RIASSUNTO MIDTERM 2

LOW POWER MEDIUM ACCESS CONTROL

S-MAC (Sensor Mac):

in generale, le metriche che andiamo a considerare quando progettiamo meccanismi di low power MAC sono:

1. ***Energy efficiency:*** quando pensiamo ad un possibile MAC protocol dobbiamo considerare il duty cycle dei device e dobbiamo anche minimizzare i casi in cui si verifica ***energy waste***, che sono:
    - *collision* -> quando un pacchetto trasmesso collide e viene corrotto, deve essere scartato e questo porta ad una ritrasmissione che incrementa il consumo di energia (e la latenza)
    - *overhearing* -> un nodo prende in considerazione pacchetti che sono destinati ad altri nodi
    - *control packet overhead*
    - *idle listening* -> listening per riceve possibile traffico che in realtà non viene inviato (*maggiore fonte di consumo di energia*), quindi ci sono lunghi periodi in cui consumo energia per "ascoltare" il canale, in cui i momenti in cui realmente uso l'energia in modo utile sono pochi.

    Un modo per contrastare la cosa potrebbe essere l'utilizzo del duty cycle per ridurre l'ammontare di tempo che un device passa on
1. ***End to end latency***
2. ***Fairness -*** *considerare se sto fornendo buona performance e poco consumo in modo equilibrato tra i device o se la cosa è sbilanciata verso qualcuno di loro*
3. ***Network capacity/capability***

In generale, abbiamo bisogno di sottoporre i device ad un duty cycle, quindi i device devono seguire un ***awake/asleep schedule***.

Nel sensor MAC i nodi condividono una schedule awake/asleep sincronizzata (scambiando informazioni periodiche in modo da rimanere sincronizzati) in modo che ogni nodo trasmettano e ricevano sempre nel momento opportuno, per cui si evitano sprechi di energia.

Note: *la sincronizzazione è importante perchè devono condividere la stessa visione del tempo. Questo approccio comporta della latenza, perchè dobbiamo aspettare che il nodo ricevitore si svegli, ma risparmiamo energia.*

In S-MAC è necessario un periodo iniziale di set up e dei lassi temporali in cui un nodo è in ascolto per ricevere lo schedule dagli altri.

prima che un nodo inizi la propria attività awake/asleep deve scegliere una schedule ed inviarla in broadcast ai suoi vicini (***schedule exchange***).
Allo start up il nodo x "ascolta" per un *random time*:

- se riceve un SYN da un altro nodo y, si sincronizza con il suo schedule (si dice che diventa un ***follower***) e fa *re-broadcast* della schedule dopo un *delay random t_d* (questo random time viene usato per far si che i follower di uno stesso sincronizzatore non collidano o collidano con una bassa probabilità)
- altrimenti il nodo x seleziona un *tempo random T* in cui va in *sleep* per voi *svegliarsi* di nuovo ed inviare questo valore T ai suoi vicini in un SYN packet (quindi x diventa ***synchronizer***)
- se invece un nodo riceve una schedule differente dopo che ha selezionato un'altra, che viene seguita da più di un device, adotta entrambe le schedules, facendo broadcast della nuova.

  Si parla in questo caso dei così detti ***border nodes***, dove due ***synchronization waves*** si incontrano e può accadere in nodi che fanno da gateway per aree differenti (questi nodi consumano più energia).

Ogni nodo mantiene anche una ***schedule table*** che conserva le schedules di tutti i vicini noti.

Quando un nodo invia i pacchetti:

se un nodo X deve trasmettere dati ad un nodo Y attendi, in base al proprio schedule, che il nodo Y sia ON ed invia i pacchetti seguendo il CSMA/CA:

- ascolta il canale per un intervallo randomico
- se non ci sono trasmissioni durante questo intervallo, trasmette RTS/CTS
- se RTS/CTS va a successo, i dati vengono inviati, seguiti da un ACK
- per evitare che i vari device con diversi schedule rimangano ON inutilmente si usa l'informazione contenuta nel vettore NAV, che indica quanto dovrebbe durare la trasmissione dei dati

Note: *per sfruttare al meglio il tempo richiesto per RTS/CTS, si inviano bursts di pacchetti se ce ne sono più in coda*
Per mantenere la sincronizzazione vengono inviati periodicamente dei SYN per fronteggiare anche il problema del clock drift, ma anche per mantenere le schedules dei vari nodi aggiornate.
Per usare questo meccanismo di SYN il tempo in cui si è ON viene diviso in una prima parte dedicata ai pacchetti SYN ed una successiva dedicata al resto.
Note:*mantenere i clock allineati per far si che i device condividano lo stesso schedule ha lo svantaggio di utilizzare effettivamente la banda di trasmissione, solo in piccola percentuale .*

LIMITS:

- ***necessita sincronizzazione****:* anche se i clock drift non sono un grosso problema per la sincronizzazione, aggiunge ***overhead*** di controllo che potrebbe portare ad una vita maggiore
- ***throughput ridotto:*** date che solo la parte attiva dei frame viene usata per la comunicazione, il throughput è limitato (utleriormente anche perchè si comunica solo quando i dispositivi sono entrambi ON, quindi si usa solo parte della capacità)
- ***latenza aumenta:*** quando un nodo genera un pacchetto, deve aspettare quando il next hop relay va on, prima che il pacchetto può essere forwardato

VARIANTI:

- una possibile alternativa è il ***T-MAC:***
  - ci sono casi in cui avere un ON-time fissato, come si usa nell'S - MAC, non conviene perchè possono verificarsi situazioni in cui non si hanno così tanti dati da trasmettere e quindi si passa del tempo ON inutilmente (o comunque, poche cose da trasmettere) e situazioni in cui si hanno troppe cose da trasmettere ed il tempo prestabilito non è sufficiente per trasmettere tutti i dati.
  - Idealmente si vorrebbe avere un tempo ON che può variare dinamicamente in base a quanti dati devono essere inviati.

  T-MAC si basa sulle stesse caratteristiche di S-MAC ovvero: i nodi sincronizzano i propri schedule sulla base di un SMAC clustering approach, per poi usare CSMA/CA nel tempo attive ed una trasmissione di pacchetti back to back per i burst di dati.
  Tuttavia, a differenza di S-MAC - *se non si ricevono trasmissioni dai vicini per un tempo TA, il tempo attivo viene abortito ed i nodi vanno in sleep.*
  Il timer TA viene resettato se:
  1. *vengono ricevuti dati*
  2. *una comunicazione viene sentita sul canale*
  3. *vengono inviati dei dati*
  4. *messaggi RTS/CTS vengono scambiati dai vicini*

  Dunque, il concetto di TA rende il tempo attivo dinamico ed adattivo in base alla necessità: si allunga se ci sono comunicazioni e decresce se lo stato è IDLE.
  ---------|| Come determinare TA ||---------
  I fattori che influiscono sulla decisioni sono:
  1. **C** - intervallo iniziale chiamato *contend* di una lungezza fissa, in cui vengono inviate informazioni come il messaggio di SYN

2. **R** - tempo necessario al nodo per trasmettere un messaggio RTS (tempo che trascorre tra l'invio del RTS ed il suo arrivo)
3. **T** - tempo che intercorre tra la ricezione del RTS e l'invio del CTS (tempo richiesto dal nodo per passare dallo stato di ricezione a quello di trasmissione)

Quindi, il tempo TA deve essere maggiore stretto alla somma dei tre valori precedenti.
Altre differenze dal S-MAC sono le seguenti:

- quando un nodo manda un RTS ma non riceve un CTS, potrebbero essersi verificati uno dei seguenti eventi:
  1. *RTS non ricevuto a causa di una collisione*
  2. *il nodo ricevente non può rispondere a causa di un overhead RTS/CTS*
  3. *il nodo ricevente è dormiente*

  e nel caso di 1 e 2, ridurre il tempo attivo sarebbe un errore in quanto: "un nodo dovrebbe riprovare mandando l'RTS almeno due volte prima di andare in sleep"
- l'early sleep potrebbe degradare il throughput (mentre decrementiamo tuttavia l'idle ed il consumo di energia)

EARLY SLEEP PROBLEM:

se avessimo 4 nodi A - B - C - D con A che invia RTS a B e B lo propaga a C, C va in sleep per NAV time, mentre A invia i dati.
Finito il tempo di NAV, B invia un pacchetto di ACK sia a C che ad A.
Infine, se C ha dei dati da inviare, può farlo inviando RTS a B e D.
Il problema è che, in base a TA, D è andato in sleep molto prima (perchè l'RTS di A non si è propagato fino a D) dell'RTS da C quindi non può rispondere a C.
Possibili soluzioni:

1. quando C riceve un pacchetto di RTS da B e sa che dovrà inviare dei dati successivamente, deve inviare un pacchetto a D per far si che resti attivo.

   Il pacchetto che viene inviato è un **FRTS**, ovvero un finto RTS che indica ai nodi di rimanere attivi. Tuttavia, il pacchetto FRTS si propagherà sia a D che a B, quindi c'è un rischio di ***collisione*** con i pacchetti che A potrebbe inviare dopo aver ricevuto il messaggio di CTS.
   Per questo motivo, A non inizia subito a trasmettere i veri dati ma invia un pacchetto **DS (Data Sent)**, ovvero un pacchetto dummy usato solo per mantenere i nodi attivi e non far scattare il timeout.

2. supponiamo che B invii un RTS ad A e C. Potrebbe succedere che C abbia il buffer pieno quindi non potrebbe ricevere altri dati. Per questo motivo, C può rispondere inviando un nuovo RTS a B e D (al posto del CTS), così da acquisire il canale per inviare i dati (questo procedimento si chiama ***Full buffer priority***).

B-MAC (Berkley Mac):

gli obiettivi che si cercano di raggiungere sono:

- *Medium Access Control*
- *operazioni low power*
- *effictive collision avoidance*
- *implementazione semplice con small code e small RAM size*
- *essere riconfigurabile dai protocolli di rete*
- *scalabile su un grande numero di nodi*

per avere un meccanismo di collision avoidance efficace, un protocollo MAC deve essere capace di determinare se un canale è libero - mediante ***Clear Channel Assessment (CCA)***:

- sostanzialmente il BMAC propone un modo per stimare il rumore del canale e determinare se un canale è libero sulla base di quello (prendendo alcuni campioni e controllando se sono al di sotto del livello di rumore medio).

per stimare il rumore utilizza una RSSI samples ed utilizza la mediana dei samples per calcolare una media mobile esponenzialmente pesata con un fattore di decay pari ad alfa.
Stimato il rumore, utilizza dei CCA samples per dire:

- se su 5 esempi non ci sono outlier (ovvero sample sotto il livello di rumore) allora il *canale è occupato*, altrimenti *libero*

L'asincronia del protocollo è dovuta al fatto che, a differenza del S-MAC, qui non vengono inviati pacchetti di sincronizzazione per allineare i duty cycle, ma viene utilizzato il così detto ***Low Power Listening (LPL):***

- *(**RECEIVER SIDE**) ogni volta che un nodo si sveglia, accende la radio e controlla se c'è qualche attività - se c'è il nodo incrementa l'energia (power up) e resta sveglio per il tempo richiesto a ricevere i pacchetti , dopodichè torna a dormire (dopo che li ha ricevuti o dopo un timeout)*
- *(**SENDER SIDE**) il mittente invia un preambolo e poi i dati - per maggiore affidabilità il preambolo viene mandato di una lunghezza che fa match con il check interval in cui il ricevitore è asleep0*

X-MAC:

per quanto l'idea del long preamble fosse buona, ci sono alcuni problemi tra cui il fatto che :

- *introduce inevitabilmente **latenza**, in quanto devo aspettare l'invio di tutto il preambolo prima di poter inviare i miei dati*
- *aumenta il **consumo di energia** perchè potrebbe accadere che il ricevitore si è svegliato ad un certo punto del preambolo, quindi potrebbe già ricevere, ma comunque bisogna attendere la fine del preambolo*
- *potrebbe succedere che, in reti dense, molti nodi sprecano tempo aspettando la fine del preambolo, per poi andare in ascolto dei dati e scoprire e che non sono loro la destinazione di quei dati*

Quello che propone X-MAC è di aggiungere l'IP della destinazione in preamboli più brevi e consecutivi (ma non continui, chiamati ***strobed preamble***, le pause tra i pacchetti fanno sì che la destinazione possa inviare un ***fast ACK***quando è up) in modo che un nodo che riceve il preambolo con il suo IP vede che invia un ACK al mittente del preambolo per indicare che è immediatamente pronto a ricevere dati. Questo porta performance molto migliori:

- riduco drasticamente lo spreco di energia
- in termi di reliability si evitano sprechi di risorse

2G + (BEYOND GSM)

nata quando si iniziò di parlare della necessità di trasmettere dati oltre che voce: necessità di supportare packet switching, high data rate e maggiore qualità della codifica dei dati, oltre alla necessità di estendere le funzionalità degli operatori mobili per permettere di fornire servizi più complessi (implica anche modificare le componenti fisiche già esistenti, tenendo conto del fatto che la soluzione migliore è fare cambiamenti di ciò che già esiste piuttosto che rifare tutto da capo).

Innovazioni:

- in termini di ***codec*** (codifica/decodifica) è stata introdotto un meccanismo full rate basato su ***ACELP (Algebric Code Excited Linear Prediction)*** che prevede una codifica a 13 kbit/s
- inoltre è stato introdotto un ***Adaptive codec***: codec di nuova generazione che si adatta (half rate o full rate) e cambia il rate in base alle condizione di propagazione del canale
- ***Tandem free operation:*** limita l'uso del transcoding (che potrebbe introdurre una qualche degradazione del segnale) permettendo multiplexing del flusso (es. usando full rate o enhanced full rate codecs) sui canali PCM in caso di comunicazione MS-MS (quindi nella stessa cellular network viene usata la codificazione standard senza fare transcoding)
- ***Enhanced data rate*** viene ottenuto apportando migliorie al livello fisico, allocando anche molteplici slot per MS, incrementando così il data rate per user

- ***Location services:*** trovare una localizzazione grezza tramite triangolarizzazione, usando il BTS come anchor
- ***Number portability:*** possibilità degli utenti di mantenere lo stesso numero telefonico, pur cambiando operatore (utilizzando un DB da interrogare quando bisogna risalire all'operatore di un certo numero)
- ***Cordless telephony system:*** possibilità di usare un MS come cordless, connettendosi ad un ***Home Base Station (HBS)*** e da lì alla rete del suo operatore
- ***SIM application toolkit:*** si è visto un cambiamento di paradigma in cui, il **ME** non è più il master e il **SIM** lo slave, ma il **SIM** ora può iniziare comunicazione con il **ME** ed i dowloand via radio link al SIM sono possibili. Inoltre, il SIM può chiedere al ME di:
  - fare set up di una call ad un numero nel SIM
  - inviare i numeri di telefono che il ME sta componendo al SIM per analizzare e modificare la chiamata
  - generare toni
  - passare al SIM informazioni
  - eseguire comandi inviati dal SIM
  - lanciare il microbrowser nel ME per ridirigerlo verso uno specifico indirizzo WEB
- costruire applicazioni al livello di SIM (scaricandole o implementandole)

EDGE

nel 2G+ non si riesce veramente a migliorare la banda disponibile ma resta comunque la necessità di aumentare il data rate ed in **EDGE** viene risolta la cosa al livello fisico, dove si trova la bottle neck delle trasmissioni radio soggette ad interferenze. Senza modificare l'architettura di sistema si è riesciti ad modificare la fase di modulazione (tre volte bit per symbol in più) e raggiungere così un data rate di gran lunga superiore.
Un altro concetto introdotto e molto usato nelle reti wireless è l'**Adaptive modulation** che fa un trade-off tra **reliability** e **data rate** perchè, in un ambiente reale, si hanno condizioni variabili ed il **SNR** cambia nel corso del tempo.
Viene implementata dunque una selezione dinamica di codifica e modulazione, basata sul SNR del canale radio ed il data rate viene *ridotto* o *incrementato* in base alla qualità del canale.
La selezione della codifica comprende anche **FEC** adattivo, in base alla qualità del segnale (cambiare dinamicamente la percentuale di bit di un burst che vengono dedicati al FEC).

GPRS

viene considerato come il core dei sistemi 2G+ ed in termini di cambiamenti nell'infrastruttura è stato realizzato per supportare non solo la voce, ma anche la trasmissione dati con Internet combinando circuit switched e packet switched communication ed utilizzando una IP backbone per il packet switching.
Dal punto di vista della trasmissione vocale, vengono usati gli stessi elementi mentre ne sono stati introdotti di nuovi per la trasmissione dati ed in particolare, sono stati introdotti i seguenti nuovi nodi di support al packet switching:

1. ***SGSN (Serving GPRS Support Node):*** *un router IP a tutti gli effetti che routa i pacchetti da/verso un set di MS sotto la sua area e le aree sono divise in* ***finer grain router areas***
2. ***GGSN (Gateway " "):*** *la versione gateway del router che, appunto, svolge funzioni di gateway per la parte di trasmissione dati piuttosto che per quella di trasmissione vocale ed interconnette la cellular network con le external packet data networks.*
3. ***PCU (Packet Control Unit):*** *al livello di BSSS, nuova entità funzionale aggiunta per gestire la trasmissione dei dati sopra il canale radio*

SGSN e GGSN sono interconnessi attraverso una IP backbone.

> ***Note:*** *quello che succede, sostanzialmente, è che il traffico mandato dal BTS attraversa una componente in cui si divide in base alla sua natura voce/dati per cui seguirà poi un path diverso in base al tipo di traffico (la voce usa le componenti già viste nel GSM, i dati usano anche le componenti SGSN e GGSN che collega le IP Network e la Backbone network alla cellular network).*

Quando una MS vuole trasmettere dati deve aver un IP address associato e deve fornire informazioni (oltre che riceverle), richiedendo l'allocazione di risorse adeguate (in base alla QoS adeguata al flusso di informazioni che deve essere trasmesso) per la trasmissione dei dati.
Quando voglio stabilire uno scambio di dati devo fare una ***PDP context activation*** con il GGSN attraverso cui ricevo un IP address dinamico (tramite DHCP) ed attraverso cui vengono scambiate le informazioni sulla QoS richiesta per il flusso di dati ed il servizio richiesto, così che si può decidere se finalizzare o meno l'attivazione in base alla presenza/assenza delle risorse adeguate al QoS richiesto.
Se l'attivazione viene finalizzata, posso iniziare a scambiare dati passando sempre per il gateway GGSN.

---------|| Dettagli PDP Context Activation ||----------

Il contesto PDP contiene:

- PDP type (es. IPv4)
- la classe QoS richiesta
- l'indirizzo del GGSN che lavora come access point all'external net

e questo contesto viene conservato in tutti i SGSN e GGSN (molteplici contesti possono essere creati per un utente, per esempio quando dobbiamo trasmettere in parallelo più traffico).

PCU - Packet Control Unit:

uno degli elementi aggiuntivi nella rete che si occupa di allocazione dinamica delle risorse tra GSM CS e GPRS ed interconnette MS e SGSN per lo scambio di pacchetti. Questo elemento serve perchè abbiamo diverse tipologie di traffico che passano sugli stessi canali fisici, richiedendo potenzialmente data rate diversi e qualità diverse, che richiedono allocazione di risorse differenti e si occupa inoltre di:

- *segmentazione e riassemblamento dei dati*
- *meccanismi di error detection e management, molto importanti in situazioni in cui è richiesta una trasmissione reliable, usando ad esempio ACK/NAK, ritrasmissioni ecc.*
- *scheduling del canale fisico*
- *access request management e resource allocation: la prima si rierisce al fatto che la rete deve essere capace di capire se dispone delle risorse richieste dall'utente in base alle informazioni contenute nel PDP, per poter dire anche "non sono in grado di gestire il traffico che vuoi inviare". La seconda si riferisce alla capacità di adattarsi della rete, in base alle necessità anche dinamiche del traffico che sta gestenti.*
- *channel management (tutti meccanismi di gestione del canale e del suo utilizzo come power control, congestion management, broadcasting ecc)*

SGNS - Serving GPRS Support Node:

altro elemento introdotto nel core della network che si occupa di:

- al posto del MSC/VLR del GSM - gestione dell'autenticazione e controllo del fatto che l'utente può effettivamente accedere al servizio, coordinamento della cifratura
- mobility management
- insieme al BSS radio resource management riserva risorse radio necessarie al supporto del QoS richiesto
- implementa routing da/verso le MS
- effettua incapsulamento e tunneling dei pacchetti
- raccoglie anche informazioni che sono utili per la fatturazione

GGNS - Gateway GPRS Support Node:

- stesse funzionalità di routing, in/decapsulamento, analisi e filtraggio dei messaggi in arrivo e raccoglie info. per la fatturazione

- oltre a conservare, si occupa anche di stabilire il contesto PDP e conserva nel suo location register l'indirizzo del SGSN che sta servendo le diverse MS, MS user profiles ed i contesti PDP attivi o in standby
- funge da interfaccia tra cellular network ed external packet network

Physical e logical channels:

le informazioni vengono scambiate usando canali logici che vengono mappati sulla stessa struttura fisica usata in GS (che usa quindi FDMA/TDMA).
Il canale logico usato è il ***PDCH - Packet Data Channel*** che viene allocato solo per il tempo necessario alla trasmissione dei dati, per poi essere rilasciato tenendo conto del fatto che: più flussi di informazioni (fino a 8) possono essere muliplexati su uno stesso PDCH e si possono allocare anche più slots per MS .
La trasmissione dei dati qui è per ***blocchi radio*** composti di 456 bits trasmessi su 4 normal burst, che vengono quindi inviati su 4 slots in frame consecutivi.
Visto che, solitamente, un utente a una richiesta variabile in termini di scambio di traffico, una volta allocato il canale logico (ed anche un altro canale logico per lo scambio di informazioni di controllo - ***Packet Associated Control Channel***), diversi subscribers possono condividere lo stesso canale radio, quindi diversi PDCH vengono multiplexati su una stessa risorsa fisica.
Quindi si ha un radio block allocato per un *gruppo di utenti*, che si danno il cambio in base ad un ***TFI (identificativo numerico che indica chi è l'owner corrente di quel blocco)***e l'owner corrente viene deciso in base alla necessità attuale dei vari subscriber, comunicata usando il canale di controllo.
Quindi, la **resource allocation** avviene con:

- la rete alloca un ***Temporary Flow Block*** che ha associata una identity ***TFI***
- una MS deve richiedere un TBF quando vuole comunicare
- una volta che il PDU ha trasmesso i dati, il TBF viene rilasciato

notare che qui, non avendo un ambiente distribuito ma un punto centrale di controllo, si può fare una sorta di scheduling dei flussi, provando ad ottimizzare l'uso di risorse facendo multiplexing degli utenti.
Al livello di ***Pacchetti di Controllo*** abbiamo:

- una variante per la trasmissione dati degli stessi visti in GSM (quindi **Packet Paging channel, Packet Random Access, Packet Access Grant, Packet Broadcast Control Channel**)
- ***PNCH - Packet Notification Channel:*** *canale di downlink che supporta i servizi di multicasting*
- ***Dedicated Control Channels:***

    - ***PACCH - Packet Associated Control Channel:*** *canale bidirezionale su cui le info. di controllo associate ad uno o più canali di traffico vengono trasmessi*
    - ***Packet Timing Advance Control Channel***

Un altro cambiamento riguarda il trade-off tra il livello di conoscenza della locazione dell'utente e la quantità di pacchetti di controllo scambiati a riguardo: mentre nel GSM il livello di granularità era la Location Area, qui viene introdotto un nuovo concetto, ovvero la ***Routing Area.***

Quindi, se un nodo è **IDLE** la locazione è nota al livello di Location Area come nel GSM (la locazione non è nota al GPRS) altrimenti, se inizia la ***PDP Context Activation***ovvero inizia con ***GPRS attach/detach*** il nodo passa a **READY** state e la posizione viene mantenuta al livello di ***Cell:***

- MS invia PDP request a SGSN
- dopo procedure di sicurezza, SGSN invia una Create PDP Context request a GGSN
- GGSN risponde con una Create PDP Context response
- SGSN attiva il contesto inviando info (come IP addr.) alla MS

Se il ***ready time***è scaduto, perchè non vengono inviati dati, si passa allo stato di **STANDBY** in cui si mantengono informazioni ad un livello tra Location Area e Cell, che è la Routing Area.

Allo scadere di uno ***standby timer***, la MS torna in **IDLE**.

3G

l'obiettivo era di superare i limiti di datarate e banda dei sistemi 2G e 2G+, arrivando almeno all'ordine dei Mps per supportare alcuni servizi e QoS (alcuni nuovi come video e streaming) integrando comunicazioni mobili e satellitari (in modo da aumentare la copertura).

Cambiamenti necessari per una richiesta di QoS molto eterogenea e diversificata in varie classi come:

- ***classi di conversazione***
- ***classi di streaming:*** *richiede di preservare relazione temporale tra informazioni dello stream*
- ***classi interattive:*** *richiede di preservare l'integrità dei dati*
- ***background***

Features:

nel 3G è stata aumenta la banda (cambiamenti nel physical layer) in modo da raggiungere un datarate maggiore ed è cambiato anche l'ammontare di banda fornito per utente (circa 5MHz di canali per utente).

Vengono inoltre usati **TDD e FDD** per dividere le risorse tra uplink e downlink e vengono supportati servizi asimmettrici, ciò significa che il datarate tra downlink ed uplink non è più lo stesso.
Infine, si ha una organizzazione gerarchica che prevede: ***Macrocells -> Microcells -> Picocells***.

UTRAN:

UTRAN è la parte di wireless access della rete 3G e svolge le seguenti funzioni:

- controlla la capacità delle celle e le interferenze, in modo da fornire un utilizzo ottimale delle risorse di interfaccia wireless
- include algoritmi di power control, handover, packet scheduling, call admission control and load control
- cifratura del canale radio
- controllo dei casi di congestione per gestire situazioni di overload della rete
- broadcasting delle informazioni di sistema
- diversità macro e micro

quindi, tra le funzioni ***network based*** vediamo:

1. *packet scheduling* - decide quando la trasmissione di un pacchetto è inizializzata e quale bit rate deve essere usato
2. *load control* - assicura la stabilità del sistema e che la rete non entri in uno stato di overload
3. *admission control* - usato per evitare overload della rete, decide se una chiamata può/non può generare traffico nella rete

MACRODIVERSITY:

nel 3G una stessa MS può essere collegata allo stesso tempo a differenti BS (in accordo alle specifiche 3G) così che uno stesso stream di dati venga inviato su canali fisici differenti.
Ciò significa che, in ***uplink*** una MS invia i dati a differenti BS che riassemblano e ricostruiscono lo stream di dati per farlo arrivare al resto della rete, mentre in ***downlink*** una MS riceve dati da differenti celle su ***spread codes*** differenti portando ad una maggiore performance e reliability.
Tutto ciò riguarda una cosa chiamata **MIMO (Multiple IN/Multiple OUT)**

CDMA:

al livello di Medium Access Control viene utilizza il Code Division Multiple Access per cui:

- *ad ogni utente viene assegnato un identificativo univoco chiamato **chipping code** che, data la sua ortogonalità, permette a utenti differenti di inviare dati contemporaneamente su uno stesso canale fisico consentendo comunque, anche in caso di interferenze, di leggere chiaramente i dati da loro inviati, distinguendoli gli uni dagli altri.*

  Quindi, i dati vengono codifica utilizzando il chipping code, vengono inviati e potrebbero subire interferenze sommandosi ad altri dati, ma a destinazione viene fatta un'operazione inversa a quella usata per la cifratura permettendo così una lettura chiara dei dati originali.

L'utilizzo del CDMA aggiunge un certo grado di complessità dovuta alla cifratura/decifrazione, alla necessaria sincronizzazione e ad altri fattori, ma è ciò che ha permesso di aumentare il datarate (dovendo rispettare comunque un limite massimo di utenti per canale fisico, non possiamo aggiungerli all'infinito).

4G

nonostante l'aumento del datarate, c'è stata grande necessità, dovuto all'incremento dell'utilizzo di internet e di dispositivi mobili, di aumentare il datarate ancora di più. Quindi, le richieste emergenti furono in questo caso:

- sostanzioso incremento del data rate, soffermandosi questa volta anche sull'aumento del data rate anche per gli utenti che si trovano più distanti dalla BS
- aumentare ancora di più la flessibilità nell'utilizzo dello spettro di frequenza, riguardo la dinamicità dell'utilizzo delle risorse
- delay ridotti
- low energy consumption
- architettura delle reti semplificata

tenendo conto del fatto che, il data rate dei cellular systems è dato da: ***bandwidth * spectral efficiency*** e visto che l'allocazione di più banda è un processo complicato ed anche costoso, è sulla parte dell'efficienza dello spettro che ci si sofferma e su cui si può lavorare, migliorando le tecnologie utilizzate.
Per questa tecnologia, ci sono state richieste di datarate differenti tra uplink e downlink e maggiore spectral efficiency anche per comunicazioni broadcast.

LTE

i requisiti per LTE sono:

1. capacità di operare in un ampio range di requency band e dimensione dell'allocazione dello spettro
2. tempo di connessione veloce
3. incremento del peak rate per una performance più uniforme
4. suporto della mobilità da picocells fino a distanze di chilometri

5. capacità di interoperare in modo flessibile con reti di altre tecnologie, anche più vecchie, ma che permettono di coprire area dove la rete in in cui si trova l'utente non arriva, così da fornire continuità del servizio anche in fase di migrazione da una rete all'altra
6. complessità dei terminali inferiore e consumo di energia inferiore
7. efficienza anche in termini di costi dei servizi, in quanto abbiamo più spettro di frequenza (maggiore banda) da allocare per ogni utente, quindi ha senso che costi di meno farlo

Tecnologie per LTE:

- ***multicarrier technology -*** *viene utilizzato OFDMA in downlink e SC-FDMA in uplink, così da avere maggiore flessibilità, adattabilità e robustezza utilizza ricevitori con minore complessità*
- ***multiple antenna technology -*** *un terminale unico può avere moltplici antenne per comunicare*
- ***packet switching -*** *si ha un sistema di pacchetti evoluto per portare pacchetti IP dal gateway della packet data network fino alle MS, utilizzando il concetto di* ***bearer*** *= flusso di pacchetti IP con un certo QoS, che passa tra gateway e UE. Una MS può richiedere di trasmettere con differenti bearers, se necessitò di requisiti di traffico differenti.*

Architettura:

in questa generazione abbiamo degli elementi di rete ben distinti:

1. user equipment (UE) collegato a
2. E-UTRAN (che sarebbe la access network) collegata a
3. EPC (Packet Core network) interconnessa a
4. reti esterne

Nella ***Access Network*** abbiamo un singolo elemento (eNB) disposto in diverse istanze collegate tra di loro in una topologia mesh, utilizzato per svolgere le funzioni tipiche di una BS e per comunicare con la EPC.

L'architettura degli UE e delle eNB è la seguente:

- ***PDCP -*** *si occupa della header compression del pacchetto IP e produce in output un PDCP-PDU*
- ***RLC protocol -*** *responsabile per la segmentazione (concatenazione in uplink) del PDCP-PDUs per la trasmissione radio. Performa inoltre correzione degli errori con il metodo Automatic Repeat Request (ARQ) protocollo usato per ritrasmettere i dati che non sono stati ricevuti correttamente*
- ***Medium Access Control -*** *il MAC layer è responsabile per lo scheduing dei dati in accordo alle priorità ed al multiplexing dei dati del Layer 1, in oltre fornisce correzione degli errori con Hybrid - ARQ*

- ***PHY -*** *esegue coding, modulation e antenna/resource mapping*

QoS e bearers:
i nodi eNodeB nella access network assicurano che il QoS per un bearer sulla radio interface venga rispettato.
Per conoscere il QoS richiesto, ogni bearer ha un QoS class identifier associato e un Allocation e Retention Priority - il livello di priorità viene usato per determinare lo scheduling dell'allocazione delle risorse, quale politica di queue management usare e con quale priorità.

Network architecture:
nella core network, per semplificare l'architettura e la gestione vi è una separazione tra gli elementi del control plane e quelli dello user plane:

1. ***Serving GW -*** *si occupa di trasportare il traffico dati IP tra UE e la rete esterna*
2. ***PDN GW -*** *si occupa di allocazione dell'IP addr per lo UE + enforcement del QoS in downlink + interworking con tecnologie non appartenenti alle tecnologie 3GPP*
3. ***MME -*** *si occupa di gestione della mobilità facendo connection setup (incluso paging) e gestione della security*
4. ***HSS -*** *gestisce i dati degli utenti, i PDNs a cui questi si possono connettere, l'identità del MME a cui sono collegati ed è anche il centro di autenticazione (HLR + AuC)*

Al livello di stack protocollare dobbiamo effettuare misurazioni del canale per poter gestire dinamicamente le risorse, in modo da poter garantire il livello di QoS (in termini di Latenza ed altre metriche) richiesto dai vari flussi di traffico.

OFDMA:
dividere le risorse disponibili in subcarriers di 15 MHz ciascuno, capace di poter trasmettere in parallelo su più subcarriers differenti, tuttavia l'unità minima allocabile per un utente, dato il numero di 20 subcarriers per un totale di 180kHz, per un tempo di 0.5ms nel time domain.

Quindi: ad ogni utente viene assegnato un certo numero di resource blocks nella griglia tempo-frequenza per cui, più blocchi riceve un utente e maggiore è la modulazione usata nelle risorse, maggiore sarò il bit-rate - tutto ciò ha portato miglioramenti anche al livello di trasmitter, perchè appunto si possono comunicare in parallelo su più carriers per uno stesso utente e supporta l'allocazione dinamica delle risorse avanzata.

Resource allocation:
ogni eNodeB, ad intervalli di 1 ms, prende una decisione di scheduling ed invia le informazioni di scheduling al set selezionato di terminali.
L'allocazione così fatta si basa su resource blocks (RBs), ognuno consistente di 12 subcarriers che occupano 180kHz.

Phy layer adaptation:
invece di are un singolo schema di modulazione o un singolo schema di correzione dell'errore, come per l'allocazione dinamica delle risorse basata su feedback e sul QoS, qui abbiamo un adattamento dinamico della modulazione e del coding, per flusso, in base alla condizioni del canale.
Abbiamo un Channel Quality Indicator che viene fornito come feedback dagli UEs che vengono usati per stimare le differenti condizioni del canale.

Femto cell concept:
portare nelle abitazione dei router che svolgono funzioni analoghe a quelle svolte dalle BS, in modo da gestire anche in casa il concetto di QoS (impossibile nelle wireless network) avendo anche un le varie features della tecnologia LTE.

5G

i fattori che hanno portato alla necessità del 5G sono sostanzialmente: maggiore connettività ad alta velocità, potenze di calcolo economiche, cambiamento nel tipo di traffico che divenne lentamente di tipo Cloud e più distribuito ed anche lo spostamento dei compenti di intelligence sempre più vicino all'utente finale.
Inoltre, sono sempre più richiesti miglioramenti in termini di achievable coverage, data rates, latency, reliability, energy consumption perchè si iniziano ad usare tipologie di device differenti, quelli IoT.
Un altro concetto importante è quello dell'interworking tra differenti infrastrutture già esistenti, in modo avere architetture multi technology che siano adatte ad ogni tipo di QoS.
Si punta molto sull'andare a frequenze sempre più alte, anche perchè con frequenze più alte si possono ottenere trasmissioni direzionali.
Richiede anche un cambiamento nel modello economico per avere dei costi che si adattano al fatto di fornire sempre più frequenze e quindi di dover abbassare i prezzi, continuando ad esplorare l'evoluzione tecnologica.
KPIs (Key Performance Indicators):

- ***througput:*** *fornisce 1000 volte più throughput disponibile in traffico aggregato e 10 volte più velocità ai singoli end users*
- latency: fino a 1 ms quando richiesto

- energy efficient: 5% del consumo di energia globale è dovuto a ICT (quindi si ha un incremento del 90% nell'efficienza energetica ed una durata delle batterie 10 volte migliore per device low power)

- coverage: copertura ovunque (planes, tranes ecc.)

System level challenges:

- viene implementata la privacy by design

- una grande challenge è costituita dalle diverse classi di QoS che la rete deve gestire, in termini delle varie metriche richieste (throughput, latency, resilience ecc.)

- simplicity challenge - necessità di fornire servizi anche per inter RAT switching

- multi - tenancy challenge: ovvero fornire servizi attraversando infrastrutture di differente proprietà, con differenti reti coesistenti , fornendo un'interazione integrata efficiente tra i sistemi mobili ed il backhaul

- density challenge, portata dall'uso degli IoT device

- diversity challenge - deve essere capace di supportare la crescente diveristà di soluzioni wireless ottimizzate, includendo la diversità nei tipi di traffico da gestire e nel numero di devices connessi

- harnessing challenge: sfruttare ogni possibile capacità di comunicazione, includendo quella device to device, per ottimizzare le comunicazioni in ogni momento

- harvesting challenge: sfruttare l'energy harvesting per poter incrementare il lifetime

- mobility challenge: mobilità discontinua tra reti/tecnologie

- location and context information challenge: challenge riguardante l'accuratezza della localizzazione dei device

- hardening challenge: rendere le i sistemi di comunicazione robusti agli attacchi ed ai disastri naturali

- resource management challenge: fornire access agnostic control, policy and charging mechanisms e protocols , configurazione/reconfigurazione e rilascio di qualsiasi tipo di risorsa
- flexibility challenge: device veramente flessibili e meccanismi di controllo/protocolli per avere reallocazione delle risorse

Phy layer advancements:
utilizzo massivo del MIMO (multiple in, multiple out), beamforming, utilizzo di array di antenne e novel modulations.
Sono state implementate operazione Full Duplex e per avere una densificazione maggiore delle risorse disponibili alla comunicazione si utilizzano reti ibride tra tecnologia VLC e tecnologia Radio.

Inoltre, utenti differente vengono fatti trasmettere su una stessa forma d'onda per cui un receiver prova a decodificare il segnale più forte, trattando gli altri come interferenze.
Il segnale così decodificato viene rigenerato in una forma d'onda e poi sottratto dal segnale composto ricevuto (questo processo viene poi reiterato per ogni segnale desiderato).

NB-IoT:

con il tempo, vedendo che i dispositivi IoT cominciarono a prendere piede, ci fu sempre maggiore interesse verso questi dispositivi che non richiedono così molto differenti dai normali device wireless (connettività, funzioni semplici, low power consumption e così via).
Date queste caratteristiche, ci si rese conto che si potevano riadattare tecnologie già esistenti per i cellular systems, modificandole per rispecchiare le necessità di reti IoT, come ad esempio l'**LTE**:
i dispositivi NB-IoT sono quella tipologia di device che trasmettono una volta ogni tanto e che sono delay tolerant (quindi rilassano alcuni requisiti di LTE) che però sono molti di più, spesso installati in posti con poca copertura e/o senza supporto energetico (quindi rinforzano alcuni requisiti).

Gli obiettivi di NB-IoT sono sostanzialmente:

- *minimizzare l'overhead di segnalazione (soprattutto sulle interfacce radio) perchè, per l'appunto, non serve scambiare molti segnali di controllo, vista la poca attività dei device*
- *utilizzare lenti duty cycles (quindi cicli aggressivi in cui il dispositivo passa poco tempo ON, solo lo strettissimo necessario) , funzioni semplici e processare velocemente i dati inviati in modo da poter stare ON il minor tempo possibile, il tutto per migliorare la vita della batteria*
- *sicurezza appropriata per completare il sistema, includendo la core network*

inoltre solo la cell reselection in idle state viene supportata, c'è una carenza di handover ed il più delle features avanzate di LTE non sono supportate.
Infine, non vi è concetto di QoS, perchè NB-IoT non viene usato per pacchetti delay sensitive.

NB-IoT permette di utilizzare anche le seguenti parti delle spettro di frequenza (occupando una banda di 180 kHz, corrispondente a un resource block di LTE):

- ***Stand alone operations -*** *un possibile scenario in cui si usano le frequenze della tecnologia GSM per NB-IoT (in quanto, con i loro 200kHz di banda allocata, vi sono anche 10 kHz rimanenti)*
- ***Guard band operations -*** *utilizzo dei resource blocks inutilizzati in un LTE guard-band*
- ***In-band operations -*** *utilizzo di resource blocks in un LTE carrier*

Al livello di performance si è osservato che: NB-IoT non è al top delle performance in termini di energy consumption rispetto a reti come LoRaWAN (rete IoT ad ampio spazio) ma comunque è molto buono.
Infatti, confrontato a LoRaWAN dimostra di avere consumi di energia simili (in termini di anni), copertura simile e delivery affidabile anche se qualche volta con alta latenza(dovuta al fatto che, per avere delivery affidabile, si utilizza la ritrasmissione come meccanismo di recovery, però la latenza è comunque abbastanza bassa per questo tipo di device).

WiseMAC

anche se in S-MAC viene scambiata la schedule in modo sincrono, si possono avere problemi di clock drifts ovvero: i dispositivi tendono lentamente a disallinearsi da un punto di vista temporale, andando ad avere una visione differente del tempo in cui è necessario prendere in considerazione un certo intervallo di tempo in cui ci

potrebbe essere questo disallineamento ed in cui mettiamo un guard period (questo problema di clock drift è il motivo per cui si scambiano tanti pacchetti di sincronizzazione in S-MAC, anche quando non è realmente necessario).
In WiseMAC si tiene conto anche del fatto che, una volta che sono riuscito a trasmettere correttamente i dati, non devo più avere le informazioni del duty cycle della destinazione, perchè basta che questa mi manda informazioni in piggybacking su quando si sveglierà di nuovo, e da lì sono sincronizzato.

Clock Drift:

in un ambiente sincronizzato di device comunicanti un concetto molto importante è l'accuratezza del clock usato dai device, che viene misurata in parts per milion (ppm) e l'errore giornaliero che affligge tale accuratezza è di 0.8 sec per day.

6loWPAN:

ci fu questa idea che i device IoT avrebbero avuto bisogno di collegarsi ad Internet ed infatti si possono avere sia reti Ad-hoc isolate sia delle mesh network in cui i device riescono a comunicare con l'Internet esterno utilizzano un device denominato sink che funge da gateway, tenendo conto del fatto che le reti IoT hanno bisogno di un'ottimizzazione di come funzionano i pacchetti IP per rispettare i requisiti da loro richiesti.
Per implementare questa cosa abbiamo bisogno di modificare lo stack protocollare:

- in primis si hanno differenti physical e MAC layers
- poi si ha un adaptation layer chiamato ***LoWPAN***
- al posto di IP si usa IPv6

Al **livello fisico** si usando diverse bande di frequenza che variano in base alle aree geografiche in cui ci troviamo e sempre al livello fisico abbiamo un PDU strutturato nel seguente modo:

- *preambolo* - primi bit utilizzati per la sincronizzazione
- *start of packet delimiter* - bit che delimitano l'inizio dei dati veri e propri
- *PHY Header*
- *PHY Service Data Unit* - payload del pacchetto che contiene i dati provenienti dai livelli superiori

Al livello di topologia non abbiamo bisogno di creare strutture con molti hop e possiamo avere:

- *star*
- *tree*
- *mesh*

in tutte e tre le situazioni abbiamo diversi tipi di elementi che possono co-esistere nella rete e sono:

- *end devices* - terminali che, nelle ad-hoc network, possono anche routare informazioni e dati
- *PAN coordinator* - nodi di controllo che si occupano di *NET ID assignment, frequency selection, handling request to join e packet relaying (ed anche di COME comunicare all'interno della rete)*
- *Co-ordinator* - si occupano sostanzialmente di supportare e fare un po' del lavoro del PAN ed in particolare: *handling request to join e packet relaying*

Una rete viene iniziata con i seguenti step:

- *elezione del PAN coordinator* - infatti questo elemento può essere un nodo dedicato o può essere anche un qualsiasi nodo della rete ed il ruolo di PAN può ruotare nel corso del tempo
- *assegnamento IP ridotti* - il PAN si assegna un indirizzo più corto rispetto a quello dello standard IPv6 (indirizzi troppo lunghi, serve compressione dei dati) così che nella rete viaggino pacchetti con IP addr di 16 bit (l'indirizzo IPv6 originale viene comunque usato per riconoscere i device della rete)
- *selezione della frequenza*
- *i nodi in ingresso alla rete eseguono ACTIVE SCAN per fare discovery del PAN*
- *mandano un association request*
- *il PAN coordinator potrebbe assegnare un addr. a 16bit al nodo in ingresso*

Al **livello MAC** si possono avere due tipi di modalità: **beaconless mode** e **beacon mode**.

Nel beaconless mode viene usato CSMA/CA mentre, nella beacon mode, usiamo il fatto di avere un punto centrale come coordinatore che può inviare un network beacon dopo cui inizia un contention access period ed un contention free period:

- nel contention free period il coordinatore alloca slot di risorse per gli utenti che devono comunicare, così da poter fornire risorse non continuamente, ma quando serve

e per farsi allocare queste risorse, richiedo in uplink l'accesso durante il contention access period, basato sul CSMA/CA e durante quel periodo posso avere informazioni in più che mi indicano di poter saltare il contention access period successivo perchè avrò delle risorse allocate per me durante il free period.

Adaptation layer 6LoWPAN:

layer introdotto nelle reti 6LoWPAN che si occupa di:

- ***addressing:*** *fornisce dei not routable local addresses e degli EUI-64 identifiers a 8 byte che identificano i device. I local address sono di 16 bit e vengono usati per avere meno overhead nella trasmissione dei pacchetti.*
- ***routing:*** *un pacchetto 6lowpan è composto da - **dispatch** iniziale che descrive il tipo/sottotipo dell'header, hop mancanti, indirizzo src, dest ...*

I vari tipi di headers che posso avere sono:

- ***fragment header:*** *utilizzato quando devo usare frammentazione dei pacchetti*
- ***mesh addressing header:*** *usato nelle topologie mesh*
- ***IPv6 Header Compression*** *sempre inserito*

Dei tre possibili formati che possono avere (solo 1 header, 2 header e 3), quello usato viene indicato nel dispatch.

Fragmentation:

usato quando si inviano PDU più grandi di 128 bytes e la frammentazione/deframmentazione viene eseguita a link level.
Nel caso del primo frammento (in cui l'offset sarebbe 0) posto del classico datagram offset per risalire alla posizione del frammento nel pacchetto originale, visto che qui abbiamo bisogno di ottimizzare e comprimere i dati il più possibile, si usa un singolo bit per indicare se il frammento è il primo o no.
I bit utilizzati nell'header sono:

- *datagram size* - descrive la dimensione del payload totale non frammentato
- *datagram tag* - identifica il set di frammenti e viene usato per fare match dei frammenti dello stesso payload
- *datagram offset* - identifica l'offset dei frammenti nel payload non frammentato

La dimensione dell'header del frammento è 4 bytes per il primo header e 5 bytes per i successivi
Possiamo avere 2 tipologie di routing in questo tipo di reti e sono:

- ***Route over forwarding -*** *routing livello 3 usato nelle reti in cui i coordinatori fanno da router ed i nodi intermedi operano su pacchetti di livello 3 (Network layer), quindi nello standard usa IP routing*
- ***Mesh under forwarding -*** *routing livello 2, quindi si hanno principalmente switch che operano su pacchetti livello 2 (Data Link Layer), nello standard usa il protocollo RPL*

Header Compression:

comprime il più possibile l'header IPv6 vista la loro dimensione, si usano header stateless chiamati *HC1*per gli header IPv6 e *HC2* per gli header UDP.

HC1:

in questo tipo di header si cercano di evitare le informazioni superflue del pacchetto IPv6 (come ad esempio la versione, che qui è sempre 6 e quindi può non essere indicata) e comunque di comprimere il più possibile le altre.
Quindi abbiamo in HC1:

- *1 bit C* - indica con 0 o 1 se le informazioni riguardanti le classi di flusso sono significative o meno, quindi se sono da leggere o no

- *NH* - la maggior parte del tempo non usiamo gli header aggiuntivi di IPv6 ma abbiamo UDP, TCP e ICMP.

  Per questo motivo utilizziamo i bit NH per indicare se la parte di next header può essere skippata (valore != 0, non includo il next header) o se deve essere usata.
- *un bit finale che indica se poi abbiamo un header HC2*

notare che la compressione viene pensata per non perdere le informazioni, quindi abbiamo la possibilità di ricostruire l'header originale agli edge routers.

HC2:
viene fatta compressione anche di porta sorgente e destinazione indicando con un campo dell'header HC2 un subset di possibile porte tra cui fare la selezione (quindi, se alloco una porta del subset indico con questi bit quale selezionare altrimenti utilizzo la rappresentazione originale).

Collection Tree Protocol
(incentrato sulla comunicazione tra device e sink)

partendo da una topologia mesh possiamo costruire una rappresentazione ad albero in cui ogni device deve scegliere un parent dell'albero a cui trasmettere basandosi su qualche sorta di metrica che indica qual è il miglior path dal device al sink.
Ogni parent riceve i pacchetti dai figli e li forwarda verso il sink (quando si hanno molteplici sink, i dati vengono inviati a quello con il costo minimo).
E' un tipo di vector protocol in cui, per selezionare il next hop, vengono usate le seguenti metriche:

- *distanza in hops dal sink*
- *qualità del canale di comunicazione locale*

Le proprietà che si cercano di avere sono:

- *reliability* - un protocollo dovrebbe consegnare almeno il 90% di pacchetti end-to-end quando esiste una rotta
- *robustezza* - dovrebbe essere capace di operare senza tuning o configurazioni in un grande range di condizioni della rete
- *efficienza energetica*
- *indipendenza dall'hardware*
- *adattabilità al cambiamento delle metriche di rete*

Parent selection metric:
la metrica usata per la selezione dei nodi parent è l'***ETX = Expected Number of Transmissions to reach the sink***che viene calcolata in base alla performance riscontrata di recente dai beacon e dai data packets per i local 1-hop ETX.

E' una media pesata usata per stimare quante ritrasmissioni servono per consegnare i pacchetti alla destinazione, così da vedere quanta latenza, consumo di energia ecc. dovute alle ritrasmissioni ci dobbiamo aspettare da quel parent (usiamo questa perchè vogliamo il percorso migliore fino ai sink, restando energy efficient).
Sostanzialmente, durante la parent selection si prende come parent il nodo che ha un ETX minore di quello attuale del pacchetto.
In questo tipo di protocollo abbiamo due tipi di pacchetti: ***Data Frame*** *e* ***Routing Frame***.
Nel primo indico chi manda il pacchetto, qual è il sink di destinazione, sequence number, metriche ETX, Time has lived (hop limit dopo cui rimuovo il pacchetto dalla rete), un *pull bit* indica, quando settato a 1, che sto costruendo l'albero e quindi mando beacon packets ed infine un *Congested bit* usato per indicare se un nodo può essere selezionato o meno come parent.

Common architecture:

nel control plane abbiamo un Link Estimator usato per calcolare il ETX e decidere quali sono i parent, così che poi un routing engine possa configurare la forwarding table.
nel data plane vi è una componente per identificare quando ci troviamo un presenza di loop ecome comportarsi a riguardo.

Routing Loops:

per controllare la consistenza del routing viene controllato un *consistency criteria*sul path andando a controllare se per ogni i tra 0 e k-1, ETX(n_i) > ETX(n_i+1).
Si esegue una datapath validation controllando che il costo nel pacchetto sia inferiore a quello del device parent scelto.
Si incontra una inconsistenza quando il costo del nodo è maggiore di quello nel pacchetto ed in questo caso si manda un signal al control plane.
Quando ciò si verifica, si ricostruisce l'albero ripartendo dai beacon frames.
Quando ricostruiamo un albero mandiamo un grande quantitativo di beacon frames, per contenere questo numero utilizziamo un algoritmo chiamato ***Trickle algorithm*** per cui: la frequenza di invio dei beacon dipende dalla stabilità del canale di comunicazione che uso e più è stabile il canale più sarà bassa la frequenza e lungo l'intervallo tra un beacon ed un altro (quindi, se abbiamo una struttura stabile, mandiamo i beacon ad intervalli nell'ordine dei minuti).

IEEE 802.15.4

la prima versione dello standard alcune limitazioni, in particolare:

- il MAC usato era basata su CSMA/CA che non forniva garanzie sul delay della comunicazione, nessuna resilienza sulle interferenze (poco support al frequency hopping), non è un Medium Access Control ideale negli scenari di traffico elevato

per applicazioni che non erano propriamente supportate, questo standard ha fornito estensioni i cui obiettivi sono:

- *Low energy* - permette ai device di operare con un duty cycle veramente basso
- *Enhanced Beacons* - estensione dei già presenti beacon frames permettono di creare application-specific beacons, includenti informazioni rilevanti in base alle necessità delle varianti
- *Multiporpose frames*
- *MAC Performance metrics* - fornire informazioni link quality ai livelli superiori
- *Fast association* - cambiare la procedura di associazione che faceva una trading off tra energia e latenza in caso di applicazioni che richiedevano fast association

IEEE 802.15.4e variants (ci sono più varianti in base alle classi di applicazioni da supportare)

- *Radio frequency Identification Blink* mode supporta uno scambio efficiente di ID per item/people identification, location e tracking
- *Asynchronous multi-channel adaptation (AMCA)* supporta l'uso di più canali dinamicamente per trasmettere dati in parallelo
- *Deterministic and Synchronous Multi-channel Extension* che supporta applicazioni time-critical per reti grandi in PANs in cui sono stati attivati i beacon.

  Il meccanismo di Time Slot garantito nello standard 802.15.4 supporta solo 7 slots per frame e non permette l'uso di canali multipli

  Il DSME migliora il GTS formando un multi-superframe e usando operazioni multi-canale, dove un multi-superframe è un ciclo di superframes dove ognuno include un beacon frame, il CAP (solo per il primo superframe nel ciclo) e il GTS.
- *Low Latency Determinist Network* - utilizzato per supportare quelle applicazioni che richiedono davvero poca latenza (sotto i msec), in cui si usa un estensione multi-channel per cui il PAN coordinator ha transceivers multiplic con cui può trasmettere e ricevere contemporaneamente su più canali, si hanno topologie a stella, pacchetti/slot più corti, indirizzi di 8-bit
- *Time Slotted Channel Hopping* - combina slotted access, support multi-channel e frequency hopping per supportare capacità delle network aumentate, maggiore affidabilità e latenza predicibile, mentre si fa uso di un duty cycle basso.

Time Slotted Channel Hopping:

si dividono i canali disponibili in time slots ed ogni comunicazione viene mappata su uno slot di un canale.
Abbiamo totale flessibilità perchè posso sia allocare molteplici comunicazioni di uno stesso nodo, su slot diversi, sia allocare comunicazioni di nodi diversi su uno stesso slot (potrebbero interferire ma uso CSMA/CA per splittare le comunicazioni).
Quindi posso allocare dinamicamente le risorse per adattarmi alle necessità ed allo stato della rete.

Current scenario:

nello scenario attuale abbiamo un insieme di devices che si interfacciano con la local network in cui uno o più gateways trasmettono le informazioni verso Internet che veicola il tutto verso back-end services dove si compiono diverse azioni come anche Business Data Analysis.
Alcuni devices IoT, non molti, sono wired perchè magari sono attuatori vecchi e direttamente collegati alla rete.
Poi abbiamo i device short range (wifi, bluetooth ecc) che hanno poca copertura e molto consumo.
Device LPWA come quelli che usano LoRa che hanno maggiore copertura e batteria ma meno datarate.
I cellular 2G/3G/4G/LTE ecc. che hanno maggiore copertura ma minor consumo.
Infine abbiamo i satellitari (1%) utilizzati per avere copertura sostanzialmente ovunque però hanno maggior costo.
Le possibili realizzazioni sono sostanzialmente: mesh network con gateway o start network con gateway in cui i gateway, per le mesh, possono essere wi-fi, 3G/4G e Ethernet mentre per le star possono essere le stesse delle mesh + Sigfox o LoraWAN.

LORA TECHNOLOGY

l'architettura tipica delle **LPWAN (Low Power Wide Area Network)** è costituita da una base-station che, in zone alte ed esposte, serve circa più di un milione di nodi. Questi nodi sono sensori piccoli ed efficienti in termini di costo, che comunicano usando ultra-low power su ultra-long distances.

Challenges:

questa tecnologia è sottoposta principalmente a questi problemi:

- ultra-low power e ultra-long distances portano ad avere alta attenuazione del segnale e quindi livelli di ricezione veramente bassi (tipicamente si assume < -140 dB)
- interferenze con altri servizi quando vengono usate le license-exempt bands

- incremento delle interferenze alla base station, dovuta al fatto che è esposta e che quindi cattura veramente tanti segnali e le interferenze
- il concetto del CSMA non funziona perchè l'area di una cell è abbastanza ampia da portare al problema dell'hidden node

Lora stack:
a differenza dello stack tipico in cui si hanno MAC e PHY layers standardizzati, con uno stack e più suppliers qui abbiamo:

- *patented PHY layer (della SEMTECH)*
- *open LoRaWAN MAC (della LoRaWAN)*

così da avere un supplier e più stack personalizzati.

PHY Layer:

per raggiungere distanze molto lontane ci sono varie possibilità basate sull'incremento dell'energia per bit:

- *incrementare la potenza di trasmissione - lede molto sulla lifetime del device*
- *usare spread spectrum technology per inviare lo stesso ammontare di dati in una quantità di tempo/bandwidth maggiore*

LoRa utilizza spread spectrum (una variante della *chirp-spread spectrum*, dove chirp indica un sengale con una variazione lineare nella modulazione di frequenza nel corso del tempo) ed è robusta alle interferenze, multipath e fading
possiamo vedere nella figura in alto che variando linearmente la frequenza di modulazione dei dati, varia anche il time symbol, cioè l'ammontare di tempo usato per inviare un dato e la codifica di questi dati dal modo in cui vengono "ruotati" i simboli che dobbiamo inviare (da quanto ho capito, possiamo vedere questa rotazione nella figura sotto, dove uno stesso simbolo parte da una certa frequenza, cresce e poi riprescipita e così via).

Un aspetto importante della modulazione di LoRa è che si tratta di una **constant envelope modulation**, ovvero una modulazione per cui la potenza inviata è costante e questo è positivo sia da un punto di vista di implementazione che di rumore sul canale.

Anche la demodulazione è molto semplice, in quanto si moltiplica (in qualche modo) il chirp ricevuto (baseband lora chirp) per il suo inverso (inverse chirp) e si ottengono differenti tones in differenti frequenze (come una normale modulazioni, ma con differenti tones).
Inoltre, questa demodulazione può funzionare anche al di sotto del **noise floor**, ovvero il livello sotto cui non si riesce a leggere il segnale.

LoRaWAN

protocollo di comunicazione ed architettura usato come MAC protocol da LoRa.
Supporta:

- *secure bi-directional communications*
- *mobility*
- *localization*

Definisce 3 differenti classi di device:

- ***Class A:*** nodi che si occupando di trasmissioni uplink e per i quali vengono aperte due piccole finestre di ricezione, solo dopo che una trasmissione uplink è stata fatta
- ***Class B:*** come la classe A, ma con finestre di ricezione esxtra schedulate nel corso del tempo
- ***Class C:*** nodi con finestre di ricezione continue, con cui possono ricevere dati eccetto quando sta trasmettendo

I classe A sono tendenzialmente sensori alimentati a batteria, quindi devono essere i più efficienti al livello energetico ed il downlink è disponibile solo dopo una trasmissione del sensore.
I classe B sono attuatori alimentati a batteria e sono efficienti per quanto riguarda l'energia, avendo comunque meccanismi per il controllo della latenza in downlink ed utilizzano comunicazioni slotted che vengono sincronizzate con un beacon
I classe C sono quelli alimentati da centri ci energia e sono device che possono ascoltare continuamente, senza avere latenza per la comunicazione in downlink.

Gateways:

tipicamente non sono alimentati a batteria perchè sono in ascolto tutto il tempo e su tutti gli Spreading Factors.
Quindi, tutti i gateway ricevono tutti i canali tutto il tempo, non avendo così bisogno di network controller o reuse planning.
I sensori possono comunicare con un qualsiasi gateway e tutti i pacchetti correttamente demodulati vengono poi mandati da loro al newtork server.

Network Server:

parte di intelligenza centralizzata della rete, che si occupa di:

1. *identificazione dei pacchetti duplicati*
2. *validazione dei dati inviati e multiplexing/demultiplexing dei diversi application servers (necessario perchè più application providers possono coesistere allo stesso momento)*

3. *possibile localizzazione dei device, grazie ad un riferimento centrale del tempo condiviso da tutti i gateways - in questo modo si possono usare i timestamp di invio messi nei pacchetti per calcolare la differenza tra tempo di invio e tempo di arrivo, così da calcolare la distanza da ogni gateway e quindi la localizzazione del device*

avere questa intelligenza centralizzata permette anche di avere gateways low cost, visto che le decisioni vengono prese dal server.

Security:

ci sono due layer che si occupano della sicurezza e sono:

- Network (newSkey): *autenticazione degli utenti ed aggiunta di controlli di integrità per i messaggi*
- Application (128 bit key length): *utilizzato per separare i dati delle applicazioni dagli altri operatori di rete*

La cosa interessante è che si possono avere due tipi di attivazione della parte di security: ***static attivation (pre configurata)*** *oppure* ***over the air.***

Over the Air Activation(OTAA):

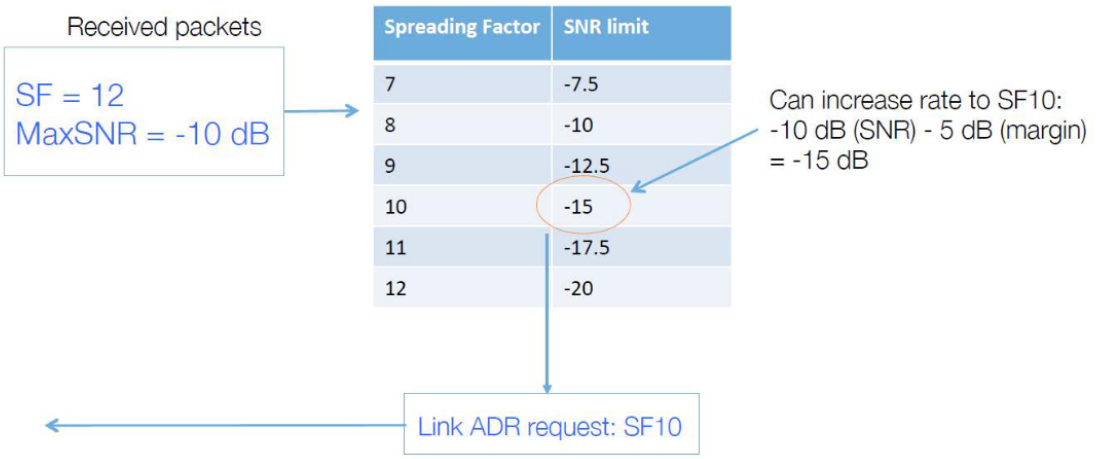
un nodo deve eseguire una procedura di join prima di inviare dati con il Network Server. Questa procedura deve essere ripetuta ogni volta che ha perso le informazioni sul contesto della sessione. Le informazioni necessarie per eseguire questa procedura sono:

- **JoinEUI**: ID globale dell'applicazione nello spazio di indirizzi IEEE EUI64 che identifica in modo univoco il Join Server (per la derivazione delle chiavi di sessione);
- **DevEUI**: identificatore di dispositivo univoco a livello globale nello spazio IEEE EUI64;
- **AppKey**: chiave di crittografia root AES-128 specifica per il dispositivo finale (specifica per ogni nodo, in modo che se uno venisse compromesso gli altri siano comunque protetti da chiavi diverse);
- **NwkKey**: chiave root AES-128 specifica per il dispositivo finale, ma fornita dall'operatore di rete.

La chiave di sessione dell'applicazione è utilizzata per crittografare il payload dell'applicazione e la chiave di sessione di rete viene utilizzata per firmare il Message Integrity Check e per autenticare gli utenti.

SF (Spreading Factor) allocation:

l meccanismo di base utilizzato per selezionare la migliore SF è quello di selezionare il più piccolo possibile, al fine di ottenere la massima velocità di trasmissione dati disponibile e di ridurre il time on air dei pacchetti.



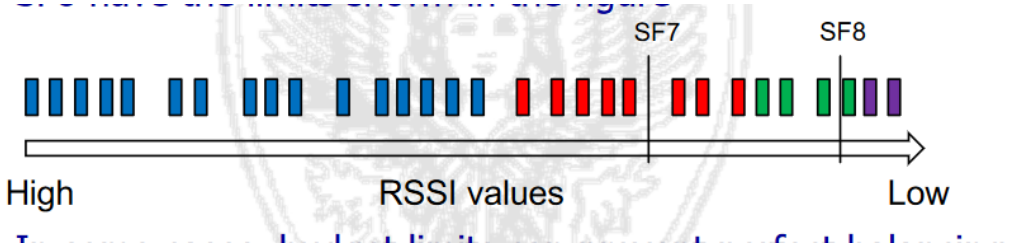
Nell'esempio mostrato possiamo vedere che l'SNR massimo misurato è -10 dB e l'SF effettivamente utilizzato è 12. Osservando la tabella si nota che il SF può essere modificato a 10 per aumentare la velocità di trasmissione dei dati ed essere ancora in grado di ricevere correttamente le informazioni.
Inoltre, se tutti i dispositivi sono vicini al gateway, lavoreranno con il minimo SF7. Pertanto, la capacità della cella non dipende solo dal numero di dispositivi, ma anche dalla loro posizione. Questo significa che possiamo aumentare la capacità della cella bilanciando gli SF tra i nodi.
Utilizzando SF diversi, i pacchetti avranno una durata diversa (aumentare SF di uno significa raddoppiare il tempo di permanenza in aria dei pacchetti). Di solito la maggior parte dei nodi
nodi utilizzano tempi di trasmissione più bassi (il che significa velocità di trasmissione più elevate e SF più bassi) e solo pochi nodi utilizzano gli SF più alti possibili.

Come distribuire i nodi con SF diversi nella cella?

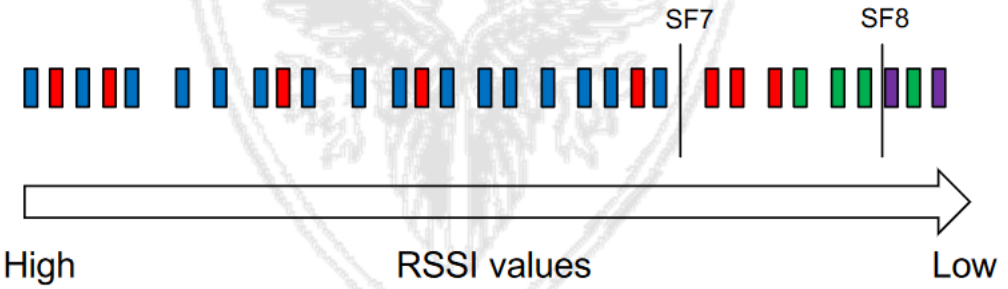
Le opzioni sono opzioni sono: **anelli circolari** o **diffusione uniforme**. La prima opzione sembra la più semplice, ma non è quella corretta. Infatti, in questo caso l'interferenza tra SF sarà sbilanciata, in quanto ne soffriranno solo gli utenti più lontani, anche se utilizzano SF più alti. La seconda opzione è la migliore in fatto di cattura (quando due pacchetti si trovano sullo stesso SF) ed evita che gli utenti lontani soffrano di un'interferenza inter- SF più elevata.
Consideriamo che nel caso di catture, si ha un maggior *Data Extraction Rate* per i vari SF.

Un'altra idea per mitigare l'interferenza potrebbe essere quella di sfruttare il controllo della potenza. Tuttavia, non c'è alcun beneficio reale nel farlo, in quanto distrugge le opportunità di cattura. È l'equivalente di spostare i nodi alla stessa distanza dal gateway.

EXPLORA:
È uno schema che distribuisce gli SF in base al loro bilanciamento, ma tenendo conto anche della potenza ricevuta (RSSI). Funziona con un approccio sequenziale di sequenziale di water filling. Per prima cosa gli utenti vengono ordinati in base all'RSSI:



Si inizia ad allocare a partire da SF=7 e dal valore massimo di RSSI. Man mano che l'RSSI si abbassa, assegniamo SF più alti.
Ora dovremmo distribuire il più possibile gli utenti che lavorano sullo stesso SF per aumentare le catture:



Teniamo conto della distanza tra i nodi consecutivi. Se l'RSSI differisce dal nodo precedente di più di 3 dB, manteniamo l'attuale SF.
Altrimenti, utilizziamo un SF diverso che sia compatibile con il budget del collegamento.
Se abbiamo M celle e di conseguenza M gateway, questo meccanismo funziona dividendo i nodi in M gruppi diversi che identificano i nodi che vedono lo stesso gateway.
stesso gateway. Ciascun gateway assegna poi i diversi SF seguendo la stessa procedura precedente

---

ROUTING IN IOT

le feature desiderabili sono:

- energy aware, small factor, lightweight solutions, low overhead
- dovrebbe essere scalabile per migliaia di oggetti
- auto configurabile e gestibile

- robusto anche in presenza di link che variano in qualità

Ripple routing protocol RPL:

- un tipo di protocollo di distance vector proattivo che specifica come costruire un grafo di destinazione aciclico ed orientato (DODAG).

L'idea è che il manager della rete può configurarla per fare routing basandosi su differenti indicatori di performance, in modo del tutto flessibile ed in base alla sua necessità.
Queste metriche potrebbero essere:

- *trovare il path con il miglior ETX ed evitare link non cifrati*
- *trovare il miglior path in termini di latenza mentre evitiamo i nodi che operano con batteria*

e dunque possibile usare differenti metriche e soprattutto usare costruzioni di metriche diverse per diversi flussi di informazione.
La composizione è importante perchè potrei avere requisiti diversi per i diversi flussi, come ad esempio: potrei volere sì il path col miglior ETX, ma allo stesso tempo potrei voler evitare alcuni hop, come quelli senza cifratura.
Quindi usiamo policy più che metriche, che sono più ricche.
Inoltre gli admin possono decidere di avere più routing topologies allo stesso tempo, per gestire requisiti differenti.
Parliamo di "lossy network" perchè la link quality e quindi i pacchetti vengono correttamente consegnati o persi in modo altamente variabile.

DODAG formation:

abbiamo un Root (ER-LowPAN Border Router) che invia uno specifico tipo di comando DIO (DODAG Information Object) e la sua propagazione viene usata per raccogliere informazioni sulle metriche.
I vicini del nodo root ricevono il DIO e decidono se unirsi o meno al DODAG:

- ognuno di essi, ricevendo il DIO, seleziona il proprio parent (in base alla metrica scelta) e, se è configurato come router, riforwarda il DIO
- lead nodes non reforwardano il DIO

una volta che un nodo si unisce al grafo, manda un DAO message (Messaggio di Advertisement) fino alla root.
Questi messaggi - DAO, DIO e DIS (usato per sollecitare proattivamente le informazioni del grafo) - compongono i nuovi messaggi di controllo ICMPv6.
Lo scambio di informazioni sulla **prefix reachability** permette anche di fare comunicazioni peer to peer:

- i dati possono essere inviati fino al primo antenato che conserva informazioni su come raggiungere la destinazione e questo li forwarda a destinazione (qui si parla di **storing mode**)

- i dati devono essere inviati fino al root node che aggiunge un source route e li manda giù fino alla destinazione (qui si parla di **non storing mode**)

I messaggio DIO vengono inviati in base ad un trickle timer dinamicamente selezionato che si basa su quanto è stabile il sistema:

- se è stabile, viene mandato seldom (raramente)
- ogni volta che si verifica una inconsistenza (come un loop o un cambiamento nel DODAG) il timer viene resettato a valori inferiori

RPL è in cima al sotto stack protocollare che si trova nel network layer, che è quindi composto da: IETF 6LoWPAN, IPv6, ICMP, IETF RPL

EH-WSNs

uno dei problemi principali nelle reti IoT è il consumo di energia: in molti applicazioni la rete deve essere operativa per decadi, ma i nodi vengono alimentati a batteria e la comunicazione tra essi è costosa.

L'approccio standard è di utilizzare il duty cycle, in cui periodicamente la radio passa tra stato di ON e OFF e per cui utilizziamo meccanismi di controllo per gestire il disallineamento temporale degli intervalli in cui un nodo è ON e l'altro è OFF.

All'atto pratico dobbiamo considerare il Trade-off tra ***Consumo di energia*** e ***Latenza*** e possiamo vedere, andando avanti con il lifetime di una network, la latenza media delle comunicazioni tende ad aumentare.

La domanda è: **possiamo utilizzare l'energia ambientale per alimentare i device IoT?**

- possiamo realmente farlo, solo se siamo veramente Low Power - se siamo low power in termini di HW e SW, possiamo ricavare dall'ambiente abbastanza energia per alimentare questo tipo di device

Il processo di accumulare energia al di fuori di quella eventualmente fornita dalla batteria (se presente) , sfruttando altre fonti di energia per alimentare direttamente il device o ricaricare l'eventuale batteria, viene chiamato ***energy harvesting.***

E' necessario fare predizioni dell'energia che potremmo accumulare perchè bisogna mitigare il più possibile l'incertezza nella disponibilità energetica, che può portare a due situazioni da evitare:

- nelle situazioni in cui l'energia accumulabile non è molta (es: giornata nuvolosa), si rischia di finire in uno stato in cui non abbiamo abbastanza energia per eseguire task critici
- nelle situazioni in cui l'energia accumulabile è molta (es. molto sole), si rischia di raggiungere un surplus energetico in cui si raggiunge il livello massimo di energia cumulabile e per cui si spreca tale energia se non viene usata

Per evitare queste situazioni si adottano metodi per pianificare in anticipo quale sarà l'utilizzo di energia, facendo ***allocazione proattiva*** dell'energia utilizzando quella

disponibile al meglio:

- *minimizzando situazioni in cui si finisce l'energia perdendo così task di massima priorità*
- *minimizzando gli sprechi di energia quando si è in surplus*
- *permettendo operazioni non considerate fattibili, quando si è in situazioni in cui abbiamo surplus*

PRO Energy:

si basa sul tracciare i vari ***enrgy profiles*** osservati durante un numero D di giornate tipiche (es di profili. giornata nuvolosa, giornata soleggiata) prendendo i profili più rappresentativi ed utilizzandoli per predirre nell'immediato futuro, quale sarà l'energy intake (guadagno di energia) guardando ai più simili profili registrati, per stimare il futuro sostanzialmente inizio ad osservare l'intake corrente della giornata e cerco tra i profili registrati, quale fitta meglio il profilo che sto registrando ora.
Sostanzialmente, il conto che facciamo è una somma pesata da fattori alpha che vanno tra 0 e 1:

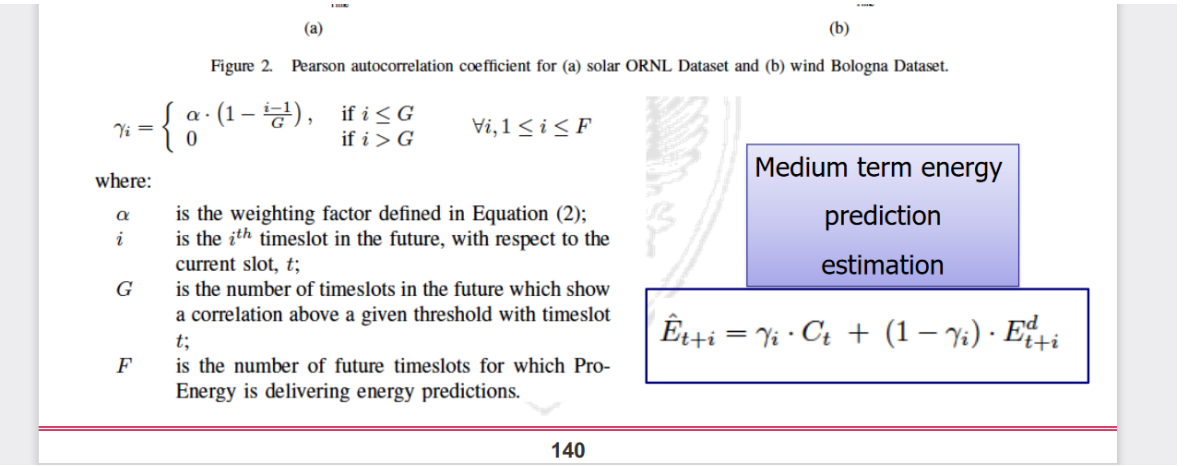
$$\hat{E}_{t+1} = \alpha \cdot C_t + (1-\alpha) \cdot E^d_{t+1} \qquad (2)$$

where:

$\hat{E}_{t+1}$ is the predicted energy intake in timeslot $t+1$ of the current day;
E^d_{t+1} is the energy harvested during timeslot $t+1$ on the stored day d;
C_t is the energy harvested during timeslot t on the current day C;
α is a weighting factor, $0 \leq \alpha \leq 1$.

The weighting parameter, α, allows to combine the value reported in the stored profile with the current energy observation, i.e., the energy observed in the last slot, C_t.

dove abbiamo che E'_{t+1} è l'intake predetto nel timeslot t+1 del giorno corrente, E^d_{t+1} è l'energia raccolta nel timeslot t+1 del profilo d e C_t è l'energia raccolta nel tempo t del giorno corrente.
Abbiamo bisogno anche di una misura di quanto sia rilevante e correlato quello che abbiamo misurato in passato rispetto a quello che vivremo nel futuro.
Questo calcolo viene fatto stimando la ***Medium term energy prediction*** per cui abbiamo:



(a) (b)

Figure 2. Pearson autocorrelation coefficient for (a) solar ORNL Dataset and (b) wind Bologna Dataset.

$$\gamma_i = \begin{cases} \alpha \cdot \left(1 - \frac{i-1}{G}\right), & \text{if } i \leq G \\ 0 & \text{if } i > G \end{cases} \qquad \forall i, 1 \leq i \leq F$$

where:

- α is the weighting factor defined in Equation (2);
- i is the i^{th} timeslot in the future, with respect to the current slot, t;
- G is the number of timeslots in the future which show a correlation above a given threshold with timeslot t;
- F is the number of future timeslots for which Pro-Energy is delivering energy predictions.

Medium term energy prediction estimation

$$\hat{E}_{t+i} = \gamma_i \cdot C_t \; + \; (1 - \gamma_i) \cdot E^d_{t+i}$$

140

Notes: è possibile rappresentare l'harvested energy di ogni nodo anche in termini di dati trasmessi/ricevuti durante il giorno, umidità e temperatura (come impatta sull'energia), misurazioni eseguite durante il giorno.
E' possibile utilizzare anche **Harversting-aware routing** per cui si tiene conto nella selezione del next hop anche di quant'è l'energia corrente sommata all'energy intake predetto, così da poter dare priorità maggiore a chi sta vivendo/vivrà una situazione di ***energy peaks*** (adottando questa tecnica è possibile vedere che il path seguito dai pacchetti varia in base alle condizioni ambientali ed anche in base al momento della giornata, in certe situazioni, portando ad un comportamento **auto-adattivo**).