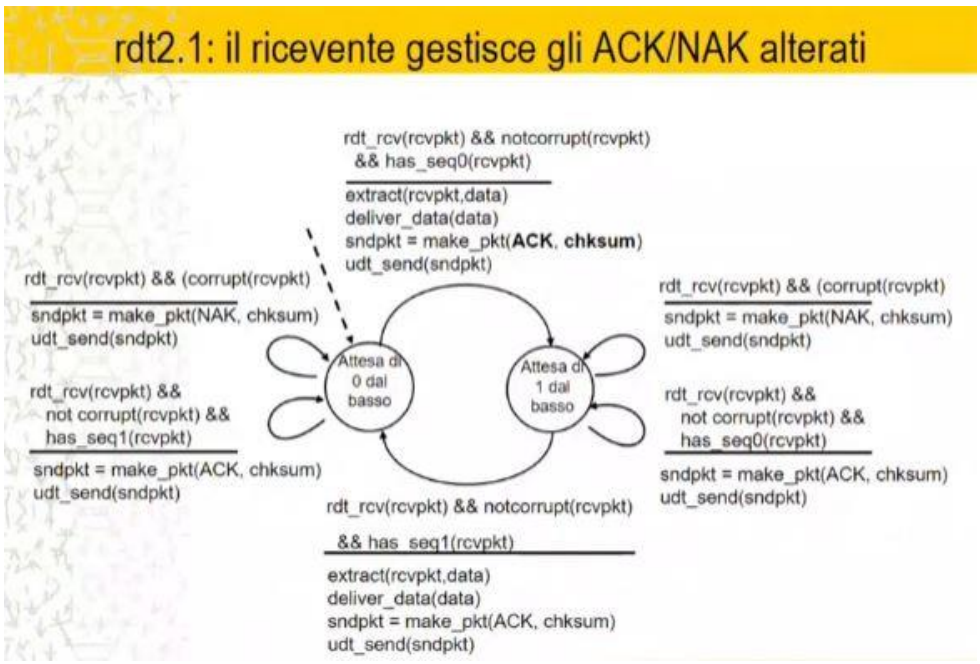
Questi protocolli sono conosciuti come **PAR** (Positive Acnowledgement with Retransmission) o anche **ARQ** (Automatic Repeat reQuest).



Il protocollo **stop-and-wait** prevede che il mittente, dopo aver inviato il frame, si fermi per attendere un riscontro. Il ricevente, una volta ricevuto il frame, invierà al mittente un frame di controllo allo scopo di avviare il mittente che può trasmettere un nuovo frame. Va osservato che il traffico dati è simplex, ma i frame devono viaggiare nelle due direzioni; quindi, il canale fisico deve essere almeno half-duplex.

Il problema che si riscontra con RDT 2.0 è se ACK o NAK si danneggiano durante il trasferimento; quindi, bisogna aggiungere il checksum anche ad ACK e a NAK. A questo punto il mittente reinvia i pacchetti quando riceve un ACK/NAK alterato. Questo metodo introduce duplicati dei pacchetti nel canale tra mittente e ricevente. Inoltre, il ricevente non sa se il pacchetto ricevuto è uno nuovo o il duplicato. Per risolvere la questione dei duplicati si aggiunge un nuovo campo che è il **numero di sequenza**: il ricevente deve solo controllare questo numero per comprendere se il pacchetto che gli è arrivato è nuovo o un duplicato.





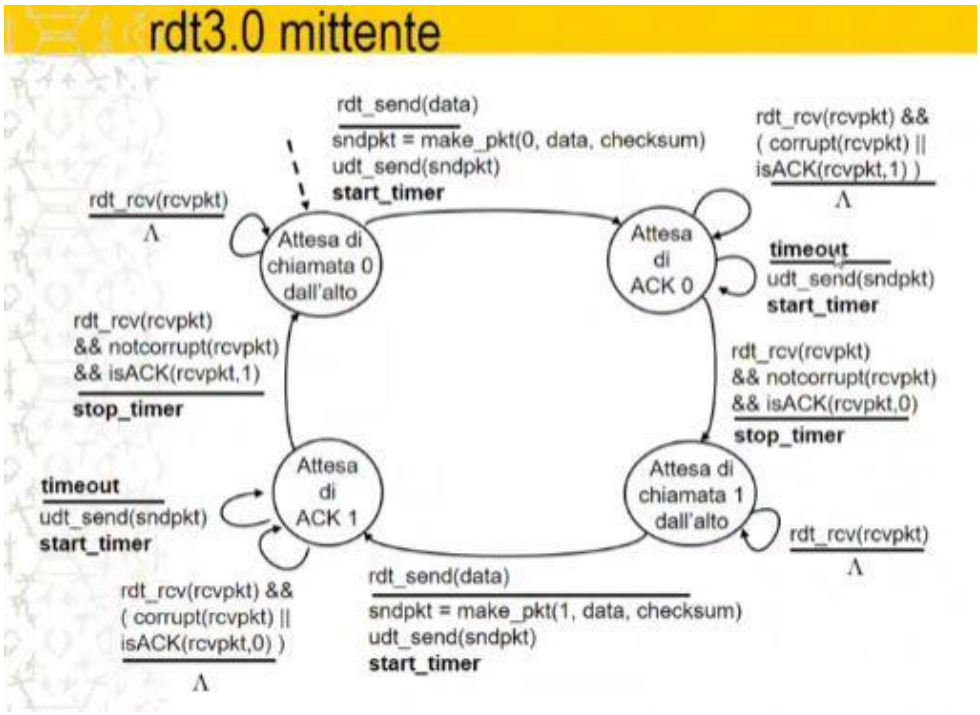
RDT 3.0

Un'altra problematica che può verificarsi è la perdita del ACK a causa del canale trasmissivo. Visto che il protocollo che si utilizza è stop-and-wait il tutto si fermerà. Per risolvere questo problema inseriamo un **timer** calibrato in base al mezzo trasmissivo che si utilizza e in base al tempo di esecuzione dei dati del DLL. Ogni volta che invio un messaggio in trasmissione attivo il timer, allo scadere del countdown rinvio il pacchetto. Se il pacchetto (o l'ACK) è soltanto in ritardo (non perso) la ritrasmissione sarà duplicata, ma questo sarà gestito con il "numero di sequenza".

Un altro problema è che quando il mittente riceve un ACK, non sa se si riferisce al pacchetto spedito più di recente o si tratta di un ACK arrivato in ritardo. La soluzione è che il ricevente deve specificare il numero di sequenza del pacchetto da riscontrare. Si inserirà nel pacchetto ACK un

campo di riscontro contenente il numero di sequenza del pacchetto dati ricevuto, così il mittente, esaminando questo campo, può individuare il pacchetto oggetto del riscontro.

Nel campo del riscontro non scrivo lo stesso numero di sequenza del pacchetto ma quello che mi aspetto di ricevere successivamente.



Prestazioni di RDT 3.0

Dopo aver implementato il protocollo per la trasmissione e la ricezione dei dati, si vuole anche analizzarlo dal punto di vista prestazionale, in particolare si vuole calcolare $\mathbf{U_{mittente}}$ cioè l'utilizzo della linea espresso come la frazione di tempo in cui il mittente è occupato solo nell'invio di bit senza il tempo impiegato per i vari controlli. *Si vuole ottimizzare il tempo in cui quella linea è impegnata per il trasferimento dei dati e non dei segnali di controllo.*

Nonostante i canali di trasmissione fanno viaggiare i dati ad una velocità pari a 200.000 km/s, c'è un ritardo che dipende dalla composizione del mezzo trasmissivo ma anche dalla lunghezza stessa.

Il **tempo di trasmissione** si calcola con la divisione tra la lunghezza del pacchetto e tra il bit rate.

Il tasso di utilizzo della linea è il rapporto tra il tempo di trasmissione diviso il tasso trasmissivo più il ritardo di propagazione. Il risultato di questa operazione sarà il tempo di utilizzo della linea per trasportare un pacchetto di una certa dimensione su un mezzo trasmissivo che viaggia ad una certa velocità e con un certo ritardo di propagazione.

Prestazioni di rdt3.0

- rdt3.0 funziona, ma le prestazioni non sono apprezzabili
- esempio: collegamento da 1 Gbps, ritardo di propagazione 15 ms, pacchetti da 1 KB:

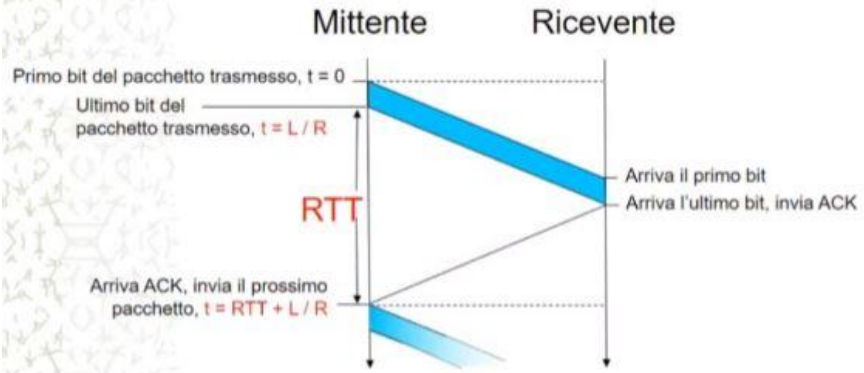
$$T_{trasm} = \frac{L \text{ (lunghezza del pacchetto in bit)}}{R \text{ (tasso trasmissivo, bps)}} = \frac{8 \text{ kb/pacc}}{10^9 \text{ b/sec}} = 8 \text{ microsec}$$

$$U_{mitt} = \frac{L/R}{RTT + L/R} = \frac{0{,}008}{30{,}008} = 0{,}00027 \text{ microsec}$$

- U_{mitt} : utilizzo è la frazione di tempo in cui il mittente è occupato nell'invio di bit
- Un pacchetto da 1 KB ogni 30 msec -> throughput di 33 kB/sec in un collegamento da 1 Gbps
- Il protocollo di rete limita l'uso delle risorse fisiche!

56

rdt3.0: funzionamento con stop-and-wait

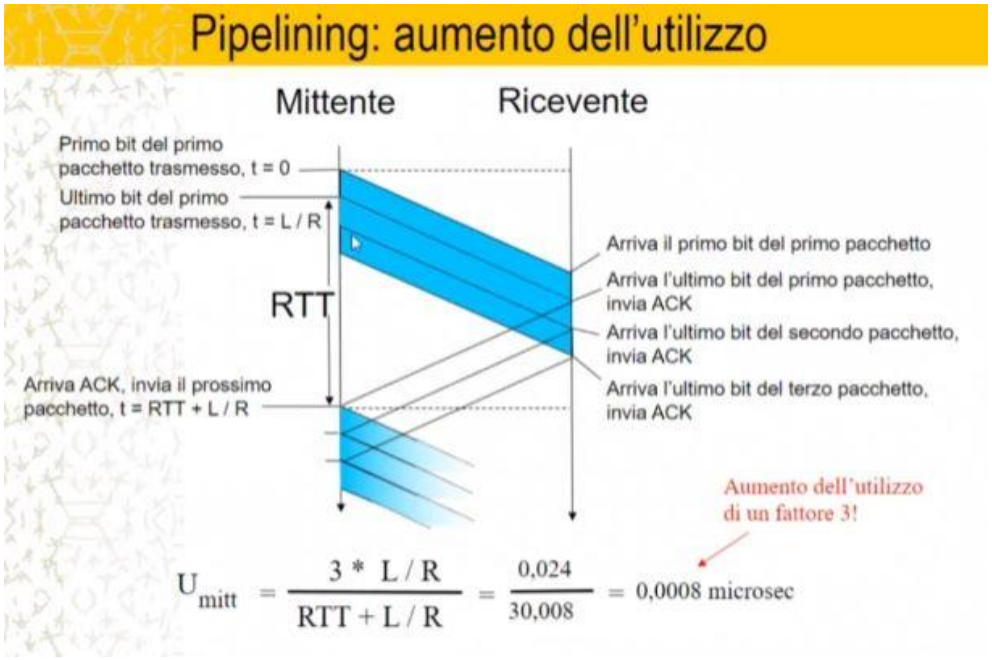


$$U_{mitt} = \frac{L/R}{RTT + L/R} = \frac{0{,}008}{30{,}008} = 0{,}00027 \text{ microsec}$$

57

Protocolli con pipeline

Il mittente non aspetta di ricevere l'ACK, ma mentre lo aspetto comincio ad inviare altri messaggi.



Protocolli a finestra scorrevole

Abbiamo visto che i frame vengono numerati con il "numero di frequenza" che viene scritto nell'header il quale avrà un numero limitato di *n* bit a disposizione per i vari flag. Con *n* bit a disposizione posso creare massimo 2^n-1 numeri di sequenza, ma se ci sono più di 2^n-1 frame occorre riprendere la numerazione da 0, ciò comporta che si possono avere più frame con gli stessi numeri di frequenza senza poter capire se è un frame nuovo o un duplicato.

Per risolvere questa problematica si usa una **sliding window**, una finestra che raccoglie una certa quantità di numeri di sequenza e permette di utilizzare solo quelli all'interno di essa (anche se sono inferiori rispetto a tutti i numeri di sequenza a noi disponibili con *n* bit).

Ma quanto dovrebbe essere grande *n*?

Vogliamo *n* piccolo per essere efficienti e occupare meno spazio nel frame, ma vogliamo un *n* grande per non creare ambiguità fra i frame.

Per il protocollo stop-and-wait basta *n=1*, è un particolare protocollo di sliding window, perché dobbiamo essere capaci solo di distinguere un frame dal successivo.

In trasmissione: si deve tener conto dei frame inviati ma non ancora riscontrati e del numero massimo di frame che possono essere ancora inviati prima di dover fermare la trasmissione: cioè ogniqualvolta io invio un pacchetto, il margine di sinistra della sliding window avanza, per dire che quel numero non lo posso più usare, però non faccio avanzare anche il lato destro perché ancora non ho ricevuto il riscontro (faccio avanzare il lato destro solo quando riceverò l'ACK). Inoltre, si fa avanzare a destra la sliding window solo se sono stati ricevuti i riscontri di tutte le caselle precedenti.

In ricezione: ogniqualvolta riceve un pacchetto, questa volta, è il margine di sinistra che si sposta a destra e ogni volta che si invia l'ACK anche il margine di destra si sposterà a destra. Ovviamente quando il margine destro si sposta in avanti non potrà mai superare la lunghezza della sliding

window impostata, ma può essere di dimensioni inferiori e questo comporta un invio di ACK prima che in trasmissione sia stata azzerata la finestra. Quando la finestra si azzera significa che si devono per forza inviare i riscontri, perché la ricezione è bloccata.

Perché devo utilizzare la sliding window? Perché i numeri di sequenze sono limitati e quindi è necessario riutilizzarli.

I protocolli a finestra scorrevole permettono di inviare più di un frame prima di fermarsi per attendere il riscontro, fino ad un valore massimo W fissato a priori. Poiché in ricezione possono arrivare più frame consecutivi, i frame devono essere numerati per garantire in ricezione che non si siano persi frame: saranno dedicati *n* bit di controllo per la numerazione, ed i frame potranno avere numero da 0 a 2^n-1. In ricezione non è necessario riscontrare tutti i frame: il ricevente può attendere di ricevere un certo numero di frame (fino a W) prima di inviare un solo riscontro cumulativo.

Un problema che si verifica è la **sovrapposizione** dei numeri identificativi tra i frame in attesa di riscontro: capiremo che la finestra scorrevole non può essere della massima grandezza ammissibile (2^n) ma dovrà essere W <= 2^n-1.

Questi protocolli richiedono maggiori risorse di buffer ed una maggiore complessità di calcolo:

- in trasmissione devono essere memorizzati i frame inviati in attesa di riscontro, per poterli ritrasmettere in caso di necessità;
- ad ogni riscontro ricevuto, vengono liberati i buffer relativi ai frame riscontrati, per occuparli con i nuovi frame trasmessi;
- a seconda del protocollo anche in ricezione si deve disporre di buffer, ad esempio per memorizzare frame fuori sequenza;
- ad ogni riscontro inviato, i frame riscontrati vengono passati allo strato di rete ed i relativi buffer vengono liberati per poter accogliere nuovi frame in arrivo.

La dimensione della finestra (W) può essere fissata a priori dal protocollo, ma esistono protocolli che permettono di modificarne il valore dinamicamente tramite informazioni di controllo del protocollo.

Sicuramente la sliding window consente di utilizzare al meglio la linea, ma complica l'implementazione dei protocolli che la usano nel momento in cui si devono gestire gli errori o anche la perdita di dati:

- il trasmittente prima di accorgersi che un frame è stato ricevuto con errore, ha già inviato altri frame;
- in ricezione possono quindi arrivare frame corretti con numero di sequenza successivo ad un frame rigettato (non ricevuto).

(*Controllo errori*)

Esistono due protocolli che gestiscono in modo differente questa situazione: **go-back-N** e **selective reject**. Questi protocolli prevedono l'invio sia di frame ACK (o RR) (per riscontrare un frame) che NAK (o REJ) utilizzato per informare il trasmittente che è stato ricevuto un frame fuori sequenza. Sia gli ACK che i REJ riportano l'indicazione del numero di sequenza del frame che è atteso in ricezione (quello successivo all'ultimo riscontrato). Questi protocolli implementano anche frame di controllo **RNR** (Receiver Not Ready) che impongono al trasmittente di fermarsi fino alla ricezione di un nuovo RR; questi possono essere utilizzati come ulteriore controllo di flusso, per gestire situazioni non di errore ma di **congestione** o temporanea sospensione dell'attività in ricezione.

Go-back-N

Quando il trasmettitore invia più pacchetti, il ricevitore non va a riscontrare un pacchetto per volta ma va a riscontrare un unico pacchetto cumulativo che indica che tutti i precedenti pacchetti ricevuti erano buoni. Come trasmettitore dovrò reinviare, dopo la fine del clock impostato, i pacchetti di cui non ho ricevuto un riscontro. Senza dimenticare che bisogna attivare i timer per tutti: in questo caso si richiede un timer per il primo pacchetto della finestra in transito. Se interviene un timeout, il trasmettitore rispedisce tutti i pacchetti già spediti ma senza un riscontro.

Perché si utilizza il go-back-N?

Dal di vista costruttivo è semplice da costruire: avendo un solo riscontro da memorizzare necessito di un solo buffer.

In ricezione se un pacchetto con un numero di sequenza "*n*" è ricevuto correttamente ed è in ordine, il ricevitore invia un ACK cumulativo per il pacchetto *n* ed invia i dati allo strato superiore. In tutti gli altri casi (ovvero quando i pacchetti non sono arrivati in ordine), il ricevitore scarta il pacchetto e rispedisce un ACK relativo al pacchetto più di recente con l'ordine giusto. Se arriva un pacchetto non in ordine, viene scartato perché si presuppone che il trasmettitore lo rimandi dopo il timeout e quindi sarebbe inutile conservarlo.

Vantaggio: il ricevitore non ha bisogno di buffering perché non ha bisogno di memorizzare alcun pacchetto fuori ordine. L'unica cosa che il ricevitore deve conservare è il numero di sequenza del prossimo pacchetto in ordine.

Esistono due possibilità:

1. **frame errato**: in questo caso il ricevitore scarta il frame:
    a. se il trasmettitore non invia i frame successivi, non accade nulla fino allo scadere del timer, quindi il trasmettitore ricomincia ad inviare frame a partire dal primo non riscontrato;
    b. se invece il trasmettitore continua ad inviare frame, il ricevitore risponde con un REJ dei frame ricevuti, in modo da notificare al trasmettitore che il frame indicato nel REJ è andato perso; al primo REJ ricevuto, il trasmettitore reinvia i frame dal primo non riscontrato.
2. **ACK errato**: in questo caso il ricevitore ha accettato il frame perché era buono:
    a. Se il trasmettitore non invia frame successivi, allo scadere del timer:

i. Il trasmettitore invia nuovamente il frame; il ricevitore lo rifiuta (duplicato) ma invia nuovamente l'ACK;
ii. Alternativamente, al timeout il trasmettitore può inviare un frame di controllo per chiedere conferma dell'ultimo frame ricevuto correttamente, a cui il ricevitore risponde con l'ACK relativo;

b. Se il trasmettitore invia i frame successivi, il ricevitore risponde con l'ACK del frame successivo; poiché gli ACK sono cumulativi, l'ACK del frame successivo riscontra anche quello di cui il trasmettitore non ha ricevuto l'ACK, quindi il trasferimento dati continua senza interruzioni.

Il problema che si riscontra con il GBN è sulla **dimensione della finestra**: il fatto che il GBN invii un riscontro cumulativo obbliga ad avere delle restrizioni sulla dimensione massima della finestra che deve essere $W <= 2^n-1$ perché ci può essere ambiguità.

Es.

$\Rightarrow$ supponiamo di avere n=3 (quindi i numeri da 0 a 7) e scegliamo per W il valore 8.

$\Rightarrow$ il trasmettitore invia il frame 7, e riceve ACK0 (riscontro del frame 7)

$\Rightarrow$ il trasmettitore non può sapere se tutti i frame sono stati ricevuti (ACK0 è il riscontro dell'ultimo frame inviato) o sono stati tutti perduti (ACK0 è il riscontro ripetuto del primo frame inviato precedentemente).

$\Rightarrow$ quindi con $W <= 2^n-1$ non c'è nessuna ambiguità.

Selective reject

Fa un REJ selettivo, può accettare anche i pacchetti fuori ordine perché non manda un ACK cumulativo bensì lo manda singolarmente. A livello di implementazione è più difficile costruirlo perché si necessita di un buffer perché si deve ricordare per quale pacchetto ha inviato l'ACK. Il mittente deve tenere, a sua volta, memoria di quale pacchetto non ha ricevuto il riscontro e soltanto quello deve inviarlo di nuovo e non tutti. Si necessita di un timer per ogni pacchetto non riscontrato. La finestra del mittente ha *N* numeri di sequenza consecutivi ma la restringe di molto.

In questo modo si riduce ulteriormente il numero di frame ritrasmessi, mantenendo la caratteristica di recapitare allo strato di rete i dati nell'ordine corretto. In ricezione i frame fuori ordine (ma dentro la finestra) vengono mantenuti nei buffer fino a che non siano stati ricevuti tutti i frame intermedi.

Quando si ha un frame perduto, il ricevitore riceverà il frame successivo fuori sequenza, al quale risponderà con un ACK relativo al frame perduto. Il trasmettitore non ritrasmette tutti i frame successivi a quello, ma solo quello perduto, quindi proseguirà con la normale sequenza. Il ricevitore ha memorizzato i frame successivi, ed alla ricezione del frame ritrasmesso libererà tutti i buffer inviando un ACK relativo all'ultimo frame ricevuto correttamente. In caso di perdita dell'ACK, sarà il timeout del trasmettitore a generare un frame di sollecito di ACK per il ricevitore, che risponderà di conseguenza.

Anche con questo protocollo andiamo incontro al problema dell'ambiguità solo che qui per risolverlo la finestra dovrà avere grandezza $W <= 2^{n-1}$.

Trasmissioni full-duplex

Quando il canale di comunicazione permette l'invio di dati in entrambe le direzioni contemporaneamente è possibile definire protocolli di comunicazione detti **full duplex**. In caso di linea full duplex il canale trasmette frame di dati in un verso e frame di ACK relativi alla comunicazione nel verso opposto, mischiati tra loro. I frame saranno distinti da un'informazione contenuta nell'header del frame, che etichetta i frame come "dati" o come "frame di controllo".

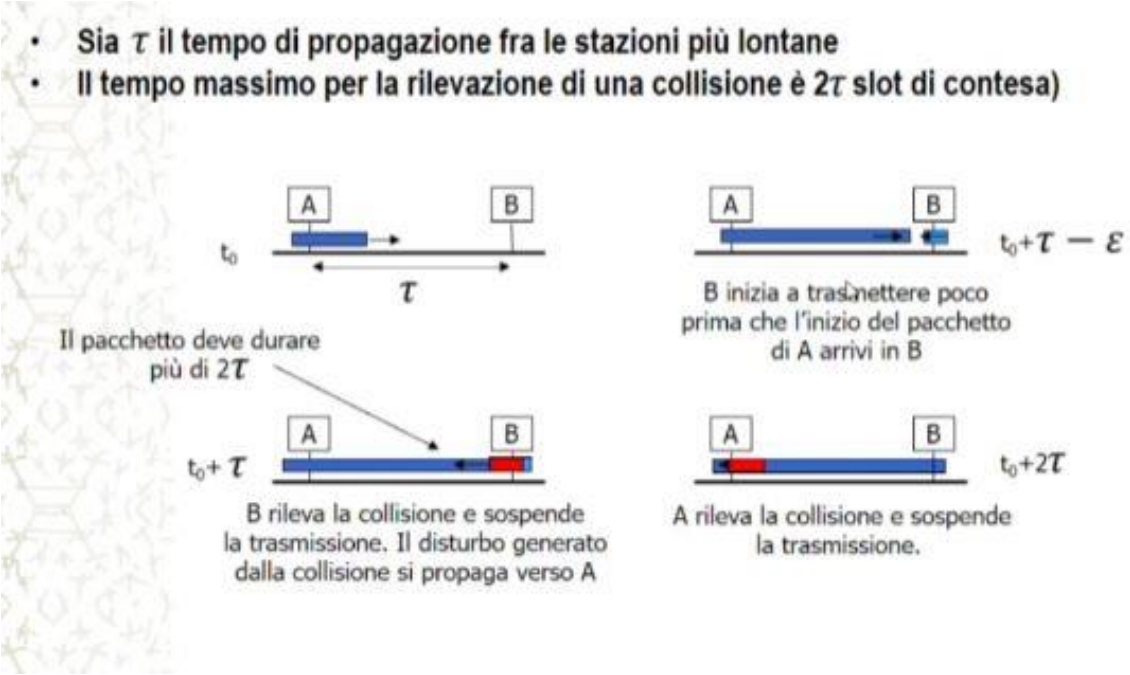
Per motivi di efficienza spesso si utilizza una tecnica, detta **piggybacking**, per evitare di dover costruire e trasmettere un frame di ACK: si dedica un campo nell'header di un frame di dati per trasportare l'ACK della trasmissione in senso inverso. Quando si deve trasmettere un ACK, si aspetta di dover trasmettere un frame di dati che possa trasportare l'informazione di ACK. Se non ci sono dati da inviare, si dovrà comunque inviare un frame di ACK prima che scada il timeout del trasmittente: questo implica di dover utilizzare un altro timer per decidere dopo quanto tempo inviare comunque l'ACK in caso di mancanza di dati da inviare in senso inverso.

È vantaggioso perché consente di sprecare meno banda in quanto sto evitando di far viaggiare sulla rete esclusivamente l'ACK perché lo inserisco all'interno del messaggio che comunque dovevo inviare.

Collegamenti di reti

Esistono due tipi di collegamenti di rete: **collegamento punto-punto (PPP)**, impiegato per le connessioni telefoniche o per collegamenti sulla rete Ethernet tra due host; **collegamento broadcast**, o canale condiviso, può essere la rete Ethernet tradizionale, un cavo coassiale o una rete wireless. Un problema del collegamento broadcast è come i vari nodi si contendono il mezzo trasmissivo che è unico: quindi c'è la necessità di protocolli per consentire la comunicazione a tutti i nodi che condividono un unico mezzo. La problematica che si viene a formare ricade sotto il nome di **collisione**: l'incontro dei messaggi di due nodi che comunicano su un unico mezzo trasmissivo, succede che i due messaggi si confondono tra di loro e i nodi che ricevono l'informazione non sono in grado di decifrarlo.

Quello che incide notevolmente sulla rilevazione delle collisioni è il **ritardo di propagazione** del segnale:



Dato che più host condividono lo stesso bus, essi ricevono tutti i messaggi che vengono scambiati sul bus stesso. Ma nel caso in cui un host debba inviare un messaggio ad un altro specifico host, nel messaggio aggiungerà l'indirizzo univoco (**MAC Address**) dell'host a cui sta inviando il messaggio. Ogni NIC confronta il suo MAC address con quello di destination address sulla trama, se corrispondono, copia il resto della trama, altrimenti la ignora, a meno che la NIC non operi in modalità promiscua (il NIC fa passare ogni trama). Però se due o più host comunicano, anche in due istanti temporali diversi, si avrà una collisione.

Protocolli di accesso multiplo

Per risolvere il problema della collisione, si devono sviluppare dei protocolli, chiamati **protocolli di accesso multiplo**, che fissano le modalità con cui i nodi regolano le loro trasmissioni sul canale condiviso; inoltre, la comunicazione relativa al canale condiviso deve utilizzare lo stesso canale (non c'è un canale esterno che utilizzano per mettersi d'accordo su chi deve comunicare e chi deve ascoltare, devono comunicare per la coordinazione dei messaggi sullo stesso canale in cui inviano

effettivamente le trame). Ovviamente il protocollo si implementa agendo sui flag degli header e del trailer.

Le caratteristiche del protocollo di accesso multiplo ideale sono:

- Se il canale ha una velocità di *R* bps, quando esiste un solo nodo che deve trasmettere, esso può utilizzare il massimo bit rate messo a disposizione dal canale;
- Se, invece, ci sono *M* nodi che devono inviare dati, allora il bit rate del canale si deve suddividere tra tutti i nodi e quindi ogni nodo avrà a disposizione *R/M* bps;
- Il protocollo deve essere *decentralizzato*, non ci devono essere moderatori o nodi master che decidono la comunicazione; vogliamo che tutti gli interlocutori sappiano quando possono parlare. In più non ci deve essere una *sincronizzazione dei clock*, cioè gli interlocutori non si devono sincronizzare secondo un clock per la comunicazione;
- Il protocollo deve essere semplice da implementare.

Esistono tre categorie di protocolli di accesso multiplo:

- **Protocolli a suddivisione del canale** (*channel partitioning*): suddivisione del canale in parti più piccole in maniera tale che tutti possono usufruirne;
- **Protocolli ad accesso casuale** (*random access*): il canale non viene diviso e si può verificare una collisione. I nodi coinvolti ritrasmettono ripetutamente i pacchetti;
- **Protocolli a rotazione** (*taking-turn* o *collision-free*): ciascun nodo ha il suo turno di trasmissione, ma i nodi che hanno molto da trasmettere possono avere turni più lunghi; non rispetta la caratteristica del protocollo ideale senza sincronizzazione.

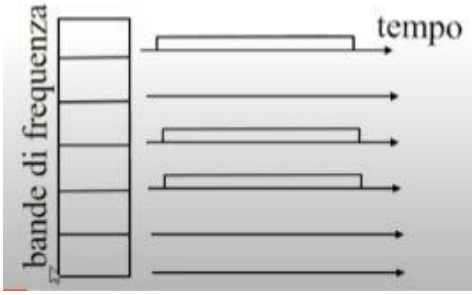
Protocolli a suddivisione del canale:

TDMA

Si suddivide il canale condiviso in *intervalli di tempo*. L'efficienza di questo protocollo non è molto alta perché se c'è un solo host che deve comunicare lo può fare solo dopo che gli altri hanno comunicato, ma se gli altri non avevano niente da comunicare lui aspetta inutilmente.

FDMA

Si suddivide il canale in *bande di frequenza* e a ciascuna stazione è assegnata una banda di frequenza.



Esiste anche un protocollo combinato tra TDMA e FDMA: per ogni sottobanda ho una serie di trasmettitori che funzionano utilizzato il TDMA (è utilizzata dai cellulari).

Protocolli ad accesso casuale

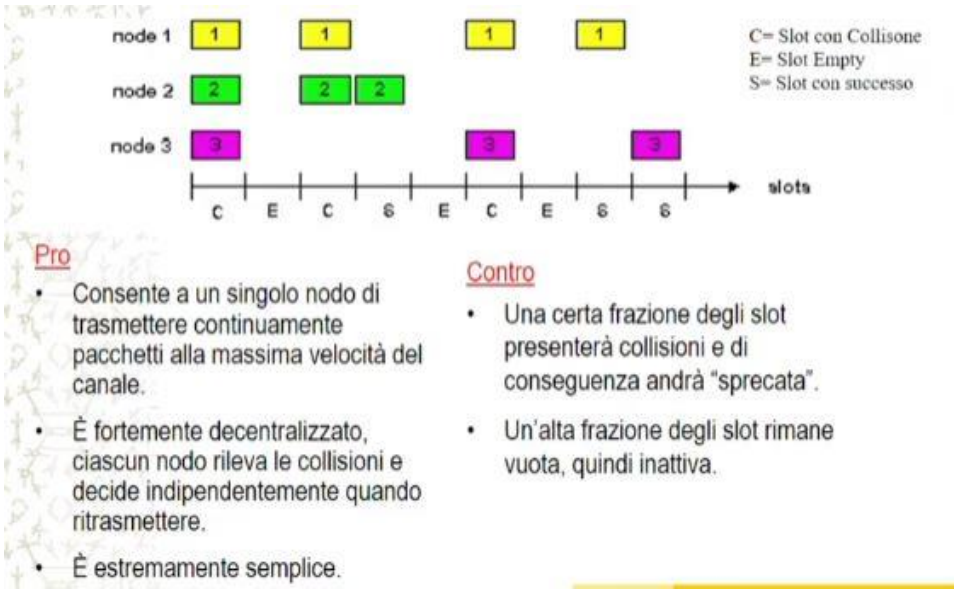
Quando un nodo deve inviare un pacchetto trasmette sempre alla massima velocità consentita dal canale senza coordinarsi con gli altri nodi: quindi c'è una grossa probabilità che ci sia una collisione. Il protocollo ad accesso casuale cerca di affrontare questo problema definendo: come rilevare un'eventuale collisione e come ritrasmettere se si è verificata una collisione.

I protocolli ad accesso casuale più usati sono: slotted ALOHA, ALOHA, CSMA, CSMA/CD, CSMA/CA.

Slotted ALOHA

Il tempo è suddiviso in **slot** e ognuno si essi equivale al tempo di trasmissione di un pacchetto: la larghezza di questi slot avrà un tempo necessario per comprendere se c'è stata o meno una collisione; quindi, dovrà essere grande almeno 2 tau. I nodi iniziano la trasmissione dei pacchetti che hanno tutti la stessa dimensione, solo all'inizio degli slot (non possono iniziare a comunicare nel mezzo di uno slot); in più tutti i nodi sono sincronizzati. Se in uno slot due o più pacchetti collidono, i nodi coinvolti rilevano l'evento prima del termine dello slot e bloccano la comunicazione.

Quando a un nodo arriva un nuovo pacchetto da spedire, il nodo attende fino all'inizio dello slot successivo. Se non si verifica una collisione, il nodo può trasmettere un nuovo pacchetto nello slot successivo. Se, invece, si verifica una collisione il nodo la rileva prima della fine dello slot e ritrasmette con una certa probabilità *p* il suo pacchetto durante gli slot successivi. Se capita che seguendo la legge di una certa probabilità di invio del messaggio, capita che non può inviarlo nello slot successivo allora dovrà aspettare il nuovo slot per poter vedere, sempre seguendo la legge di probabilità, se può inviare o meno il pacchetto.



Non c'è certezza di quando un host deve comunicare e di quando comincerà effettivamente a farlo.

L'efficienza di Slotted Aloha

L'**efficienza** è definita come la frazione di slot vincenti in presenza di un elevato numero di nodi attivi, che hanno sempre un elevato numero pacchetti da spedire.

- Supponiamo N nodi con pacchetti da spedire, ognuno trasmette i pacchetti in uno slot con probabilità *p*.
- La probabilità di successo di un dato nodo = $p(1-p)^{N-1}$ *
- La probabilità che un nodo arbitrario abbia successo = $Np(1-p)^{N-1}$

- Per ottenere la massima efficienza con N nodi attivi, bisogna trovare il valore p* che massimizza $Np(1-p)^{N-1}$
- Per un elevato numero di nodi, ricaviamo il limite di $Np^*(1-p^*)^{N-1}$ per N che tende all'infinito, e otterremo 1/e = 0,37

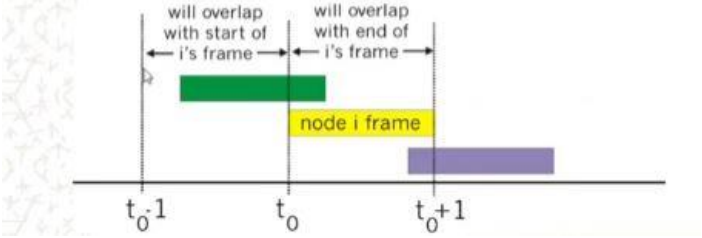
Nel caso migliore: solo il 37% degli slot compie lavoro utile.

*La probabilità di successo di un dato nodo è uguale alla sua probabilità di successo moltiplicata per l'insuccesso di tutti gli altri N-1 nodi. La probabilità di insuccesso è 1-p.

ALOHA puro

L'ALOHA puro, a differenza dello slotted, non è sincronizzato quindi trasmette senza aspettare gli slot successivi: in questo modo potrei trovarmi a cavallo di due frame. Con questo protocollo c'è un'elevata probabilità di collisione.

- Il pacchetto trasmesso a t_0 si sovrappone con la trasmissione dell'altro pacchetto inviato in $[t_0-1, t_0+1]$.



L'efficienza peggiora notevolmente infatti mentre con lo slotted l'efficienza era del 37%, con l'ALOHA puro l'efficienza è uguale al 18%!

P(trasmissione con successo da un dato nodo) = P(il nodo trasmette)·

P(nessun altro nodo trasmette in $[t_0-1,t_0]$ ·

P(nessun altro nodo trasmette in $[t_0,t_0+1]$

$= p \cdot (1-p)^{N-1} \cdot (1-p)^{N-1}$

$= p \cdot (1-p)^{2(N-1)}$

… scegliendo p migliore e lasciando n -> infinito …

$= 1/(2e) = 0,18$

Peggio di prima !

CSMA (accesso multiplo a rilevazione della portante)

Si pone in ascolto prima di trasmettere: se rileva che il canale è libero trasmette l'intero pacchetto, altrimenti, se sul canale stanno già trasmettendo, il nodo aspetta un altro intervallo di tempo. Il problema è che i messaggi si trasmettono con una certa velocità impiegando un certo tempo prima di arrivare a destinazione, quindi, può succedere che il nodo non ha rilevato nessuna portante in quell'istante temporale solo perché, a causa del ritardo di propagazione, ancora non l'ha percepita e inizia a trasmettere causando una collisione.

Per rilevare la portante si utilizza la codifica Manchester per i segnali: i livelli in presenza di segnale sono standardizzati IEEE 802.3: -0.85, +0.85.

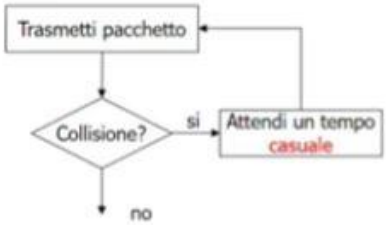
Per rilevare la collisione, i sistemi ascoltano mentre parlano; quindi, ascolta sul canale mentre sta trasmettendo: quindi i sistemi sanno cosa hanno trasmesso e nello stesso tempo cercano di ascoltare: se il segnale che ha ascoltato è esattamente lo stesso di quello che ha trasmesso non c'è nessuna collisione, altrimenti c'è collisione.

CSMA con trasmissioni in collisione

Le collisioni *possono* ancora verificarsi:
Il ritardo di propagazione fa si che due nodi non rilevino la reciproca trasmissione

collisione:
Quando un nodo rileva una collisione, cessa immediatamente la trasmissione.

nota:
La distanza e il ritardo di propagazione giocano un ruolo importante nel determinare la probabilità di collisione.



La collisione viene rilevata ascoltando il canale e verificando che il segnale ricevuto corrisponda a quello trasmesso senza interferenze. Basta una minima sovrapposizione dei due pacchetti per farli andare persi.

Esistono diverse tipologie di questo CSMA:

1-persistente: ascolta il mezzo trasmissivo per capire se c'è qualcuno che sta comunicando, se non comunica nessuno trasmette il pacchetto se c'è la collisione aspetta un tempo casuale e poi ritorna ad ascoltare la portante. Se la portante è occupata rimane in ascolto continuamente, ed appena il canale si libera trasmette con probabilità 1 (trasmette sempre).

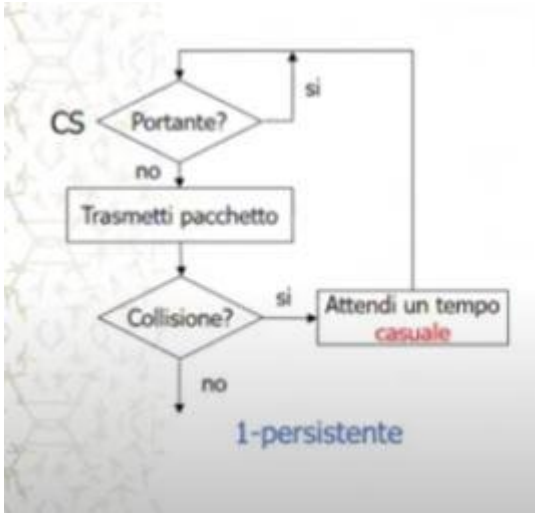
Con questo protocollo acquista grande importanza il ritardo di propagazione del segnale tra due stazioni:

- Quando una stazione inizia a trasmettere una seconda stazione potrebbe voler trasmettere, ed ascolta il canale;
- Se il segnale trasmesso dalla prima stazione non ha ancora avuto il tempo di propagarsi fino alla seconda stazione, questa troverà il canale libero e trasmetterà, generando una collisione.

Maggiore è il ritardo di propagazione, più numerose saranno le collisioni.

- Questa situazione di collisione si presenterà sempre ed indipendentemente dal ritardo di propagazione qualora due stazioni volessero trasmettere mentre una terza sta trasmettendo: alla fine della trasmissione della terza stazione le due stazioni in attesa si metteranno a trasmettere contemporaneamente, e questo succede sempre.

Il CSMA 1-persistente non richiede la sincronizzazione delle stazioni connesse alla rete.



Non persistente: si differenzia dall'1-persistente per il fatto che una stazione, quando vuole trasmettere ma trova il canale occupato, non resta ad ascoltare in continuazione ma attende un tempo casuale e poi si rimette in ascolto.

Questo meccanismo riduce sensibilmente le collisioni dovute al fatto che due stazioni vogliano trasmettere durante la trasmissione di una terza:

- Ora le stazioni attenderanno generalmente tempi diversi prima di ritentare;
- La prima che ritenta troverà il canale libero e trasmetterà;

- La seconda troverà nuovamente il canale occupato, quindi non interferirà ed aspetterà ancora.

Questo protocollo alza notevolmente l'efficienza di utilizzo del canale con l'aumento del carico, cioè con l'aumento delle stazioni connesse. Il problema principale di questo protocollo è che in condizioni di elevato carico il tempo che intercorre tra l'istante in cui la stazione vuole trasmettere e l'istante in cui riesce a trasmettere può crescere enormemente (non c'è certezza della trasmissione).

p-persistente: in quest'ultima versione del protocollo a rilevamento della portante, il tempo è suddiviso in slot temporali come nello slotted ALOHA. In questo caso, chi desidera trasmettere ascolta il canale continuamente e quando lo trova libero:

- Trasmette con probabilità *p*, oppure attende lo slot successivo con probabilità 1-p;
- Allo slot successivo, se libero, trasmette nuovamente con probabilità *p* e aspetta lo slot successivo con probabilità 1-p, e così via;
- In caso di collisione, o se durante i tentativi di trasmissione qualche altra stazione inizia a trasmettere, la stazione attende un tempo causale e ripete l'algoritmo.

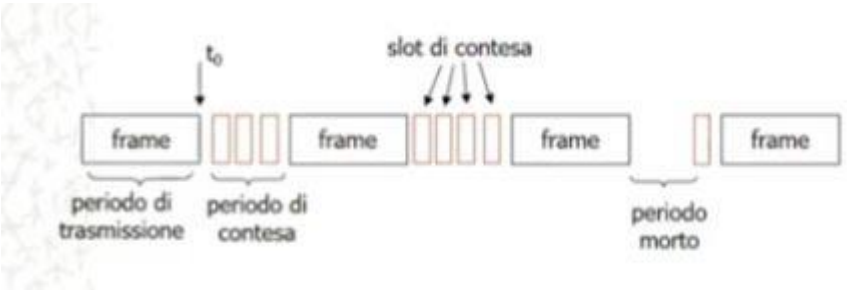
Questo protocollo è una via di mezzo tra il protocollo 1-persistente e quello non persistente. Come nel caso di CSMA non persistente, ad elevato carico e per bassi valori di *p* cresce l'efficienza di utilizzo della linea ma cresce il ritardo di trasmissione rispetto all'arrivo dei dati dallo strato di rete. Per alti valori di *p* l'efficienza di utilizzo della linea decresce rapidamente con l'aumentare del carico.

CSMA/CD (Collision Detection)

Il protocollo opera in tre diverse fasi:

- ***Carrier sense*** (rilevazione della trasmissione): ogni stazione che deve trasmettere ascolta il bus e decide di trasmettere solo se questo è libero (*listen before talking*);
- ***Multiple access***: nonostante il carrier sense è possibile che due stazioni, trovando il mezzo trasmissivo libero, decidano contemporaneamente di trasmettere; la probabilità di questo evento è aumentata dal fatto che il tempo di propagazione dei segnali sul cavo non è nullo, e quindi una stazione può credere che il mezzo sia ancora libero anche quando un'altra ha già iniziato la trasmissione;
- ***Collision detection***: se si verifica la sovrapposizione di due trasmissioni si ha una "collisione"; per rilevarla, ogni stazione, mentre trasmette un pacchetto, ascolta i segnali sul mezzo trasmissivo, confrontandoli con quelli da lei generati (*listen while talking*). Quanto più grande è il tempo di propagazione maggiori saranno i problemi di collisione. Il tempo minimo per accorgerci di una collisione è di 1 tau, mentre il tempo massimo è 2 tau.

Il tempo in cui un host non sta comunicando, ma sta aspettando se può trasmettere o meno, è detto **tempo di contesa**: un tempo all'interno del quale stanno scegliendo chi deve comunicare.



Jamming

Ci sono alcuni protocolli che utilizzano una sequenza di jamming per comunicare a tutte le stazioni di rilevare l'avvenuta collisione. In poche parole, l'host che ha rilevato la collisione invia un segnale a tutti gli altri nodi per comunicare la collisione.

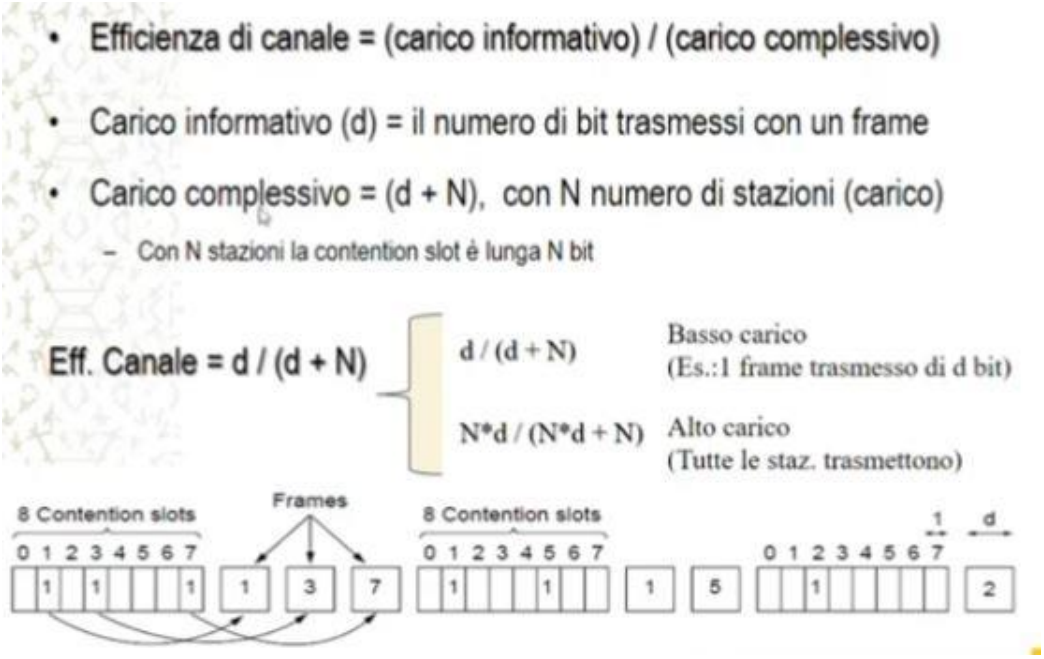
Protocolli collision free

Questi protocolli utilizzano i tempi di contesa per effettuare una prenotazione di chi deve parlare.

- Sulla rete ci sono *N* stazioni, numerate da 0 a N-1;
- Alla fine della trasmissione di un frame inizia un *periodo di contesa*, in cui ogni stazione, andando per ordine di indirizzo (dal più piccolo al più grande), **trasmette un bit che vale 1 se la stazione deve trasmettere, altrimenti trasmette 0**;
- Al termine del periodo di contesa (privo di collisioni in quanto ogni stazione aspetta il suo turno) tutti hanno appreso quali stazioni devono trasmettere, e le trasmissioni procedono un frame alla volta sempre andando per ordine;
- Se una stazione riceve dati da trasmettere quando la fase di prenotazione è terminata, deve attendere il successivo periodo di contesa per prenotare la propria trasmissione.

L'efficienza di questo protocollo è bassa per grandi valori di *N* e basso carico trasmissivo, cioè ci sono tanti nodi nella rete ma sono pochi quelli che vogliono comunicare: in queste condizioni una stazione deve attendere tutti gli *N* bit delle altre stazioni prima di poter trasmettere.

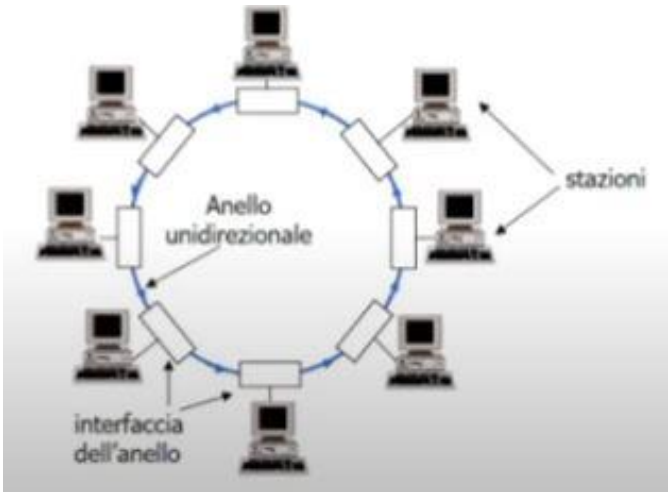
In condizioni di carico elevato il tempo di contesa lo si distribuisce su tutti i nodi della rete che devono trasmettere, riducendo l'inefficienza complessiva del protocollo.



Token ring

Gli host sono posti ad anello e viene fatto passare un **token** tra loro, solo chi è in possesso di questo token può inviare i dati.

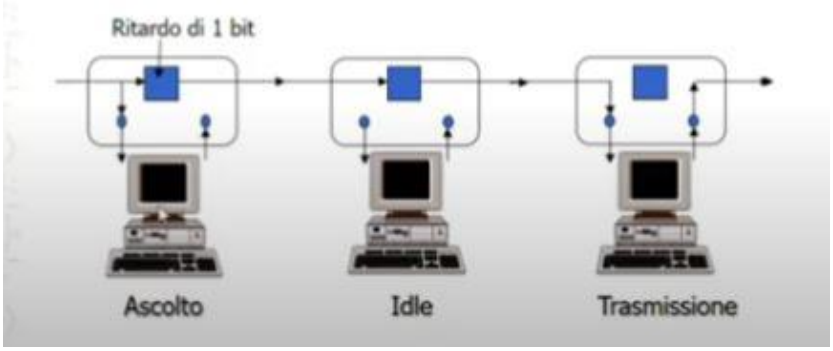
Nella pratica c'è un moderatore al centro (*centro-stella*) che di volta in volta fa passare il token tra i vari host.



Esiste una versione modificata del token ring standardizzata per la trasmissione su doppio anello in fibra ottica, detto FDDI (Fiber Distributed Data Interface) a 100 Mbps.

Visto che questo protocollo fa uso del token, ci sarà l'overhead per la gestione di questo token.

- Il token è composto da una sequenza di bit di una certa lunghezza, ciò significa che se devo inviarlo all'host successivo la mia rete è impegnata nell'invio di questo token;
- L'host riceve il token ma decide di non trasmettere nulla, quindi, dovrà passare il token all'host a lui successivo;
- Quindi se non si inviano dati, si occupa la rete per l'invio reciproco di questo token;
- Esiste un modo per velocizzare il passaggio del token da un host all'altro?
- Ogniqualvolta un host riceve i bit appartenenti al token, l'host valuta il bit e lo passa immediatamente al successivo. Al massimo si dovrà **aspettare il tempo di 1 bit** affinché possa inviare questo tipo di informazione all'host successivo;
- Velocizziamo questa operazione perché non dovremmo aspettare di valutare tutti e 24 i bit del token ma ne analizzeremo solo 1 di esso (che lo identifica) e poi lo passeremo al successivo host.

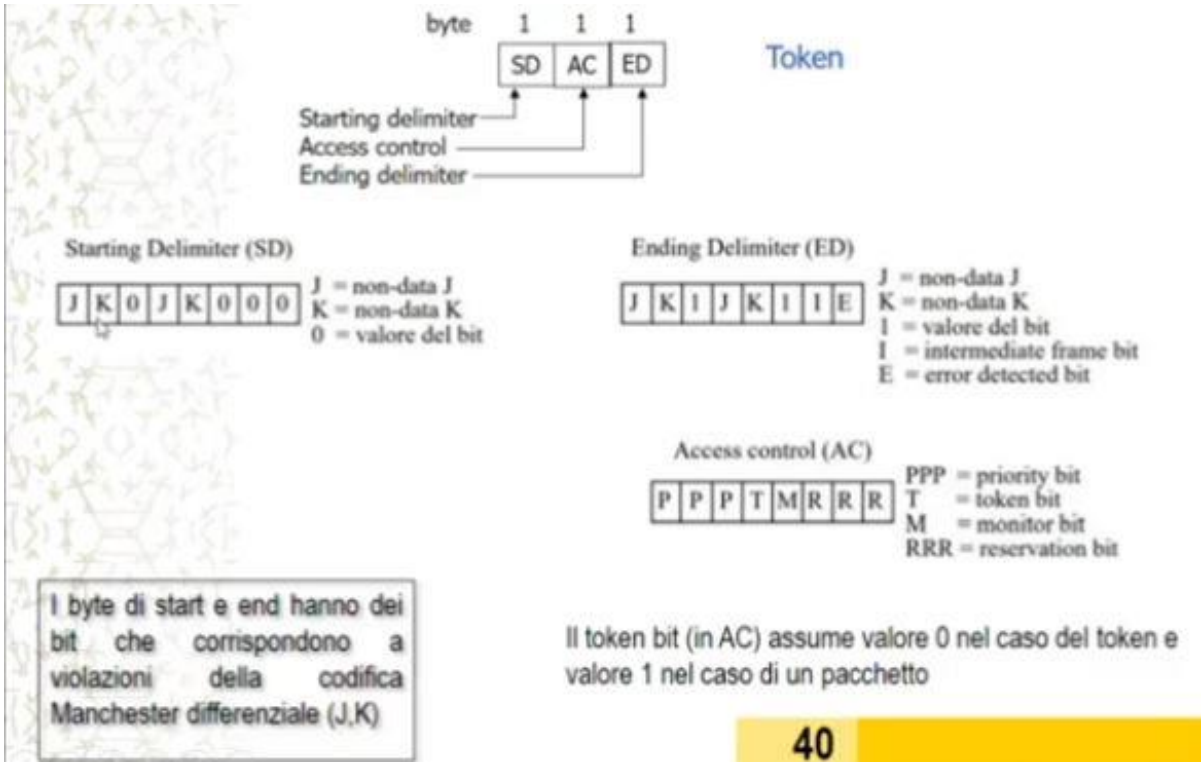


In "ascolto" il bit viene solamente letto; in "idle" non fa nulla, l'host è in modalità passiva; in "trasmissione" prende il bit e lo può modificare per poi rimetterlo sul bus.

Il token

Il token è una sequenza particolare di bit che circola sull'anello quando tutte le stazioni sono inattive. Quando una stazione vuole trasmettere, si impossessa del token e lo rimuove dall'anello. È composta da 3 byte: 1 byte per lo "*Starting delimiter*", 1 byte per "*Ending delimiter*" e 1 byte per "*Access control*". Lo starting e l'ending servono a capire l'inizio e la fine del token ma soprattutto servono per la sincronizzazione (permettono di far capire quando inizia e quando finisce il token in fase di ascolto). Il byte dell'access control è quello che viene modificato quando il token viene acquisito.

I byte di start e end hanno dei bit che corrispondono a violazioni della codifica Manchester:



Il bit **T** nell'AC è il bit del token: se vale 0 vuol dire che il frame che sta passando sul bus si tratta del token; se, invece, vale 1 vuol dire che sta passando un pacchetto di dati. Per ripristinare il token l'host imposta di nuovo questo bit a 0.

Il bit **M** nell'AC è un bit che se posto ad 1 significa che il token è comandato dal monitor (il centro-stella): può capitare che il token sia "orfano", non si capisce più a chi appartiene a causa di una corruzione di esso; andrebbe rimosso dal monitor per poi rimettere in circolo quello giusto.

Il protocollo token ring è una rete che prevede l'utilizzo delle priorità: i bit **RRR** (da 0 a 7 diverse tipologie di prenotazione) vengono chiamati *reservation bit*, sarebbero i bit che ogni host riempie inserendo la priorità della sua prenotazione (a seconda che sia urgente o meno la sua trasmissione). I bit **PPP** rappresentano la priorità attuale: quando il token gira tra i vari host, ognuno di essi confronta la priorità che lui aveva inserito in RRR con quella che è attualmente possibile. Se corrispondono i due campi allora significa che l'host con quella priorità può trasmettere.

Il pacchetto

Il pacchetto

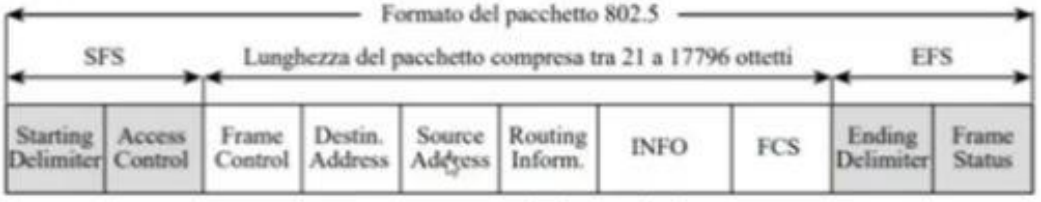
Il token bit (in AC) assume valore 0 nel caso del token e valore 1 nel caso di un pacchetto.

Il paccheto è delimitato da due sequenze:

- SFS (Start of Frame Sequence), indica l'inizio del paccheto
- EFS (End of Frame Sequence) , indica la fine del pacchetto

Il pacchetto vero e proprio inizia dopo la SFS.

- INFO contiene i dati da inviare.
- DA e SA sono rispettivamente il Destination ed il Source Address
- FCS contiene il CRC
- FC definisce il contenuto del pacchetto
- RI contiene le informazioni d'instradamento per una WAN



Esempio di funzionamento

Es. "A" vuole inviare un'immagine al nodo "E"

- "A" sovrascrive il token, inserendo la propria "priorità" di invio all'interno di AC
- Il token gira tra tutti i nodi e torna ad A
- Se la priorità di A risulta essere superiore a quella inserita dagli altri, allora A è autorizzato a trasmettere.
- A sovrascrive il token, che diventa un pacchetto, inserendo DA, SA, … Dati.
- Il pacchetto gira sull'anello fino ad arrivare a destinazione.
- I nodi con indirizzo diverso da DA reinviono il pacchetto al nodo successivo.
- Il nodo con indirizzo DA cattura il pacchetto, preleva i dati e sovrascrive il token con un «riscontro di avvenuta consegna»
- In DA inserisce SA ed invia il pacchetto.
- Il pacchetto girando sull'anello arriva ad A che riceve il riscontro.
- A ripristina il token

Da ciò si evince che questo protocollo non è molto veloce. Un altro problema è **la lunghezza dell'anello**: l'anello deve avere un ritardo sufficiente per contenere un token completo circolante quando tutte le stazioni sono inattive. Considerando che la velocità di propagazione tipica è di 200m/μs, se la velocità di trasmissione è di R Mbps, ogni bit occupa 200/R m (metri). Se R fosse stata 200 Mbps la lunghezza del ring sarebbe dovuta essere di almeno 1 metro.

In **termini di efficienza**, il token ring è poco efficiente in condizioni di basso carico perché la stazione che deve trasmettere deve attendere di ricevere il token (o in generale deve attendere il suo turno) prima di poterlo fare, anche se il canale non è occupato.

In condizioni di carico elevato, quando tutti vogliono trasmettere, l'efficienza del protocollo sfiora l'unità perché il solo overhead è dovuto alla necessità che ha una stazione di identificare il token prima di poter trasmettere.

La rete Ethernet (ethernet è uno standard)

Esistono tanti protocolli, se un host ne utilizza uno e un altro host ne utilizza uno diverso, ognuno parla in un modo suo e nessuno capirebbe un cazzo.

La **IEEE**, che è un ente internazionale per le standardizzazioni, ha dato vita ad un progetto **IEEE 802** che definisce un insieme di standard per le LAN e le MAN, relativamente ai livelli data link e fisico. Quando le prime LAN cominciarono a diffondersi, l'IEEE decise di costituire sei comitati per studiare il problema della standardizzazione delle LAN e delle MAN, complessivamente raccolti nel progetto IEEE 802.

Tali comitati sono:

- 802.1, *Overview*, *Architecture*, *Bridging* and *Management*: parla in generale di come devo essere fatte le reti a livello datalink;
- 802.2, *Logical Link Control*;
- 802.3, *CSMA/CD*;
- 802.4, *Token Bus*;
- 802.5, *Token Ring*;
- 802.6, *MAN – DQDB* (Distributed Queue, Dual Bus).

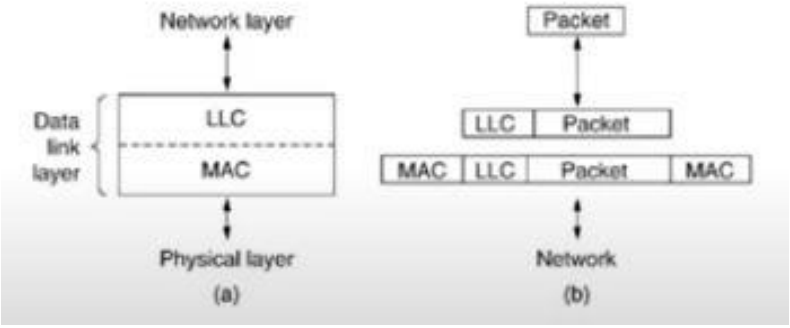
Esistono tanti altri comitati tra i quali quello che ci interessa è 802.11 *Wireless LAN Networking*.

MAC e LLC

Lo standard IEEE 802 introduce l'idea che le LAN e le MAN devono fornire un'interfaccia unificata verso il livello Network. Per rendere trasparente il fatto che sto utilizzando un mezzo fisico diverso a tutti i livelli superiori, allora suddivideremo il datalink in due livelli: **LLC** e il **MAC**. LLC (Logical Link Control) fa tutto quello che abbiamo studiato fino ad ora del livello datalink e lo fa senza accorgersene di come è fatto il mezzo trasmissivo; mentre il MAC (Media Access Control) si occupa di come accedere al mezzo trasmissivo.

LLC è un'interfaccia unificata verso il livello rete, comune a tutte le LAN, invece il MAC è peculiare di ciascuna LAN, così come il livello fisico. Lo strato di rete passa i suoi dati al LLC, che aggiunge un suo header con le informazioni di numerazione del frame, riscontro etc. LLC passa al MAC il campo dati che il MAC gestisce con le sue specifiche. In ricezione, il MAC recapita il frame al LLC che rimuove l'header e passa i dati allo strato di rete.

La funzione principale del LLC definito da IEEE è di mascherare allo strato di rete le specifiche dei protocolli 802 utilizzati a livello di MAC, in modo da offrire allo strato superiore un'interfaccia uniforme.



Il **sottolivello MAC** è specifico di goni LAN e risolve il problema della condivisione del mezzo trasmissivo. Il MAC è indispensabile in quanto a livello Data Link le LAN implementano sempre una sottorete trasmissiva di **tipo broadcast** in cui ogni sistema riceve tutti i frame inviati dagli altri. Trasmettere in broadcast implica la soluzione di due problemi:

1. **in trasmissione**, verificare che il canale sia libero prima di trasmettere e risolvere eventuali conflitti di più sistemi che vogliono utilizzare contemporaneamente il canale;
2. **in ricezione**, determinare a quali sistemi è effettivamente destinato il messaggio e quale sistema lo ha generato.

La soluzione del primo problema è data dai vari algoritmi di MAC che, per poter soddisfare il requisito "apparecchiature indipendenti", devono essere algoritmi distribuiti su vari sistemi e non necessitare di un sistema master.

La soluzione del secondo problema implica la presenza di **indirizzi a livello MAC** che trasformino trasmissioni broadcast in:

- trasmissioni punto-a-punto, se l'indirizzo di destinazione indica un singolo sistema;
- trasmissioni punto-gruppo, se l'indirizzo di destinazione indica un gruppo di sistemi;

- trasmissioni effettivamente broadcast, se l'indirizzo di destinazione indica tutti i sistemi.

Questo MAC Address è l'indirizzo della scheda di rete e una macchina può avere più schede di rete. Quindi non si riferisce al computer ma alla scheda di rete e in particolare non sto identificando il processo software che gira sul computer. Con il MAC Address non sto indirizzando un processo che gira sul computer: i computer comunicano tra di loro ma alla fine quelli che comunicano sono i processi che però non usano il MAC Address, con il MAC Address comunicano solo le NIC.

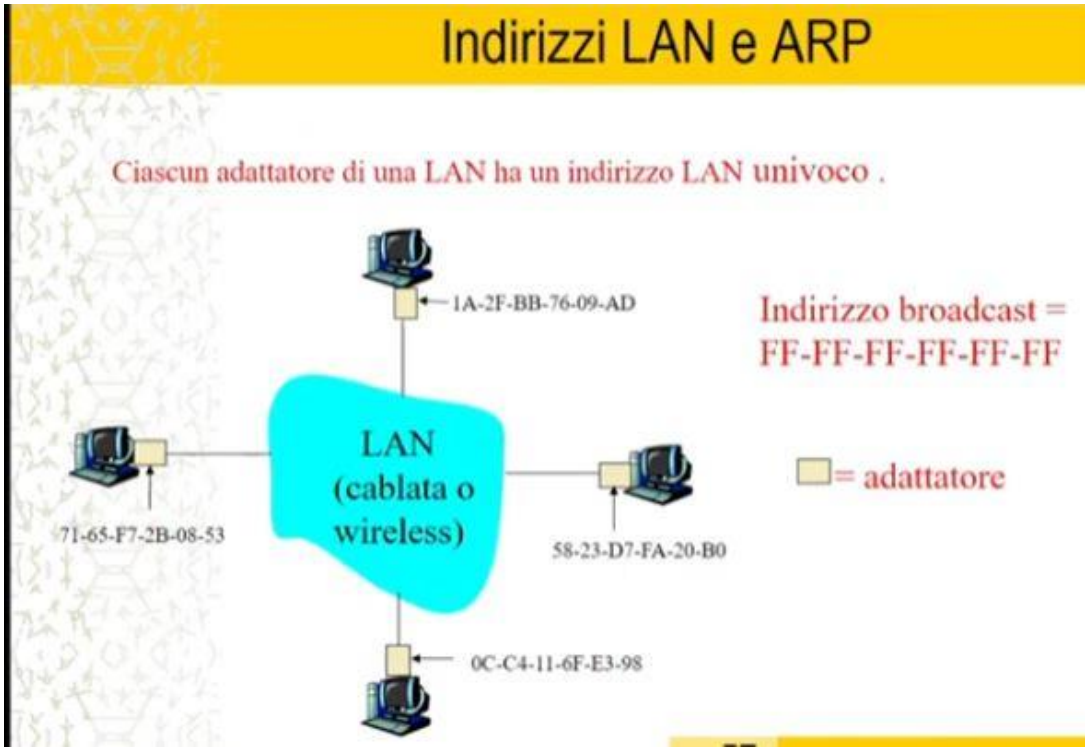
Indirizzi MAC

L'indirizzo MAC è anche noto come indirizzo LAN o fisico o Ethernet. È univoco nel mondo: ha una struttura orizzontale e non varia a seconda del luogo in cui la persona si trasferisce. È formato da **48 bit** rappresentato tipicamente in esadecimale in due modi:

- sestetto separato da ":", per esempio OC:00:1B:F1:18:01
- terzina puntata: 0c00.1bf1.1801

Da questo numero posso risalire al tipo di NIC che si riferisce e quindi a che mezzo fisico sto utilizzando (wireless, ethernet, wired, …).

A livello di rete esiste un altro indirizzo detto **indirizzo IP** a 32 bit che identifica la rete. Esso identifica la posizione attuale di dove si trova una macchina, è analogo all'indirizzo postale di una persona: ha una struttura gerarchica e deve essere aggiornato quando una persona cambia residenza. Invece l'indirizzo MAC è analogo al numero di codice fiscale di una persona.

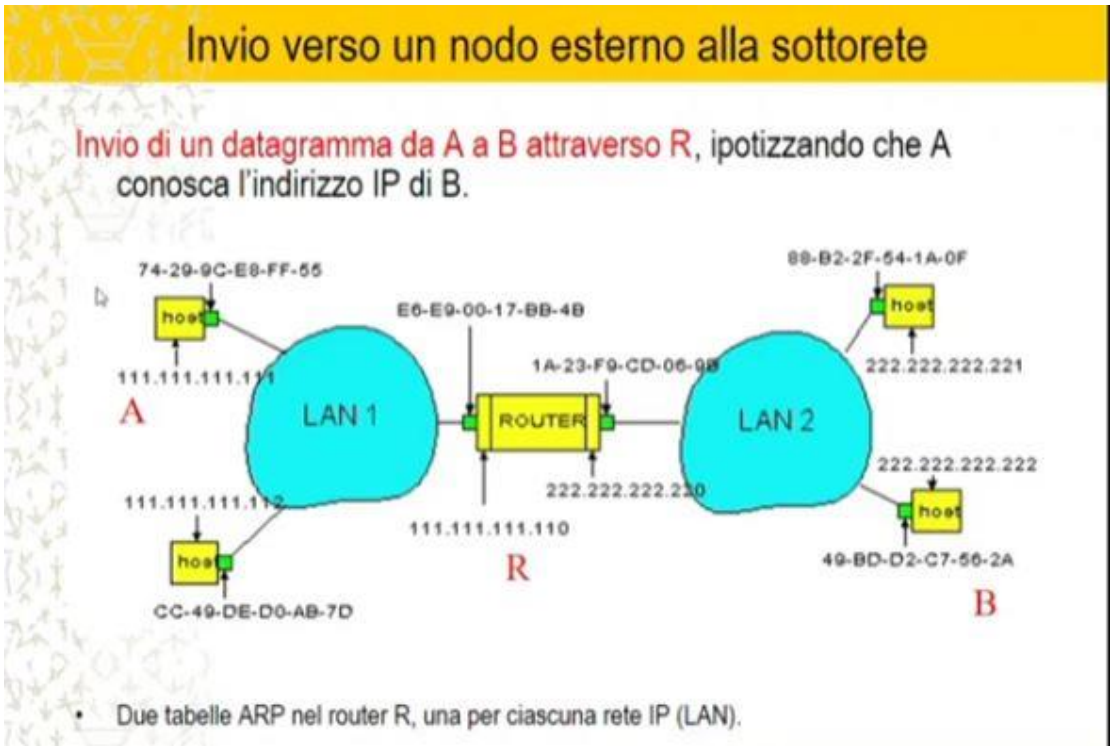


Se l'indirizzo è composto da tutte F allora significa che invierò il messaggio a tutti.

Spesso gli host devono scambiare messaggi identificando il *destination address* non attraverso il MAC Address bensì lo fanno attraverso l'IP.

E come si determina il MAC di un host se si conosce solo l'IP? Dato che il MAC si trova al livello 2 mentre l'IP è al livello.

Ogni host al suo interno ha una tabella di conversione, detta **tabella ARP**, in cui ogni riga contiene l'IP, il MAC e un TTL (tempo di vita, tipico di 20 minuti) che rappresenta il tempo di validità di questa associazione, perché come si sa l'IP può cambiare.



In questo esempio abbiamo due reti LAN (1 e 2) che possono essere connesse attraverso un **router** (dispositivo che mette in comunicazione diverse reti) che offre diverse interfacce tante quante sono le reti che deve connettere (in questo caso ha due interfacce): queste interfacce si comportano che un host.

Ora *A* vuole comunicare con *B*:

- *A* va a leggere la sua ARP e capisce che il suo MAC è associato a un IP;
- Deve inviare il messaggio a *B* che risiede in un'altra rete LAN e quindi ha un IP diverso rispetto ad *A*;
- Visto che *A* non ha associato il MAC di *B*, manda la sua informazione all'interfaccia del router;
- Nel router c'è un'altra tabella ARP (in realtà ha una tabella ARP per ogni interfaccia, in questo caso ne ha due), e si accorge che l'IP appartiene all'altra interfaccia, quindi passa il messaggio all'altra interfaccia e tramite la sua ARP riesce a trovare l'associazione dell'IP ricevuto con il MAC dell'host a cui inviare il messaggio.

Ethernet

La rete Ethernet è un **LAN**, fa uso del protocollo CSMA/CD 1-persistente. Inizialmente era un bus, un cavo elettrico, in cui si affacciavano i vari host composti da un **transceiver** (ricetrasmettitore) sul mezzo fisico usato (se uso le onde radio per comunicare, il transceiver sarà composto da antenne), un'**interfaccia** che sarebbe la NIC ed infine un **controller** che gestisce il tutto.

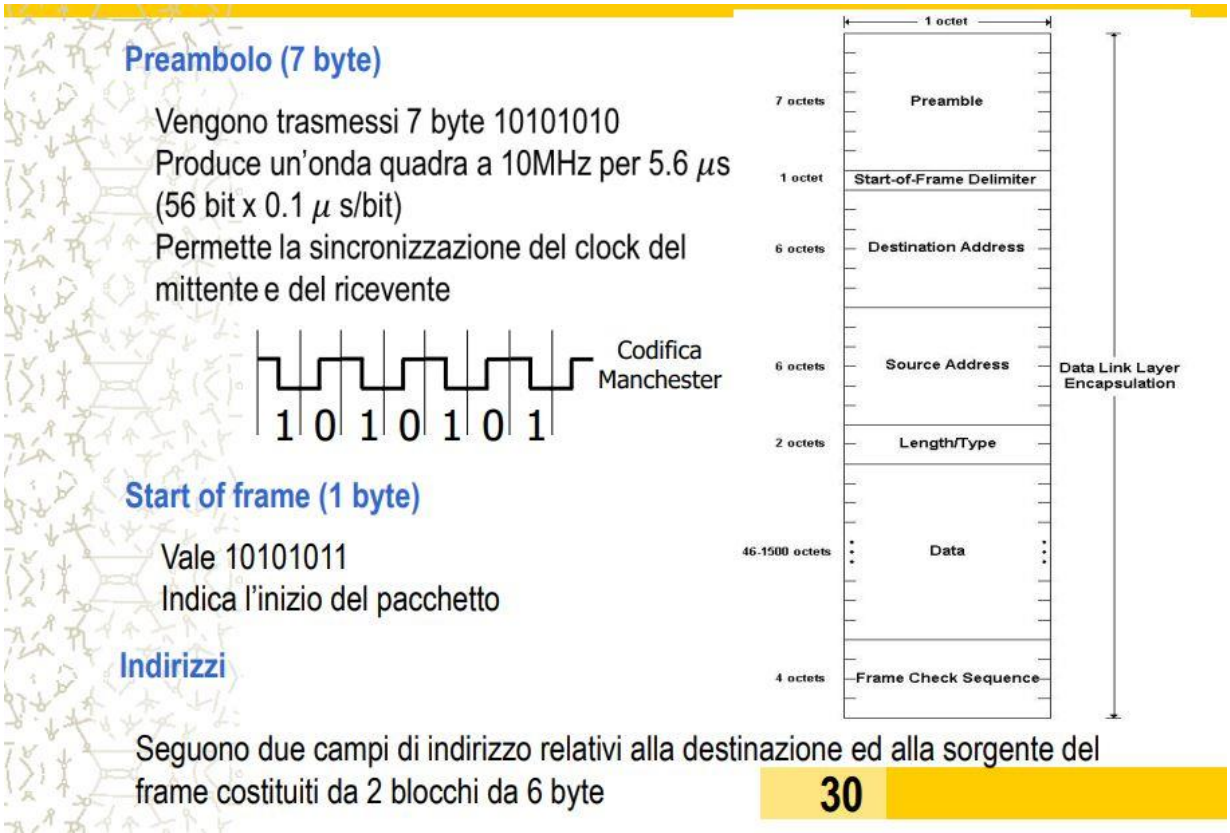
(tutto il modello ISO OSI è implementato parte a livello fisico e parte a livello software che è gestito dal processore della macchina, quindi è un overhead)

Generalmente una rete Ethernet è una rete a stella in cui c'è un **hub** o switch dove tutti gli host sono collegati. A seconda della rete, l'hub può gestire sia le reti cablate che quelle wireless.

Struttura dei pacchetti Ethernet

I pacchetti Ethernet iniziano con un **preambolo** che serve a capire quando inizia e quando finisce un pacchetto ma serve anche per la sincronizzazione. Poi c'è il **destination address** a 48 bit; c'è source **address**, poi c'è il **type**, il campo **data** e il **CRC**.

Il preambolo è una sequenza di bit 10101010, e siccome si utilizza una codifica di tipo Manchester alla fine il tutto diventa un'onda quadra di periodo pari a tau. Questa onda quadra consente ai dispositivi, che si mettono in ascolto in uno specifico momento casuale, di sincronizzarsi.



Indirizzamento Ethernet

Gli indirizzi dei pacchetti Ethernet (destination e source) sono rappresentati su 6 byte (48 bit). Il bit più significativo, cioè il 47esimo, si chiama **bit IG** sta per "individuale" o "gruppo" perché sappiamo che il MAC doveva garantire sia una comunicazione punto-a-punto (imposto il bit IG a 0) e sia una comunicazione multipunto (imposto il bit IG a 1).

Mettendo il bit IG a 1 come faccio poi ad indirizzarmi a più persone?

Si adottano degli altri accorgimenti, cioè si fa sì che più indirizzi appartengano ad una lista che viene scambiata tra i vari host. Quindi nel momento in cui il bit IG è uguale a 1, indipendentemente da ciò che sta nel resto dei 47 bit successivi, se l'host appartiene alla lista riceverà il messaggio.

Ovviamente se tutti e 48 bit sono uguali a 1, questo è il messaggio di broadcast.

Il 46esimo bit invece, viene chiamato **bit GL** che sta per "globale" o "locale": noi sappiamo che il MAC è univoco globalmente e quindi questo bit sarà uguale a 0, però esistono dei software che consentono di cambiare il MAC localmente e quindi il bit GL sarà impostato a 1.

Campo type

Il campo type è composto da 2 byte e consente alla rete Ethernet di supportare vari protocolli di rete.

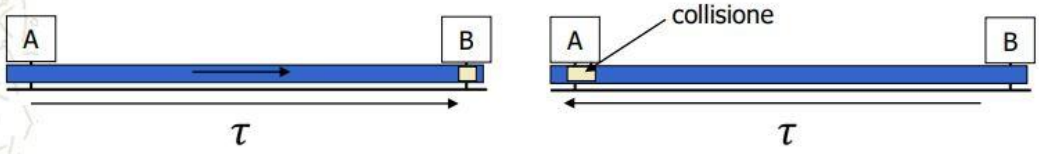
Campo data

Il campo data trasporta le informazioni del protocollo di livello 3 ed ha dimensione variabile, con un limite superiore. La sua dimensione massima è di **1500 byte**, e fa sì che la lunghezza massima del frame Ethernet sia 1518 byte (preambolo escluso): il valore massimo è determinato dal fatto che il transceiver deve ospitare l'intero frame in RAM, ed al momento della definizione dello standard la RAM era più costosa di oggi.

Lo standard prevede che un frame Ethernet non possa essere inferiore a 64 byte e quindi se togliamo i 18 byte dei vari campi, rimane che il campo data deve essere minimo di **46 byte**: se capita che non ho nessun dato da trasmettere o i dati non arrivano a 46 byte, dopo il campo data devo inserire un **campo di riempimento** affinché si arrivi a 46 byte.

Lunghezza del frame

- **Un frame valido deve essere lungo almeno 64 byte**
- **Se si tolgono i 6+6 riservati agli indirizzi, i 2 per il campo length e i 4 del checksum, il campo dati deve avere almeno 46 byte (eventuale padding)**
- **La lunghezza minima di un pacchetto deve garantire che la trasmissione non termini prima che il primo bit abbia raggiunto l'estremità più lontana e sia tornata indietro una eventuale collisione (per rilevare la collisione)**

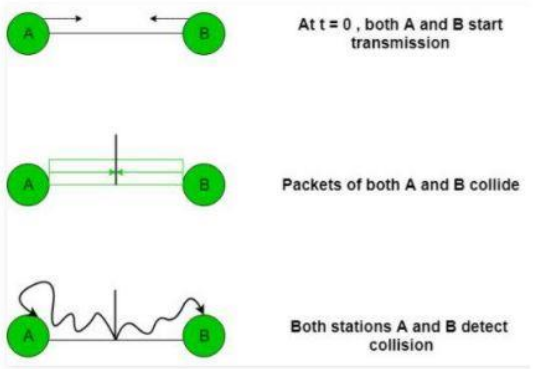


Per una LAN a 10 Mbps di 2.5 Km con 4 ripetitori un pacchetto deve durare almeno 51.2 μs (64 byte)

Exponential Back-off

È l'algoritmo usato dal CSMA/CD Ethernet per calcolare il tempo di attesa dopo una collisione.

- Dopo una collisione, il tempo è diviso in slots di durata $T_{slot}=2\tau$
- A e B scelgono un numero casuale da K={0, 1}. K ≡ finestra di contesa, e aspettano un numero di time slots pari al numero scelto prima di trasmettere
- Se avviene una nuova collisione, perché hanno scelto lo stesso numero, allora la finestra si raddoppia e diventa K={0, 1, 2, 3}
- A e B scelgono un numero dall'insieme K e aspettano un numero di time slots pari al numero scelto prima di iniziare ad ascoltare il canale e trasmettere se libero.



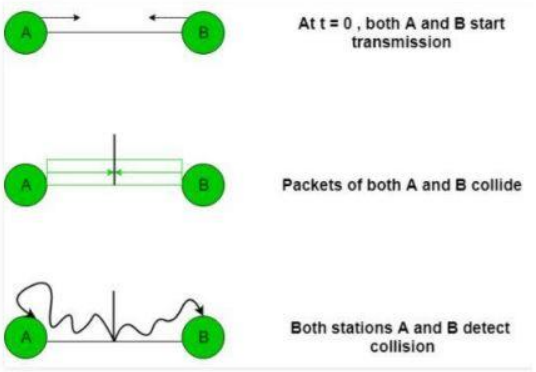
τ: massimo ritardo di propagazione tra A e B

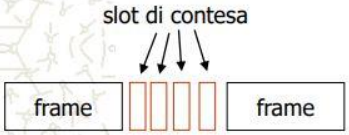
L'host che ha scelto il numero più piccolo trasmette per primo

- Posto n il numero di collisioni, allora
- Il tempo di attesa = $K*T_{slot}$,
  con $K=[0, 2^n-1]$

N.B. il tempo di attesa rappresenta anche lo slot nel quale si riprova a trasmettere

- Nella pratica $K=[0, 2^r-1]$, con $r = \min\{n,10\}$
- Se n =16 si notifica l'errore di trasmissione



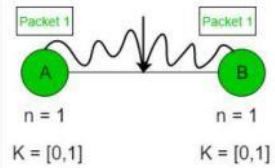


$T_{slot}=2\tau$
Pari a 512 bit - 51,2 µs per 10 Mbps

37

Esempio 1:

- Alla prima collisione K={0, 1}
- Probabilità di trasmettere di A = 1/4 (caso 2 della Tabella)
- Probabilità di trasmettere di B = 1/4 (caso 3 della tabella)
- Probabilità di collisione 2/4 (casi 1 e 4 della tabella)

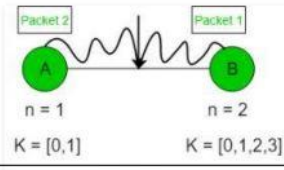


Value of K

	A	B
1	0	0
2	0	1
3	1	0
4	1	1

Esempio 2:

- A è riuscito ad inviare il pkt_1 e prova ad inviare il pkt_2, mentre B tenta di inviare ancora il pkt_1
- Arriva una collisione. Per A è la prima collisione, quindi $K_A=\{0,1\}$.
- Per B è la seconda collisione, quindi $K_B=\{0,1,2,3\}$
- Pr. tras. di A = 5/8 (casi 2,3,4,7,8)
- Pr. tras. di B = 1/8 (caso 5 della tabella)
- Pr. coll. 2/8 (casi 1 e 6 della tabella)



Value of K

	A	B
1	0	0
2	0	1
3	0	2
4	0	3
5	1	0
6	1	1
7	1	2
8	1	3

La probabilità di collisione diminuisce nell'esempio 2

La probabilità di collisione diminuisce in modo esponenziale

La stazione che vince continua a vincere

Prestazioni di Ethernet

Come tutti gli altri protocolli CSMA anche Ethernet presenta le seguenti caratteristiche:

- In condizioni di **basso carico** i tempi di ritardo sono contenuti e l'efficienza assomiglia al CSMA 1-persistente con la miglioria legata al fatto che c'è rilevazione della collisione;

- In condizioni di **carico elevato** crescono le collisioni, ma l'algoritmo di back-off esponenziale fa si che le stazioni mutino il loro comportamento rendendo il protocollo simile ad un CSMA p-persistente con p sempre più piccolo;
- Quindi al crescere del carico l'andamento dell'efficienza tende ad appiattirsi su una percentuale di valore non nullo;
- C'è una forte dipendenza dalla dimensione media dei frame trasmessi: più piccolo è il frame, più pesa l'overhead del periodo di contesa rispetto al periodo di trasmissione riuscita.

Tecnologia Ethernet

L'insieme di protocolli Ethernet domina tuttora saldamente il mercato delle LAN. La velocità di trasmissione originariamente era di 10 Mbit/s su cavo coassiale. Ethernet è evoluta su diversi mezzi trasmissivi fino a 10 Gbit/s, passando da trasmissioni nel dominio elettrico a trasmissioni su fibra. Ethernet, alle diverse velocità e per i diversi mezzi trasmissivi, è sempre stata standardizzata per permettere schede di interfaccia a basso costo, pensate per essere utilizzate in un PC.

Codifica

Sul mezzo condiviso la condizione di "**assenza di trasmissione**" è necessariamente identificata da **assenza di segnale**. Non sono quindi possibili codifiche che utilizzino il segnale a 0 volt per identificare un bit. Proprio per questo o standard Ethernet utilizza la codifica Manchester con segnali a +0.85 V e -0.85 V.