

IOT (Internet of things) =insieme di oggetti con abilità computazionali e di comunicazione, connessi tra loro.

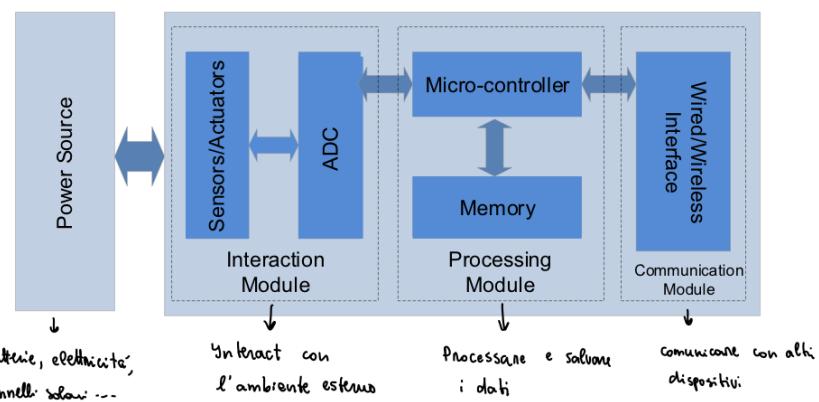
### Smart objects

Venne definito smart object un qualsiasi oggetto reale con le seguenti caratteristiche:

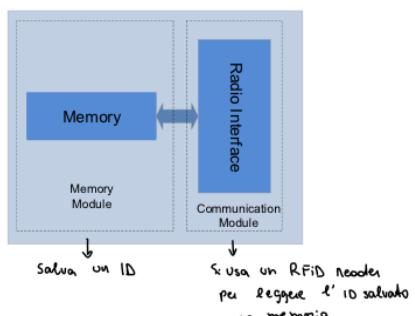
- 1) Capacità di comunicazione
- 2) Capacità computazionali
- 3) Capacità sensoriali / attive

Per fornire le seguenti caratteristiche, gli oggetti hanno bisogno di ENERGIA

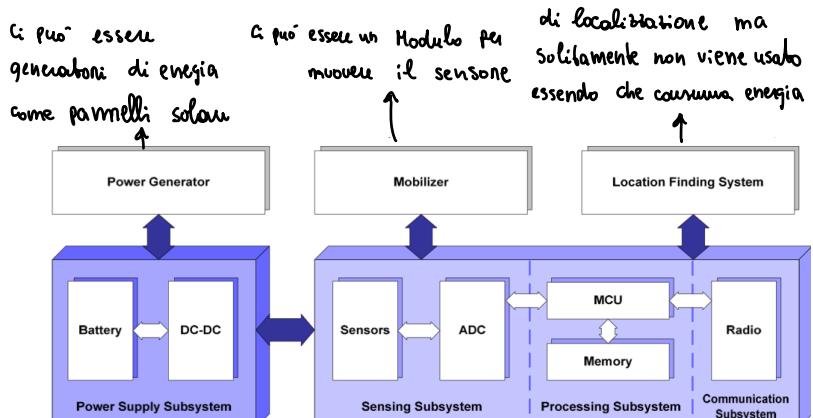
Tipica forma di uno smart object



### RFID



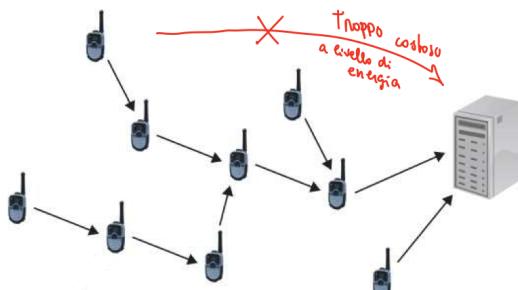
### SENsori



## Smart Object Networks

Sono reti di smart object connessi tra loro per eseguire un task specifico.

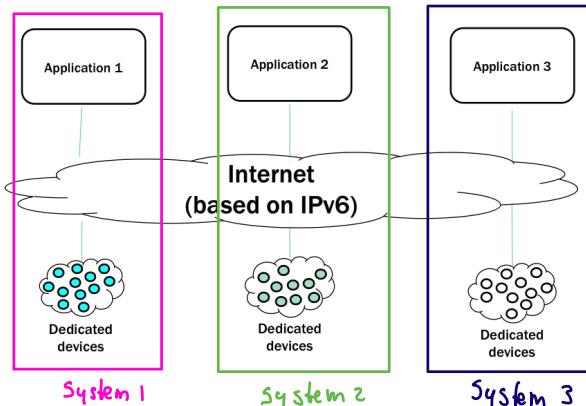
Le connessioni possono essere Wireless o Cabilate.



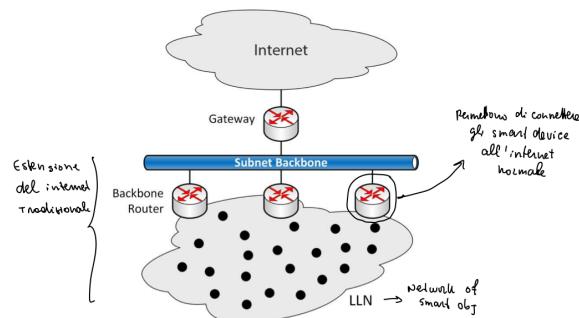
## IoT communication

L'obiettivo dell'IoT è interconnettere tutti gli smart object tra loro.

Per fare ciò viene utilizzato il protocollo IPv6.



## IoT ARCHITECTURE



## Smart Environments

- Smart Cities
- Smart Mobility
- Smart Parking
- Smart Lighting
- Smart Waste/Water Management
- Smart Buildings
- Smart Energy (Smart Grid)
- Smart Healthcare
- Smart Factory (Smart Industry)
- Smart Manufacturing
- Smart \*

The keyword «smart» is extremely popular

Sono ambienti chiamati: cyber-Physical Systems

Cioè sistemi costituiti da:

- Real space (people, cars, machines...)
- Cyber-Space (hw, sw, algorithms...)

ATT: - Oggi sistema IoT è un C-P system

- Non tutti i C-P system sono IoT

Non è detto sia connesso a internet  
Non è detto usi IoT protocols

Per effettuare la comunicazione, l'IETF e ETSI hanno definito degli standard

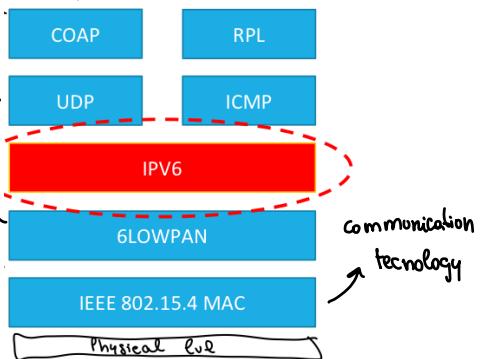
IP for smart obj → insieme di IPv6 based solutions (IETF)

Machine-to-Machine (M2M) → Service architecture (ETSI)

HTTP troppo complesso  
quindi si usa una versione semplificata

UDP perché  
TCP troppo complesso

Adaptation layer  
con lo scopo di  
adattare IPv6 per  
uso IoT  
(comprimere header  
e applicare  
segmentation)



## SMART OBJECTS

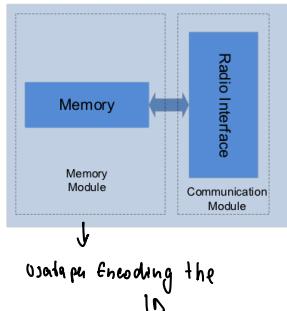
Gli smart object si dividono principalmente in:

- Sensors  $\Rightarrow$  singolo sensore
- Sensor Nodes  $\Rightarrow$  piattaforma con più sensori, microcontrollore, memoria ...
- Sensor / Actuators Node  $\Rightarrow$  piattaforma con più sensori in grado di svolgere azioni

### Passive Tags

#### RFID

è un dispositivo elettronico che necessita di un RFID reader per poter restituire l'ID in esso contenuto.



#### QR-Codes

two dimensional Bar code che salva l'informazione in punti bianchi e neri



USE CASES: Supply management, baggage tagging, animal tagging, smart cities ...

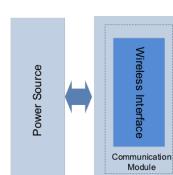
### Active tags

#### Beacons

I beacon permettono agli utenti di performare azioni quando sono vicini ad un beacon

- 1) I beacon trasmette segnali periodici
- 2) I dispositivi mobile ricevono il segnale e capiscono la loro posizione
- 3) I dispositivi mobile performano un'azione sulla base della posizione

I segnali periodici si chiamano Advertisement messages e vengono mandati tramite bluetooth low Energy (BLE).



Gli ADV messages consistono in 6 pezzi di informazione:

### 1) UUID (universal Unique Identifier)

È una stringa di 16 byte usata per identificare un grosso gruppo di beacon, solitamente l'identità di riferimento per quel beacon (es: UNIPI)

### 2) Major

È una stringa di 2 byte usata per identificare un sottogruppo di beacon all'interno del gruppo principale (es: Edificio A)

### 3) Minor

È una stringa di 2 byte usata per identificare il singolo beacon (es: Stanza 10)

### 4) Tx Power

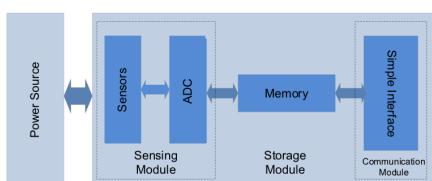
Usato per determinare la distanza dal beacon

USE CASES : localization in buildings, Retail, tracking, Indoor Navigation ..

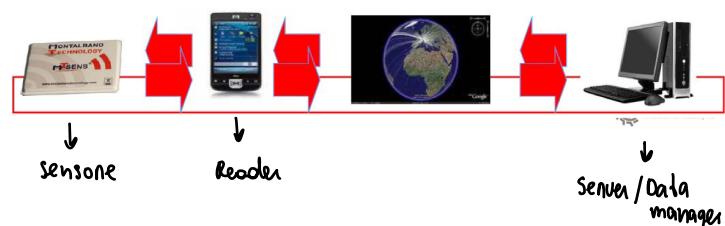
## Semi - Passive sensors (Data loggers)

Sono sensori alimentati da energia (solitamente batterie) e misurano e salvano quantifici fisiche (temperatura, umidità...)

Device Architecture



System architecture

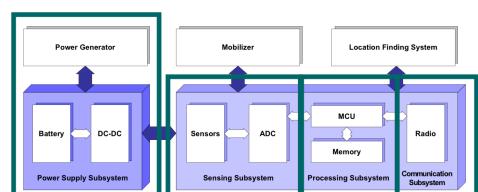


## Sensor nodes

Sono nodi che contengono più sensori, CPU, memoria ...

Hanno bisogno di gestire la concorrenza (event arrival e data processing)

Inoltre devono considerare risorse limitate, la reabilità ..



## Tiny OS (operating system per i sensor nodes)

È stato creato specificatamente per le wireless sensor network.

### Component base architecture

Componente: Entità computazionale composta da un'interfaccia e un frame  
↓  
Per effettuare/ricevere commands/events      ↓  
private variables

Commands: Richiesta di un servizio (Non bloccante)

Event: Command completion ( message o interrupt)

task: Contesto dell'esecuzione, numba fino a completamento

### Concurrency Management

La concorrenza è data da i tasks (dati dai components) e gli eventi (interrupts).

I task sono schedulati in modo FiFo e possono essere anticipati solo dagli eventi.

## Contiki: operating System (operating system per i sensor nodes)

- Limited Memory Footprint
- Event-driven Kernel
- Portability
  - Many different platform supported
    - ⇒ Tmote Sky, Zolertia, RedBee, etc
- C Programming
- Academic and Industrial support
  - Cisco and Atmel are part of the *Contiki project*
- Protothread (optional multi-threading)
- Dynamic Memory Allocation
- TCP/IP stack (*uIP*)
  - Both IPv4 and IPv6
- Power profiling
- Dynamic loading and over-the-air programming
- IPsec
- On-node database Antelope
- Coffee file system

## Low - Power and lossy Networks (LLN)

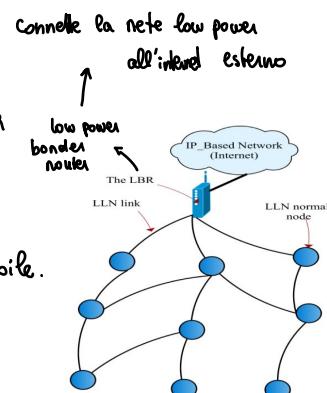
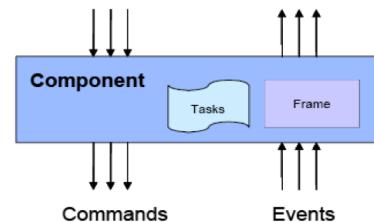
Sono reti composte da molti embedded device con potenza, memoria e risorse computationali limitate.

Sono solitamente caratterizzate da bandwidth limitata e imprevedibile.

Sono quindi caratterizzate dall'essere constrained (vincolate).

**Constrained Nodes:** low power, limited memory, risorse computationali scarse.

**Constrained Networks:** lossy Networks, limited and unpredictable bandwidth, Dynamic topology.



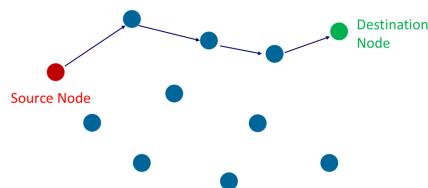
## Communication in LLNs

4 communication pattern dipendono dallo specifico scenario ma sono tutti caratterizzati da:

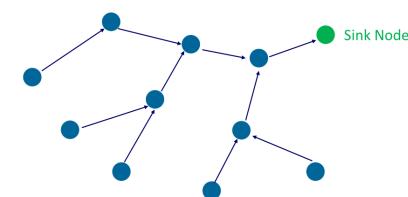
- 1) Collegamenti: inaffidabili (solitamente sono wireless)
- 2) 4 metti di comunicazione sono condivisi e portano a collisioni (Random access protocols)
- 3) la topologia delle reti (networks) è dinamica (ostacoli, meteorological conditions, power consumption)
- 4) la potenza disponibile è limitata e quindi richiede energy-efficient communication  
meccanismi per risparmiare energia

## Communication patterns

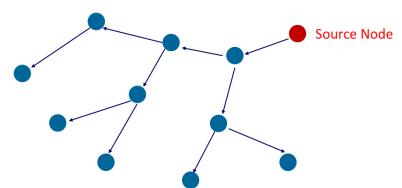
### One - to - One



### Many - to - One



### One - to - Many



### 1) Wireless Sensor Networks (WSNs)

Sono reti composte da un grande numero di piccoli sensor nodes i quali integrano 3 differenti capacità: Sensing, Computing and communication.

La rete processa i dati seguendo il seguente pattern:

- 1) Sensor nodes catturano le physical information
- 2) Processano i dati localmente e/o li inviano a uno o più "collection points"

## Classificazione WSNs

Sink node | base station | gateway | border router

### Topologia

- 1) Static WSN: tutti i sensor nodes sono statici
- 2) Quasi-Static WSN: Alcuni sensori hanno mobilità limitata (es: Relocatable nodes)
- 3) Hobile: Alcuni (o tutti) i sensori sono mobili

## Densità

1) **Dense WSNs**: la distanza tra i sensori vicini è minore del transmission range.

Richiede molta densità  $\Rightarrow$  molti sensori

2) **Spanse WSNs**: la distanza tra sensori vicini è molto più del transmission range.

Il numero dei sensori dipende dalla necessità dell'applicazione.

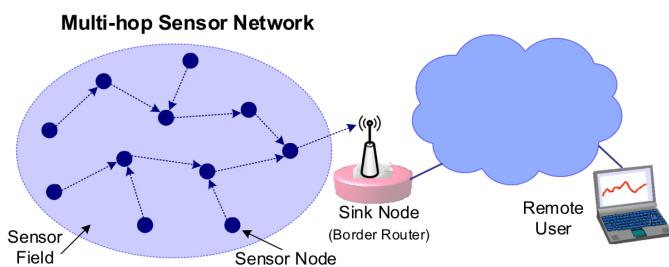
## Data Collection Paradigm

1) **Multi-hop Communication**

a) **Flat**: ogni nodo trasmette i dati

b) **Hierarchical**: solo i super-nodes trasmettono ai border router, gli altri sensori raggiungono il più vicino super-node con 1-hop.

### Flat Multi-hop



Tutti i sensori possono comunicare tra loro.

Richiede una rete densa e con topologia statica.

Svantaggi:

Il numero di hops dal sensore al sink è alto, di conseguenza:

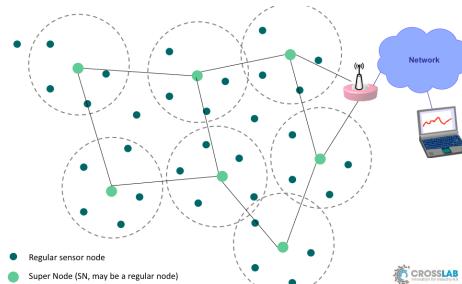
a) Delay sensor  $\rightarrow$  sink alto

b) Bassa affidabilità

c) Alto dispendio di energia

d) I nodi vicini al sink node potrebbero ricevere troppe info congestione

### Hierarchical Multi-hop



I sensori possono comunicare solo con il super node assegnato.

Richiede una rete densa e con topologia statica.

Vantaggi:

a) tipicamente solo 1-hop dal sensore ai super nodes

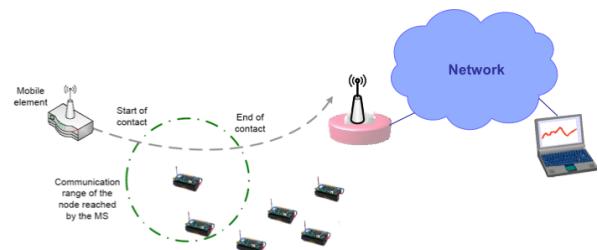
b) Delay sensor  $\rightarrow$  sink limitato

c) Affidabilità maggiore

d) Non abbiamo problemi di congestione essendo che solo i super nodes comunicano tra loro.

## Mobile Data Collection

Senzore che si muove e raccoglie i dati dai vari sensori quando arriva in range



→ Application ⇒ Areas

- Military Applications
- Environmental Monitoring
- Precision Agriculture
- Location/Tracking
- Industrial applications
- Health Monitoring
- Smart Buildings
- Smart Grid
- Smart Cities
- Smart \*
- ...

## 2) Wireless Sensor / Actuator Networks (WSANs)

Sono reti composte da sensori e/o attuatori i quali solitamente comunicano tramite wireless link il quale dà più flessibilità e facilità nel deployment. (In caso di specifiche appl anche il cablaggio può essere utilizzato)

### Network issues

Quali sono i principali Network issues?

- 1) Application specific: I protocolli dovrebbero adattarsi all'application behavior
- 2) Environment - driven: Il data traffic ci si aspetta sia diverso rispetto al human - driven traffic
- 3) Scalability: Alto numero di sensori
- 4) Energy - efficiency: Sensor nodes limitati in potenza, capacità computazionali e memoria
- 5) Dependability: Sensor nodes facile fallimento ⇒ Cambio frequente di topologia
- 6) Data - centricity: L'importanza di un particolare nodo è ridotta (ridondanza). Ci interessano dati specifici non i nodi.

## Energy Management

Abbiamo bisogno di gestire il trade-off tra il network life-time e il consumo di energia.

### Energy harvesting

Processo attraverso il quale viene catturata energia dall'esterno e convertita per essere utilizzata.

L'energia può essere raccolta dall'ambiente esterno in modi differenti:

- 1) Thermal gradients : generare energia dal cambio di temperatura. L'efficienza delle conversioni è il problema principale, soprattutto quando la differenza è piccola.

use-case: wearable sensor

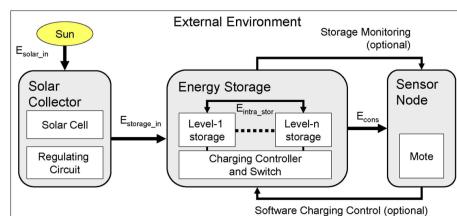
- 2) Thermoelectric generators : convertono direttamente il calore in energia. L'efficienza è limitata dovuto alla high thermal conductivity dei materiali convenzionali.
- 3) Radioactivity : Sorgente di energia per piccoli device, la limitata grandezza del radiating material evita problemi di salute / sicurezza.

- 4) Kinetic Energy: convertire energia dal movimento delle persone

- 5) Radio frequency signals: viene usata la Wireless Energy Transfer per alimentare i sensori

- 6) Vibration Energy

- 7) Solar Cells  $\Rightarrow$



### Helio Mote

Il sistema impone il suo energy environment  
e adatta il suo consumo di energia.

### Concluding Remarks

- Energy harvesting is a very promising approach
  - But, currently the conversion process is not efficient enough
- Can be used to power
  - very simple devices
  - as a complementary power source, e.g., to replenish a battery in the background.
- Energy that can be collected is virtually infinite
  - However, available power is quite limited
- Even when using energy scavenging
  - energetic resources are limited and must be used judiciously
  - Energy harvesting and energy conservation must be used jointly

$$\text{Energy consumption} = \text{Power Consumption} \times \text{Time}$$



## Energy Conservation

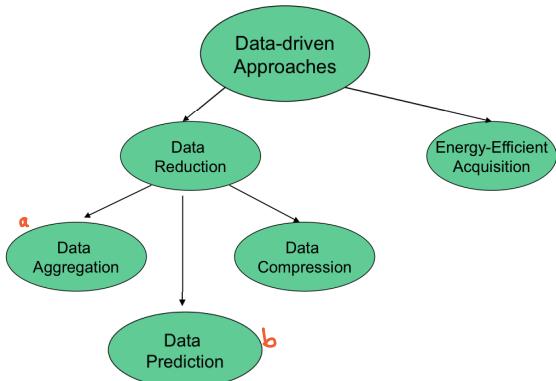
componente che contribuisce maggiormente.

↑ ricevere o trasmettere consumo uguale.

l'obiettivo è ridurre il più possibile le **radio activity**, cioè mantenere il più possibile la radio in sleep mode.

Ci sono tre differenti approcci: **Duty-Cycling**, **Data-Driven**, **Mobility-based**

- 1) **Data Driven Approaches** → ridurre la quantità di dati trasmessi → riduce il tempo di attività della radio



### Energy Efficient acquisition

L'acquisizione deve essere il più basso possibile e dovrebbe soddisfare i requisiti dell'applicazione.

Ci permette di ridurre la quantità di dati raccolti.

- a) **Data Aggregation** (Ci permette di ridurre la quantità di pacchetti sulla rete)

- 1) **Mac-layer Data Aggregation**: Combinare molti pacchetti in un singolo MAC frame

↓

Possibile drawback: delay nell'aspettare; pacchetti degli altri nodi per fare l'aggregazione

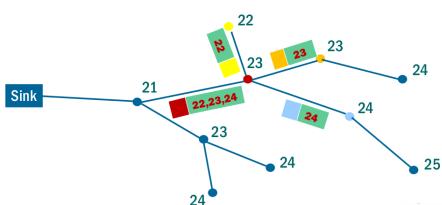
Non richiede la conoscenza del network ed è application-independent

- 2) **Cluster-based Data Aggregation**: Viene fatta l'assunzione che i sensor nodes sono organizzati in clusters.

Tutti i membri all'interno di un cluster mandano i dati al cluster head, il quale aggredisce i dati e li invia al border sensor. Potrebbe usare la conoscenza dell'applicazione

- 3) **Tree-based Data Aggregation**: I dati sono aggregati dalle foglie dell'albero verso la radice.

Questo permette di ridurre l'overhead nelle comunicazioni e ridurre la energy consumption, ma aumentare il delay dovuto all'aspettare i pacchetti. Potrebbe usare la conoscenza dell'applicazione.



Aggregation Factor è il ratio tra i # di bits che devono essere trasmessi con

aggregation e il # bits che devono essere trasmessi senza agg.

$$\alpha = \frac{D_{agg}}{D_{no-agg}} = \frac{H + N \cdot P}{N(H + P)} = 1 - \frac{H}{H + P} \cdot \frac{N-1}{N}$$

$$h = \frac{H}{H + P}$$

$$\alpha = 1 - h \cdot \frac{N-1}{N}$$

$0 < \alpha \leq 1 \rightarrow$  riduce sempre la quantità di dati inviati

dove:

N = packages

P = payload

H = header

Assuming H = 9 bytes (IEEE 802.15.4 MAC Frame)

Aggregated packets	Payload Size (bytes)							
	1	2	4	8	16	32	64	118
1	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
2	0.55	0.59	0.65	0.74	0.82	0.89		
3	0.40	0.45	0.54	0.65	0.76	0.85		
4	0.33	0.39	0.48	0.60	0.73			
5	0.28	0.35	0.45	0.58	0.71			
6	0.25	0.32	0.42	0.56	0.70			
7	0.23	0.30	0.41	0.55				
8	0.21	0.28	0.39	0.54				
9	0.20	0.27	0.38	0.53				
10	0.19	0.26	0.38	0.52				

### b) Model Driven Data Prediction



rinché non cambia il trend viene usato il modello, da parte del sink, per prevedere i dati

Invece di mandare tutti i dati al sink, vengono

inviai solo e soltanto quando il trend cambia.

Venne esatto un modello che descrive i dati.

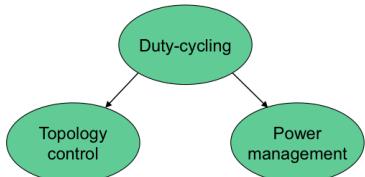
limitazioni del data-driven approach → Non è detto che il lifetime ottenuto sia significativo, dipende dalle appl.

- 1) Trasmettere un messaggio richiede la stessa energia, non dipende dalla grandezza
- 2) Il costo energetico nel mantenere la senson network non può essere evitato
- 3) La riduzione dei dati elimina la ridondanza, abbiamo quindi bisogno di una 100% communication reliability.

$$\text{Duty cycle} = \frac{\text{Pulse active time}}{\text{signal period}}$$

2) Duty Cycling → Con duty cycle facciamo riferimento alla frazione di tempo che i nodi sono attivi rispetto al totale periodo del segnale

L'idea principale del duty cycling è di spegnere i componenti di un sensore quando non necessari



### →) Topology Control

Utilizza la network redundancy. Seleziona un minimo insieme di nodi che garantiscono la connettività, gli altri vengono mantenuti in

Sleep mode. Aumenta il network lifetime a seconda del grado di ridondanza (solitamente 2-3).

Naturalmente, avere pochi nodi attivi comporta più distanza tra nodi attivi "vicini" (aumento dei packet loss) oppure richiede una tx power maggiore (che richiede più energia). Allo stesso modo,

avere molti nodi attivi comporta uno spazio non necessario di energia e possibili interferenze tra i nodi.

## 2) Power Management

le radio vengono spente quando non ci sono network activity. Il power management può essere implementato sia come protocollo indipendente eseguito "in testa" al MAC Protocol (tipicamente nel network o application layer) oppure può essere direttamente integrato nel protocollo MAC.

Il primo caso garantisce una flessibilità maggiore permettendo di esserne adattato all'esigenza dell'applicazione. Nel secondo caso abbiamo una ottimizzazione maggiore.

Le due tecniche (Topology control e Power Management) sono complementari e implementano il duty-cycling con differente granularità.

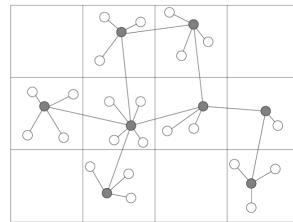
### 1) Topology control (+ delivery ratio, + energy saving)

Trovare il minimo subset di nodi che è capace di assicurare la network connectivity.

#### 1) Location Driven Protocols

Necessitano di conoscere l'esatta posizione dei nodi.

GAF (Geographic Adaptive Fidelity)



Ogni nodo conosce la sua posizione (GPS).

La sensing area nella quale i nodi sono distribuiti è divisa in piccole virtual grids.

Tutti i nodi di due griglie adiacenti devono essere in grado di comunicare tra loro.

Viene mantenuto + solo nodo attivo per griglia in ogni momento. (Non fa differenza quale esso sia)

Stati dei nodi: Active (grid leader), Sleeping (non grid leader), Discovery (leader election).

Leader election: Viene usato un algoritmo distribuito per la leader election di ogni cella.

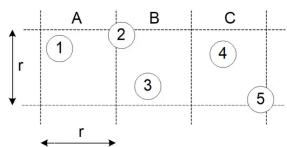
Il nodo leader viene scelto a rotazione per avere una uniform energy consumption

Gli altri nodi dormono e vengono svegliati solo quando il leader riceve un evento.

La scelta è basata su un rank-based election algorithm il quale considera l'energia rimanente dei nodi.

Hierarchical routing: I dati vengono inviati con questo ordine

Nodes  $\Rightarrow$  Leader  $\Rightarrow$  BS



La distanza max tra due nodi è data da  $\sqrt{r^2 + (2r)^2}$ , di conseguenza il

transmission range deve essere  $\geq$  di questa distanza.

$$R \geq \sqrt{r^2 + (2r)^2}$$

Manipolando un po' la formula, otteniamo quanto devono essere grandi le guglie, dato il tx

$$\text{range} \quad r \leq \frac{R}{\sqrt{5}}$$

Svantaggio: Tutti i nodi devono conoscere la loro posizione iniziale ed avere questo dispositivo aggiuntivo consuma energia

## 2) Connectivity Driven Protocols

Più flessibilità, non dipende dal routing protocol.

### ASCENT (Adaptive Self-Configuring Sensor Networks Topologies)

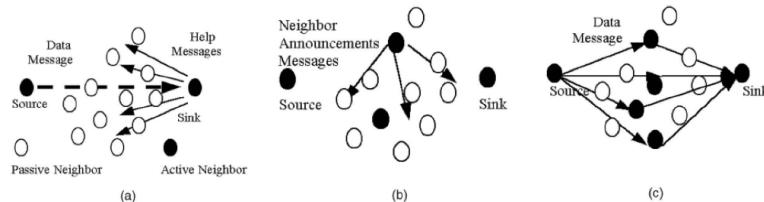
Un nodo decide se joinare un network o continuare a dormire basandosi sulle info di connettività e packet loss misurate localmente dal nodo stesso.

Y nodi possono essere:

**Attivi**: Y nodi attivi si occupano di inoltrare i pacchetti. Potrebbe inviare un help message per sollecitare i nodi vicini a diventare attivi se sta rilevando un'alta perdita di pacchetti

**Passivi**: Può dormire o ricevere pacchetti. Non inoltrano i pacchetti.

Un nodo che joins a network manda un announcement message. Questo processo continua fin quando il numero di active nodes che hanno perso il msg è minore di una data soglia. Il processo inizia quando alcuni network events o env. changes causano un aumento di msg loss.

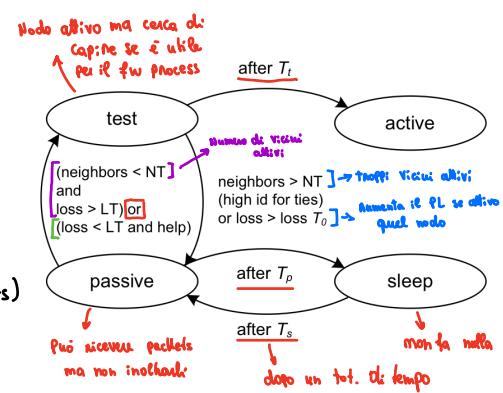


(a) A communication hole is detected (il sink node rileva una perdita di pacchetti) e invia help msgs

(b) Transition from passive to active state

(c) Final State

(solo se non interagisce con gli altri nodi, senno torna in sleep mode)



## Performances:

- Vicino al 100% delivery ratio (rispetto ad avere tutti i nodi attivi che interferiscono tra loro)
- Viene salvata energia e ne viene salvata di più se i parametri (le soglie) non sono fissati ma sono adattivi nel tempo.

## 2) Power Management (Mettere in sleep la radio quando non necessaria)

Ideia generale: + pacchetto di size L ogni T seconds

$$\text{Avg power consumption } (P) = \frac{E_T}{T} \quad \text{dove: } E_T = \text{Energia consumata durante il periodo } T$$

### 1) General Sleep-wakeup schemes (costruiti "on top" al MAC protocol)

Quando dovremo svegliare un nodo per comunicare con i suoi vicini?

a) On-demand (STEH): Quando un altro nodo vuole comunicare con esso

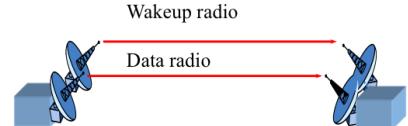
Tipicamente questi schemi utilizzano due radios (o uno singolo con frequenze differenti)

a) Wakeup radio = Usata per segnalare che un nodo vuole effettuare una comunicazione.

Soltanente low rate and low power (essendo che deve essere sempre attivo)

b) Data radio = Usato per la comunicazione dei dati.

Soltanente high rate and high power



Nelle pratica, possiamo assumere che entrambi i radios hanno range simili.

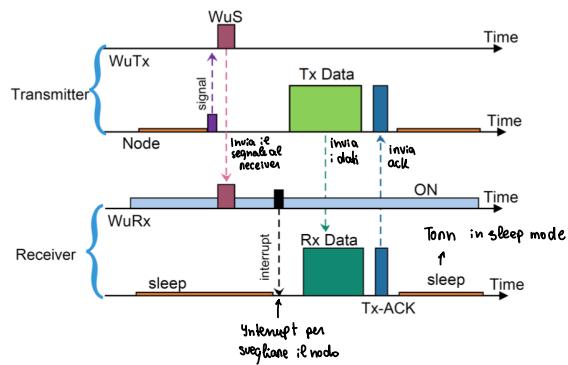
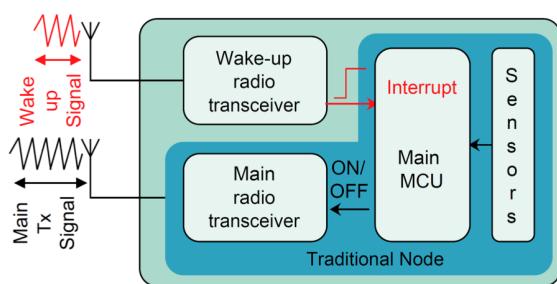


TABLE I: ACRONYMS FOR WAKE-UP RADIO TECHNOLOGY

WuR	wake-up radio, the secondary low-power module
WuRx	wake-up receiver
WuTx	wake-up transmitter
WuS	wake up signal, the message sent by the WuTx

## Design del WoR

- 1) Deve consumare poco in active mode, non più di 10 micro-watts
  - 2) Un nodo con il wake-up radio (WuR) si deve svegliare con la minima latenza quando riceve un wake-up signal (WuS) per:
    - Evitare la latenza dovuta ai multi-hops verso il sink node
    - Aumentare la responsiveness di un network puramente attivato.
  - 3) I nodi devono evitare i false wake-up che causano consumo di energia.

I false wake-up possono essere causati da:

    - a) WuS destinati a altri nodi  $\Rightarrow$  WuR può usare il node addressing per triggerare solo il nodo voluto.
    - b) Interferenze da nodi vicini su ugual frequenza  $\Rightarrow$  WuRx deve avere abbastanza capacità computazionali per filtrare i WuS erronei (interferenze/norme)
  - 4) WuR idealmente dovrebbe avere lo stesso range del data radio, solitamente è di circa 30m e può essere migliorato attraverso delle antenne.
  - 5) Alto data rate  $\Rightarrow$  Minore consumo, faster wakeup      basso data rate  $\Rightarrow$  Range maggiore, miglior affidabilità       $\left\{ \begin{array}{l} \text{Nou e' richiesto strettamente un alto data rate per WuR soprattutto se e' usato solo come triggering device, essendo richiesti pochi bytes per fare cio.} \\ \text{Per esempio se ci permette di risparmiare piu energia e' conveniente!} \end{array} \right.$
  - 6) Introdurre il WuR su nodi esistenti dovrebbe costare poco (il 5/10% del costo totale).
- 
- 2) Scheduled rendez-vous: Nello stesso momento dei suoi vicini, si mettono d'accordo sul momento nel quale attivare il radio per la comunicazione.

$\downarrow$

E' richiesto un clock di sincronizzazione  $\downarrow$  Consuma energia!  $\rightarrow$  Però se ci permette di risparmiare più energia è conveniente!  
Gestito tramite scambio di pacchetti di sincronizzazione (solitamente con un master node)
  - a) Fully Synchronized scheme (tinyDB)

Tutti i nodi si svegliano nello stesso momento seguendo un certo periodo.

$T_{\text{wake up}} = \text{Ogni nodo si sveglia ogni } T_{\text{wake up}} \text{ Time}$  \  $T_{\text{active}} \ll T_{\text{wake up}}$

$T_{active}$  = Ogni nodo rimane attivo per  $T_{active}$  Time

Vantaggio: Semplicità

Svantaggi:

- Tutti i nodi si svegliano e cercano di comunicare nello stesso momento  $\Rightarrow$  Alto n° di collisioni
- Non flessibile (statico)  $\rightarrow$  Active time statico, definito a "programmazione"
- Duty cycle uguale per tutti i nodi  $\rightarrow$  Per alcuni nodi potrebbe essere più grande del necessario!

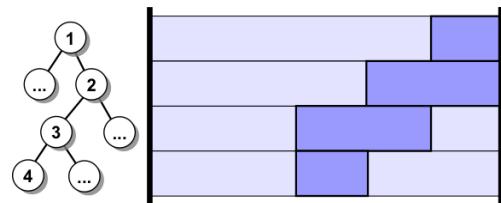


### b) Fixed Slotted Scheme (TAG, Task)

g nodi su diversi livelli dell'albero si svegliano in tempi diversi.

le active part di due lvl adiacenti si devono sovrapporre un minimo per poter permettere la comunicazione tra i nodi e i suoi nodi figli.

Vantaggio: Riduce il numero di nodi attivi allo stesso tempo, di conseguenza riduce sia le collisioni sia l'active time



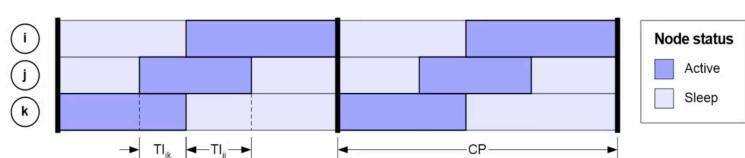
Svantaggi:

- Non flessibile (statico)  $\rightarrow$  Active time statico, definito a "programmazione"
- Duty cycle uguale per tutti i nodi  $\rightarrow$  Per alcuni nodi potrebbe essere più grande del necessario!

### c) Adaptive Slotted Scheme (ASLEEP)

Migliora lo slotted approach considerando active period variabili, che si adattano alle operating conditions:

- Numero di figli
- Network traffic
- Channel conditions
- Nodi che entrano / escono nella rete

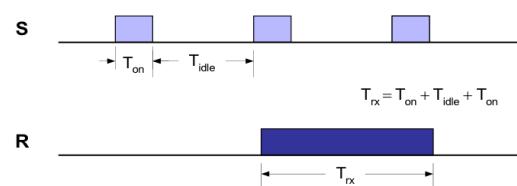


Svantaggio: Complessità nella coordinazione

3) **Asynchronous:** permettono ad ogni nodo di svegliarsi indipendentemente dagli altri nodi garantendo che i nodi vicini hanno active periods sovrapposti all'interno di uno specifico numero di cicli.

Funziona se sono periodici!

**Variante 1:** Il sender invia uno stream di discovery messages periodici. Il receiver listening time deve essere almeno uguale a  $T_{on} + T_{idle} + T_{on}$  dove  $T_{on}$  = Tempo di trasmissione di un discovery message e  $T_{idle}$  = tempo tra due discovery messages consecutivi, per ricevere correttamente il message da parte del sender!



Consumo più energetico sul sender e meno sul receiver

**Variante 2:** Il sender invia un singolo discovery message e il receiver ascolta periodicamente per un breve lasso di tempo.

Il tempo del discovery message deve essere almeno maggiore di quello di listening.



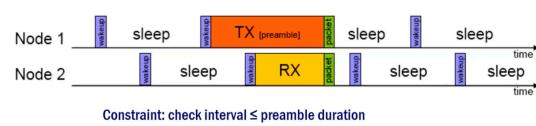
Migliore della variante 1 perché tipicamente il sender è 1 ma i receiver sono molti!

### Low Power listening (LPL) - implementazione della seconda variante

- 1) Il sender usa un lungo preamble (dati non importanti), con tempo  $\geq T_{rx}$ , prima di ogni pacchetto, per comunicare al receiver l'invio di un pacchetto.
- 2) Finita la trasmissione del preamble, viene trasmesso il pacchetto.

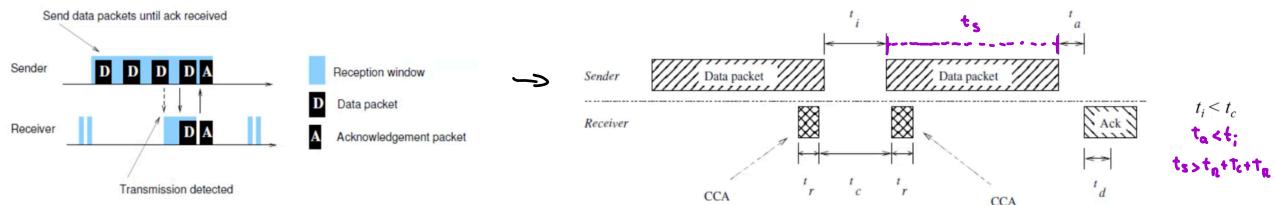
#### Svantaggi:

- a) Il sender deve inviare un lungo preamble e il nodo 2 deve aspettare di ricevere tutto il preamble prima di ricevere il pacchetto
- b) Ogni trasmissione sveglia tutti i vicini



## Contiki MAC Duty Cycling (Versione migliorata del LPL implementata in Contiki)

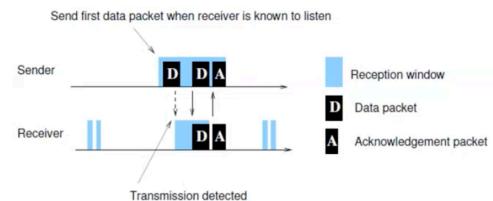
Contiki migliora la trasmissione non inviando il preamble, ma inviando ripetutamente il pacchetto fin quando non riceve un Ack dal receiver.



Dopo aver effettuato la prima trasmissione, il sender può calcolare quanti data packets ha inviato prima di ricevere l'Ack, e di conseguenza ha imparato la wake-up phase del receiver.

Alla prossima trasmissione necessita meno trasmissioni per inviare il pacchetto.

Questo meccanismo è chiamato "transmission phase-lock".



### Riassunto

- Data-driven approaches can significantly reduce the amount of data to be transmitted
  - Up to 99% and beyond
- However, this does not necessarily result in energy consumption reduction, due to
  - Energy costs introduced by transmission overhead, network management
  - Additional costs due to communication reliability

Are they really useful in practice?



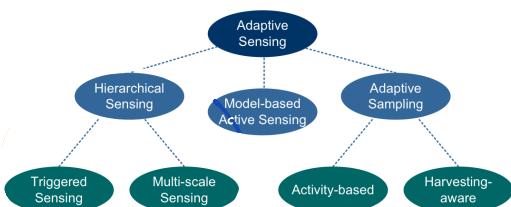
Sì, ma se usati in concatenazione con altri approcci di duty cycling

- Topology Management** exploits node redundancy
  - The increase in the network lifetime depends on the actual redundancy, and is limited in practice (some %)
  - It allows a longer lifetime at the cost of increased redundancy (i.e., larger economic costs)
- Power Management** removes idle times
  - May provide very large energy reductions
  - with limited costs (in terms of additional complexity)
- Energy Efficiency vs. Robustness**
  - Simple approaches → high robustness/limited energy efficiency
  - Complex approaches → higher energy efficiency but less robustness
  - Very complex solutions cannot work in practice

## Sensor energy Management

Consideriamo adesso il caso in cui il radio non sia l'unica componente che consuma molta energia e dobbiamo gestire anche il consumo degli altri componenti (sensori) pressure, humidity ...

## Adaptive sensing strategy (livello applicativo)



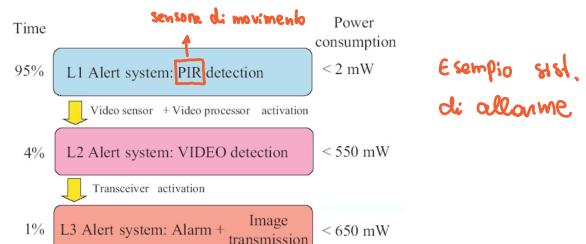
OSS: i sensori hanno un warm-up time prima di essere utilizzati correttamente

### 1) Hierarchical Sensing

Queste tecniche assumono che più sensori sono installati sul sensor node e osservano lo stesso evento con differenti consumi di energia e accuratezza.

Idea: L'idea è di scegliere dinamicamente quali sensori devono essere attivati facendo un trade off tra accuratezza e consumo di energia.

! I sensori con meno accuratezza e costo energetico sono usati per rilevare qualche attività e attivare i sensori con maggiore accuratezza e costo energetico.



Esempio sist. di allarme

### 2) Adaptive Sampling

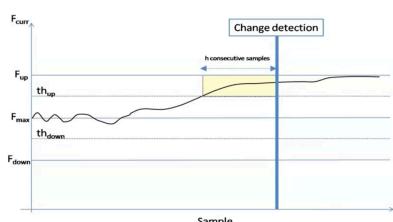
Idea: le sampling rate viene adattato alle dinamiche del fenomeno sotto monitoraggio.

Se evolve velocemente allora viene aumentato il sampling rate, e viceversa.

Come e quando cambiano il sampling rate?

#### Adaptive Sampling Algorithm (ASA)

È un algoritmo basato sul Nyquist Theorem, il quale dinamicamente computa la massima frequenza e dinamicamente si adatta al sampling rate basato sul signal's dynamics.



→ se  $f_{curr}$  è vicina a  $f_{up}$  o  $f_{down}$  per h samples consecutivi, viene rilevato un cambio nella  $f_{max}$  del segnale

## The frequency change detection



- Modified CUSUM → basic idea:

- Estimate the maximum frequency  $\bar{F}_{\max}$  of the signal by using a training sequence ( $W$  samples)
- Define two alternative hypothesis  $F_{up}, F_{down}$  for the maximum frequency of the signal during the operational life
- If the current maximum frequency  $F_{curr}$  of the signal ( $W$  samples) during the operational life is closer to  $F_{up}$  or  $F_{down}$  than  $\bar{F}_{\max}$  for  $h$  consecutive samples, a change is detected in the maximum frequency of the signal
- A new sampling frequency is defined

- Parameter  $c$

- confidence** parameter for the maximum frequency detection ( $c > 2$ , Nyquist)

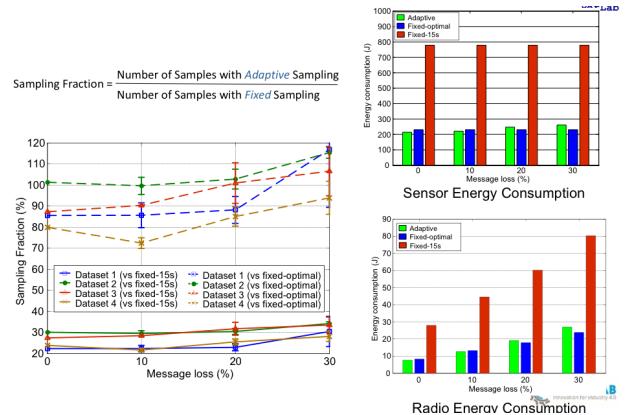
- Parameter  $h$

- critical to the **robustness** of the algorithm
  - low values (e.g., 1 or 2): quick detection but possible false positives
  - high values (e.g., 1000): few false positives but less promptness in detecting the changes

- Parameter  $W$

- critical to the **accuracy** of the algorithm
  - low values: not accurate estimation but low energy consumption
  - high values: accurate estimation of  $F_c$  but energy consumption

A-priori information about the process could provide the designer with a suitable parameter



la convenienza dell' ASA dipende dall'app.

In questo caso riduce sia l'energia consumata dal sensore, sia quella consumata dalla radio, ma ci potrebbero essere dei casi dove l' ASA provoca troppe riasmissioni dovute alla riduzione dei samples e di conseguenza un maggiore consumo da parte della radio.

## Hawesing-aware

È un decentralized Adaptive Sampling.

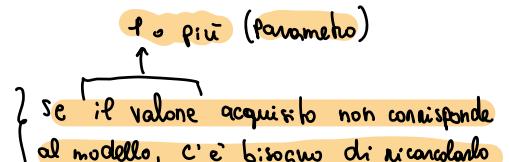
Il sampling rate è adattato sulla base dell' energia disponibile (Nodi alimentati dalle solar cells) obiettivo. Minimizzare il total uncertainty error, dato dal massimo numero di samples che un sensore può raccogliere per giorno.

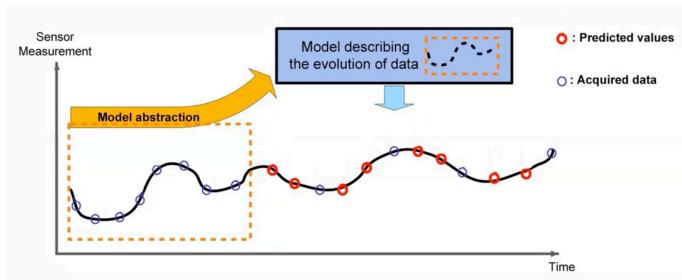
## 3) Model-based Active Sensing

Idea: imparare la relazione spatio-temporale tra le misure e usare questa conoscenza per rendere il sensing process efficient.

Ad ogni step il sensore decide se:

- Acquisire un nuovo sample attraverso una misurazione
- Stimare il nuovo esempio tramite un modello





## Communication Technologies

### (Low-duty cycle) MAC Protocols

Low duty because it includes techniques of power management. These techniques are integrated in the MAC protocol.

#### Time-Slotted Access Protocols

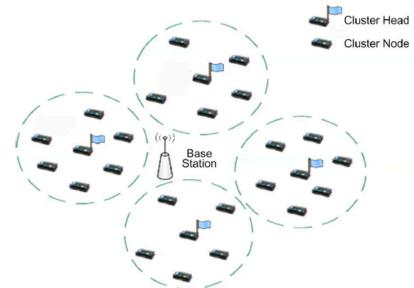
These protocols divide time into slots.

Characteristics:

- 1) High Energy Efficient: Each node is active only during its slot.
- 2) Guaranteed Bandwidth: Each node takes one or more fixed-length slots each round.
- 3) Bounded and Predictable Latency: Each node accesses the same slot each frame.

Characteristics: disadvantages:

- 1) Limited Flexibility
  - A topology change may require a different slot allocation pattern.
- 2) Limited Scalability
  - Finding a scalable slot allocation function is not trivial, especially in multi-hop (i.e., hierarchical) networks.
- 3) Interference-prone
  - Finding an interference-free schedule may be hard.
  - The interference range is larger than the transmission range.
- 4) Tight Synchronization Required
  - Clock sync introduces overhead.



### LEACH (low Energy Adaptive Clustering Hierarchy)

The nodes are organized in clusters and each cluster has a cluster-head (CH), which coordinates all activities within the cluster.

Each node has a predefined slot and wakes up during its slot, in this time it sends data to the CH.

The CH has the highest energy consumption, so it is rotated among the various nodes.

## CSMA-based MAC protocols

Vantaggi:

- 1) Non c'è richiesta la sincronizzazione
- 2) Grande flessibilità: Un cambio nella topologia non richiede nessuna reconfigurazione

Svantaggi:

- 1) Scalabilità limitata: Un grande numero di nodi può causare un grande numero di collisioni e ritrasmissioni
- 2) Low energy efficient: I nodi potrebbero avere conflitti e c'è un magg. consumo dovuto all'overhearing

## B-MAC (Bentley MAC)

È il protocollo più popolare tra quelli content-based ed è considerato "low complexity and low power". È usato nel tinyOS operating system.

### Caratteristiche principali

- 1) Semplicità
- 2) Possibilità di configurazione
- 3) Minimizza il listening a riposo (idle listening) per risparmiare energia

### Access Protocol

- 1) Genera un delay random prima di trasmettere [15-68,3] ms
- 2) Effettua il channel assessment
  - a) Se il canale è libero → Trasmetti (e aspetta l'ACK)
  - b) Altrimenti → random backoff [+2.08-19,3] ms
- 3) Ritorna al punto 2

### Componenti principali:

- 1) CSMA senza RTS/CTS

- 2) Low power listening optionale

- 3) ACK optionali

Per il power management

Attivarlo o no dipende dalla affidabilità richiesta dall'applicazione

## IEEE 802.15.4 Standard

È lo standard all'interno dell'802.15 che si occupa delle PAN (Personal area Network).

Esso definisce il livello fisico e il MAC layer.

Gli obiettivi di questo standard sono i seguenti:

1) low-rate: la velocità di trasmissione dei dati

dove essere bassa.

2) low power

3) low complexity: Per ridurre il costo di implementazione

## Network Topologies

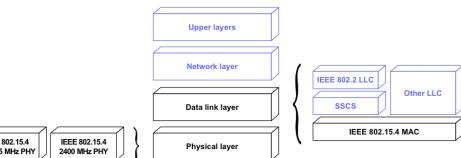
1) A stella: la comunicazione avviene tra un dispositivo e un gestore centrale.

Non è ammessa la comunicazione tra gli altri dispositivi.

2) Peer-to-peer: la comunicazione avviene direttamente tra dispositivi che si trovano all'interno del raggio di copertura.

Reti di questo tipo possono essere multi-hop e

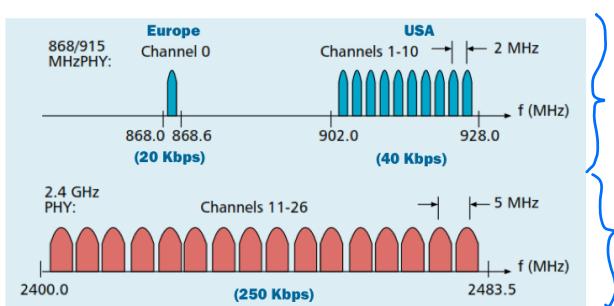
possono estendere l'area di copertura delle PAN.



## Livello Fisico

Si occupa della trasmissione e ricezione dei dati, entrambi basati sulla modulazione a spettro espanso.

## Channel Frequencies



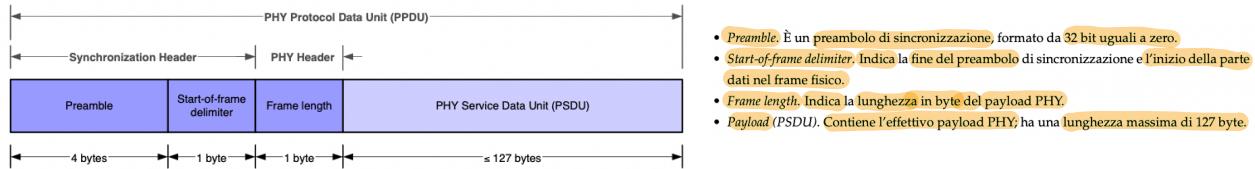
## PHY 868/915

Ha a disposizione 11 canali, di cui uno intorno agli 868 MHz e gli altri nell'intervallo compreso tra 902 e 928 MHz (figura 3.3a). La banda a 868 MHz può essere utilizzata in Europa, mentre quella a 915 MHz può essere utilizzata negli Stati Uniti. Le velocità disponibili sono, rispettivamente, di 20 kbps e di 40 kbps. In entrambi i casi ogni bit viene codificato in modo differenziale e rappresentato da una sequenza di 15 chip. La sequenza così ottenuta viene moltiplicata per +1 o -1 e modulata con tecnica Binary Phase Shift Keying (BPSK).

## PHY 2450 MHz

Ha a disposizione 16 canali nella banda a 2.4 GHz (figura 3.3b). In questa banda - disponibile quasi ovunque nel mondo - è possibile raggiungere una velocità di 250 kbps. I bit sono divisi in gruppi di 4 e codificati in un simbolo. Ciascun simbolo viene mappato in una sequenza pseudo-casuale, a massima lunghezza e quasi ortogonale, formata da 32 chip. La sequenza così ottenuta viene modulata con tecnica Offset Quadrature Phase Shift Keying (O-QPSK).

## FRAME FORMAT LVL FISICO



## Livello MAC

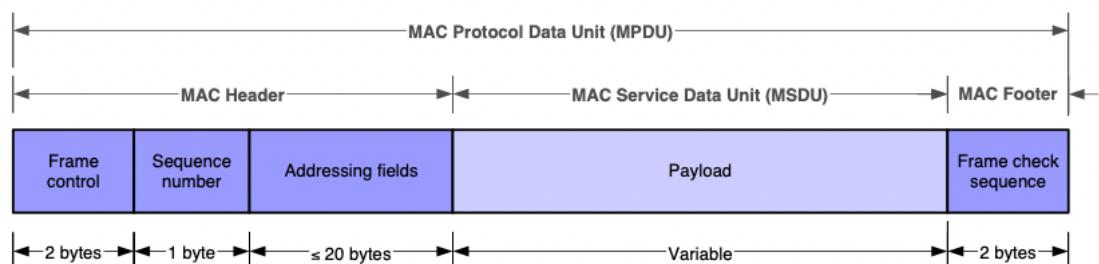
Il livello MAC si occupa delle modalità di accesso al canale, includendo gli aspetti di affidabilità e sicurezza della trasmissione.

### MAC addresses

Esistono due tipi di indirizzi:

- 1) **PAN address**: Identifica in modo univoco una PAN all'interno di una certa area, consentendo la comunicazione tra PAN differenti.  
È composto da 16 bit con ind. di broadcast **0xffff**.
- 2) **Device Address**: Può essere di due tipi:
  - a) **Esteso**: 64 bit utilizzati per l'identificazione univoca del dispositivo  
*Introduce overhead* ↴
    - a) 24 bit più significativi assegnati dall' IEEE
    - b) gli altri 40 bit assegnati dal manufacturer
  - b) **Ridotto**: 16 bit può essere negoziato con il PAN coordinator durante l'associazione e utilizzato per tutte le comunicazioni (al posto dell'esteso).

## FRAME FORMAT



- **Frame control**. Specifica il tipo di frame, le modalità di indirizzamento e altri flag di controllo.
- **Sequence number**. Rappresenta un identificatore univoco per il frame.

- **Addressing fields**. Contengono:

- l'indirizzo della PAN mittente;
- l'indirizzo del dispositivo mittente nella PAN indicata dal campo precedente;
- l'indirizzo della PAN destinataria;
- l'indirizzo del dispositivo destinatario nella PAN indicata dal campo precedente.

La presenza e la lunghezza di questi campi varia secondo le modalità descritte nel paragrafo precedente.

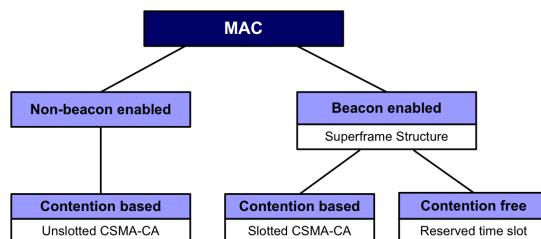
- **Frame payload**. Contiene l'effettivo MAC payload; la sua lunghezza è variabile.
- **Frame check sequence (FCS)**. Contiene un codice a ridondanza ciclica (CRC) a 16 bit conforme allo standard ITU-T.

## Modalità di Accesso AL MEZZO Fisico (channel)

802.15.4 prevede due modalità di accesso al mezzo (channel):

- 1) **Beacon-enabled**
- 2) **Non-Beacon Enabled mode**

### BEACON - ENABLED MODE



Essa si basa sulla struttura a superframe. Il superframe è delimitato dai beacon, inviati dal PAN coordinator, ed è diviso in una parte attiva e in una inattiva.

L'accesso al mezzo trasmissivo avviene durante la parte attiva, composta da 16 slot.

Durante la parte attiva possiamo contraddistinguere due ulteriori posizioni:

- 1) **CAP** (contention access period): è sempre presente nel superframe ed è sempre dopo il beacon.

Prevede un accesso basato su contesa con protocollo CSMA-CA.

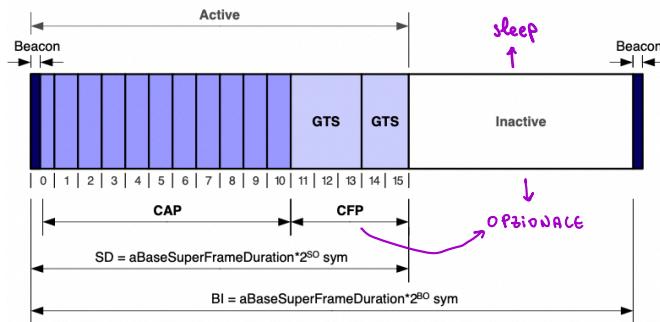
- 2) **CFP** (Contention Free Period): è opzionale ed occupa l'ultima posizione della parte attiva. Esso è composto da un insieme di Guarantee Time slot (GTS)

cioè intervalli di comunicazione unidirezionale dove solo un dispositivo può accedere al canale.

la durata del superframe è data dall'intervallo tra i due beacon ed è definita in base a un parametro  $BO$  (aMAC BeaconOrder).

$BO$  ammette valori tra  $0 \leq BO \leq 14$  mentre se  $BO = 15$  allora il coordination non trasmette beacon.

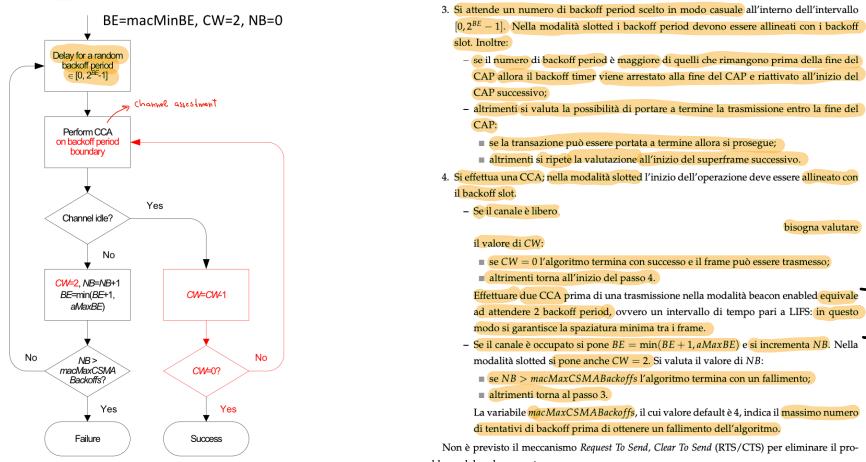
la durata della posizione attiva del superframe è definita dal parametro  $SO$  (aMAC superframeOrder), dove  $0 \leq SO \leq BO \leq 14$  mentre se  $SO = 15$  allora il superframe è formato solo dalla parte attiva.



### CSMA-CA (Carrier Sense Multiple Access with Collision Avoidance) algorithm

È l'algoritmo usato per l'accesso al mezzo trasmissivo. Ogni dispositivo attende un intervallo di tempo casuale prima di accedere al canale, se il canale è occupato attende nuovamente una quantità di tempo casuale.

La variante slotted è usata durante il CAP del superframe, in PAN che operano in modalità beacon enabled.



3. Si attende un numero di backoff period scelto in modo casuale all'interno dell'intervallo  $[0, 2^{BE} - 1]$ . Nella modalità slotted i backoff period devono essere allineati con i backoff slot. Inoltre:

- se il numero di backoff period è maggiore di quelli che rimangono prima della fine del CAP allora il backoff timer viene arrestato alla fine del CAP e riattivato all'inizio del CAP successivo;
- altrimenti si valuta la possibilità di portare a termine la trasmissione entro la fine del CAP:
  - = se la transazione può essere portata a termine allora si prosegue;
  - = altrimenti si ripete la valutazione all'inizio del superframe successivo.

4. Si effettua una CCA: nella modalità slotted l'inizio dell'operazione deve essere allineato con il backoff slot.

- Se il canale è libero bisogna valutare

il valore di  $CW$ :

- = se  $CW = 0$  l'algoritmo termina con successo e il frame può essere trasmesso;
- = altrimenti torna all'inizio del passo 4.

Effettuare due CCA prima di una trasmissione nella modalità beacon enabled (equivale ad attendere 2 backoff period, ovvero un intervallo di tempo pari a LIFS: in questo modo si garantisce la spazio-temporale minima tra i frame).

- Se il canale è occupato si pone  $BE = \min(BE + 1, aMaxBE)$  e si incrementa  $NB$ . Nella modalità slotted si pone anche  $CW = 2$ . Si valuta il valore di  $NB$ :

= se  $NB > macMaxCSMABackoffs$  l'algoritmo termina con un fallimento;

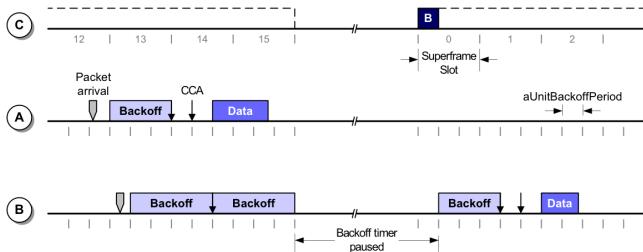
= altrimenti torna al passo 3.

La variabile  $macMaxCSMABackoffs$ , il cui valore default è 4, indica il massimo numero di tentativi di backoff prima di ottenere un fallimento dell'algoritmo.

Non è previsto il meccanismo Request To Send, Clear To Send (RTS/CTS) per eliminare il problema del nodo nascosto.

Vengono effettuati 2 CCA per assicurarsi che il mezzo sia veramente libero perché può succedere che alcuni nodi non siano perfettamente sincronizzati e quindi un nodo potrebbe effettuare il CCA assestandosi e trova il mezzo libero, ma in realtà un altro nodo sta per trasmettere con leggero delay. Con il secondo CCA il nodo si accorge di questa situazione.

## Esempio algoritmo CSMA/CA



Per chiarire il funzionamento dell'algoritmo CSMA/CA consideriamo la situazione seguente (adattata da [Mis 04] e illustrata nella figura 3.11): due dispositivi A e B effettuano l'accesso con contesa in modalità slotted all'interno di una PAN il cui coordinatore è il dispositivo C. Supponiamo che la trasmissione avvenga senza il riscontro dei dati e con la modalità battery life extension disabilitata. La parte attiva del superframe è rappresentata da una riga tratteggiata in corrispondenza del coordinatore; sotto questa riga sono indicati anche i superframe slot.

Consideriamo il dispositivo A. Un pacchetto viene inviato al MAC dal livello superiore nell'istante contrassegnato dalla freccia grigia. L'algoritmo slotted CSMA/CA inizializza le variabili  $NB$ ,  $BE$  e  $CW$  ai loro valori iniziali e attende l'inizio del backoff period slot successivo. Supponiamo che sia  $BE = macMinBE = 2$  e che il periodo di backoff sia scelto pari a 3 slot. Il dispositivo attende 3 backoff period slot poi valuta se il tempo residuo a disposizione nel superframe è sufficiente a portare a termine la trasmissione. In questo caso la valutazione ha esito positivo e si procede ad una CCA (rappresentata come una freccia verso il basso nella figura). La CCA trova il mezzo libero, quindi si sottrae uno a  $CW$  e si procede ad una nuova CCA allineata con il backoff period slot successivo. Poiché il canale è di nuovo libero,  $CW$  viene decrementato e raggiunge il valore zero: al backoff period slot successivo il dispositivo A inizia la trasmissione.

Consideriamo ora il dispositivo B. Anche in questo caso un pacchetto viene inviato al MAC dal livello superiore nell'istante contrassegnato dalla freccia grigia. Assumiamo gli stessi parametri considerati per il dispositivo A, ma con un periodo di backoff scelto pari a 4 slot. Il dispositivo B attende 4 backoff period slot poi valuta se il tempo residuo a disposizione nel superframe è sufficiente a portare a termine la trasmissione. Anche in questo caso la valutazione ha esito positivo e si procede ad una CCA. Poiché il canale è occupato  $CW$  viene reimpostato a 2,  $NB$  viene incrementato ad 1 e  $BE$  viene portato al valore 3. Supponiamo ora che il nuovo periodo di backoff sia scelto pari a 7 slot. Il conto alla rovescia inizia immediatamente; al termine del superframe il timer viene arrestato e riattivato all'inizio del CAP nel superframe successivo. Al termine del periodo di attesa si procede ad una CCA. La CCA trova il mezzo libero, quindi si sottrae uno a  $CW$  e si procede ad una nuova CCA allineata con il backoff period slot successivo. Poiché il canale è di nuovo libero,  $CW$  viene decrementato e raggiunge il valore zero: al backoff period slot successivo il dispositivo B inizia la trasmissione.

## ACK Mechanism

È un meccanismo opzionale per l'affidabilità della connessione. Quando non viene usato, il mittente considera la trasmissione del frame sempre avvenuta con successo.

- Sender dell' ACK
  - 1) Ritrasmette l' ACK se non correttamente arrivato entro il timeout
  - 2) A ogni tentativo di ritrasmissione la backoff window viene riinizializzata
  - 3) Solo un numero max di ritrasmissioni è consentito
- Destination Side
  - 1) ACK inviato in caso di ricezione riuscita del data frame

## Security

Per la sicurezza della comunicazione ci sono tre modalità:

- 1) Unsecured mode: Nessun servizio di sicurezza è fornito
- 2) ACL mode: Il controllo degli accessi è basato sulla Access Control List (ACL).

L'ACL è una tabella usata dai dispositivi per determinare quali dispositivi sono autorizzati a svolgere determinate funzioni.

3) Secured Mode: Offre la sicurezza completa, garantendo l'ACL, la protezione anti-replay, la confidentialità e l'integrità dei dati. Questa modalità usa l'AES con chiave a 128 bit.

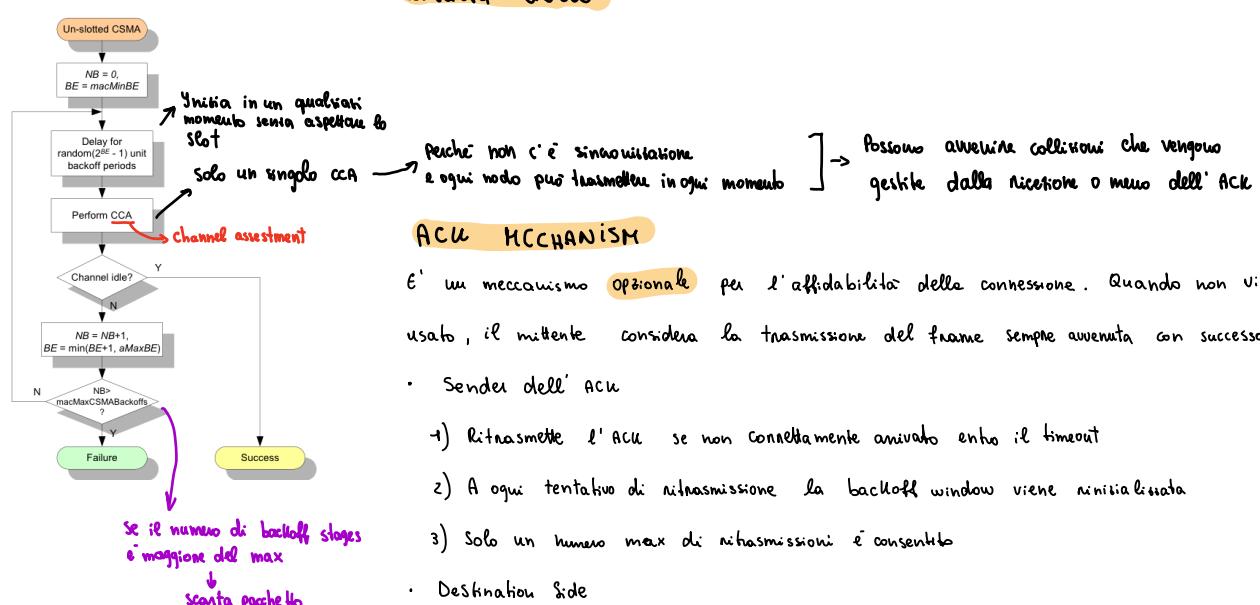
### Performances

Abbiamo visto che questo protocollo non è adatto per app. che richiedono affidabilità perché in questo di fornire affidabilità alta con parametri molto alti di valore. In più abbiamo visto che un'affidabilità alta richiede un aumento di latenza. È buono per app. non critiche che non richiedono affidabilità e latenza.

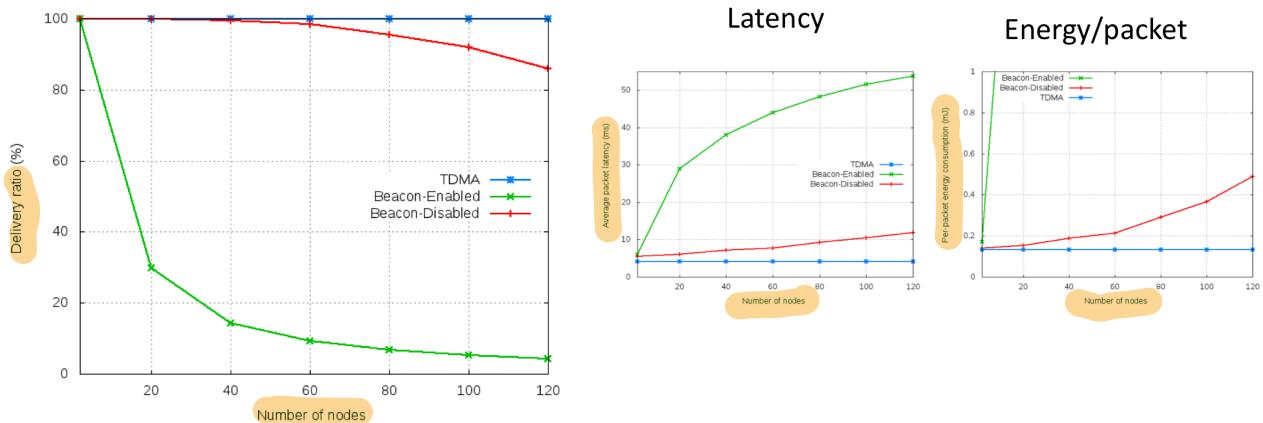
### NON-BEACON ENABLED MODE

La Non-beacon mode, come dice il nome, non utilizza beacon e non è composto da active/inactive pants. Semplicemente usa il protocollo unslotted CSMA per l'accesso al channel. Ogni nodo può svegliarsi in qualsiasi momento ed effettuare l'accesso se il medio è sulla libera.

#### Unslotted CSMA



## CONFRONTO TRA BEACON ENABLED e DISABLED



Come si può vedere dai grafici, la modalità **beacon disabled** è migliore.

## Limiti della tecnologia IEEE 802.15.4

- 1) Communication range limitato  $\Rightarrow$  Multi-hop comm. per distanze più lunghe e di conseguenza una necessità di maggiore di relay nodes che aumentano il deployment cost
- 2) Sensitivity to comm. interferences dovuto alle wireless network che operano nello stesso range di frequenze (vicino le 2.4 GHz)
- 3) Sensitivity to multi-path fading

## BLUETOOTH

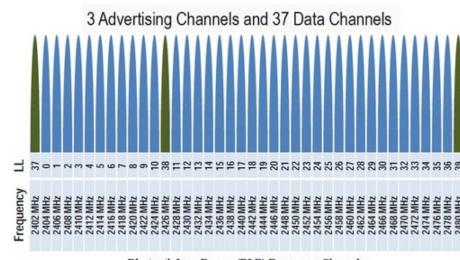
Per poter utilizzare il bluetooth sui sensori nodes abbiamo bisogno di una versione low energy

### BLE (Bluetooth low Energy)

#### Confronto con il bluetooth

	Bluetooth V2.1	Bluetooth Low Energy
Standardization Body	Bluetooth SIG	Bluetooth SIG
Range	-30 m (class 2)	-50 m
Frequency	2.4–2.5 GHz	2.4–2.5 GHz
Bit Rate	1–3 Mbit/s	<200 kbit/s
Set-Up Time	>6 s	<0.003 s
Voice Capable?	Yes	No
Max Output Power	+20 dBm	+10 dBm
Modulation Scheme	GFSK	GFSK
Modulation Index	0.35	0.5
Number of Channels	79	40
Channel Bandwidth	1 MHz	2 MHz

#### BLE channels



Dopo aver trasmesso un pacchetto i dispositivi cambiano il canale di trasmissione successivo per ridurre le interferenze

## Communication Modes

la comunicazione è device-to-device e avviene con un approccio master-slave.

Per la ricerca di nuovi dispositivi viene usata la advertise mode, nella quale c'è un advertiser che manda beacon periodici e un observer che li riceve.

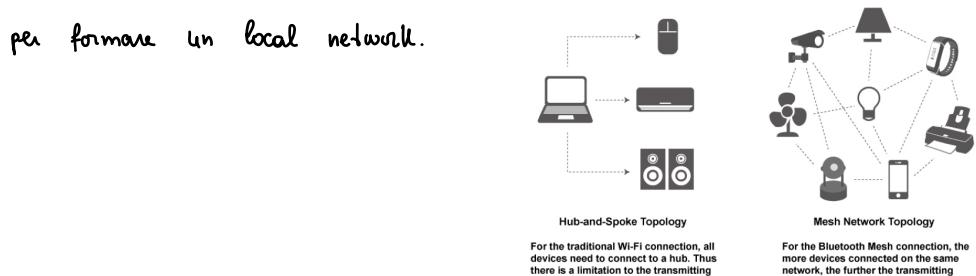
La comunicazione nell' advertise mode è broadcast.

## Topologie

1) Hub e Spoke: 4 device sono connessi a un hub centrale.

La comunicazione non avviene se l'hub è out of coverage.

2) Mesh Topology: 4 dispositivi sono connessi direttamente tra loro, senza l'uso di internet per formare un local network.



## IEEE 802.15.4g

È un emendamento allo standard IEEE 802.15.4 con target le Wireless Smart Utility Networks.

### Caratteristiche:

1) Permette connessioni a range più lunghi rispetto all' IEEE 802.15.4

2) Opera a "sub GHz" (< 1 GHz) il quale offre:

- Distanze più lunghe con la stessa potenza
- Meno interferenza rispetto al 2.4 GHz

3) Nuovi layer fisici:

- 1) Frequency Shift Keying (FSK)
    - data rates in the range 5–400 Kbps
      - depending on the radio parameter setting
    - Forward Error Correction (FEC) to reduce the bit error rate
      - reduces the # of re-transmissions
  - 2) Offset Quadrature Phase-Shift Keying (OQPSK)
    - shares some characteristics with the original IEEE802.15.4 standard
    - data rates in the range 6–500 Kbps
  - 3) Orthogonal Frequency Division Multiplexing (OFDM)
    - data rates in the range 50–800 Kbps
      - in challenging environments with multi-path fading
- Many Operating Modes for each PHY
    - Depending on the parameter setting
    - Up to 31 different modes
    - Data rates from 6.25 Kbps to 800 Kbps
  - Default Mode
    - FSK at 50 Kbps
    - The default mode is mandatory
      - must be implemented in any IEEE 802.15.4g compliant product
  - Maximum frame size is always 2047 bytes

Yl communication range dipende fortemente sull'ambiente esterno ed è minore di 1 km.

Potrebbe non essere abbastanza  $\Rightarrow$  Multi-hop comm. richiesta.

Come connettere la LPL a internet?

- 1) Non-IP solution: i nodi non hanno un IP associato e non si connettono direttamente a internet. L'intera rete può essere più connessa tramite l'application gateway.  
I protocolli usati nella sensor network differiscono dal protocollo IP.
- 2) IP-based solution: è basato sull'IPv6 e i protocolli IoT. I nodi sono connessi direttamente a internet e hanno un IPv6 associato.

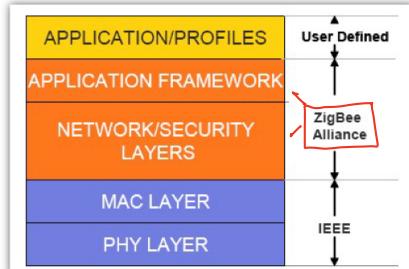
### Non-IP solutions

Diverse sono le tecnologie disponibili. Di queste noi studieremo la tecnologia ZigBee.

#### ZigBee

La Tecnologia zigbee è stata sviluppata dalla ZigBee Alliance, la quale è una corporazione non-profit.

La ZigBee Alliance ha definito alcune specifiche che devono essere rispettate per poter utilizzare la tecnologia ZigBee.



#### ZigBee Device Types

In una rete zigbee ci sono tre tipologie di device:

- 1) ZigBee Coordinator (zC): È il nodo che coordina tutte le attività della rete.  
Almeno uno per network zigbee.
- 2) ZigBee Router (zR): Sono i nodi che gestiscono routing e forwarding.  
I router sono nodi che sono sempre attivi.

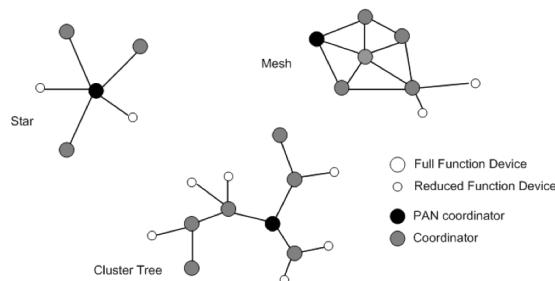
3) ZigBee End device (ZED): Sono nodi terminali che non hanno la capacità di fare operazioni di routing. Essi generano il traffico e i pacchetti. Possono essere attivi o inattivi a seconda se devono fare qualcosa.

gli dispositivi sopra descritti si possono dividere in due ulteriori categorie:

- 1) Full function Device: Sono i device che implementano l'intero protocol stack. ZC e ZR sono per forza di questo tipo. ZED devices possono esserlo come non esserlo.
- 2) Reduced Function Device: Sono i device che implementano una parte del protocol stack.

### ZigBee Network Topologies

- 1) Star: Sono reti dove i nodi sono connessi direttamente allo ZC
- 2) Cluster-tree: Sono reti che ammettono un solo path dal source node a una centrale destinazione.
- 3) Mesh: Sono reti che permettono più path da un source node ad una centrale dest



### Zigbee Routing

- 1) Tree-based Routing: È un algoritmo di routing parent-child.
  - a) Viene usato nella tipologia cluster-tree
  - b) Può usare sia la Beacon-enabled che la Non-Beacon Enabled

- 2) Mesh routing : Viene usato nella tipologia Mesh.
- Un algoritmo di routing usato è l' AODV, il quale è un protocollo distance vector on-demand.
  - È basato sulla Non-Beacon Enabled Mode
- 3) Nella tipologia Mesh, il Tree-based e mesh routing possono coesistere .  
Il routing algorithm può switchare tra le due modalità

### AODV (Ad-hoc On-demand Distance Vector)

È un protocollo, di tipo Distance vector, reattivo, cioè il percorso per inoltrare il pacchetto verso una certa destinazione viene scoperto solo quando necessario. Non vengono fatte operazioni iniziali per popolare la routing table ecc... come avviene nei protocolli di routing standard (appuccio proattivo), perché consumerebbe energia aggiuntiva inutilmente.

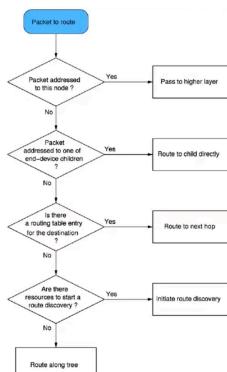
Una volta scoperto il route verso un nodo, può essere riutilizzato (naturalmente necessita di manutenzione in caso cambiasse)

Questo protocollo utilizza:

- Sequence number: Usato per selezionare il path più recente ed evitare i loop
- Routing Table : Struttura dati contenente le info sul prossimo hop da fare per raggiungere una certa destinazione.  
Ad ogni entry ha una lifeTime associata per mantenere la routing table aggiornata.

Operazioni svolte quando il packet arriva a un intermedio router e deve essere inoltrato:

Destination Address	Next-Hop Address	Entry Status
16 bits	16 bits	Active/ <sup>Info valid</sup> Discovery/ Inactive
		Info non valida
		Discovering l'optimal path



- 3) Route discovery table: usata da ogni nodo per gestire le informazioni sul routing nella rete (+ per nodo)

Field Name	Description
RREQ-ID	Sequence number given to every RREQ message being broadcasted
Source Address	Network address of the RREQ initiator
Sender Address	Network address of the device that sent the most recent lowest-cost RREQ
Forward Cost	The accumulated path cost from the RREQ originator to the current node
Residual Cost	The accumulated path cost from the current node to the RREQ destination
Expiration time	Number of milliseconds until this entry expires

### Route Discovery

Per poter performare l'operazione di route discovery vengono usati due control packets:

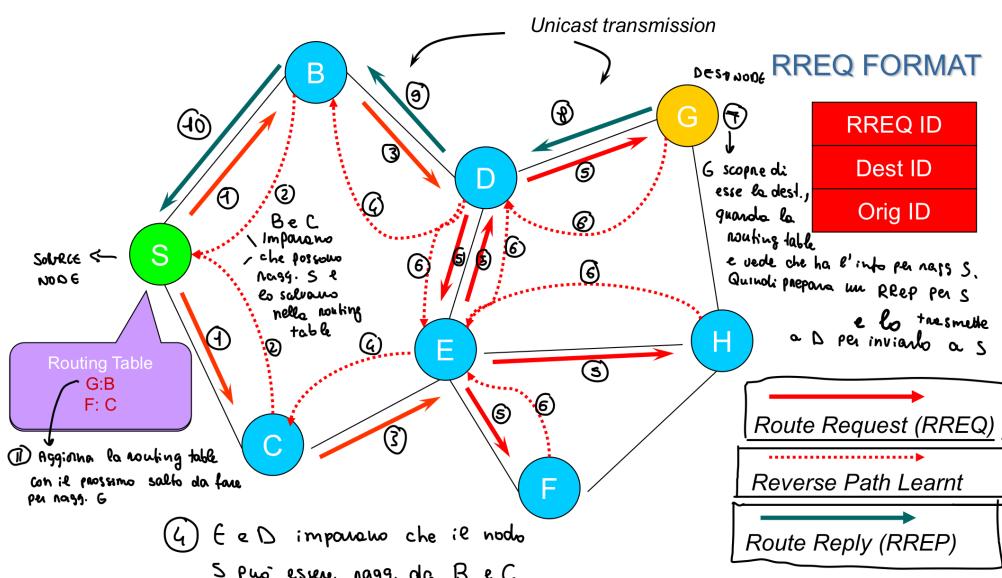
- 1) ROUTE REQUEST (RREQ): Usato per scoprire il path da parte del router
- 2) ROUTE REPLY (RREP): Mandato in risposta al router

Le roule che inizializza la richiesta manda in broadcast un RREQ packet e aspetta di ricevere la corrispondente RREP.

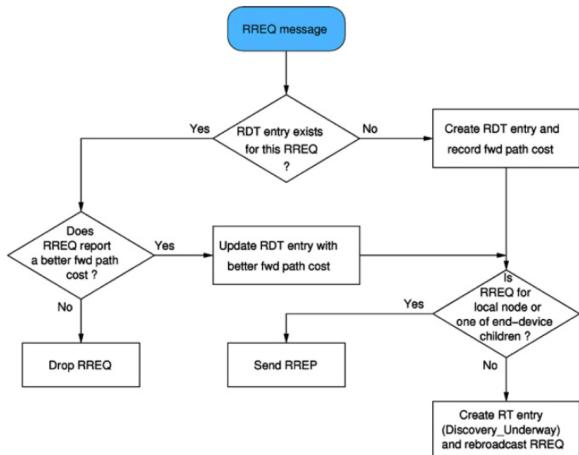
Un nodo intermedio X:

- 1) Se riceve un RREQ da Y: a) Crea un reverse-path dove Y è il prossimo hop verso S usato da X per inviare la RREP indietro a S  
b) Fa nuovamente il broadcast del RREQ
- 2) Se riceve un RREP da Y: a) Crea un forward path dove Y è il prossimo hop verso D  
b) Invia il RREP verso S (usando il reverse path)

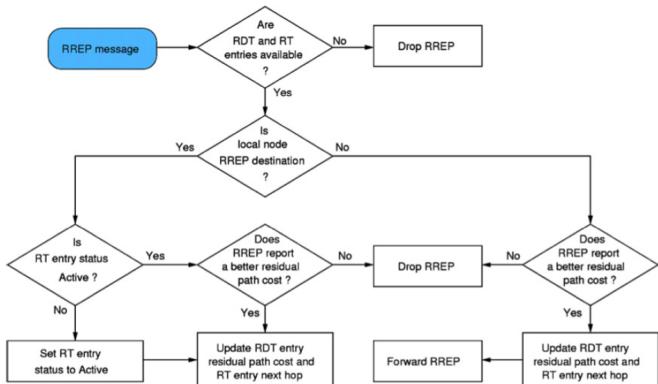
OSS: Se ci sono più path possibili, l'algoritmo decide il migliore (min. num. di hops)



## RREQ Processing



## RREP Processing



## Route Maintenance

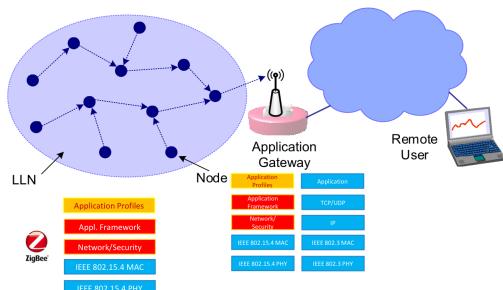
Possono esserci dei possibili errori nella rete e bisogna aggiornare i route path.

Quando avviene un link failure al nodo  $X$  per il path  $X - X_1$ :

- 1) X invia ai nodi vicini un Unsolicited Route Reply (URR) il quale invalida tutti i path che usano il link X-X'.
  - 2) Il source node S che riceve l'URR inizia una route discovery phase per trovare un nuovo path verso la dest.

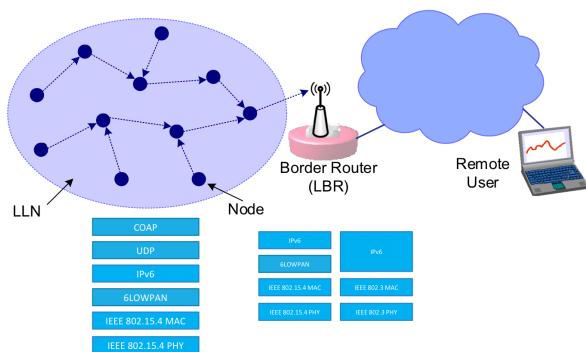
## Come connettere a internet?

Per le soluzioni "NON-IP" viene usato un application gateway, il quale implementa sia lo stack protocollo Zigbee che quello TCP/IP ed effettua la traduzione dei pacchetti da uno stack all'altro.



Questo non succede nelle soluzioni full-IPv6 perché tutti i nodi hanno associato un indirizzo IPv6 e inviano/ricevono datagrammi. Di conseguenza non avviene una traduzione

da parte del border router, ma viene semplicemente analizzato e inoltrato.

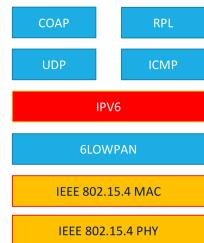


## IPv6 nelle LLN (IP-based Solution)

Per le reti LLN, come già specificato, viene usato il protocollo IPv6.

### Funzionalità chiave

- 1) Spazio di indirizzamento più grande
- 2) I nodi sono capaci di auto-configurarsi
- 3) L'header è stato modificato per velocizzare il processo ai router
- 4) Viene supportato il QoS
- 5) Authentication e privacy
- 6) Sicurezza



### Cambiamenti nell'header

- 1) Rimozione di alcuni campi dell'IPv6
  - a) Fragmentation: Era time consuming dovuto alla divisione in frammenti, l'aggiunta di informazioni e l'invio di tutti i frammenti
  - b) Checksum: Era time consuming essendo che andava ricalcolata ad ogni hop  
↳ Reso obbligatorio per TCP e UDP (a differenza dell'IPv4)  
Per i protocolli di trasporto invece che IPv6

2) Grandezza dell' header fissa (40 bytes)

a) Rimossi i fields optionali

b) Aggiunto "daisy-chained extended headers" optionali

3) Nuovi field per il supporto del QoS

a) Priority

b) Flow label

### IPv6 datagram format

• Version (4 bits) = Specifica la versione del protocollo IP. (= 6)

• Priority (4 bits) = Determina la priorità del datagram rispetto agli altri.

32-bits			
4-bit version	4-bit priority	Flow label	
16-bit payload length		Next header	Hop limit
128-bit source IP address			
128-bit destination IP address			
Data (variable length)			

• Flow label (20 bits): Identifica i datagram nello stesso flow. Comunica ai router come gestire i datagram appartenenti a flow differenti.

Il concetto di flow non è ben definito, i router lo gestiscono in modo diverso.

• Payload length (16 bit): Descrive la payload size in bytes, da questa dimensione esclude la fixed header size ma include la extended header size

• Next header (8 bits): Identifica il prossimo extended header (se ce ne sono).

Permette di aggiungere header additionali usando una daisy chain.

• Hop limits (8 bits): Specifica il numero max di hops permesso per un datagram.

Decrementa ad ogni router prima dell'inoltro.

• Source Address (128 bits): Indirizzo sorgente del datagram

• Destination Address (128 bits): Indirizzo destinazione del datagram

## Extended Headers

Gli extended headers seguono il fixed header e precedono il transport header.

### 1. No Extended Header

Next Header=6 (TCP) TCP Header + Payload

### 2. With a Routing Header

Next Header=43 (Routing) Next Header=6 (TCP) TCP Header + Payload

### 3. With a Routing Header and Authentication Header

Next Header=43 Next Header=51 (AH) Next Header=6 (TCP) TCP Header + Payload



Il field next header indica la tipologia del prossimo header.

In questo modo l'header IPv6 rimane fixed a 40 bytes.

## Hop - by - hop header

Sono le opzioni che devono essere processate da tutti i router lungo il path (inclusi source e destination)

Hop-by-Hop Option and Destination Option Header Format

Offsets	Octet	0	1	2	3		
Octet	Bit	0 1 2 3 4 5 6 7	8 9 10 11 12 13 14 15	16 17 18 19 20 21 22 23	24 25 26 27 28 29 30 31		
0	0	Next Header	Hdr Ext Len	Options and Padding			
4	32			Options and Padding			
8	64				Optional: more Options and Padding ...		
12	96						

## Destination Header

Sono le opzioni che devono essere processati dal destination node

## Routing Header

Specifica un insieme di nodi che devono essere attraversati dal datagram per raggiungere la destinazione.

Il Path non viene calcolato dinamicamente dai router ma è fixed e specificato nell'header.

Routing Header Format

Offsets	Octet	0	1	2	3		
Octet	Bit	0 1 2 3 4 5 6 7	8 9 10 11 12 13 14 15	16 17 18 19 20 21 22 23	24 25 26 27 28 29 30 31		
0	0	Next Header	Hdr Ext Len	Routing Type	Segments Left		
4	32			Type-specific Data			
8	64				Optional: more Type-specific Data ...		
12	96						

lista dei router che devono essere attraversati

## Fragmented Header

Viene usato solo se necessario cioè quando la grandezza del datagram eccede la minima MTU (maximum transmission unit) del path.

Il fragment extension header trasporta l'info necessaria per riassamblare il datagram originale.

Extension Header	Type	Description
Hop-by-Hop Options	0	Options that need to be examined by all routers on the path Only header processed by all the routers
Routing	43	Methods to specify the route for a datagram (loose source routing)
Fragment	44	Contains parameters for fragmentation of datagrams
Destination Options	60	Options that need to be examined only by the destination of the datagram
Authentication Header (AH)	51	Contains information used to verify the authenticity of most parts of the datagram
Encapsulating Security Payload (ESP)	50	Carries encrypted data for secure communication

- ⇒ Next Header as in the original datagram
- ⇒ Identification value assigned by the source node
- ⇒ Frag. Offset indicates the offset wrt the start of original packet
- ⇒ M: More Fragment (1) or Last Fragment (0)



## Authentication Header

L' AH fa parte dell' IPsec ed è usato allo stesso modo che nel IPv4

## Encapsulation Security Payload (ESP)

L' ESP fa parte dell' IPsec ed è usato allo stesso modo che nel IPv4

## IPv6 Addressing scheme

Un indirizzo IPv6 è composto da 128 bit divisi in due parti:



### Address Representation

Un indirizzo IPv6 è rappresentato nel seguente formato: X:X:X:X:X:X:X:X

Esempio  
=> 2020:0db8:0000:0000:ff00:0042:0826

Ogni X rappresenta da 1-4 hexadecimal digits

### Short Notation

Uno o più zeri iniziali da ogni gruppo di hexadecimal digits vengono rimossi.

Questo viene applicato solitamente o a tutti o a nessuno. 2020:db8:0:0:ff00:42:0826

Sezioni consecutive di zeri sono riappiattite con le double colon (::)

Le double colon possono essere usate solo una volta in un indirizzo. 2020:db8::ff00:42:0826

### Loopback addresses

0000:0000:0000:0000:0000:0000:0001 => ::1  
abbreviato in

### Unspecified Address

0000:0000:0000:0000:0000:0000:0000 => ::  
abbreviato in

Questo indirizzo è usato come source address da un nodo che non ha ancora imparato il suo indirizzo di unicast.

NON deve essere usato come destination address

## Addressing Methods

### 1) Unicast Address

Questo indirizzo identifica univocamente una singola interfaccia network.

Un datagram inviato ad un unicast address è consegnato solo alla specifica interfaccia.

#### Struttura

Un unicast address è composto da:

1) Subnet Prefix : Identifica la subnet



2) Interface ID (suffix) : Identifica la specifica interfaccia, deve essere univoco all'interno della subnet e potrebbe essere localmente

Esistono tre tipi di unicast address:

1) Link Local Addresses : Identificano univocamente l'host all'interno della subnet.

↓  
usati solo localmente

È usato su una singola subnet per l'auto-config, neighbor discovery e in assenza di router nella subnet

	10		54 bits		64 bits	
+-----+		+-----+	+-----+	+-----+	+-----+	+-----+
1111111010	0		interface ID			

L'interface ID è assegnato dal nodo stesso senza un DHCP server

Viene usato un hash-based method per calcolare l'indirizzo.

2) Global unicast Address : Identifica in modo univoco l'host in internet.

- IPv6 che contengono IPv4 iniziano per 000
- IPv6 regolari hanno i bit più a sinistra = 001

	n bits		m bits		128-n-m bits	
+-----+		+-----+	+-----+	+-----+	+-----+	+-----+
global routing prefix   subnet ID   interface ID						

Global routing prefix: Valore assegnato al "silo" (insieme delle subnet)

Subnet ID : Identifica la subnet

3) Unique local Addresses (ULAs) : Usato per le comunicazioni in locale.

È comparabile all'indirizzo privato IPv4.

▪ Global routing prefix: typically hierarchically-structured value assigned to a site (a cluster of subnets/links)

▪ Subnet ID is an identifier of a link within the site

## Any Cast

Identifica un gruppo di interface.

Un pacchetto inviato ad un anycast address è consegnato a solo un indirizzo del gruppo di interface.

n bits	128-n bits
subnet prefix	00000000000000

è composto dal subnet prefix (n bits) seguito da 128-n uguali a zero

## Multicast Address

Identifica un gruppo di interface.

Un pacchetto inviato a un multicast address è consegnato a tutti gli indirizzi del gruppo di interface.

8	4	4	112 bits	group ID
11111111	flgs	scop		

- all-nodes address: FF02::1
- all-routers address: FF02::2
- solicited-node address: FF02:0:0:0:1:FFXX:XXXX
  - ⇒ computed as a function of a node unicast/anycast address
  - ⇒ formed by taking the low-order 24 bits of the node (unicast or anycast) and appending those bits to the prefix FF02:0:0:0:1:FF00::/104

## IPv6 Neighbor Discovery Protocol (NDP)

NDP fornisce un insieme di funzioni per l'autoconfigurazione.

### Servizi offerti:

- Router discovery
  - Hosts can locate routers residing on attached links
- Prefix discovery
  - Hosts can discover address prefixes that are on-link for attached links
- Parameter discovery
  - Hosts can find link parameters (e.g., MTU, hop limits, ...)
- Address autoconfiguration
  - Process by which a node can compute its unique global address
- Address resolution
  - mapping between IP addresses and link-layer addresses
- Next-hop determination
  - Algorithm to find the next-hop to forward a for a specific destination
- Neighbor Unreachability Detection (NUD)
  - Process by which a node determines that a neighbor is no longer reachable on the link
- Duplicate Address Detection (DAD)
  - Verification process to check whether an address is already in use
- Packet redirection
  - Process allowing a node to find a better next-hop route for a certain destination

## ICMPv6 Packets

NDP per offrire i servizi descritti usa dei pacchetti ICMPv6 :

- 1) Router Solicitation (RS, type 133) : Usato dall'host per localizzare i router nella subnet

- 2) Router Advertisement (RA, type 134): Può essere usato in due modi
- Usato dai router per avvertire della loro presenza periodicamente.
  - Invia in risposta a un RS message
- 3) Neighbor Solicitation (NS, type 135): Usato dagli host per
- Address resolution, il NS message è inviato per ottenere o confermare il link-layer address (subnet) di un nodo per il quale conosce l'IP address
  - Neighbor Unreachability Detection (NUD)
  - Duplicate Address Detection
- 4) Neighbor Advertisement (NA, type 136): Usato in due modi
- Dall'host in risposta ad un NS message
  - Informare un nodo del link-layer address (subnet)
- 5) Redirect (type 137): Invia dai router per informare un host di un migliore first-hop router sul path di destinazione.

L'autoconfigurazione è importante negli ambienti IoT, dovuto il grande numero di nodi e device incostituiti. Essa permette di auto-generare i link local addresses e global Addresses.

### Creatura dell'unicast IP

#### 1) Creazione del link local address

L'interface ID viene generato tramite un hash function dal MAC address



## 2) Ricezione delle prefix information dal router

Il nodo ha 2 opzioni:

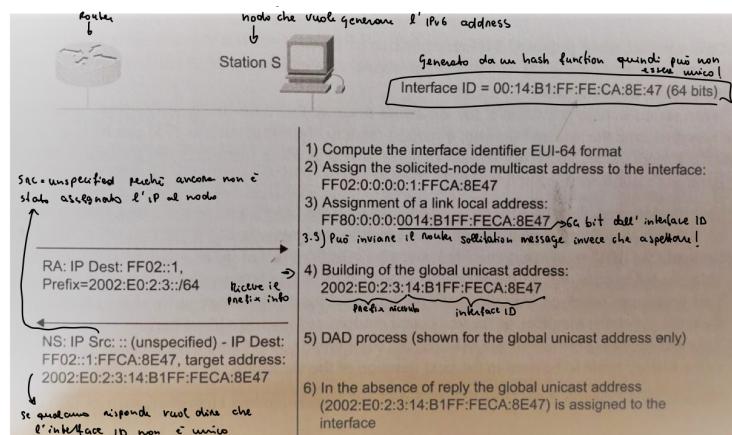
- a) Aspetta per un messaggio periodico RA  
oppure  
b) Il nodo invia un RS message per ricevere un RA message

Una volta ricevuto l'insieme dei prefissi, viene generato il global address usando l'unicast address e il prefix ricevuto.

## 3) Verifica dell'unicità (DAD procedure)

Viene usato un NS message per verificare se qualcun altro ha lo stesso indirizzo (può accadere perché l'interface ID è generato tramite Hash function).

Se qualcuno risponde con un NA message, un altro interface ID deve essere assegnato al nodo.



## DHCPv6

È simile al DHCPv4 ed è un meccanismo centralizzato per fornire:

- Indirizzo di rete
- Altre info utili come le DNS address

Ci sono due versioni:

- Stateful DHCP (RFC 3315): Il DHCP fornisce sia l'indirizzo che le info utili.

Per richiedere questi dati, il requesting node invia una richiesta all'indirizzo FF02::1:2

- 2) Stateless DHCP (RFC 3736): Il DHCP fornisce solo le info addizionali solo dopo che l'IPv6 global address è stato ottenuto.  
Non assegna il global address.

## 6 LowPAN Adaptation layer

È l'adaptation layer usato nello standard IEEE 802.15.4 (usato nelle lowPANs)

4 suoi obiettivi sono:

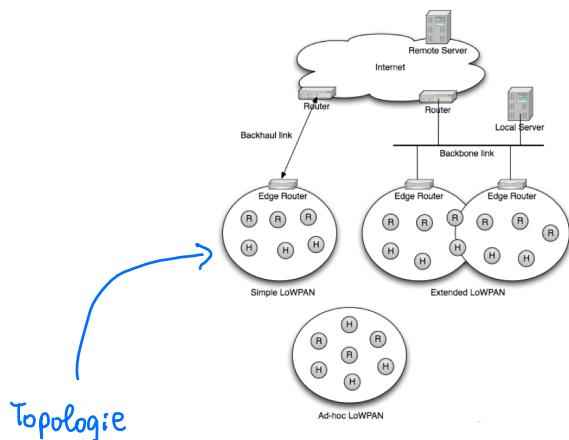
- 1) Efficient header compression
- 2) Fragmentation
- 3) Optimized Neighbor discovery

## 6 LowPAN architecture

### Nodes

4 nodi possono essere di tre tipi

- 1) Hosts
- 2) Routers (GLR)
- 3) Edge Routers (GLBR)



Topologie

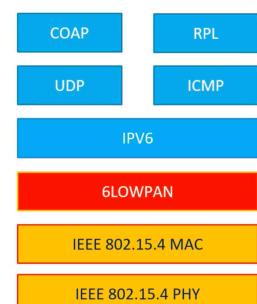
Ci sono tre tipi di topologie

- 1) Simple : c'è un singolo edge router
- 2) Extended : più edge routers connessi a un backbone link
- 3) Ad hoc: Non ci sono edge router.

la rete è isolata, non connessa a internet.

## 6 Low PAN caratteristiche

- 1) Packet Fragmentation e Riassemblamento
- 2) Header Compression
- 3) Supporto per indirizzi: a) 64 bit IEEE EUI 64- addresses  
b) 16 bit short addresses



4) Optimized Neighbor Discovery (Network autoconfig)

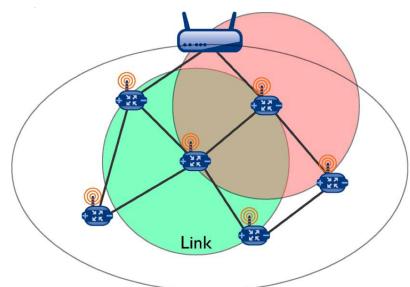
5) Supporto per link-layer Routing

## 6 LowPan Network architecture

Viene usato il concetto di IP Link (Nodi raggiungibili, cioè all'interno del transmission range) con un single ip hop

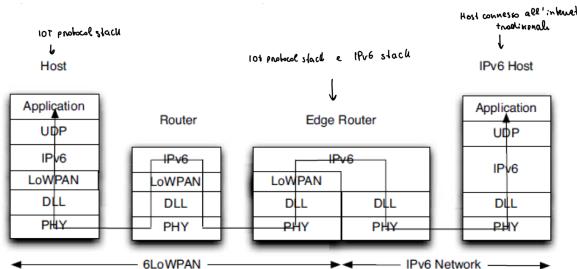
Il broadcast domain viene considerato come l'insieme di nodi all'interno del transmission range.

Il forwarding viene fatto secondo il "Route-over" invece che "mesh-under".



### Route-over

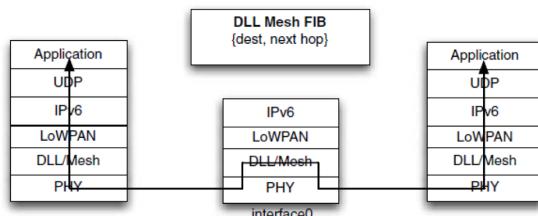
Il forwarding è eseguito al IPv6 layer.



### Mesh-under

Viene usato da alcuni standard.

Il forwarding è eseguito dall'underlying network.



## 6 LowPan Fragmentation e header Compression

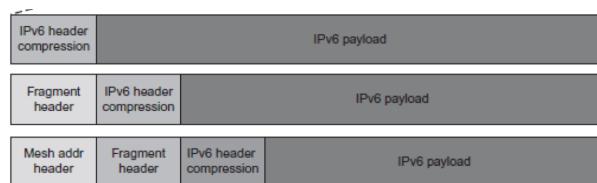
IPv6 richiede al link di trasportare un payload grande fino a 1280 bytes.

I low-power links non supportano solitamente certe dimensioni, infatti i frame dello standard IEEE 802.15.4 supportano 127 bytes di payload.

Per questo motivo viene usato un adaptation layer, per poter implementare le tecniche di:

1) IPv6 header compression

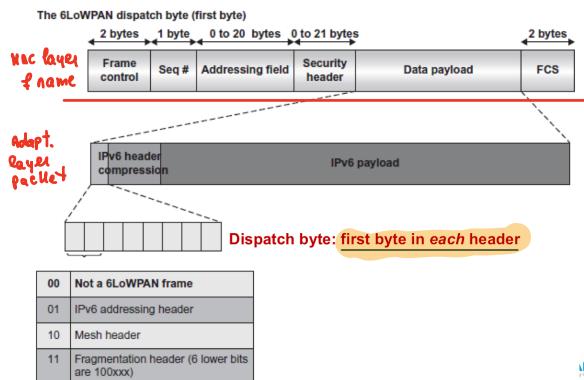
2) Fragment header



### 3) Mesh address header

L'ordine degli header, nel caso ne venisse usato più di uno, è quello nell'immagine

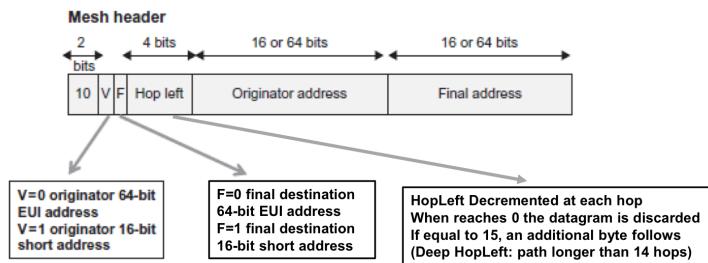
Ogni header ha un dispatch byte (4° byte) il quale indica il tipo di header.



Come mostrato dall'immagine, gli header e il payload (che formano l'adaption layer packet) sono incapsulati all'interno del payload del MAC layer frame.

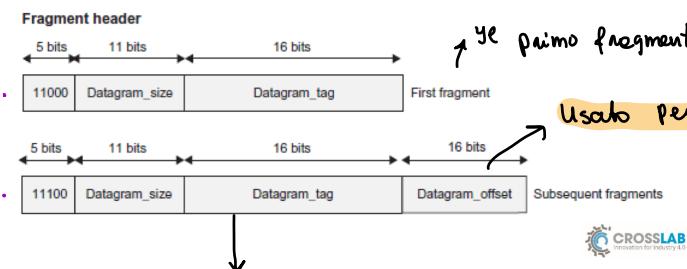
### Mesh addressing header

Viene usato solo nel caso del mesh-under routing approach.



### Fragmentation header

La frammentazione viene fatta ogni qualvolta l'ipv6 supera la MTU dell' IEEE 802.15.4



Tra il primo frammento e i seguenti, cambia il terzo bit da 0 a +

## Header compression

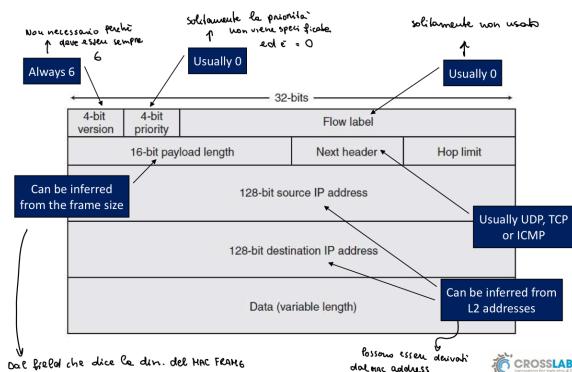
La header compression offre due caratteristiche:

1) **Efficienza**: Trasmettere un minor numero di dati permette di ridurre il costo energetico e, a volte, evitare la frammentazione.

2) **Ridondanza**: Evitare di inviare info inutile o

che possono essere inferite da altri livelli.

Ad esempio nella foto si possono notare molti campi ridondanti o che possono essere inferiti da altre info.



Ci sono tre tipologie di header compression:

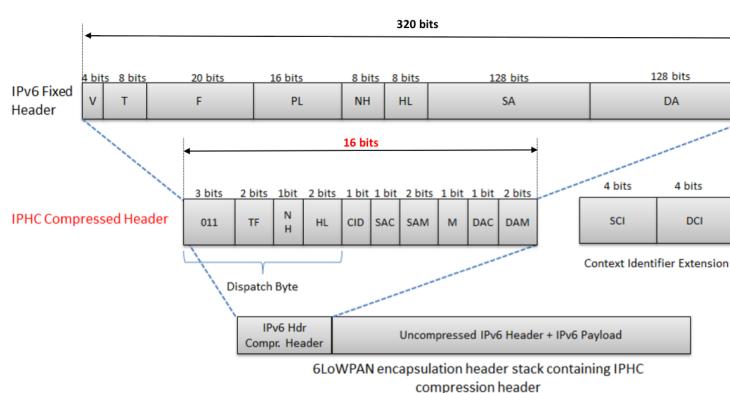
1) **Stateful header compression**: Usata quando si ha un flow  $\rightarrow$  Non adatta per le flow pairs essendo controllate da short lived flows.

2) **Stateless header compression**: Rimuove info ridondanti tra i livelli.

Effetto limitato su pacchetti destinati al di fuori della lowPan.

3) **Context based Compression**: Spesso la migliore opzione, usa le context info condivise dal source node al destination node.

## Come viene compresso l'header?



- **Version Field (V)** always elided
- **Payload Size (PL)** inferred from the 802.15.4 frame
- **Traffic Class (TC) + Flow Label (FL) → TF (2 bits)**
  - 00: Both TC and FL are carried in line (subsequent 32 bits)
  - 01: TC field compressed (2-bit ECN), FL in line
  - 10: TC uncompressed and in line, FL elided
  - 11: Both TC and FL compressed
- **Next Header (NH) → NH (1 bit)**
  - 0: original NH field (8 bit) in line
  - 1: original NH field elided



- **Source Address Mode (SAM, 2 bits)**
  - SAC=0: Stateless address compression
    - ⇒ 00: full 128-bit source address in line
    - ⇒ 01: first 64 bits elided, remaining 64 bits carried in line
    - ⇒ 10: first 112 bits elided, remaining 16 bits carried in line
    - ⇒ 11: address fully elided; the first 64 bits are the local prefix, the remaining 64 bits are inferred from the IEEE 802.15.4 frame
  - SAC=1: Stateful address compression, based on context
    - ⇒ 00: the address is the unspecified address (::)
    - ⇒ 01: the 64-bit prefix is derived from context information, the remaining 64 bits are inline (64 bits)
    - ⇒ 10: the 64-bit prefix is derived from context information, the remaining 16 bits are inline (16 bits)
    - ⇒ 11: the address is derived from context information and, potentially, the link-layer frame (0 bits)



- **Destination Address Mode (DAM, 2 bits)**
  - M=0, DAC=0 (unicast address, stateless compression)
    - ⇒ 00: the full 128-bit address is in line (128 bits)
    - ⇒ 01: The first 64-bits of the address are elided. The value of those bits is the link-local prefix padded with zeros. The remaining 64 bits are carried in-line (64 bits)
    - ⇒ 10: The first 112 bits of the address are elided. The value of the first 64 bits is the link-local prefix padded with zeros. The following 64 bits are 0000:00FF:FE00:XXXX, where XXXX are the 16 bits carried in-line (16 bits)
    - ⇒ 11: The address is fully elided. The first 64 bits of the address are the link-local prefix padded with zeros. The remaining 64 bits are computed from the encapsulating 802.15.4 frame header (0 bits)

- **Destination Address Mode (DAM, 2 bits)**
  - M=1, DAC=0 (multicast address, stateless compr.)
    - ⇒ 00: the full 128-bit address is in line (128 bits)
    - ⇒ 01: The address takes the form
      - FFXX::0XXX:XXXX:XXXX (48 bits)
    - ⇒ 10: The address takes the form
      - FFXX::0XXX:XXXX (32 bits)
    - ⇒ 11: The address takes the form
      - FF02::00XX (8 bits)

#### ▪ Context

- Shared information between the node that compresses a packet and the node(s) that need(s) to expand it
  - ⇒ up to 16 contexts
- The context used to encode the source address does not have to be the same as the context used to encode the destination address



#### ▪ CID =1: additional byte for context

- following the DAM bits but before the IPv6 header fields that are carried in-line
- identifies the pair of contexts to be used when the IPv6 source and/or destination address is compressed
  - ⇒ 4 bits for each address → 16 different contexts (0 is default)



- **Hop Limit (HL) → HL (2 bit)**
  - 00: original HL field in line
  - 01: original HL field elided, hop limit = 1
  - 10: original HL field elided, hop limit = 64
  - 11: original HL field elided, hop limit = 255
- **Context Identifier extension (CID, 1 bit)**
  - 0: No additional context information
  - 1: 1-byte context information immediately after the DAM field
- **Source Address Compression (SAC, 1 bit)**
  - 0: Stateless address compression
  - 1: Stateful address compression, based on context



- **Multicast compression (M, 1 bit)**
  - 0: the destination address is *not a multicast address*
  - 1: the destination address is a *multicast address*
- **Destination Address Compression (DAC, 1 bit)**
  - 0: the compression of the destination address is *stateless*
  - 1: the compression of the destination address is *stateful*
    - ⇒ based on context

#### ▪ **Destination Address Mode (DAM, 2 bits)**

- **Destination Address Mode (DAM, 2 bits)**
  - M=0, DAC=1 (unicast address, stateful compression)
    - ⇒ 00: Reserved
    - ⇒ 01: The address is derived using context information and the 64 bits carried in-line (64 bits)
      - Bits covered by context information are always used. Any IID bits not covered by context information are taken directly from the corresponding bits carried in-line. Any remaining bits are zero.
    - ⇒ 10: The address is derived using context information and the 16 bits carried in-line (16 bits)
      - Bits covered by context information are always used. Any IID bits not covered by context information are taken directly from their corresponding bits in the 16-bit to IID mapping given by 0000:00FF:FE00:XXXX, where XXXX are the 16 bits carried in-line. Any remaining bits are zero.
    - ⇒ 11: The address is fully elided and is derived using context information and the encapsulating header (e.g. 802.15.4 or IPv6 destination address, 0 bits)
      - Bits covered by context information are always used. Any IID bits not covered by context information are computed from the encapsulating header as specified. Any remaining bits are zero.



#### ▪ **Destination Address Mode (DAM, 2 bits)**

- M=1, DAC=1 (multicast address, stateful compr.)
  - ⇒ 00: This format is designed to match Unicast-Prefix-based IPv6 Multicast Addresses (48 bits)
    - FFXX:XXLL:PPPP:PPPP:PPPP:XXXX:XXXX
    - X are the nibbles that are carried in-line, in the order in which they appear in this format
    - P denotes nibbles used to encode the prefix itself.
    - L denotes nibbles used to encode the prefix length.
    - The prefix information P and L is taken from the specified context
  - ⇒ 01: Reserved
  - ⇒ 10: Reserved
  - ⇒ 11: ReservedF

#### ▪ **Source Context Identifier (SCI)**

- Identifies the prefix that is used when the IPv6 source address is statefully compressed

#### ▪ **Destination Context Identifier (DCI)**

- Identifies the prefix that is used when the IPv6 destination address is statefully compressed

## 6LoPAN Neighbor discovery

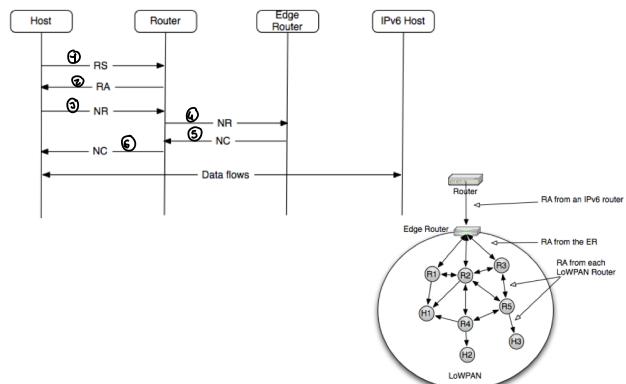
Ye Neighbor discovery usato nell' IPv6 non funziona bene per le lowPANs perché ci sono multiple overlap broadcast domains e la comunicazione multicast è troppo costosa.

Ye 6LoPAN Neighbor discovery utilizza un approccio "centralizzato" dove gli edge routers conoscono tutti gli address usati dai nodi all'interno della lowPAN.

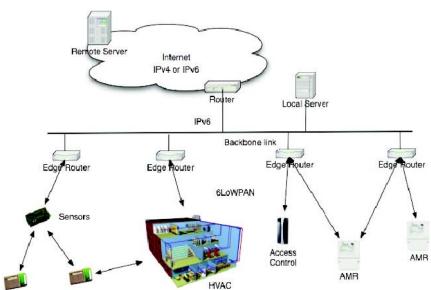
Dovuto a ciò, il processo di DAD viene fatto dall' edge router invece di utilizzare un multicast approach.

NR : Messaggio di registrazione

NC : Messaggio di conferma



## Esempio di una rete



### Edge Router

- Has both a 6LoWPAN interface and a regular IPv6 interface
- and it is responsible for
  - ⇒ Diffusing RAs to the 6LoWPAN network
  - ⇒ Address Resolution of a node
  - ⇒ Maintaining a whiteboard of IPv6 addresses
  - ⇒ Duplicate Address Detection for the addresses it defends
  - ⇒ Listening for RS and Node Registration messages
  - ⇒ Can also generate IPv6 addresses using IEEE 802.15.4 short addresses on behalf of the registering nodes

### Edge Router forwards packets

- from one 6LoWPAN to another 6LoWPAN or to the backbone network
  - ⇒ When packets are forwarded to the backbone network, the 6LoWPAN adaptation layer is stripped off, the header is uncompressed and it makes sure that global IPv6 source address is used for the outgoing packets
- from the backbone network to one 6LoWPAN
  - ⇒ For incoming packets, ER adds 6LoWPAN specific adaptation layer and possibly 6LoWPAN IPv6 header compression mechanism and then forwards them to the 6LoWPAN

### LowPAN Router

- forwards data packets across links and relays control packets between the Edge router and the 6LoWPAN nodes
- Responds to the RS messages from hosts on the same link with RAs
- 6LoWPAN routers are aware of the Edge Router address and may be able to forward a packet addressed to the 6LoWPAN\_ER anycast address

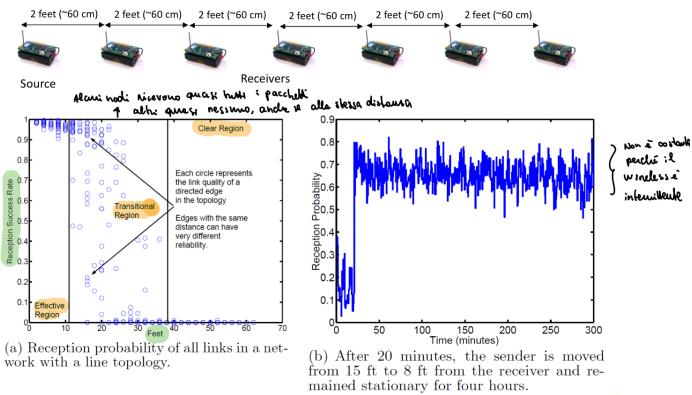
### LowPAN Host (Nodes)

- Are required to do very little ND signaling
- Do not need to perform
  - ⇒ DAD
  - ⇒ Address Resolution
  - ⇒ Neighbor Solicitation
- They just have to periodically renew their registration with one or more Edge Routers
- **LowPAN Host (Node)**
  - After bootstrap, forms an optimistic IPv6 link local address from the EUI64 bit MAC address
  - sends a RS to its own link router (or ER)
    - ⇒ The node is recommended not to use multicast address
    - ⇒ Since the IPv6 address contains the EUI64 MAC address in its lower 64 bit address, NS to the ER for address resolution is not required
  - Registers with the ER
  - Can send data packets to a 6LoWPAN node's address via the 6LoWPAN routers that act as default routers
  - The node should configure a global IPv6 address to communicate with nodes outside the 6LoWPAN 

## Routing protocol per LLN

Dobbiamo usare un routing protocol che sia adatto alle caratteristiche delle LLN:

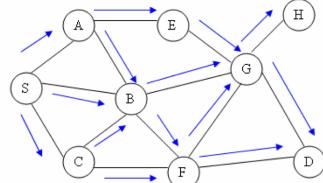
- 1) Resource constraints
- 2) Intermittent connectivity
- 3) lossy links



## Routing schemes

- 1) Flat routing: Tutti i nodi hanno la stessa importanza

- Approssimativi: Flooding, gossiping, AODV...
- È un approssimativo molto costoso in termini di bandwidth e energia.



Soprattutto dovuto all'alto numero di interferenze che portano a retransmissioni.

Di conseguenza non è molto adatto alle LLN.

- 2) Hierarchical Routing: Viene rispettata una gerarchia che spesso può venir costituita inizialmente

- attraverso il flooding.
- Approssimativi: LEACH
- Tipologia di nodi:
  - 1) Sensor nodes normali
  - 2) Relay nodes
- Two-step Forwarding:
  - 1) I sensor nodes inviano i dati ai relay nodes
  - 2) I relay nodes inoltrano i dati al sink node
- Possibili approssimativi:
  - 1) Tree Topology: Solo un path da un nodo al sink node
  - 2) Mesh Topology: Più path da un nodo al sink node



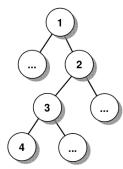
## Tree based Routing

Ogni nodo invia i dati al suo upstream parent.

Come costruire un tree path?

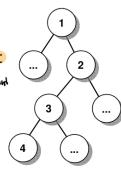
### 1) Broadcast and reverse path

- a) The sink periodically broadcasts beacons
  - b) The other nodes re-broadcast beacon messages
    - Possibly after a random delay to avoid collisions
  - c) Nodes use the source address of the first received beacon to select the parent node
- simplest way, ma non è il più efficiente
- g. beacon  
odeviamo  
l'hop  
count
- Node viene usato il primo  
beacon ricevuto per decidere  
di parent
- Limits?**
- The selected path may not be the shortest path
  - Link quality is not considered
  - Lead to convoluted trees with many long and unreliable links



### 2) Shortest Path (SP)

- Conventional Distance Vector (DV) protocol
  - A node is a **neighbor** if a packet is received from it
  - Each node selects as its **parent** the **minimum hop count neighbor**
  - hop\_count(node) = hop\_count(parent) + 1
- l'hop count parte da 0 dal sink node
- value di hop count minimo
- Limits?**
- Link quality is not considered
    - Hop count is not an adequate metric in lossy links
    - The expected # of transmissions would be more appropriate
  - Possible Variant: Shortest Path with link-quality Threshold (SP(t))
    - a node is a neighbor if the link quality exceed a pre-defined threshold t (e.g., 70% or 40%)



### 3) location based routing : usa le info di posizione

RPL (Ripple) = Routing Protocol for LLNs

RPL è il routing protocol creato dall'IETF Roll WG, nel 2008, appositamente per le LLN. È un Distance Vector and source routing protocol, layer -3.

#### RPL Network

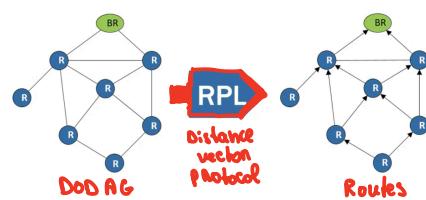
Il Network è composto da 1 o più DODAG (Destination-Oriented Directed Acyclic Graph).

Una DODAG è una struttura basata sui tree, dove ogni nodo però può avere più parents e tra essi ne viene scelto uno chiamato "preferred parent".

I path vengono calcolati con il distance vector protocol (RPL).

Quindi i passi sono:

- 1) Creare la DODAG
- 2) Usare l'RPL per creare le routes dalla DODAG
- 3) Usare le routes create per il routing



## Type of Nodes

1) Low Power and lossy Border Router (LBR) : Sono la radice di una DODAG.

Essi permettono la connessione tra la LLN e internet.

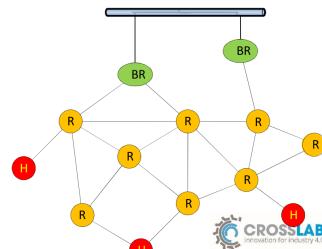
In una LLN ci possono essere più di un LBR e ognuno di essi ha una DODAG associata.

la creazione di una DODAG parte dai LBR.

2) Router : È un dispositivo che può generare e inoltrare traffico.

Esso non può creare una DODAG, ma può associarsi a una DODAG esistente.

3) Host : È un end device che può solo generare traffico, non inoltrarlo.



## RPL topology

La topologia del RPL è sia:

1) Hierarchical topology : I nodi sono fatti a formare le DODAGs che sono basate su una parent-child relationship.

2) Mesh Topology : La comunicazione può avvenire anche tra "fratelli", non solo figlio - genitore.

## Communication paradigms

1) Multi - Point-to-Point (MP2P) : Many To one communication } più frequente

2) Point - to - Multi - Point (P2MP) : One to many communication

3) Point - to - Point (P2P) : One - to - One communication

## RPL Main Features

### 1) Auto-configuration

- RPL-based LLNs benefit from basic IPv6 routing characteristics
  - ⇒ Neighbor Discovery

### 2) Self-healing

- Adaptation to network topology changes and node failures
  - ⇒ links and nodes in LLNs are not stable and may vary frequently
  - ⇒ mechanisms that choose more than one parent per node in the DAG to eliminate/decrease the risks of failure

### 3) Loop avoidance and detection

- RPL includes reactive mechanisms to detect loops
- RPL triggers recovery mechanisms (global and local repair) when the loops occur

### 4) Independence and Transparency

- RPL can operate over multiple link layers
  - ⇒ independent from data-link layer technology
- RPL is designed to operate in LLNs
  - ⇒ constrained networks
  - ⇒ highly constrained devices

### 5) Multiple edge routers

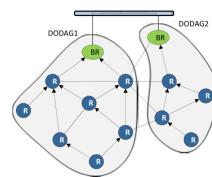
- It is possible to construct multiple DAGs, each with a root
- A node may belong to multiple instances, and may act different roles in each instance
  - ⇒ High availability
  - ⇒ Load balancing

## RPL instances

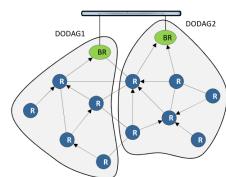
Possiamo avere più RPL instances nella stessa LLN per gestire diversi routing requirements.

Esse sono identificate dal RPL instance ID e più DODAG possono far parte della stessa istanza.

RPL Instance 1



RPL Instance 2



Tutte le DODAG sotto la stessa istanza condividono le stesse metriche e vincoli (e come vengono usate queste info per costruire la DODAG)

I nodi RPL possono appartenere a più RPL instances diverse contemporaneamente.

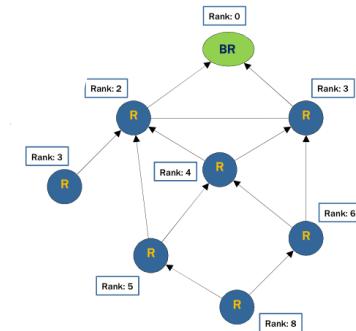
## RPL Rank

Ad ogni nodo viene assegnato un rank, il quale è un intero che rappresenta la posizione di un nodo all'interno di una DODAG. Il valore aumenta man mano verso le foglie e un valore minore del rank indica un rank maggiore nella DODAG.

Usato per:

- 1) Rilevare possibili loop
- 2) Selezionare il prossimo hop per il forwarding (tra i nodi con  $\text{rank} \geq$ )

{  
• Rank > : Parent node  
• Rank = : Sibling node



## Objective Function

È la funzione usata per definire:

- 1) Come calcolare il rank dalle metriche usate
- 2) Come calcolare il path cost
- 3) Come selezionare i parents (quando, chi e quanti)
- 4) Come "pubblicizzare" il path cost

- Node State and Attributes
  - CPU overload, available memory
    - ⇒ 1 bit to signal congestion
- Node energy
  - Node power mode (2 bits)
    - ⇒ main powered
    - ⇒ battery powered
    - ⇒ Scavenger
  - Estimated residual lifetime
  - Other metrics (TBD)
- Hop count
- Link Throughput
  - Range of throughput the link can handle
- Link Latency
- Link reliability
  - Link Quality Level (LQL: 0-7)
    - ⇒ 0: unknown
    - ⇒ 7: highest level
  - Expected Transmission Count (ETX)
- Link color
  - 10-bit encoded colors (implementation specific)
  - Allow to attract/avoid specific links for specific traffic types

## Tipologie di Objective Functions:

1) Objective Function zero (OF0) : Non vengono usate le metriche, fa riferimento solo alle config di default.

2) Minimum Rank with Hysteresis Objective Function (MRHOF) :

Calcola il rank in base alle metriche (node o link metrics).

Soltamente viene usato "l'expected number of transmission di uno specifico link" come metrica.

## RPL control Messages

I control messages usati da RPL sono basati sugli ICMPv6 messages.

Sono composti da due parti:

octets: 1 1 2			variable	
Type	Code	Checksum	Message Body	
Header			Body	

1) Message header:

a) Type : È settato a "155" nel caso di RPL

b) Code : Identifica la tipologia del RPL control message

c) Checksum

2) Message Body:

a) Message base

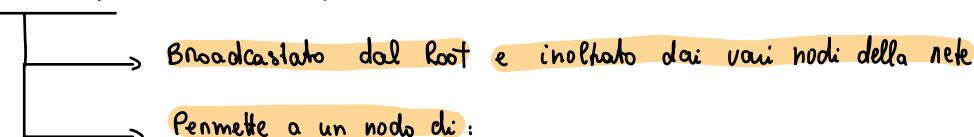
b) Options

RPL Type	Description
0x00	DODAG Information Solicitation (DIS)
0x01	DODAG Information Object (DIO)
0x02	Destination Advertisement Object (DAO)
0x03	Reserved

## Type of Control Messages

1) DIS Message : Viene usato per sollecitare un DIO message da parte di un RPL node

2) DIO Message : Viene usato per costruire una nuova DODAG.

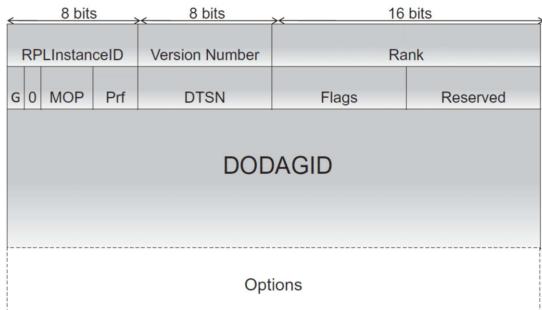


a) Discover a RPL instance

b) Imparare i suoi conf. parameters

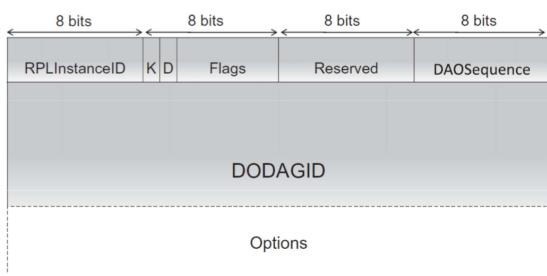
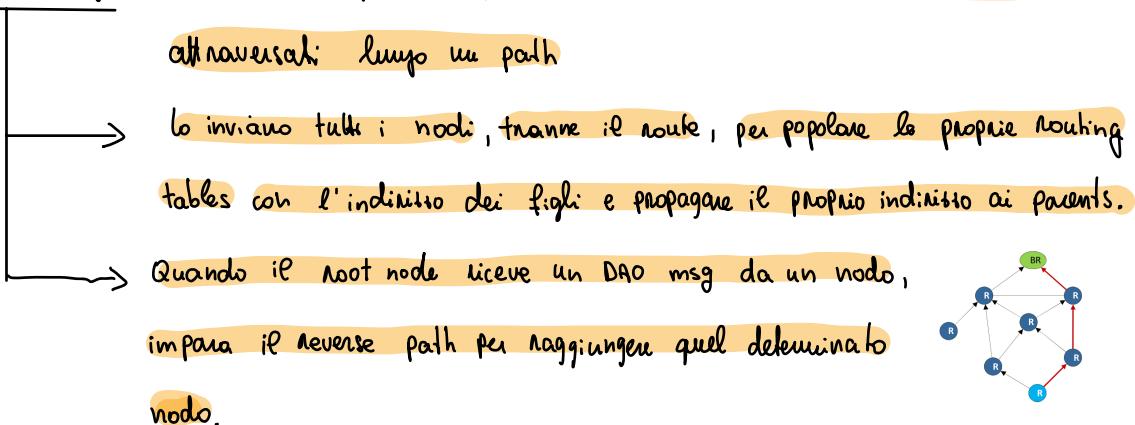
c) Selezionare un parent set nella DODAG

d) Mantenere la DODAG

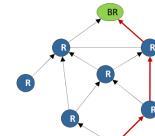


- RPLInstanceID
  - ID of the RPL instance the DODAG is part of
- Version Number
  - Version # of the DODAG  $\Rightarrow$  viene aggiornata ogni tot tempo da parte del root facendo partire una nuova carabinazione
  - Incremented at each network information update
- Rank
  - The rank of the node sending the DIO message
- Destination Advertisement Trigger Seq Number (DTSN)
  - Used to maintain downwards routes
- Grounded (G)
  - A flag indicating whether the DODAG satisfies the application-defined objective
- Mode of Operation (MoP)
  - Sent by the root, identifies the mode of operation
  - A node must be able to cope with the MoP to join as a router
- DODAG Preference (Prf)
  - Preference degree of the DODAG root, with respect to other DODAG roots
    - $\Rightarrow$  00: least preferred degree (default)
    - $\Rightarrow$  07: most preferred degree
- DODAG ID
  - Uniquely identifies the DODAG
  - Set by the DODAG root

3) DAO Message: Viene usato per propagare le reverse route info ricordando i nodi attraversati lungo un path



- RPLInstanceID
  - ID of the RPL instance the DODAG is part of
    - $\Rightarrow$  Learned from the DIO
- K Flag
  - Indicates whether an ack is required in response to a DAO
- DAOSequence
  - Sequence Number, incremented at each DAO message
- DODAG ID
  - Set by the DODAG root
  - Uniquely identifies the DODAG
  - This field is present only when flag D is set to 1



4) DAO ACK Message : Viene inviato in risposta a un DAO message

→ Ha alcuni fields :

- RPL instance ID
- DAO sequence
- status : Uno status code > 128 indica un reject  
e il nodo dovrebbe selezionare un parent alternativo

### RPL DODAG construction

La costruzione di una DODAG è basata su due fasi:

- 1) Trasmissione broadcast dei DIO messages ⇒ Per costituire il path dai nodi al root
- 2) Trasmissione unicast dei DAO messages ⇒ Per costituire il path dal root ai nodi

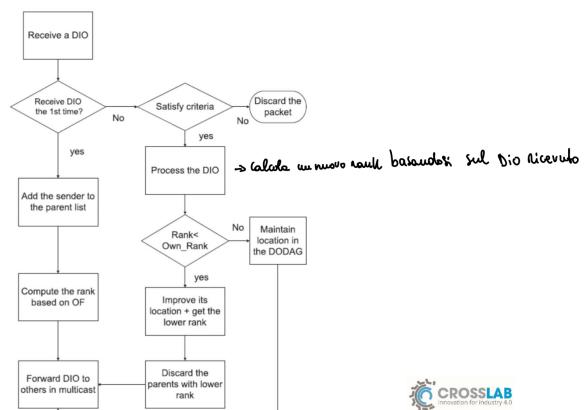
#### FASE 1

Il root播播ca il DIO message per annunciare:

- 1) Il DODAG ID
- 2) Il RANK ⇒ Permette ad ogni nodo di determinare la sua posizione all'interno della DODAG
- 3) l' Objective Function ⇒ Usata per calcolare il rank  
→ È contenuta nel Option field del DIO msg

Questi messaggi possono essere ricevuti da:  
a) Un nodo che vuole Joinare la DODAG  
b) Un nodo che ne fa già parte

Quando viene ricevuto un DIO message :



## Parent selection

Ricevendo un DIO message, un nodo impara l'insieme di nodi che stanno nel one-hop neighborhood (nodi raggiungibili con 1-hop), questo è il suo neighbor set.

Usando la obj function, un nodo:

- 1) Determina il suo rank basandosi sui rank ricevuti dai vicini (neighbor nodes)
- 2) Seleziona uno o più parents dal suo neighbor set
- 3) Seleziona un preferred parent dal parent set (quello di default verso il DODAG root)

Il route formato dai preferred parent di ogni nodo è chiamato Default route.

Il default route viene usato dai nodi per inviare i pacchetti verso il DODAG root.

4) Dio msgs vengono re-broadcastati ogni tot tempo (regolato dal trickle algorithm) per aggiornare le routing info.

## Fase 2

Ogni nodo invia un unicast DAO message verso il root.

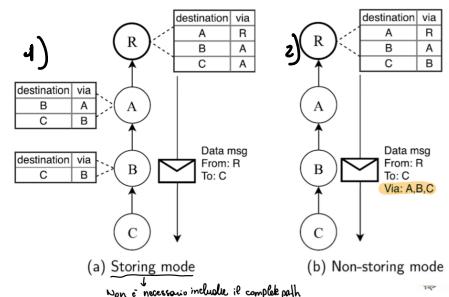
Quando un nodo riceve un DAO message:

- 1) inserisce il suo indirizzo nel DAO message
- 2) ritrasmette il DAO message aggiornato al preferred parent.

Quando il DAO message arriva al root, esso impara il downward route verso il nodo.

Ci possono essere due modes of operation:

- 1) Storing mode: Ogni nodo salva il route fino al source node in memoria.
- 2) Non-Storing mode: Non viene salvato il route in memoria



## RPL Network management

Vengono svolte tre operazioni per la gestione della rete:

- 1) DODAG Repair operation:

Viene eseguita per adattare la Topologia della rete a possibili link/node failures.

Possono avvenire due riparazioni:

a) **locale**: Fatta quando vengono rilevate delle network inconsistency.

Il path viene riempiatato da un path alternativo anche se non ottimale (nuovo preferred parent).

b) **global**: Fatto dal root della DODAG quando più inconsistenti vengono rilevate.

Viene generata una nuova versione della DODAG.

2) **Rooting loop**:

Un loop avviene quando un nodo perde tutti i suoi parents e cerca di attaccarsi a un nodo in una sua sub-DODAG.

a) **loop avoidance**: basata sul name

b) **loop detection**: basata sul path. Controlla le info nei data packet per assicurarsi che un pacchetto si stia muovendo nella forward direction.

3) **Security Modes**:

a) **Unsecured**: Senza sicurezza

b) **Pre-installed**: I nodi hanno una chiave pre-installata per mettere in sicurezza gli RPL control messages in termini di confidentialità e integrità.

c) **Authenticated**: Viene usata la pre-installed key per JOINARE una RPL instance come foglia.

## RPL Performance Evaluation

Per valutare le performance di RPL possiamo valutare:

1) **DODAG Convergence time**: Tempo necessario per la creazione della DODAG.

2) **Power Convergence**: Energia consumata in media da ogni nodo per la costituzione della DODAG.

3) **Path length**: Numero di hops per arrivare da un nodo al Root.

4) **latency**: Tempo per inviare un pacchetto da un nodo a un altro.

## Industrial IoT

le applicazioni industriali richiedono caratteristiche specifiche rispetto alle normali applicazioni:

- **Affidabilità** nella collezione dei dati
- **Vincoli in tempo** reali nella collezione dei dati
- **Efficienza energetica**
- **Scalabilità**

Inoltre, possono essere divise in alcune categorie:

### 1) Safety - critical applications

- Class 0 : Emergency actions

### 2) Control applications

- Class 1 : Closed-loop regulatory control
- Class 2 : Closed-loop supervisory control
- Class 3 : Open-loop control

### 3) Monitoring applications:

- Class 4 : Alerting
- Class 5 : Logging and monitoring

class 0,1,2 solitamente

usano una connessione cavo

High reliability

Fault tolerance

Deterministic latency

Bounded jitter

Security

class 2,3,4,5 possono usare

una connessione wireless

Flexibility

Mobility

Easy of deployment

Limited Cost

## Limiti dello standard 802.15.4

- 1) **Unbounded delay**: Usa il CSMA-CA, di conseguenza non può garantire nessun limite massimo sul tempo impiegato dai dati per raggiungere la dest. finale.
- 2) **Limited communication reliability**: Basso delivery ratio dovuto all'inefficienza del time slotted CSMA-CA algorithm, usato per l'accesso ai canali.
- 3) **No protection against interferences/fading**: L'802.15.4 MAC usa un singolo canale e non ha un meccanismo per il cambio delle frequenze.
- 4) **Powered relay nodes**: Le topologie multi-hop richiedono meccanismi complessi di sync. e

beacon scheduling. Per risolvere questo problema, in molti applicazioni i relay nodes intermedi mantengono il loro radio sempre attivo, causando un grande consumo di energia.

Per questi motivi, lo standard 802.15.4 non è adatto per molti scenari critici.

### IEEE 802.15.4 e

Questo standard venne creato per sopperire alle problematiche del 802.15.4. L'obiettivo era di definire un low-power multi-hop MAC protocol, capace di soddisfare le emergenze richieste dalle applicazioni industriali.

Questo standard prende alcune idee dai precedenti industrial standards (WirelessHART e ISA 100.11.a) tra le quali: slotted access, shared and dedicated slots, multi-channel communication, and frequency hopping.

Lo standard ha definito 5 diverse MAC behavior Modes, ma solo la Time slotted Channel Hopping (TSCH) ha avuto successo.

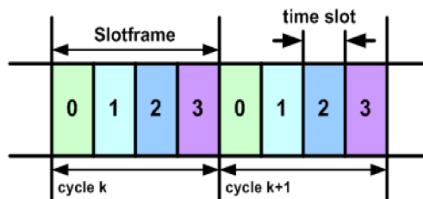
**TSCH** (Time slotted Channel hopping)  $\rightarrow$  MAC protocol

TSCH si concentra su tre caratteristiche:

- 1) Time-slotted access: Aumenta il potenziale throughput eliminando le collisioni tra competing nodes e fornendo deterministic latency all'applicazione.
- 2) Multi-channel communication: Più nodi possono comunicare nello stesso momento usando channels differenti.
- 3) Channel hopping: La frequenza usata dai nodi cambia ad ogni timeslot, mitigando l'interferenza e il multipath fading e migliorando l'affidabilità.

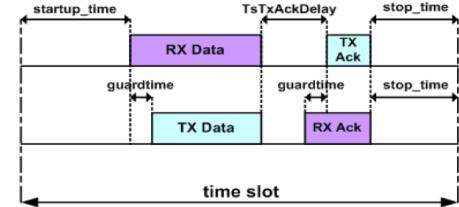
In TSCH il tempo è diviso in timeslots e l'insieme di  $n$  timeslots fanno uno slotframe.

Gli slotframe hanno tutti lo stesso numero di timeslots e fanno che si ripetano ad ogni ciclo.



Un singolo timeslot è composto da:

- 1) **Startup time**: Tempo di attesa dopo l'inizio di uno slotframe.
- 2) **Guard time**: Tempo che intercorre tra l'inizio dell'ascolto del receiver e invio del transmitter.
- 3) **TsTxAckDelay**: Tempo di attesa di un ACK.
- 4) **StopTime**: Tempo di attesa prima della fine dello slotframe.



## TSCH links

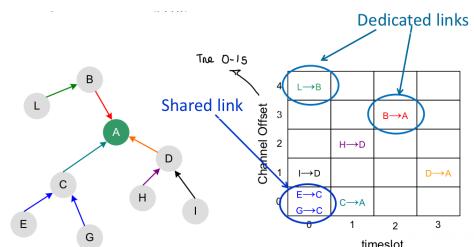
Un link è un assegnazione a coppie di una comunicazione diretta tra dispositivi in uno specifico slot, con un dato offset.

Esistono due tipi di link:

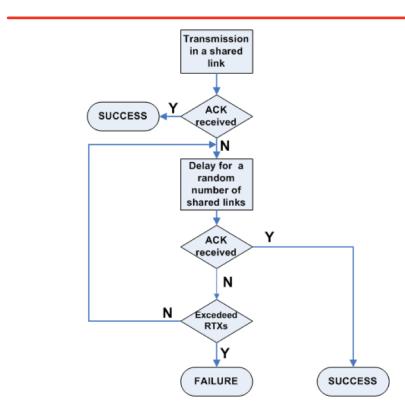
- 1) **Dedicated links**: Sono link dedicati a un transmitter e un receiver.

Permettono:

- a) Traffico deterministico
- b) Trasmissioni periodiche
- c) Accesso diretto



- 2) **Shared links**: Sono link con transmitter e receivers multipli.



Caratterizzati da:

- a) Traffico spontaneo e imprevedibile  $\Rightarrow$  Discovery e routing messages
- b) CSMA-CA based access

TSCH CSMA-CA  
vs  
802.11s, 6

- Backoff mechanism
  - is activated only after the node has experienced a collision (to avoid repeated collisions)
- Backoff unit duration
  - TSCH: a shared slot
  - 802.15.4: 320 µsec
- Clear Channel Assessment (CCA)
  - CCAs are not used to prevent collisions among nodes, but to avoid transmitting a packet if a strong external interference is detected
- Packet dropping
  - a packet is dropped only if it reaches the maximum number of retransmissions

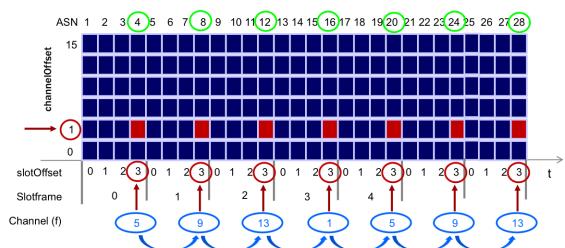
## TSCH channel hopping

Il channel offset di un link =  $(t, chOf)$  e' tradotto in una operating frequency  $f$

$$f = F \{(\text{ASN} + chOf) \mod n_{ch}\}$$

$$f = F \{(\text{ASN} + chOf) \mod n_{ch}\}$$

- ASN: total # of slots elapsed since the network was deployed
- $n_{ch}$ : number of used physical channels (16)
- $F$  is implemented as a look-up-table containing the set of available channels



OSS: Slot frame size e  $n_{ch}$  dovrebbero essere coprimenti per usare il maggior numero di frequenze, perché senno si rischia che solo un piccolo numero di freq. venga usato.

## TSCH Network formation

La costituzione della rete e' basata sugli Enhanced Beacons (EBs).

EBs sono frame speciali che contengono:

- Synchronization information = Permette ai nuovi device di sincronizzarsi alla rete.
- Channel hopping information = Permette ai nuovi dispositivi di imparare la channel hopping sequence.
- Timeslot information = Descrive quando aspettare un frame transmission e quando inviare un ACK.
- Initial link and slotframe info = Permette ai nuovi dispositivi di conoscere:
  - Quando ascoltare per una trasmissione dagli advertising device
  - Quando trasmettere agli advertising device

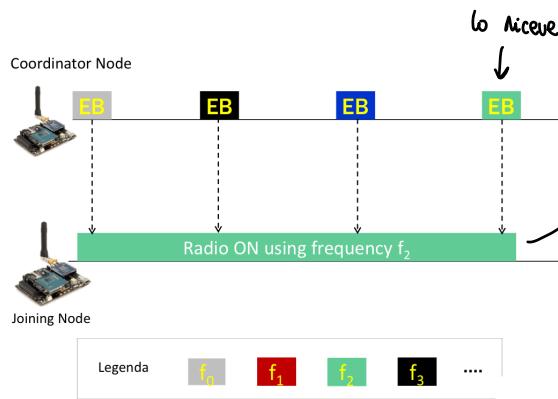
OSS: L'EB advertising policy non fa parte del TSCH protocol.

## Implementation

- 1) Un joining device inizia ad ascoltare per possibili EBs su una specifica frequenza

?) Quando riceve un EB:

- Il MAC layer notifica l'higher layer
- l'higher layer initializza lo slotframe e i links usando le info ricevute dagli EB messages e switcha la modalità del device in TSCH mode.  
In questo momento il dispositivo è connesso alla rete.
- A questo punto, il device alloca le communication resources (slotframes e links)
- Dopo tutte queste operazioni, inizia l'advertising quando è il suo turno.



Lo riceve perché è sulla stessa frequenza.

Durante tutto il tempo di ascolto, il Joining node mantiene il radio ON e consuma molta energia.

Potremmo quindi inviare gli EB con freq. maggiore ma vorrebbe dire occupare bandwidth e consumo di energia per i nodi già all'interno della rete.

Quando trasmettere gli EBs

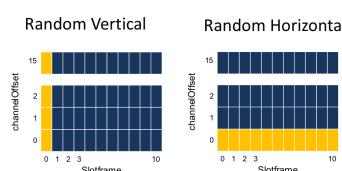
le advertising policy non sono definite dallo standard, quindi possono essere usate diverse strategie.

l'obiettivo è minimizzare il joining time quindi mandare gli EB frequentemente ma aumenta l'energia consumata dai nodi che sono già all'interno della rete, quindi c'è bisogno di un trade-off.

1) Usare un insieme di celle  $\geq 1$  (o orizzontalmente o verticalmente)

- Orizzontale: Vengono usati diversi channel offset
- Verticale: Vengono usati diversi slotframe

2) Model-based Scheduling Algo

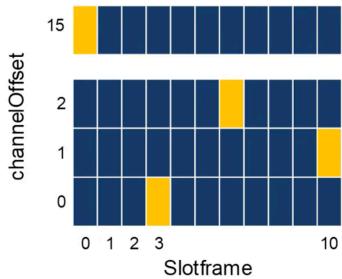


Dato il numero totale di EB, fornisce la loro allocazione in modo da minimizzare il joining time.

Questo è possibile perché ciò che è importante per il joining time è la distanza tra due

slot consecutivi allocati per la tua trasmissione degli EB.

Questa distanza dovrebbe essere più o meno sempre la stessa, cioè gli EB dovrebbero essere distribuiti uniformemente lungo lo slotframe.



## Wireless Hant

Molte idee nell' IEEE 802.15.4e sono state prese da WirelessHart.

Caratteristiche :

- 1) Open-standard wireless networking technology  $\Rightarrow$  Rilasciato dalla HART Communication foundation (2007)
- 2) Usa l' IEEE 802.15.4 nella 2.4 GHz ISM band come PHY
- 3) TDMA-based MAC protocol
  - $\Rightarrow$  Network-wide time synchronization
  - $\Rightarrow$  Pre-scheduled 100-ms time slots } durata,
  - $\Rightarrow$  Frequency hopping
  - $\Rightarrow$  Channel blacklisting } se non conviene usare alcuni channel
  - $\Rightarrow$  AES-128 ciphers and keys
- 4) Mesh networking self-organizing e self-healing

## Components

- **Wireless Field Devices**
  - Devices connected to process or plant equipment.
  - This device could be a device with WirelessHART built in or an existing installed HART-enabled device with a WirelessHART adapter attached to it.
- **Gateways**
  - Enable communication between these devices and host applications connected to a high-speed backbone or other existing plant communications network.
- **Network Manager**
  - Responsible for configuring the network, scheduling communications between devices, managing message routes, and monitoring network health.
  - The Network Manager can be integrated into the gateway or the application host

## Routing

### Graph Routing

- A graph is a collection of paths that connect network nodes.
- Paths in each graph are explicitly created by the network manager and downloaded to each individual network device.
- To send a packet, the source device writes a specific path ID in the network header.
- All network devices on the way to the destination must be pre-configured with graph information that specifies the neighbors to which the packets may be forwarded.

### Source Routing:

- **supplement of** the graph routing aiming at network diagnostics.
- The source device includes in the header an ordered list of devices through which the packet must travel.
- Each device utilizes the next network device address in the list to determine the next hop

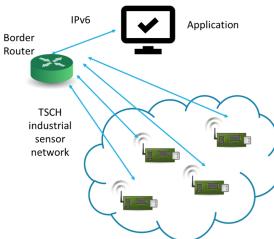
## 6TiSCH

6TiSCH ha come scopo l'abilitare le comunicazioni IP-based sulle reti LPZ per applicazioni industriali.

Esso definisce l'architettura della rete e un insieme di protocolli per permettere l'integrazione di una TSCH wireless network in una infrastruttura IPv6 esistente.

### Architettura

L'architettura di una 6TiSCH network include uno o più border routers per connettere la rete TSCH industriale a un'applicazione esterna, attraverso IPv6.



### Protocol stack

Al classico stack protocollare, in 6TiSCH è stato aggiunto un nuovo layer chiamato 6TOP con lo scopo di integrare i layer più alti con il layer IEEE 802.15.4 TSCH.  
(la rete TSCH)

Nello specifico:

- 1) 6TOP : Definisce gli allocation algorithms e protocolli per la negoziazione delle communication resources
- 2) IEEE 802.15.4E TSCH MAC : Definisce i meccanismi di comunicazione, ma non l'allocation e gestione delle network resources (come visto precedentemente)



### Scheduling (delle celle del timeslot)

6TiSCH usa diverse tipologie di scheduling:

- 1) Static Scheduling : Usato durante l'avvio della rete oppure quando uno scheduling migliore non è disponibile, e preconfigurato.
- 2) Centralized Scheduling : Basato su un'entità centrale (Path Computation Element - PCE)
- 3) Distributed scheduling : Basato sulla negoziazione di nodi vicini (Non centralizzato)

- 4) **Autonomus**: Ogni nodo decide "da solo", senza negoziazione con i vicini
- 5) **Hop-by-hop**
- 6) **Hybrid**: Combinazione diversi scheduling

### Static Scheduling

È pre-configurato e imparato dai nodi quando entrano nella rete.

### Centralized Scheduling (Non adatto per reti dinamiche o di grandi dimensioni)

È basato su un'entità centrale (PCE) la quale:

- 1) Collezione le info sullo stato della rete e i requisiti di traffico dei nodi
- 2) Conviene il communication schedule
- 3) Installa lo schedule sulla rete

Può usare due algoritmi diversi per convivere lo scheduling:

#### a) TASA (Traffic Aware Scheduling algo)

- Tree network topology
- Converge-cast communication model
- The coordinator has a single radio interface
- Heterogeneous traffic conditions

#### b) MODESA (Multi-channel Optimized Delay Slot Assignment)

- Tree network topology
- Converge-cast communication model
- The coordinator has multiple radio interfaces
- Homogeneous traffic conditions (heterogeneous conditions in [4])

### TASA (Traffic Aware Scheduling algo)

- 1) Ogni nodo aggiorna regolarmente il manager con:
  - La lista degli altri nodi che può ascoltare
  - La quantità di dati che genera
- 2) Il manager:
  - "Disegna" il connectivity graph con le info ricevute dai nodi

- Assegna gli slot ai vari link del grafo
- Informa ogni nodo dei link nei quali è coinvolto.
- Aggiorna la schedule e i nodi interessati in caso di cambiamenti nel connectivity graph

### **MODESA ( Multi-channel Optimized Delay Slot Assignment )**

- 1) Itera su un insieme di nodi (ordinato per priorità) che hanno dati da trasmettere e hanno l'interfaccia disponibile con il loro parent in quello slot
- 2) Sceglie il nodo con la priorità più alta (cioè il nodo che ha il maggior numero di pacchetti da trasmettere in quel momento).
- 3) Schedula la sua prima trasmissione sul timeslot e nel primo channel offset
- 4) Sceglie poi un altro nodo:
  - a) se è in conflitto con il precedente  $\Rightarrow$  la sua trasmissione avviene su un channel offset differente
  - b) Sennò viene usato lo stesso channel offset
- 5) Continua così fin quando non sono soddisfatte tutte le trasmissioni dei nodi scheduled.

### **Distributed Scheduling**

Il distributed scheduling non si appoggia su un nodo centrale, ma lo scheduling viene calcolato da ogni nodo, basandosi sulle info locali e trasmesse dai nodi vicini. Solitamente lo scheduling non è il migliore possibile, ma viene computato con un consumo minimo di risorse (quindi adatto per constrained nodes).

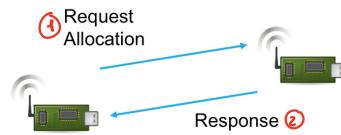
I vari nodi si mettono d'accordo su una schedule comune, tramite la negoziazione tra i nodi vicini la quale viene basata su quanti e come allocare le celle per soddisfare i Quality of Service (QoS) requirements.

Per far funzionare lo scheduling distribuito abbiamo bisogno di **due componenti**:

### 1) 6Top Protocol (6P)

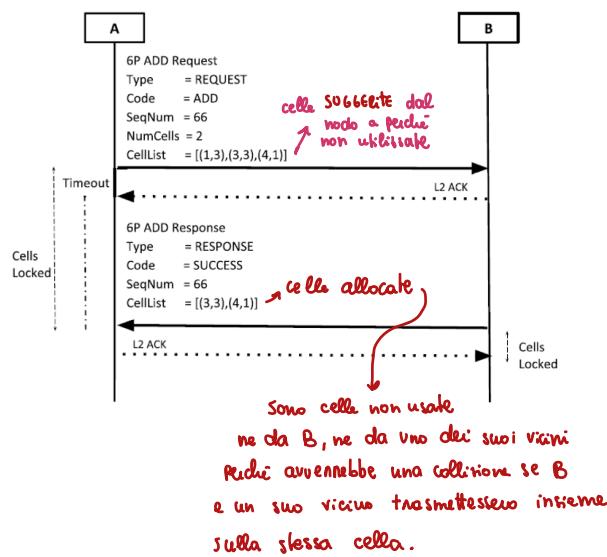
È il protocollo usato per la **negoziazione**.

L'idea si basa sulla **richiesta di allocazione delle celle** da parte del nodo che invia i dati, e l'**accettazione** di allocazione da parte del nodo che li riceve.

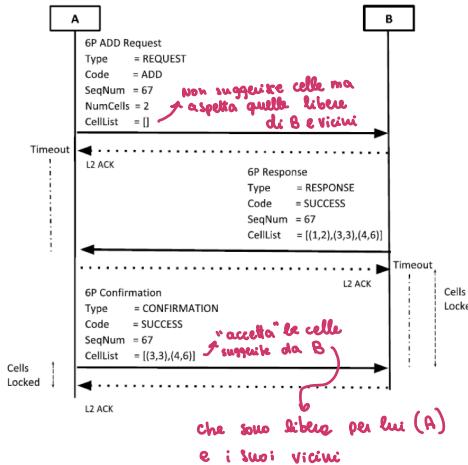


Esempio:

1)



2)



### 2) Scheduling Function

Vengono usate da parte dei nodi per determinare **quante celle sono necessarie** per inviare i pacchetti (Generati e/o inoltrati).

Naturalmente, il numero di celle necessarie è dinamico in base alla bandwidth requirements corrente e le condizioni di rete, quindi le celle possono essere **allocate** ma anche **deallocate**!

1) Minimal Scheduling Function (MSF)  $\rightarrow$  si basa sul numero di celle usate  
 è una scheduling function, definita dallo standard, la quale è hybrida e  
 usa un approccio Autonomus + Negotiated.

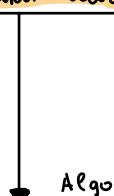
Ci possono essere due tipologie di celle:

a) Autonomous Cells: Allocate senza negoziazione, richiedono una quantità minima di bandwidth. Sono allocate in modo fixed.  
 Alcune delle celle autonome vengono usate per trasmettere, altre per ricevere.

Sono usate solitamente per trasmettere / ricevere control msgs,  
 ad esempio 6P msgs.

b) Negotiated Cells: Allocate con la negoziazione tramite il 6P protocol.

Possono essere dinamicamente aggiunte / rimosse dipendentemente dal traffico e le network conditions.




---

**ALGORITHM 1: MSF Algorithm (NumCellsElapsed, NumCellsUsed)**

---

**Input:**  
 $NumCellsElapsed$  = Number of elapsed negotiated cells  
 $MAX\_NUMCELLS$  = Max number of elapsed negotiated cells  
 $NumCellsUsed$  = Number of negotiated cells used for transmission  
 $LIM\_NUMCELLSUSED\_HIGH$  = Threshold to add a new negotiated cell  
 $LIM\_NUMCELLSUSED\_LOW$  = Threshold to delete an unnecessary negotiated cell

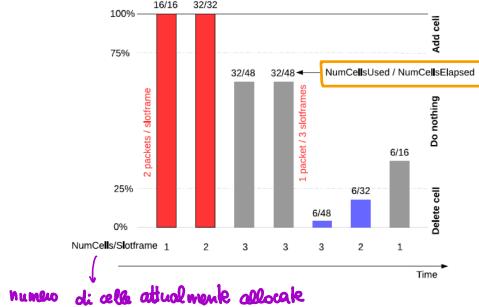
**Output:**  
 ADD/DEL one negotiated cell

---

```

1 if  $NumCellsElapsed > MAX\_NUMCELLS$  then
  if  $NumCellsUsed > LIM\_NUMCELLSUSED\_HIGH$  then
    trigger 6P to ADD one negotiated cell
  2 else if  $NumCellsUsed < LIM\_NUMCELLSUSED\_LOW$  then
    trigger 6P to DEL one negotiated cell
  
```

---



2) On-the-Fly Scheduling Function (OTF)  $\rightarrow$  si basa sulla quantità di traffico da gestire (bandwidth)  
 è una scheduling function, usata precedentemente nello standard, la quale usa un approccio basato solamente sulla negoziazione

---

**ALGORITHM 2: OTF Allocation Algorithm ( $S_c, R_c, T$ )**

---

**Input:**  
 $S_c$  = Number of scheduled cells  
 $R_c$  = Number of required cells  
 $T$  = Hysteresis Quantum

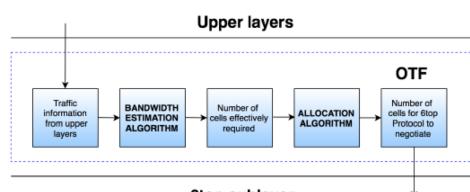
**Output:**  
 $\Delta S$  = Number of cells to add/delete

---

```

1 if  $R_c > S_c$  then  $\Delta S = R_c - S_c + \lceil T/2 \rceil$   $\Rightarrow$  ADD CELLS
2 else if  $R_c < S_c - T$  then  $\Delta S = S_c - R_c - \lceil T/2 \rceil$   $\Rightarrow$  DELETE CELLS
3 else  $\Delta S = 0$   $\Rightarrow$  DO NOTHING
  
```

---



## Distributed Scheduling Performance Evaluation

Per considerare le performance dobbiamo tenere di conto la scheduling function, il GP protocol e l'interazione con RPL.

### RPL impact

- 1) Il dinamismo di RPL può risultare in frequenti cambi di pattern
- 2) Il meccanismo di "link-quality assessment" può portare alla scelta di un parent node caratterizzato da un link quality pessima.

Entrambi questi fattori possono portare a un incremento eccessivo della queue size, risultando in grandi ritardi e/o perdità di pacchetti.

### GP impact

GP transaction failures causano inconsistenti nella schedule, la quale deve essere resettata e ri-negotiata.

Questo porta a una queue size maggiore risultando in grandi ritardi e/o perdita di pacchetti.

### Congestion

Congestion periods sono inevitabili essendo causati dagli altri IoT protocols (RPL, GP, TCSH)

- 1) DTF non è capace di reagire alla congestione
- 2) MSF reagisce meglio, ma il tempo richiesto per rilevare e recuperare dalla congestione può essere nell'ordine di minuti o decine di minuti.

## Enchanted DTF ( $\epsilon$ -DTF)

$\epsilon$ -DTF, a differenza di DTF, considera la link quality nel calcolo del numero di celle necessarie (invece di avere la bandith e il parametru  $T$  per l'overprovisioning). Inoltre include un meccanismo per reagire in tempo alla aumentare della queue size sopra una certa soglia.

**ALGORITHM 3: Enhanced-OTF Allocation Algorithm ( $S_c, R_c, T, B, Q, U, ETX, \beta, \alpha$ )**

**Input:**

- $S_c$  = Number of scheduled cells
- $R_c$  = Number of required cells
- $T$  = Hysteresis Quantum
- $B$  = Congestion Bonus (CB)
- $Q$  = Average Queue Occupancy
- $U$  = Average Cell Utilization
- $ETX$  = Estimated Link Transmissions

**Output:**

- $\Delta S$  = Number of cells to add/delete

```

1  $R'_c = R_c * ETX$ 
2 if  $Q > \beta$  then  $\Delta S = B \Rightarrow ADD\ CELLS$ 
3 if  $R'_c > S_c$  then  $\Delta S += R'_c - S_c + \lfloor T/2 \rfloor \Rightarrow ADD\ CELLS$ 
4 else if  $R'_c < S_c - T$  then  $\Rightarrow DELETE\ CELLS$ 
5 if  $U > \alpha$  then  $\Delta S = B \Rightarrow REMOVE\ CB$ 
6 else  $\Delta S = S_c - R'_c - \lfloor T/2 \rfloor$ 
7 else  $\Delta S = 0 \Rightarrow DO\ NOTHING$ 

```

## Limits del Distributed Scheduling

Fa affidamento sul 6P protocollo:

- 1) 6P transaction hanno un avg delay nell'ordine di secondi
- 2) Una percentuale significativa delle 6P transaction fallono
- 3) 6P negotiations aumentano il network overhead
- 4) 6P transactions sono deboli a attacchi

## Autonomous Scheduling

le celle sono allocate in modo autonomo dai nodi, usando una hash function sull'indirizzo dei nodi. Non usa negoziazione e 6P protocol.

I vantaggi di questo approccio sono:

- Non c'è l'overhead dovuto alla negoziazione
- Non c'è delay aggiuntivo introdotto dalle 6P transactions
- Non ci sono 6P attacks

## Come allocare le celle?

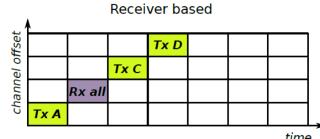
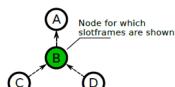
### 1) Node based approach

a) Receiver based Approach: Alloca una cella per ricevere i pacchetti dai vicini e una

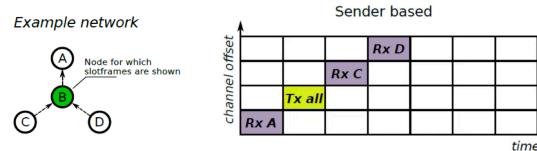
↓  
prone to collisions!

celle per ogni nodo vicino per trasmettere

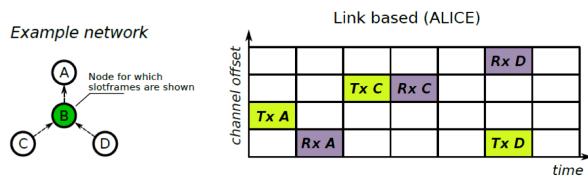
Example network



b) **Sender based approach**: Alloca una cella per trasmettere i pacchetti dai vicini e una cella per ogni nodo vicino per ricevere



2) **Link based Approach**: Alloca due celle per ogni nodo, una per ricevere e una per trasmettere



## Performance

- 1) Receiver-based performance male dovuto alla singola cella per ricevere che porta a molte collisioni
- 2) Link based (Alice) performance meglio del sender-based dovuto al fatto che più slot frame sono allocati per ogni nodo