$P_{succ} = N p (1-p)^{N-1}$

$n(t)$ = numero nodi che hanno un pacchetto all'inizio dello slot t.

$0 \leq n(t) \leq N$ $\quad n(t)\uparrow$ ogni volta che "arriva" un nuovo pacchetto

$n(t)\downarrow$ ogni volta che un nodo trasmette con successo

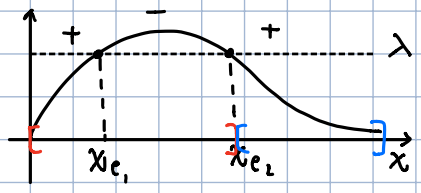
Un modo per calcolare la stabilità: approx $n(t)$ a una f continua nel tempo $n(t) \sim x(t)$ $\downarrow \forall$ succ, $\uparrow \forall$ nuovo pacchetto

$x(t)$

$\frac{dx}{dt} = \lambda = \frac{1}{T} x p (1-p)^{x-1} = f(x) = \lambda - g(x)$ quando $=0$ p. staz.

(eq. diff. che rappr. la dinamica del sistema)

T = durata $\quad P_{succ} = n(t) p (1-p)^{n(t)-1}$



- in questa zona il sistema → 0 (stabile)
- in questa zona il sistema → ∞/N (diverge)

se l'impulso viene dato a x_{e_1}, il rimane nell'intorno ↓ se è troppo forte il finisce a divergere

SLOTTED ALOHA è intrinsecamente instabile, allora il utilizza un algoritmo adattivo che stima n

$n^*(t+1) = \max\{1, n^*(t) - v\}$ se IDLE o succ.

$n^*(t+1) = n^*(t) + v$

SLOTTED ALOHA:
- RANDOMIZZAZIONE
- STABILIZZAZIONE ADATTIVA

Portata slottata in base al livello

MAC $\quad \lambda_{MAC} = \lambda < \frac{V_e}{T} = \frac{1}{e} \cdot \lambda_{PHY}$

PHY $\quad \lambda_{PHY} = 1/T$

slotted ALOHA $\lambda \sim \frac{1}{e}$

CSMA (Carrier Sense Multiple Access)

Visto che slotted aloha ha una portata limitata, si usa quest'altro algoritmo.
Il tempo non è necessariamente slottato, è variabile. Si assume una regola fondamentale "ascolta prima di trasmettere".
In slotted aloha si trasmette con probabilità p, in questo caso prima di trasmettere verifico se il canale è occupato.

1) CCA = clear channel assestment (controllo il canale)

D1) come li fa CCA ?

D2) cosa faccio se il CCA → BUSY ?

Quando arriva un pacchetto la stazione fa CCA per controllare se il canale è occupato. Per fare CCA il livello MAC chiede al livello fisico, quest'ultimo risponde busy o idle.
Come fa l'entità di livello fisico a dire se il canale è libero o occupato?
Ascolta e conclude che il canale è libero se non c'è rumore o comunque è abbastanza basso per trasmettere. Misura il livello medio della potenza ricevuta e confronta questa con potenza di rumore di fondo.

$$\overline{P_{rx}} \gtrless 2 \cdot P_N$$

Se si conclude che il canale è occupato si aspetta che torni libero.

1) CCA
2) id CCA busy wait for IDLE
3)Trasmetti

Una conseguenza di queste regole è che rischiano di essere troppo aggressive. Ogni stazione è indipendente quindi se più stazioni ascoltano nello stesso momento quello stesso canale e contemporaneamente concludono che devono aspettare che torni IDLE si rendono conto insieme di quando io canale torna libero (in quanto il ritardo è trascurabile) e quindi entrano per forza in COLLISIONE.
Se si usano queste regole si ritiene che è poco probabile che accada una cosa del genere, ossia c'è poco traffico, oppure se si pensa che la collisione non porti danni.

Questo metodo viene utilizzato principalmente nelle reti Ethernet.

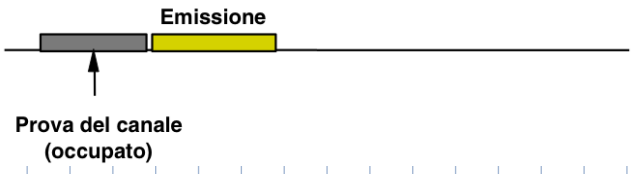
Come possiamo rivedere queste regole per renderle più prudenti?

Si modifica la regola 3, non devo necessariamente trasmettere subito con certezza. Si può trasmettere con probabilità p una volta che si conclude che il canale è idle.

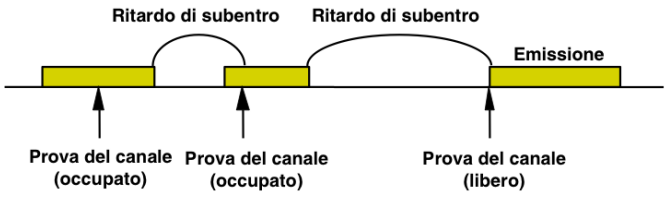
Algoritmo di persistenza

1)CCA
2) if CCA busy wait for idle
3) w.p p to subito, w.p (1-p) aspetti uno “slot” δ e poi riprovi (sempre sondando il canale, l’ascolto continua sempre).

- persistente
- la stazione aspetta che il canale torni libero, quindi trasmette



- non persistente:
- la stazione ritarda l’ emissione di un intervallo di tempo calcolato in base ad un algoritmo di subentro



Come si rendono conto se chi trasmette sta collidendo con un altro?
In Ethernet c’è la collision detection, ossia mentre si trasmette si ascolta anche.
In Wi-Fi non c’è collision detection.
Nel caso non ci sia collision detection , quindi chi trasmette non capisce se c’è qualcun altro a trasmettere nello stesso canale e quindi occupano il canale inutilmente, si aumenta la prudenza modificando il passo 3 e utilizzando una probabilità p più piccola, quindi si aspetta di più. Ogni volta che si va in collisione si diventa più prudenti la volta dopo.
Questo algoritmo di chiama **backoff**.
Un backoff in cui la probabilità diminuisce dimezzandosi ogni volta si chiama **binary exponential backoff (BEB).**

non-pers. CSMA (Wi-Fi)

1. Wait for a new packet from upper layer
2. $k \leftarrow 0$, done $\leftarrow$ FALSE
3. WHILE ($k \leq$ max_retry) OR (NOT done)
   1. CCA $\leftarrow$ channel_state(δ)
   2. IF (CCA == IDLE) THEN IF (rand $\leq p_k$)
      1. $k \leftarrow k + 1$
      2. send(packet)
      3. set(ACK_timer)
      4. check(ACK)
      5. IF (ACK) THEN done $\leftarrow$ TRUE
4. GO TO step 1

nel caso di persistenza c’è la collision detection

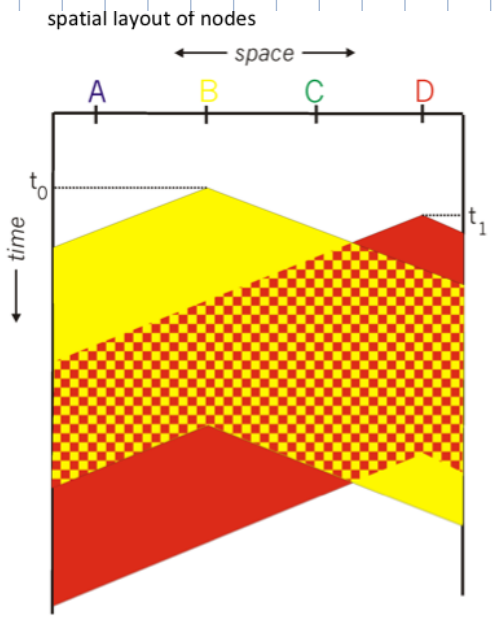
Per entrambi fare binary exponential backoff se si collide poi si riprova con probabilità più piccola.

CSMA COLLISIONS

collisions can still occur:
propagation delay means two nodes may not hear each other's transmission

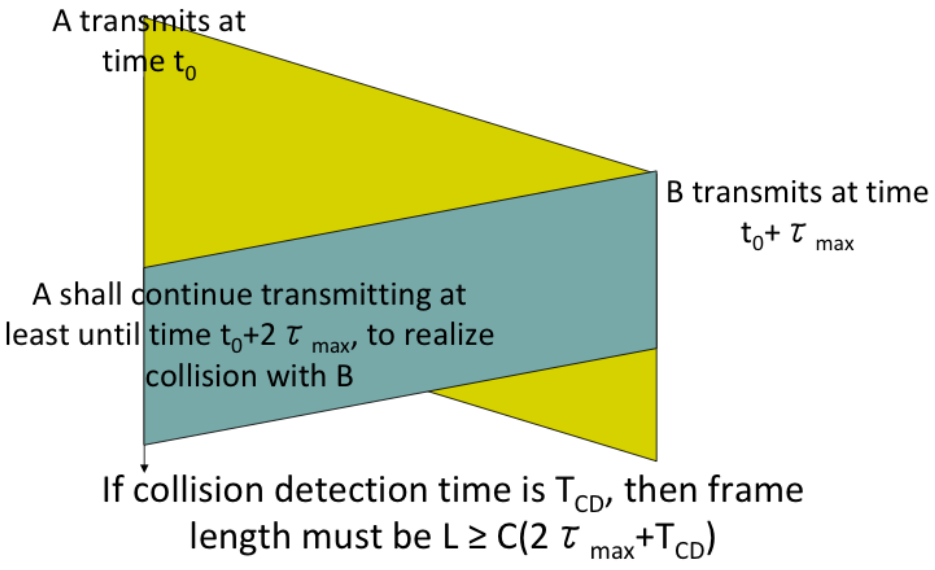
collision:
entire packet transmission time wasted

note:
role of distance & propagation delay in determining collision probability



La trasmissione di B viene indicata con delle linee gialle inclinate perché rappresentano il tempo di propagazione (che contiene ritardi). b sonda il canale al tempo t0 è libero e quindi trasmette. Al tempo stesso D in t1 trova il canale libero perché b ancora non è arrivato e quindi poi entrano in collisione. Non iniziano mai a trasmettere nello stesso istante.

Quindi per quanto tempo bisogna ascoltare per capire se il canale è occupato o comunque rendersi conto che sta per arrivare un segnale?



C'è un intervallo di durata 2taumax, chiamato intervallo di vulnerabilità: ossia se c'è una collisione questa avviene sono in questo intervallo.
Se si rivelano collisioni si fa bindery exponential backoff, in altri casi non ci saranno più collisioni.
La collision detection implica che:

$$T_{Tx} > T_{max}$$

$$T_{tx} = \frac{L}{C_{PHY}}$$

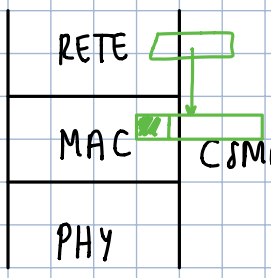
$$\tau_{max} = \frac{d_{max}}{V}$$

$$T_{tx} > 2\tau_{max}$$

Non si possono trasmettere pacchetti troppo piccoli affinché la collision detection avvenga correttamente → in quanto non "aspetterei" abbastanza per vedere se ci sono o no collisioni.

$$\frac{L_{PACC}}{C_{PHY}} \geq 2\tau_{max} \Rightarrow L_{pacc} \geq 2\tau_{max} C_{PHY} \Rightarrow L_{pacc} \geq 2\tau_{max} C_{PHY}$$

Ma come si impone questa condizione di lunghezza del pacchetto?



CSMA/CD $\Rightarrow T_{tx} \geq 2\tau_{max}$

$\Downarrow$

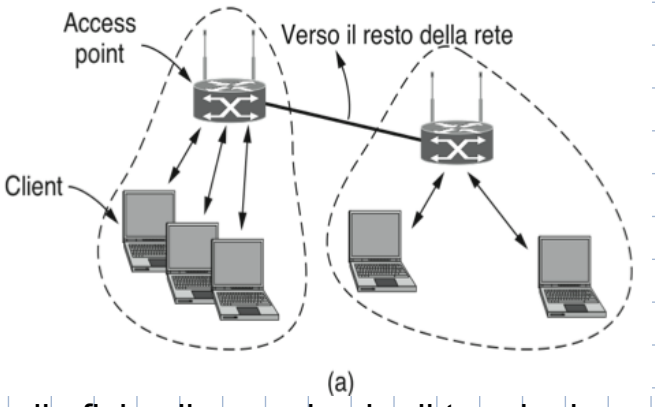
$L \geq L_{min} = 2\tau_{max} C_{PHY}$

Se il livello di rete manda pacchetti troppo piccoli si aggiungono byte di ridondanza, chiamati padding. In questo modo il livello MAC garantisce che la collision detection avvenga correttamente.

Architettura 802.11 p, implementazione CSMA non persistete nel Wi-Fi

CSMA

- Ascolta prima di tx $\Rightarrow$ CCA
  - CD persistente → ethernet
  - NO CD non persistente → Wi-Fi



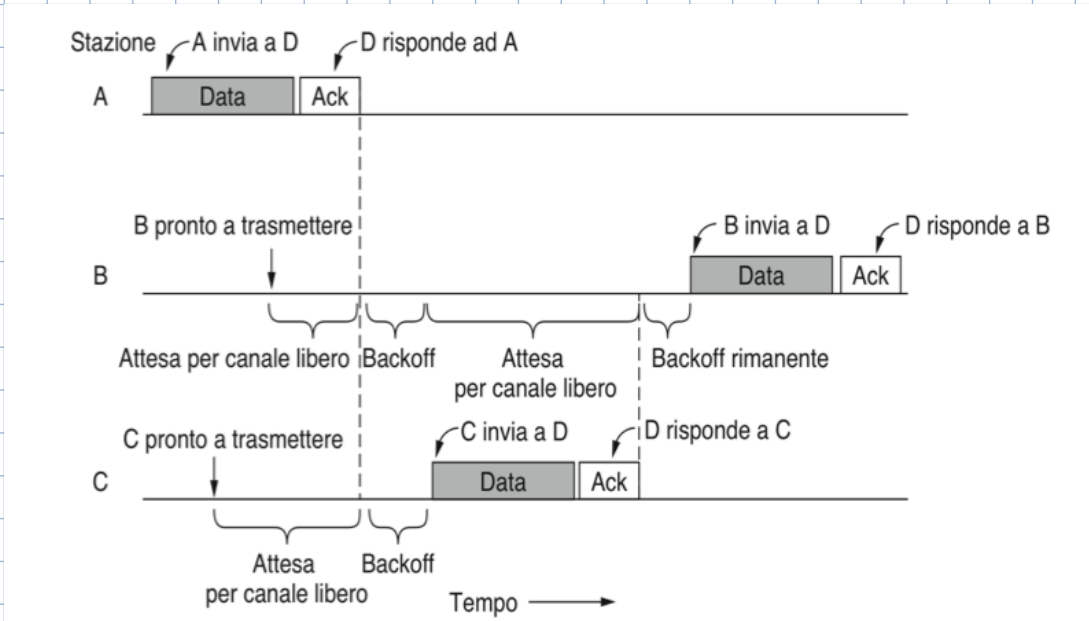
In Wi-Fi ogni terminale comunica con l'access point a livello MAC e riceve pacchetti , a livello fisico il segnale che il terminale manda per comunicare con l'access point viene ricevuto da tutti gli altri terminali a lui intorno che capiscono che il canale è occupato.
Come in una casa dove ci sono cellulare, pc, iPad ecc...
Nei pacchetti che si scambiano c'è anche un campo di intestazione di errori per verificare se i bit sono stati decodificati correttamente.
Perché allora a volte è preferito slotted aloha rispetto a CSMA?

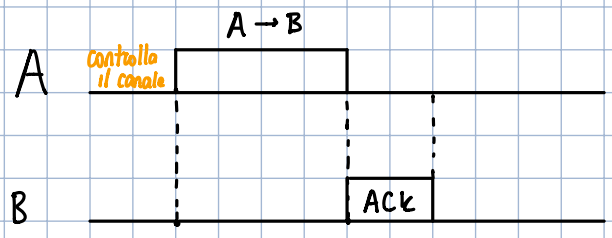
Portata massima normalizzata: $\frac{\lambda_{max\,MAC}}{C_{PHY}}$

- ALOHA: $< \frac{1}{e} \sim 0.37$
- CSMA: < 1

Si può fare sempre CCA affidabilmente? A volte NO quindi si utilizza slotted ALOHA.

In Wi-Fi è un protocollo MAC CSMA non persistente





A controlla il canale, manda il pacchetto destinato a B (ma a livello fisico tutti possono leggerlo), quest'ultimo una volta ricevuto manda un ACK.