Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE

Corso di Laurea Magistrale in Informatica

Curriculum B - Informatica per il Management

Droni e sensori low-power per applicazioni IoT di monitoraggio: una valutazione sperimentale

Relatore:
Chiar.mo Prof.
Marco Di Felice

Presentata da:
Paride Martinelli

Sessione III
2017 - 2018

Introduzione

Al giorno d'oggi la tecnologia sta diventando sempre più pervasiva e tutto ciò che ci circonda sempre più intelligente. A partire dagli oggetti più comuni che utilizziamo quotidianamente all'interno delle nostre case, come termostati, impianti elettrici e lampadine; tutto ciò che troviamo nelle strade, come macchine, semafori, spartitraffico e tanto altro ancora. Anche strutture più impensabili, come le campagne, gli allevamenti e l'agricoltura stanno man mano diventando sempre più smart. Si stanno creando delle vere e proprie città intelligenti, composte da case, strade e campagne intelligenti; che a loro volta sono composte da oggetti smart più piccoli. Questo mondo vene chiamato Internet Of Things. Un mondo popolato da oggetti digitali di piccole dimensioni, capaci di raccogliere una grandissima quantità di dati, di elaborarli e di creare da essi intelligenza.

Gli oggetti smart appena citati comunicano tra di loro per mezzo della tecnologia LPWAN, una tecnologia di comunicazione wireless che fornisce connettività a dispositivi caratterizzati da una bassa potenza e da una bassa velocità di trasmissione dati. In particolare LoRa definisce il livello fisico e LoRaWAN definisce il livello MAC. Queste tecnologie, per ambienti outdoor, sfruttano lo standard IEEE 802.15.4, uno standard che fornisce supporto sia a livello fisico che a livello MAC della rete.

Uno dei principali utilizzi della tecnologia LPWAN è in agricoltura con la smart agriculture. Ogni singola fase della produzione agricola potrebbe es-

sere migliorata ed agevolata dalle nuove tecnologie: dalla gestione del suolo, alla minimizzazione del consumo di acqua; dalla protezione delle piante; fino ad arrivare alla salute degli animali e all'automazione degli allevamenti. Lo smart farming permette infatti alle aziende di essere più efficaci, di ottimizzare i processi, di massimizzare i rendimenti e di minimizzare lo spreco di risorse. Tutto questo grazie all'utilizzo di sensori intelligenti che raccolgono informazione e ne interpretano l'evoluzione. Le nuove tecnologie permetteranno, in futuro, di offrire ai campi coltivati, alle piante e agli animali proprio quello di cui avranno bisogno al momento giusto.

È proprio dalla smart agriculture, e in generale dagli scenari outdoor, che prende spunto questo progetto. Infatti questa tesi si basa su un'analisi qualitativa di una rete wireless composta da sensori, in particolare mette a confronto una soluzione "classica", basata su un livello di comunicazione a terra, e una soluzione alternativa basata su comunicazioni aeree. L'obiettivo è quindi quello di valutare l'efficacia di scenari in cui il tradizionale sistema di multi-hop forwarding viene sostituito da un drone per il recupero dei dati. Per effettuare le analisi si è utilizzato un ambiente simulato implementato in OMNET++.

La rete sfrutta lo standard IEEE 802.15.4 per definire i livelli fisici e MAC dei vari sensori. Vengono fatti quattro tipi di esperimenti, per ogni esperimento viene cambiata la configurazione della rete al fine di migliorarla. Il miglioramento viene fatto analizzando quattro diversi parametri: il delay, che è il tempo impiegato da un pacchetto per arrivare a destinazione, calcolato in secondi; il throughput, che misura la capacità effettiva della rete, calcolato in bytes/s; il packet delivery ratio, che è la percentuale dei pacchetti che effettivamente arrivano a destinazione a partire dal totale inviato; e infine la lifetime, che è la durata complessiva della rete, ovvero quanto tempo dura la batteria dei sensori prima che muoiano e lascino quindi la rete scollegata.
Il primo esperimento consiste nel determinare il numero ottimale di sensori

che compongono la rete. L'analisi consiste nello studio delle metriche prese in considerazione e si varia il numero di nodi che popolano la rete, partendo da un totale di 4 nodi fino ad arrivare a 144. Una volta trovato il numero ottimale di sensori, si passa al secondo esperimento. Il secondo esperimento prende appunto la configurazione con il numero ottimale di sensori, ovvero 36 e si cerca di ottimizzarne il packet delivery ratio cercando di variare il traffico generato. Per variare il traffico sono effettuate svariate simulazioni al variare dell'intervallo di tempo con cui i sensori inviano i loro pacchetti. Il terzo e quarto esperimento invece sono finalizzati al tentativo di prolungare la lifetime della rete. In particolare il terzo esperimento sfrutta un meccanismo chiamato sleep and wake up sul duty cycle della radio della rete per il prolungamento della vita della rete, invece il quarto ed ultimo esperimento utilizza un drone. Nel quarto esperimento il drone viene utilizzato prima esclusivamente per la raccolta dati e poi come sistema per svegliare i sensori oltre che per la raccolta.

Prima di fare questo tipo di analisi e di trarne le relative conclusioni, è stato redatto un primo capitolo che parla dell'IoT: cos'è, come nasce, le sue caratteristiche principali, le conseguenze che comporta e alcuni scenari applicativi. Dopo una breve introduzione sull'IoT vengono elencati i principali standard di rete e le principali schede di prototipazione utilizzati in questo mondo smart. Viene poi fatto un approfondimento sulle tecnologie LoRa, LoRaWAN e più in generale su LPWAN.

Il secondo capitolo invece descrive lo stato dell'arte, infatti vengono discussi i vari articoli, già scritti aventi come tema le reti wireless di sensori che sfruttano dispositivi mobili UAV (Unmanned Aerial Vehicle) , mettendoli in relazione tra di loro e confrontando i risultati ottenuti dalle loro analisi.

Un terzo capitolo descrive le varie fasi della progettazione e della creazione della struttura per l'analisi dei dati. Vengono quindi descritte le specifiche del

progetto, elencate le tecnologie utilizzate per l'implementazione dell'ambiente simulato, e vengono descritti i moduli INET utilizzati per configurare la rete.

Il quarto ed ultimo capitolo descrive in dettaglio l'analisi svolta sezionandola per le sue varie fasi; partendo dalla descrizione delle metriche prese in considerazione fino ad arrivare alla descrizione dei vari esperimenti effettuati. Di ogni esperimento viene spiegato lo scenario, vengono elencati i dati ottenuti, i quali verranno poi rappresentati graficamente, ed infine verranno tratte specifiche conclusioni in merito ai risultati ottenuti.

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Internet of things, componenti e standard di rete

In questo capitolo verrà introdotto il mondo dell'Internet of Things (IoT), ne verranno descritte le caratteristiche principali, i suoi componenti e le conseguenze che comportano le innovazioni tecnologiche che stanno alla base dell'IoT. Si parlerà delle varie dashboard in commercio, dei protocolli di comunicazione ed in generale degli standard utilizzati per l'IoT. Infine verranno elencate tecnologie e strumenti di analisi che permettono la gestione della grande quantità di dati prodotta da questi strumenti innovativi.

1.1 Introduzione all'IoT

Al giorno d'oggi la tecnologia sta diventando pervasiva. L'informatica in generale, il rilevamento di informazioni e dati e l'utilizzo di questi dati per la propagazione della conoscenza e per il marketing sono ovunque. Si è passati da una rappresentazione approssimativa di questi dati ad una vera e propria rappresentazione semantica, più precisa per poterli sfruttare al massimo.

Tutto ciò che ci circonda sta diventando intelligente. A partire dagli oggetti più comuni che utilizziamo quotidianamente all'interno delle nostre case,

come termostati, impianti elettrici e lampadine; tutto ciò che troviamo nelle strade, come macchine, semafori, spartitraffico e tanto altro ancora. Anche strutture più impensabili, come le campagne, gli allevamenti e l'agricoltura stanno man mano diventando sempre più smart. Si stanno creando delle vere e proprie città intelligenti, composte da case, strade e campagne intelligenti; che a loro volta sono composte da oggetti smart più piccoli.

Questo mondo vene chiamato Internet Of Things. Un mondo popolato da oggetti digitali di piccole dimensioni, capaci di raccogliere una grandissima quantità di dati, di elaborarli e combinarli tra di loro per venire incontro alle esigenze dell'utente. Questi oggetti, grazie alla loro capacità intrinseca di creare valore dai dati, stanno rivoluzionando le nostre vite quotidiane.

Smart Things

Gli oggetti intelligenti sono dispositivi digitali che forniscono funzioni di servizio realizzate dalla sinergia tra sensori, attuatori e controllori (eventualmente implementati da piattaforme di esecuzione locali / distribuite e comunicazioni Machine to Machine (M2M)/ Internet). Un sensore è un dispositivo collegato che consente di rilevare i parametri fisici dello scenario o dell'ambiente controllato, i cui valori vengono trasformati in dati digitali. Un attuatore è un dispositivo connesso che consente l'attivazione di azioni sull'ambiente controllato. Un controller è un dispositivo connesso che implementa un algoritmo per trasformare i dati di input in azioni. L'unione di più Smart Things da origine a Smart Environments.

Smart Environments

Il più grande esempio di ambiente intelligente è la Smart Home. Una generica Smart Home include una moltitudine di sensori e oggetti interagenti (attuatori). All'interno di una casa intelligente non è necessaria nessuna interazione umana per l'assemblaggio e la configurazione del sistema. Gli oggetti si auto-scoprono e si coordinano in modo autonomo. Infatti, gli oggetti intel-

ligenti che popolano la Smart Home sono in grado di apprendere in maniera autonoma le esigenze e i bisogni degli esseri umani che vi vivono. Questo è il paradigma che sta alla base dell'IoT. Alcuni esempi di Smart Things che popolano la Smart Home possono essere: sistema di riscaldamento, automazione porte/finestre, sicurezza, allarme anti intrusione, consumo monitorato di energia, frigorifero intelligente, vestiti intelligenti, casse audio intelligenti, luci intelligenti e tanto altro ancora.

1.1.1 Pervasive communication and sensing

Come accennato nell'introduzione, il mondo delle comunicazioni sta diventando sempre più pervasivo. Se si pensa che Internet prende le sue origini da un progetto chiamato ARPANET nel 1969, come rete locale, utilizzata poi dal 1971 per lo scambio delle prime email solo per i dipendenti del CERN di Ginevra che collaboravano al progetto. Oggi, invece, non solo internet è diventata una rete pubblica, ma abbiamo la possibilità di restare sempre connessi con qualsiasi tipo di dispositivo, possiamo scegliere la tipologia di rete da strutture per le nostre comunicazioni e sfruttare servizi cloud che la rete mette a disposizione. Si è partiti da una rete che metteva in comunicazione pochi individui e si è arrivati ad una rete che mette in comunicazione ogni singolo individuo. Quello che si vuole ottenere, in un futuro non troppo lontano, è una rete smart machine to machine (M2M), ovvero che mette in comunicazione terminali, computer e macchine in generale in modo del tutto autonomo, senza l'uomo come intermediario.

La pervasività della rete non si dimostra solo in maniera virtuale attraverso la comunicazione, ma anche in maniera fisica. Le informazioni digitali, infatti, vengono recuperate da un'ampia gamma di dispositivi che incorporano la possibilità di comunicare tra di loro per mezzo della rete. Questi dispostivi, chiamati NoC (Network on Cip) li possiamo trovare ovunque. Un

facile esempio sono i braccialetti intelligenti che permettono di recuperare informazioni sullo stato di salute dall'utente che intraprende attività sportive. Questi braccialetti, dotati di NoC, sono in grado di recuperare informazioni, inviarle ad un server centrale e distribuirle per il web. L'esempio del braccialetto intelligente è solo uno dei tanti oggetti smart in grado di sfruttare queste potenzialità della rete e ci fa capire come la rete può essere ovunque. Non solo la rete è pervasiva, ma anche i dispositivi di rilevamento lo sono diventati. Un semplice smartphone, grazie ai sensori che lo compongono, è in grado di fornire molte informazioni sul suo possessore e sulle sue attività. È in grado di determinare i tuoi spostamenti, grazie ai sensori GPS; conosce i tuoi contatti e le interazioni che hai con loro e tanto altro ancora. Basta pensare che un semplice cellulare di seconda generazione dispone di 14 sensori: accelerometro, giroscopio, magnetometro, barometro, sensore di prossimità, di luce, di pressione, GPS, WiFi, Bluetooth, GSM/CDMA Cell, NFC e due telecamere. Un'automobile economica di bassa cilindrata dispone di 200 sensori, tra cui freni, cinghie, dispositivi per il sistema airbag e dispositivi per la chiusura di sicurezza delle portiere. Un'auto di lusso può arrivare fino a 600 sensori. I sensori, infatti, permettono di migliorare la sicurezza e i confort del veicolo.

Mettendo in relazione quelle che sono le principali caratteristiche dell'IoT studiate fino adesso, ovvero la capacità di raccogliere i dati e di sfruttarli in maniera intelligente, la possibilità di comunicarli attraverso la rete e l'utilizzo dei sensori per migliorare la sicurezza e i confort dei veicoli, è possibile dare vita a veri e propri veicoli autonomi.

Non solo dispositivi smartphone possono rilevare informazioni sui loro possessori, ma tutto ciò che ci circonda oggi è in grado di rilevare abitudini, preferenze alimentari e sessuali, prevedere i nostri spostamenti e venire a conoscenza degli amici, dei componenti della famiglia, del nostro stato di salute e delle nostre condizioni economiche. Per assurdo, il tuo frigorifero

intelligente e il tuo forno intelligente ti potrebbero conoscere meglio del tuo dottore, il tuo smartphone ti conosce meglio della tua fidanzata e in generale la tua casa ti conoscerà meglio dei tuoi genitori.

Possiamo quindi affermare che "le cose", comprese quelle più inaspettate, avvertono, raccolgono e condividono le informazioni in modo proattivo. Queste informazioni producono un'incredibile quantità di dati e scaturiscono un nuovo paradigma di programmazione più semplice per gli utenti, i quali dovranno semplicemente fornire obiettivi. Gli oggetti, in completa autonomia e coalizione tra di loro, riusciranno a portare a termine questi obiettivi in maniera del tutto trasparente all'utente. Ovviamente questo porterà non solo dei vantaggi ma anche dei rischi per la privacy e la sicurezza dell'utente.

1.1.2 Industry 4.0

Adottando questi nuovi paradigmi al mondo dell'industria è possibile trarne innumerevoli benefici. Infatti è proprio qui che nasce l'Industria 4.0. I paradigmi dell'IoT e i sensori hanno migliorato i processi di produzione anche grazie al concetto di manutenzione predittiva, ovvero alla loro capacità di predire lo stato del macchinario e prevederne la sua usura.

Possiamo dire che nella storia sono avvenute quattro rivoluzioni industriali. La prima del 1784, attraverso l'introduzione di impianti di produzione meccanici che sfruttavano l'ausilio di acqua e vapore. La seconda nel 1870, attraverso l'introduzione di una nuova suddivisione del lavoro e della produzione di massa grazie all'introduzione dell'energia elettrica. La terza, nel 1969 attraverso l'uso di sistemi elettronici e informatici che automatizzarono ulteriormente la produzione. E l'ultima, che sta avvenendo ora, che vede l'utilizzo di sensori e sistemi cibernetici.

Le sfide che sta portando avanti l'industria 4.0 sono sostanzialmente tre:

1. Ridurre i tempi di consegna del prodotto;
2. Aumentare la flessibilità;
3. Ridurre il dispendio energetico delle risorse.

Smart Shop

Se prima abbiamo parlato di Smarth Things e Smart Home, con l'industria 4.0 entra in gioco il concetto di Smart Shop. Grazie a tecnologie come iBeacon che permette di effettuare un'analisi del campo geomagnetico e il tracciamento dei segnali wireless è possibile tenere traccia degli interessi dell'utente e proporgli offerte mirate per attività di marketing.

1.1.3 Big data concept

Dati raccolti, ma anche l'evidenza stessa, dimostrano che dagli anni 2000 ad oggi, il numero delle persone che utilizza internet aumenta progressivamente. Nel 2000 la popolazione mondiale raggiungeva circa sei miliardi di individui, di questi solamente circa quattro milioni utilizzavano internet. Ad oggi, esattamente cinque miliardi di persone utilizzano regolarmente internet.

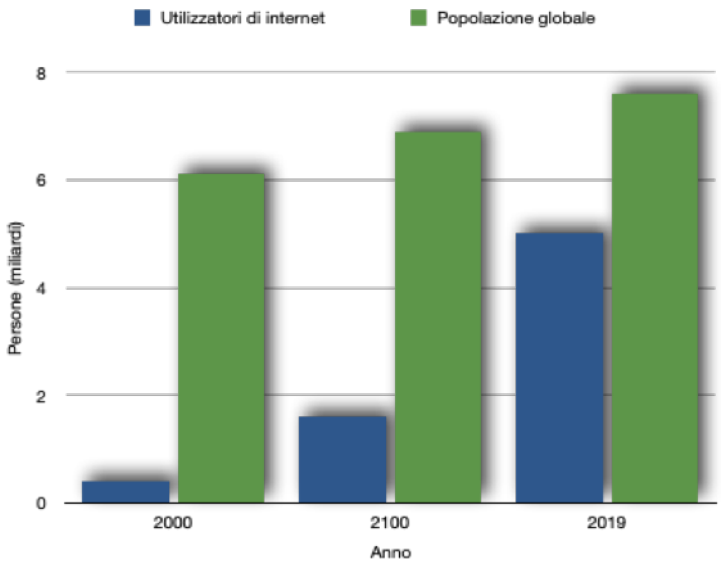

Figura 1.1: Popolazione globale e utilizzo di internet

Non solo il numero delle persone connesse ad internet sta aumentando con il passare degli anni, ma anche il numero dei dispositivi connessi ad internet o dotati di un dispositivo che permette la connessione sta aumentando. Nel 1992 il numero di dispositivi connessi ad internet era 1 milione in tutto il mondo, oggi superano addirittura i 42 miliardi.

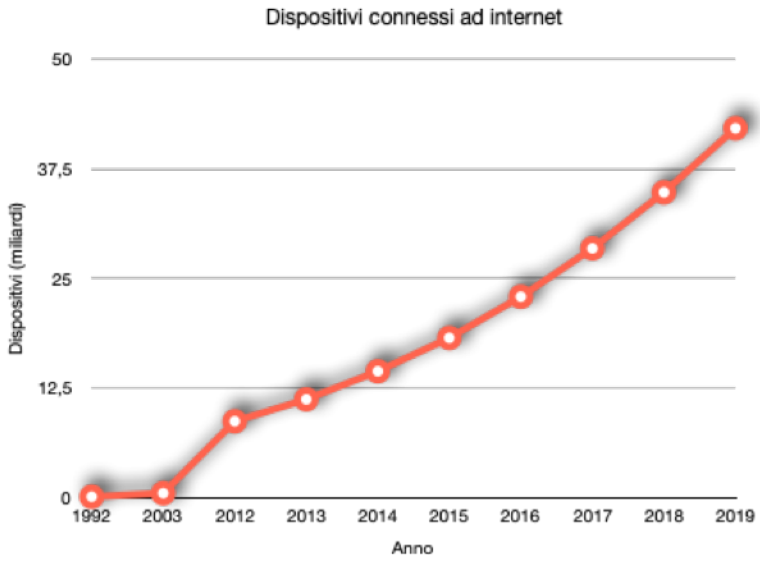

Figura 1.2: Dispositivi connessi ad internet

Dalla Figura 1.2 si può notare che nel 2012 inizia il boom dei dispostivi connessi: è proprio in quegli anni, con precisione nel 2009, che si da inizio all'era dell'IoT. L'aumento della popolazione connessa ad internet e l'aumento dei dispositivi connessi alla rete producono un conseguente aumento del numero dei dati prodotti ogni anno.

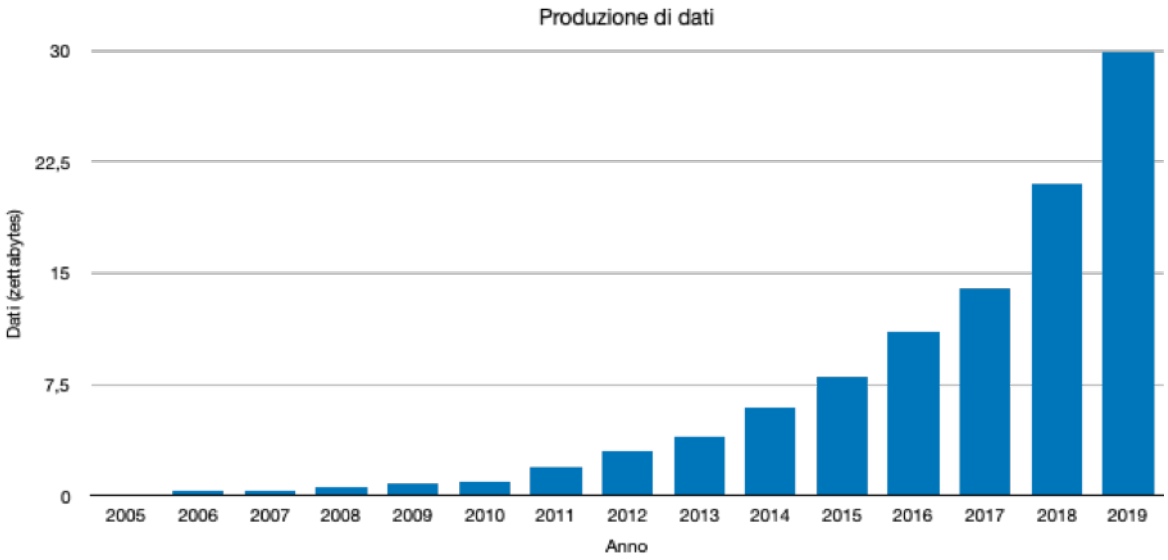

Figura 1.3: Quantità di dati prodotta

La Figura 1.3 mostra sull'asse delle ordinate la quantità di dati prodotta in zettabytes. Considerando che 1 zettabytes è 2^{70} bytes, stiamo raggiungendo delle cifre davvero impressionanti nel giro di pochissimi anni. Una vera e propria crescita esponenziale in costante aumento. Ecco dimostrata la provenienza dei cosiddetti Big Data: un enorme quantità di dati eterogenei, possibilmente correlati, complessi, non necessariamente strutturati che richiedono complesse metodologie di calcolo e analisi, basate sull'apprendimento, su grandi volumi e in tempo reale. Riuscire ad analizzare questi dati consente la creazione di valore aggiunto. Attraverso l'identificazione di caratteristiche complesse come schemi, cluster, regole, relazioni e somiglianze tra i set di dati, si possono prevedere comportamenti e scenari futuri. Tutto questo però a patto che le tecnologie utilizzate per l'analisi siano scalabili, precise e riescano a fornire risultati in tempo reale.

1.2 I componenti dell'IoT

Questo capitolo descrive le componenti principali dell'IoT che vengono utilizzate in questa tesi. In particolare vengono descritti gli standard di rete

a partire dalla struttura di LPWAN e dai sui componenti LoRa e LoRaWAN, fino ad arrivare allo standard IEEE 802.15.4[1]. Non viene trattata la tecnologia 6LoWPAN in quanto è una tecnologia utilizzata in applicazioni indoor, mentre questo progetto di tesi è finalizzato ad ambienti outdoor; non vengono neanche descritti i diversi tipi di indirizzi IP tipici del mondo IoT o i messaging protocols, in quanto l'analisi è incentrata solo a livello di rete. Infine vi è un breve approfondimento sui principali use-cases delle tecnologie LPWAN.

1.2.1 Schede di prototipazione

Come spiegato nei paragrafi precedenti, il mondo dell'IoT è fatto da oggetti intelligenti. Un oggetto intelligente è un oggetto fisico digitalmente aumentato con uno o più dei seguenti componenti: sensori intelligenti (ad esempio sensori per il rilevamento di temperatura, luce o movimento), attuatori intelligenti (ad esempio display, suoni o motori), programmi per il calcolo e infine, interfacce di comunicazione, cablate o wireless.

Sensori

Un sensore è un dispositivo che è in grado di rilevare eventi o cambiamenti all'interno dell'ambiente fisico in cui si trova. A sua volta, un sensore intelligente è un dispositivo in grado di recuperare gli input analogici dall'ambiente fisico e renderli digitali utilizzando alcune risorse integrate.

Attuatori

Un attuatore è un dispositivo che converte energia in movimento. Un attuatore intelligente è un dispositivo in grado di trasformare gli input digitali in azioni fisiche.

IoT Boards

Le piattaforme hardware, più comunemente usate per costruire prototipi, si possono dividere in microcontroller boards e single board computers. La

microcontroller board è un sistema su un chip (SoC), contenente core di elaborazione, RAM e EPROM per l'archiviazione di programmi personalizzati che vengono eseguiti sul microcontrollore. È un PCB con circuiti aggiunti. Una delle più famose schede microcontrollore è l'Arduino. Invece le single board computers (SBC) sono computer completi su un singolo circuito, compresi di microprocessori, memoria, porte di input e output e molte altre funzionalità. I computer SBC forniscono in genere una soluzione di calcolo a bassa potenza. Il più famoso SBC è il Raspberry Pi.

Figura 1.4: Arduino

1.2.2 La tecnologia LPWAN

Low Power Wide Area Networks (LPWAN) è una nuova tecnologia di comunicazione wireless che fornisce connettività wide-area per dispositivi low power e low data rate, ovvero dispositivi a bassa potenza e bassa velocità di trasmissione dati.

Questa rete può essere utilizzata in maniera complementare alle tecnologie WLAN, soprattutto per applicazioni IoT, come ad esempio per reti di sensori su aree rurali. Oppure può essere utilizzata come alternativa alle reti WLAN in alcuni casi specifici, per riduzione dei costi nella implementazione

dell'infrastruttura.

La principale caratteristica di LPWAN è quella di essere una rete a lungo raggio. Infatti permette una copertura molto estesa, nell'ordine di chilometri, e un miglior segnale di propagazione nelle applicazioni indoor. Ad esempio, è utile per mantenere una connessione in ambienti interni difficili da raggiungere come scantinati o ripostigli. LPWAN utilizza una banda di circa 1 GHz per ottenere una comunicazione affidabile e mantiene un consumo energetico molto limitato. Inoltre utilizza una tecnica di modulazione molto robusta; i ricevitori LPWAN possono raggiungere fino a -130dBm.

La seconda caratteristica di questa tecnologia è quella di essere una rete a basso consumo. Questo è dovuto principalmente alla sua topologia, per il meccanismo del duty-cycling[1] utilizzato e per il particolare protocollo utilizzato a livello MAC, che lo rende molto leggero e per il fatto che riesce a rendere gli end-devices il più economici possibile, in quando sposta ogni tipo di complessità sul back-end del sistema. La topologia della rete è quasi sempre a stella: vi è sempre collegamento diretto tra i dispositivi e il gateway, e non vi è quindi nessun tipo di meccanismo multi-hop[2]. Il meccanismo del duty-cycling permette di disattivare la ricetrasmittente della radio quando e se necessario.

Le reti LPWAN sono reti a basso costo. Il basso costo è dovuto in primo luogo al fatto che viene ridotta la complessità dell'hardware dei singoli dispositivi e in secondo luogo perché basta una singola base-station a fornire una copertura su una vasta area.

[1] Il ciclo di lavoro o ciclo di lavoro utile (a volte detto duty cycle D, dall'inglese) è la frazione di tempo che un'entità passa in uno stato attivo in proporzione al tempo totale considerato.

[2] Il routing multi-hop (o routing multihop) è un tipo di comunicazione nelle reti radio in cui l'area di copertura della rete è più ampia della gamma radio dei singoli nodi. Pertanto, per raggiungere qualche destinazione un nodo può utilizzare altri nodi come relè.

Un'altra caratteristica di questa tecnologia di rete è la scalabilità. Vengono infatti utilizzate tecniche diverse, come comunicazioni multi-canale e multi-antenna per parallelizzare le comunicazioni dei device LPWAN.

Come ogni cosa, anche la tecnologia LPWAN presenta alcuni inconvenienti. In alcuni casi il lungo raggio e la bassa potenza vengono a discapito di: una velocità di trasmissione ridotta, non maggiore di 50 kbps; una lunghezza limitata del payload [3], non più grande di 250byte; un ritardo di consegna che potrebbe essere più o meno elevato a seconda della specifica configurazione.

Tra i principali casi di utilizzo delle tecnologie LPWAN abbiamo: il monitoraggio della natura, l'agricoltura, smart greed e smart metering, monitoraggio delle infrastrutture e attività di logistica. Vedremo qualche esempio nel capitolo successivo.

Tra le tecnologie che definiscono la rete LPWAN troviamo LoRa e LoRaWAN.

1.2.3 Gli standard di rete

Gli standard utilizzati nel mondo dell'IoT a livello di rete possono essere raggruppati in due macro gruppi: quelli che aderiscono al progetto 3GPP e quelli esterni. Quelli che non aderiscono agli standard 3GPP sono: LoRa, LoRaWAN, Sigfox, RPMA. Invece quelli che aderiscono sono: LTE-M, EC-GSM. Vediamoli nel dettaglio.

LoRa

LoRa (Long Range) è una tecnologia di comunicazione dati wireless brevet-

[3]Payload è un termine mutuato dalla lingua inglese che può essere tradotto come "carico utile" e in una trasmissione informatica, indica la parte di dati trasmessi destinata all'utilizzatore, in contrasto con gli altri elementi del messaggio necessari a far funzionare il protocollo di comunicazione.

tata e sviluppata da Cycleo di Grenoble, in Francia, ed acquisita da Semtech[4] nel 2012. LoRa è un protocollo di comunicazione wireless a lungo raggio che compete con altre reti wireless a bassa potenza (LPWAN) come NB-IoT o LTE Cat M1. Rispetto a quelle, LoRa raggiunge la sua connettività a lungo raggio, 100 km possibili, scambiando dati ad alta velocità. Poiché la sua velocità di trasmissione dati è inferiore a 50kbps e poiché LoRa è limitato dal ciclo di lavoro e da altre restrizioni, è adatta ad applicazioni non realtime e ad applicazioni tolleranti a queste restrizioni. Una di queste limitazioni è il fatto che LoRa contiene solo un protocollo a livello link layer. Questa limitazione viene superata da LoRaWAN.

LoRa definisce il livello fisico. I ricetrasmettitori LoRa sono mappati su diverse frequenze Sub-GhZ, basate su regolazioni nazionali. Il livello fisico di LoRa è basato sulla Chirp Spread Spectrum (CSS) modulation. In CSS la potenza del segnale aumenta o diminuisce con il tempo; la banda viene sfruttata completamente per la trasmissione dei segnali; ed è resistente al Doppler Shift[5].

LoRaWAN

LoRaWAN, a differenza di LoRa, include anche il livello di rete, quindi è possibile inviare le informazioni a qualsiasi Base Station già collegata a una piattaforma cloud. I moduli LoRaWAN possono funzionare anche a frequenze diverse semplicemente collegando l'antenna giusta alla sua presa. LoRaWAN è una rete a bassa potenza, la sua modulazione occupa generalmente 125 KHZ

[4]Semtech è un fornitore leader di semiconduttori analogici e a segnale misto ad alte prestazioni e algoritmi avanzati. Attraverso le sue piattaforme tecnologiche avanzate, Semtech offre applicazioni mission critical per tre dei mercati in più rapida crescita del settore: Internet of Things, Data Center e Mobility. [6]

[5]L'effetto Doppler è un fenomeno fisico che consiste nel cambiamento apparente, rispetto al valore originario, della frequenza o della lunghezza d'onda percepita da un osservatore raggiunto da un'onda emessa da una sorgente che si trovi in movimento rispetto all'osservatore stesso.

di banda. È una rete ad alta sensibilità, gira attorno ai 140 dBm. Consente una comunicazione a lungo raggio, fino a 15km e ha una forte penetrazione indoor, fino a 20dB di penetrazione interna. LoRaWAN è resistente all'effetto Doppler e all'eventuale indebolimento del segnale.

LoRaWAN definisce il livello MAC e l'architettura generale della rete. È uno standard libero, non proprietario, definito da LoRa Alliance[6]. L'architettura LoRaWAN è formata da quattro componenti di rete: end-devices, gateways, network server e application server. Gli end-device sono gli "oggetti" IoT, come ad esempio: pet tracking, smoke alarm, water meter, trash container, vending machine o gas monitoring. Questi end-device si connettono al server di rete per mezzo di gateway e all'interno della rete sono presenti i server delle singole applicazioni ai quali accedono per il recupero e l'invio dei dati.

Al livello MAC viene utilizzato un semplice schema di accesso casuale chiamato ALOHA. Grazie a questo meccanismo più end-device possono comunicare allo stesso tempo utilizzando diverse frequenze o diversi sistemi. Gli end-device non sono associati a nessun gateway specifico; i dati trasmessi possono essere ricevuti da un qualsiasi gateway e a più di uno. Gli end-device possono richiedere un ACK[7] di conferma del messaggio inviato. In un protocollo ALHOA-like: gli intervalli di trasmissione hanno lunghezza fissa; i dati vengono trasmessi sul canale ogni volta che la radio è attiva dopo un numero casuale di slot di inattività.

[6]La LoRa Alliance è l'alleanza tecnologica in più rapida crescita. E' una associazione senza scopo di lucro di oltre 500 aziende associate, impegnata a consentire l'implementazione su vasta scala di IoT con reti a bassa potenza (LPWAN), attraverso lo sviluppo e la promozione dello standard aperto LoRaWAN. I membri beneficiano di un vivace ecosistema di collaboratori attivi che offrono soluzioni, prodotti e servizi che creano opportunità di business nuove e sostenibili. [7]

[7]ACK, in ambito telecomunicazioni e informatico, è il simbolo che identifica un segnale di Acknowledge emesso in risposta alla ricezione di un'informazione completa.

I gateway ricevono i dati da più client (end-device connessi tramite LoRa) e li inoltrano ad un server di rete tramite connessione cablata. Il gateway deve avere una capacità molto elevata per mantenere una connessione a lungo raggio e per sostenere un volume elevato di dati che gli arrivano in input. I gateway supportano per questo motivo la tecnologia multicanale, ovvero riescono ad ascoltare più canali simultaneamente e decodificare pacchetti ricevuti da diversi end-device LoRa based.

Il compito del server di rete è quello di applicare filtri su eventuali pacchetti duplicati e di decifrare i messaggi per reindirizzarli al giusto application server. Quest'ultimo si preoccupa invece di eseguire l'elaborazione dei dati specifici dell'applicazione.

Infine, LoRaWAN utilizza due livelli di sicurezza: uno a livello di rete e uno per l'applicazione. Prima che un end-device possa comunicare tramite rete LoRaWAN deve essere attivato; deve quindi avere: un Device Address (DevAddr), un Network Security key (NwkSkey) e una Application Security key (AppSKey). Il DevAddr deve essere unico, che lo identifichi all'interno della rete, a 32 bit. Questo indirizzo deve poi essere condiviso, quindi noto al server di rete e all'application server. La NwkSkey è una chiave crittografata a 128 bit, condivisa tra l'end-device e il server di rete; il possesso di questa chiave garantisce l'integrità del messaggio all'interno dell'infrastruttura di rete. Infine, la AppSKey è una chiave crittografata a 128 bit, condivisa tra l'end-device e l'application server; questa chiave viene utilizzata per crittografare e de-crittografare l'oggetto del messaggio. Viene utilizzato questo meccanismo per garantire riservatezza all'interno della rete.

Con questo sistema di sicurezza l'end-device riesce a recuperare i dati utili tramite due metodologie: la prima consiste nell'inviare una JOIN Request all'application server con un identificativo univoco a livello globale (DevEUI),

un identificativo a livello di applicazione (AppEUI) e un application key (AppKey); la seconda consiste in una chiamata di tipo JOIN REPLAY dalla server application, in questo caso sono necessarie una DevAddr, una NwkSKey e una AppSKey.

Sigfox

Questa tecnologia è caratterizzata da uno strato fisico basato su una modulazione wireless della banda ultra sottile. Ha un bassissimo rendimento, inferiore a 100bps, una bassa potenza di segnale ed è a corto raggio, non supera infatti i 50km di estensione. È possibile applicarla a dispositivi utilizzati per inviare al massimo 140 messaggi al giorno per dispositivo. Supporta il roaming dei dati.

RPMA

RMA è l'acronimo di Random Phase Multiple Access. È una tecnologia a bassa potenza, ma riesce a coprire un'ampia area. È uno standard di rete utilizzato solo per comunicazioni M2M e consente di utilizzare un'ampia banda, fino a 2,4 GHz.

LTE-M

È una tecnologia ottimizzata per il mondo dell'IoT. LTE è caratterizzata dall'essere a basso consumo energetico, di facile implementazione e ha un basso costo computazionale complessivo. Inoltre, questa tecnologia ha un'ottima copertura, infatti prevede un raggio d'azione di oltre 10km e ad una velocità di addirittura 1Mbps.

EC-GSM

Sfrutta la rete 2G e la adatta per fornire un maggiore grado di efficienza, connettività e copertura per oggetti del mondo dell'IoT. Garantisce una lunga durata della batteria di chi ne usufruisce, fino a 10 anni di autonomia con batterie da 5Wh. Offre una velocità di scambio dati variabile: da circa

350 bps a 70 kbps in base al livello di copertura. Permette una connessione stabile fino ad oltre 50.000 dispositivi.

1.2.4 IEEE 802.15.4

Per regolare il livello fisico ed il livello mac delle reti LoRa, LoRaWAN e in generale delle reti WPANs (Wireless Personal Area Networks[8]) è stato concepito lo standard IEEE 802.15.4.

IEEE 802.15.4 è uno standard sviluppato per fornire una struttura e gli strati inferiori nel modello OSI per reti di connettività wireless a basso costo e bassa potenza. La bassa potenza è uno degli elementi chiave di 802.15.4 poiché viene utilizzata in molte aree in cui i sensori remoti devono funzionare a batteria. IEEE 802.15.4 non ha lo scopo di competere con i sistemi orientati all'utente finale più comuni come IEEE 802.11 dove i costi non sono così importanti e sono richieste velocità più elevate e la potenza potrebbe non essere altrettanto critica. Invece, IEEE 802.15.4 fornisce comunicazioni a costi molto bassi di dispositivi vicini con un'infrastruttura sottostante minima o nulla. Il concetto di IEEE 802.15.4 è quello di fornire comunicazioni su distanze fino a circa 10 metri e con velocità di trasferimento dati massima di 250 kbps. Lo standard IEEE 802.15.4 ha subito un certo numero di versioni. Oltre a questo ci sono un certo numero di varianti dello standard IEEE 802.15.4 per far fronte a diverse forme di livello fisico. La tecnologia IEEE 802.15.4 viene utilizzata per una varietà di diversi standard di livello superiore. In questo modo i livelli fisici e MAC di base sono già definiti, consentendo agli strati superiori di essere forniti dal singolo sistema in uso. Anche se lo standard IEEE 802.15.4 potrebbe non essere noto come alcuni

[8] In telecomunicazioni una rete personale, in lingua inglese Personal Area Network, in sigla PAN, è una rete informatica utilizzata per permettere la comunicazione tra diversi dispositivi (telefono, PC tascabile, ecc.) vicini a un singolo utente. I singoli dispositivi possono anche non appartenere all'utente in questione. Il raggio di azione di una PAN è tipicamente di alcuni metri. [8]

standard e sistemi di livello superiore come Zigbee che utilizzano la tecnologia IEEE 802.15.4 come sistema di livelli inferiori sottostanti, è comunque molto importante. Si estende su una varietà di sistemi diversi, e come tale fornisce un nuovo approccio, fornendo solo gli strati inferiori e consentendo ad altri sistemi di fornire gli strati superiori che sono adattati per l'applicazione pertinente.

Le bande di frequenza IEEE 802.15.4 si allineano con le bande radio prive di licenza disponibili in tutto il mondo. Tra le bande disponibili, la banda a 2,4 GHz (2 400 MHz) è la più utilizzata in considerazione del fatto che è disponibile a livello globale e ciò porta a molte economie di scala. Le principali caratteristiche di questo standard sono quindi: la bassa complessità, la bassa potenza e la bassa velocità di trasmissione dati tra dispositivi fissi e mobili. In particolare la Spectrum Bands[9] ha una dimensione di 2.4 GHz, 915 MHz o 868 MHz. La velocità di trasmissione dati raggiunge appena i 250 Kbs (2.4 GHz). Il raggio d'azione copre fino a poco meno di 30m. il canale ha una dimensione di 2.4 GHz e utilizza il protocollo di accesso CSMA/CA[10].

[9]Nelle telecomunicazioni con il termine banda radio (o spettro radio) si indica la sezione dello spettro elettromagnetico utilizzata per la trasmissione di dati e informazioni. Essa identifica la suddivisione spettrale del mezzo trasmissivo per poter essere utilizzato da più operatori e utenti per l'effettuazione di servizi e che deve essere adeguatamente ripartito tra di essi al fine di evitare conflitti di utilizzazione o interferenza tra segnali radio di più sorgenti. [9]

[10]In telecomunicazioni CSMA/CA (acronimo inglese di Carrier Sense Multiple Access with Collision Avoidance, ovvero Accesso Multiplo tramite Rilevamento della Portante con Evitamento delle Collisioni) è un protocollo di accesso multiplo che utilizza il rilevamento della portante ma in cui i nodi tentano di evitare a priori il verificarsi di collisioni. Una volta iniziata, la trasmissione prosegue fino al termine del pacchetto. È di particolare importanza nei casi in cui il rilevamento delle collisioni non è realizzabile (o la cui eventuale realizzazione non sarebbe affidabile), come avviene in pratica nel campo delle reti senza fili. Infatti, oltre all'esistenza del problema dei nodi nascosti, il ricevitore radio di un nodo in una rete senza fili non può rilevare in modo affidabile eventuali trasmissioni provenienti da altri nodi mentre il relativo trasmettitore è attivo. [10]

Lo scopo del livello MAC IEEE 802.15.4 è di fornire un'interfaccia tra il PHY o livello fisico e il livello dell'applicazione. Come IEEE 802.15.4 non specifica un livello applicativo, questo è generalmente un sistema applicativo come Zigbee, RF4CE, MiWi, ecc. Il MAC IEEE 802.15.4 fornisce l'interfaccia al livello dell'applicazione utilizzando due elementi:

- Servizio di gestione MAC: si chiama MAC Layer Management Entity, MLME. Fornisce le interfacce di servizio attraverso le quali è possibile chiamare o accedere alle funzioni di gestione dei livelli. Il MAC MLME IEEE 802.15.4 è anche responsabile del controllo di un database di oggetti per il livello MAC. Questo database viene definito base di informazioni PAN o PIB del livello MAC. L'MLME ha anche accesso ai servizi MCPS per le attività di trasporto dei dati.

- Servizio dati MAC: questo si chiama MAC Common Port Layer, MCPS. Questa entità all'interno del MAC IEEE 802.15.4 offre servizi di trasporto dati tra i MAC peer.

Esistono due forme principali di topologia di rete che possono essere utilizzate all'interno di IEEE 802.15.4. Queste topologie di rete possono essere utilizzate per diverse applicazioni e offrono diversi vantaggi. Le due topologie di rete IEEE 802.15.4 sono:

- Topologia a stella: il nome indica che il formato di avvio per una topologia di rete IEEE 802.15.4 ha un nodo centrale chiamato coordinatore PAN con cui comunicano tutti gli altri nodi.

- Topologia di rete peer-to-peer: in questa forma di topologia di rete, c'è ancora ciò che viene definito un coordinatore PAN, ma le comunicazioni possono anche avvenire tra diversi nodi e non necessariamente tramite il coordinatore.

Esistono inoltre tre diversi tipi di dispositivi in una rete:

1. FFD: Full Function Device - un nodo con livello completo di funzionalità. Può essere utilizzato per inviare e ricevere dati, ma può anche instradare i dati da altri nodi.

2. RFD: Reduced Function Device - un dispositivo che ha un livello ridotto di funzionalità. Tipicamente è un nodo finale che può essere tipicamente un sensore o interruttore. Gli RFD possono comunicare solo con gli FFD in quanto non contengono funzionalità di routing. I dispositivi possono essere di alimentazione molto bassi perché non hanno bisogno di instradare altro traffico e possono essere messi in modalità di sospensione quando non sono in uso. Questi RFD sono spesso noti come periferiche figlie in quanto necessitano di altri dispositivi principali con cui comunicare.

3. Coordinatore: questo è il nodo che controlla la rete IEEE 802.15.4; è una forma speciale di FFD. Oltre alle normali funzioni FFD, imposta anche la rete IEEE 802.15.4 e funge da coordinatore o gestore della rete.

Queste definizioni sono state originariamente generate per l'uso in Zigbee, ma il loro uso è stato ora introdotto con la terminologia di rete IEEE 802.15.4. [1]

1.3 Smart Agriculture

Uno dei principali casi d'uso delle tecnologie LPWAN è nel mondo dell'agricoltura smart. Contesto applicativo principale di questo progetto di tesi.

Tutte le dimensioni della produzione agricola sono interessate dalle nuove tecnologie: gestione del suolo, dell'acqua, delle colture, protezione delle piante, gestione degli allevamenti, salute degli animali, automazione. Lo smart farming permette alle aziende di essere più efficaci, ottimizzare i processi, aumentare i rendimenti e minimizzare il loro impatto sull'ambiente.

Il futuro dell'agricoltura non si basa solo sullo sviluppo dei macchinari e dei robot più performanti, ma anche su sistemi di sensori intelligenti che seguono con precisione determinati parametri e ne interpretano l'evoluzione. Le nuove tecnologie permetteranno, in futuro, di offrire ai campi coltivati, alle piante e agli animali proprio quello di cui avranno bisogno e al momento giusto. Lo smart farming promette quindi una maggiore efficacia, un uso ridotto di prodotti fitosanitari, il rilevamento precoce delle malattie negli animali, prodotti di maggiore qualità e tanto altro ancora.

Le nuove tecnologie rivoluzioneranno quindi l'agricoltura, ma anche tutta la catena alimentare. Tuttavia lo smart farming non può sostituire l'esperienza, la percezione e la formazione professionale degli agricoltori. I numerosi punti di forza dell'evoluzione tecnologica non impediscono ai ricercatori (ad esempio quelli di Agroscope[11] e a tutti i loro partner) di mantenere l'attività umana al centro della loro attenzione.

[11] Agroscope è il centro di competenza della Confederazione Elvetica per la ricerca nel settore agroalimentare e ambientale; è aggregato all'Ufficio federale dell'agricoltura (UFAG). [11]

Figura 1.5: Smart Agriculture

1.3.1 Scenario

Smart Agriculture, Smart Farming, Precision Agriculture sono tutti termini validi che stanno a significare cose a volte diverse ma legate allo stesso tema, ossia le soluzioni applicative volte al monitoraggio, alla gestione e all'ottimizzazione di diversi processi relativi all'agricoltura. Lo scopo principale è quello di aumentare la produttività, non solo a livello di business. Il parallelismo con quanto si sta facendo in campo industriale con le stesse nuove tecnologie è evidente. Tanto che, se nel manufacturing si parla di Quarta Rivoluzione industriale, qui si parla di Terza Rivoluzione Verde (le prime due sono l'incrocio delle specie vegetali e l'uso della genetica nelle coltivazioni). Uno degli elementi chiave nelle applicazioni di Smart Farming è la raccolta di più informazioni possibili sul terreno. La logica è quella dei sensori disseminati dovunque sia necessario, sfruttando il fatto che grazie alla miniaturizzazione della componentistica elettronica è possibile avere sensori

periferici di dimensioni e consumo contenuto per tutte le variabili ambientali d'interesse: temperatura, umidità, concentrazione di elementi chimici nel terreno, intensità e direzione del vento, precipitazioni, eccetera. [3]

In agricoltura da sempre si cerca di capire ed interpretare i segnali che provengono dal terreno, dalle colture e dal clima. Per interpretare questi segnali ci si può basare sull'esperienza degli agricoltori e sulle conoscenze scientifiche degli agronomi, ma per quanto attendibili la maggior parte delle informazioni sono presunte. La Smart Agriculture, al di là delle diverse tecnologie utilizzate, crea un canale di comunicazione diretto con il terreno e con le colture. Oggi finalmente è possibile calcolare la quantità di umidità del terreno grazie a particolari sensori. È il terreno stesso che fornisce il dato, non più l'interpretazione del contadino. Tutto comincia da questi dati finalmente "diretti" e raccolti in tempo reale. Poi, a seconda dell'esigenza dell'agricoltore è possibile elaborarli nei modi più diversi per poter fornire applicazioni e supporti utili.

Droni e sensori IoT raccolgono e storicizzano dati sulle condizioni del suolo, acidità e temperatura del terreno, l'esposizione solare, il livello di crescita delle coltivazioni, i danni causati da siccità, grandine e alluvioni. Il sistema per l'Internet of Things integra questi dati con quelli sulle condizioni meteo e li trasforma in statistiche. Tutte queste informazioni e le analisi predittive che ne conseguono sono utilissime per gli agricoltori, perché li aiutano a usare solo le risorse strettamente necessarie (acqua, fertilizzanti, pesticidi etc.) dove c'è veramente bisogno (aree del campo poco irrigate, piante deboli o malate).

Smart Agriculture significa quindi migliorare qualità, quantità e sostenibilità ambientale ed economica della produzione agricola.

I sensori IoT, posizionati sul territorio da coltivare, raccolgono dati in tempo reale sull'umidità del terreno, sullo stadio di crescita delle piante e sul clima.

Nelle stalle, i sensori monitorano quantità e qualità del mangime. Nei silos, tengono sotto controllo i parametri che preservano le proprietà organolettiche dei materiali conservati. In caso di cambiamenti repentini, il sistema IoT li anticipa e avvisa immediatamente l'agricoltore.

Gli agricoltori possono usare i loro Smartphone per monitorare da remoto attrezzature, coltivazioni, bestiame e ottenere statistiche sull'alimentazione dei loro animali e sulla produzione di latte, uova e carne. Possono anche usare le tecnologie IoT per compiere analisi predittive su coltivazioni e animali.

I droni sono diventati uno strumento indispensabile per sorvegliare i terreni agricoli e raccogliere dati sulle coltivazioni. Possono acquisire immagini, fare rilevamenti sullo stato di salute delle piante, sulle zone del terreno poco o troppo irrigate e spargere prodotti sul suolo. [3]

Figura 1.6: Drone

1.3.2 Inevitabile conversione alla Smart Agricolture

Secondo la FAO, l'Organizzazione delle Nazioni Unite per l'alimentazione e l'agricoltura, entro il 2050 la popolazione mondiale arriverà a oltre nove miliardi. Quindi, per riuscire a sfamare tutti, dovremo produrre il 70% di cibo in più. A fronte di risorse sempre più scarse, condizioni metereologiche estreme, la popolazione in aumento e la riduzione delle superfici coltivabili, il settore agricolo ha di fronte a sé una sfida non indifferente. La soluzione è convertirsi alla Smart Agriculture.

Come già accade nelle città, con la rivoluzione Smart City, per affrontare in modo sostenibile l'aumento della popolazione mondiale e i cambiamenti climatici, molte imprese agricole stanno già adottando soluzioni per l'Internet of Things. Si tratta di sistemi che partendo dai dati su suolo, meteo, coltivazioni ed altri parametri, restituiscono informazioni preziosissime per gli agricoltori, per esempio, dove e quando irrigare, in quali zone spargere fertilizzanti, quali piante stanno per ammalarsi e quali possono essere raccolte. L'incontro fra Internet delle cose e agricoltura si traduce in riduzione degli sprechi e ottimizzazione delle risorse.

Smart Agriculture è l'insieme di applicazioni che aiutano le aziende agricole a monitorare, gestire e ottimizzare più efficacemente la loro attività, sfruttando solo le risorse effettivamente necessarie, attraverso:

- Mappatura del territorio e monitoraggio delle colture con sensori e droni.
- Irrigazione selettiva.
- Monitoraggio dei parametri ambientali: microclima, compost, tossicità.
- Tracciatura e rilevamento degli animali.
- Monitoraggio delle condizioni metereologiche.

È un'agricoltura "intelligente" che unisce l'esperienza secolare di contadini e allevatori alle informazioni ottenibili solo con tecnologie avanzate.

1.3.3 Esempi di sistemi IoT applicati all'agricoltura

Dalla vigna al bestiame, ecco alcuni esempi di come la Smart Agriculture può migliorare le attività agricole.

WaterPlan

SWAP (Smart Watering Planner) è il prototipo realizzato da Soonapse[12], un'applicazione cloud per l'ottimizzazione "predittiva" dell'uso dell'acqua in agricoltura. Le innovazioni che presenta questo progetto sono molte. La principale è quella di fornire delle "strategie" di utilizzo dell'acqua per ottimizzarne il consumo, distribuendolo in modo ottimale (per la coltura e per i costi dell'agricoltore) nella finestra temporale di cinque giorni in cui le previsioni meteo sono attendibili. Gli esperimenti effettuati con questo progetto mostrano un risparmio fra il 30% ed il 50%.

I principali guadagni ottenuti da questo progetto sono: risparmio di denaro e ottimizzazione dell'utilizzo dell'acqua. Il risparmio di denaro avviene grazie al fatto che grazie a questo sistema non è più necessario un esperto che dica la composizione e lo stato del terreno. Specialisti di questo genere possono costare fino a 3.000 euro al mese e non tutti possono permetterselo. SWAP in poco tempo calcola automaticamente la curva di assorbimento idrico del terreno e definisce il bilancio idrico. Invece l'ottimizzazione dell'utilizzo dell'acqua avviene in quanto SWAP è in grado di generare una vera e propria agenda che stabilisce i turni di irrigazione. In un azienda agricola risparmia-

[12]8. Soonapse è una startup che nasce dalla grande esperienza dei suoi soci fondatori, con una visione innovativa che si basa sui molti progetti di ricerca effettuati, sia nazionali che europei, e sulla profonda frequentazione di quella fucina di sperimentazione collaborativa internazionale che è l'Open Source, dove sono nate tutte le tecnologie e le metodologie che oggi rendono Internet l'esperienza planetaria che conosciamo. [12]

re anche solo due o tre turni di irrigazione, sui 14/15, può fare una grande differenza a livello di profitto.

SWAP è stato premiato come miglior progetto italiano di Smart Agriculture. [2]

Rumi Watch

Un altro esempio di progetto è RumiWatch realizzato in collaborazione con Vetsuisse dell'Università di Berna e l'azienda Itin Hoch. Grazie a una cavezza e a un pedometro muniti di un sensore, vengono raccolti dati sul comportamento di una mucca, sui suoi spostamenti e sulla sua alimentazione. Questi dati vengono poi analizzati e interpretati dal programma RumiWatch. L'obiettivo dei ricercatori è far sì che il programma possa inviare una notifica all'agricoltore quando uno dei suoi animali adotta un comportamento anomalo. Ciò permetterebbe di curare prima un animale malato e quindi di limitare la somministrazione di antibiotici: un vantaggio per il benessere degli animali e per la lotta contro la resistenza agli antibiotici. Inoltre questi dati permettono di trovare la formula migliore per ottimizzare i prodotti di alta qualità, come eccellenti formaggi.

Piattaforma web Agrometeo

Questa piattaforma viene utilizzata per monitorare l'evoluzione della peronospora, per monitorare gli insetti, lo sviluppo fenologico del grano e tanto altro ancora. Tutte le informazioni utili alla protezione delle piante sono raggruppate su un sito destinato agli agricoltori. Le varie applicazioni servono a prendere decisioni per gestire i problemi fitosanitari in grandi colture, viticolture e frutticolture. Grazie alla condivisione dei dati meteorologici, delle conoscenze frutto della ricerca e di osservazioni, i ricercatori che hanno contribuito a questo esperimento sono riusciti a sviluppare modelli di previsione e algoritmi, che fanno di questa piattaforma un vero strumento di lavoro per lottare con maggiore efficacia contro le malattie e i parassiti. Anche in que-

sto caso la tecnologia permette di limitare l'utilizzo di concimi preservando la natura e permettendo di diminuire le spese degli agricoltori.

Vigneti intelligenti

Nella Mosel Valley in Germania, le aziende vitivinicole usano la tecnologia IoT per raccogliere dati sull'umidità dell'aria e del terreno, la temperatura, l'intensità dei raggi solari e sullo stato di salute della vite. In questo modo possono compiere analisi predittive e capire qual è il momento (giorno e ora) migliore per la vendemmia, usando i pesticidi solo dove necessario.

Mucche connesse

Moocall è una soluzione IoT pensata per gli allevamenti di bovini: un'applicazione per monitorare le mucche gravide. Rileva le contrazioni e quando la mucca ha le doglie invia un SMS all'allevatore per informarlo del parto imminente.

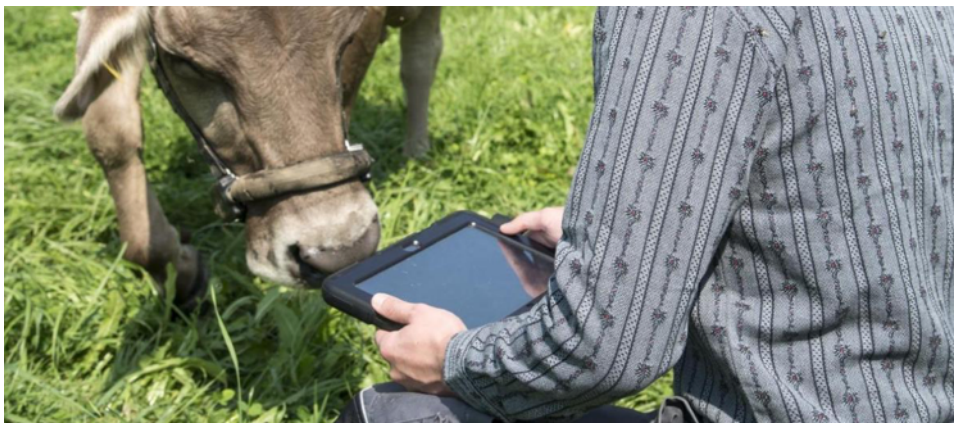

Figura 1.7: Mucche connesse

Smart Agricolture in Italia

Rispetto ad altri Paesi, in Italia la Smart Agriculture e Agrifood ha trovato terreno fertile tra alcuni imprenditori agricoli "illuminati". Le applicazioni riguardano soprattutto sistemi IoT per i vigneti e per il monitoraggio delle

mucche da latte e dei prodotti lattiero caseari DOP. Stando all' Osservatorio Smart Agrifood del Politecnico di Milano, però, la diffusione dell'agricoltura intelligente stenta a decollare a causa degli investimenti troppo elevati per i singoli agricoltori e al limitato coinvolgimento di tutta la filiera e delle politiche agricole. La sfida è quella di passare quindi da iniziative a sé stanti alla progettazione di iniziative sistemiche, per arrivare ad un Sistema Paese di Agricoltura Intelligente. L'agroalimentare Made in Italy infatti mantiene ancora oggi un primato all'estero in termini di prestigiosità, ma c'è bisogno di un continuo investimento in innovazione per evitare che il fermento attuale possa svanire presto.

Capitolo 2

UAV e reti di sensori

Nel corso degli ultimi anni il mondo dell'IoT sta prendendo sempre più piede e una delle principali aree di interesse di questo settore si basa sull'utilizzo di sensoristica e droni. Stanno quindi aumentando gli studi per cercare di sfruttare le reti di sensori e droni in diversi ambiti, come nella sorveglianza di zone non accessibili, nell'esplorazione di zone ignote o pericolose, nell'agricoltura o nelle reti stradali. Proprio per questo motivo parecchi studi sono stati fatti per migliorare la qualità della rete di sensori in modo da ottimizzare il processo di recupero dei dati e la durata della vita della rete. Proprio come è stato fatto in questa tesi. In questo capitolo verranno quindi riportati diversi studi aventi come tema principale l'utilizzo degli UAV (Unmanned Aerial Vehicle) nelle reti WSN (Wireless Sensor Network). Questi studi sono stati presi da diversi articoli e verranno raggruppati a seconda delle problematiche analizzate, delle soluzioni proposte e delle conclusioni ottenute.

2.1 Reti WSN-UAV

Con l'innovazione tecnologica si è pensato di sfruttare delle reti di sensori, connessi tramite tecnologia wireless, per l'ottimizzazione del processo del recupero dati. Con il termine Wireless Sensor Network (WSN) si indica

quindi una determinata tipologia di rete che, caratterizzata da una architettura distribuita, è realizzata da un insieme di dispositivi elettronici (sensori) autonomi in grado di prelevare dati dall'ambiente circostante e di comunicare tra loro. [13]

Una rete WSN si compone di due o più sensori e uno o più droni, fino a formare dei veri e propri sciami di droni collaborativi. Un drone è un apparecchio volante caratterizzato dall'assenza del pilota a bordo. Il suo volo è controllato dal computer a bordo del mezzo aereo oppure tramite il controllo remoto di un navigatore o pilota, sul terreno o in un altro veicolo. Il loro utilizzo è ormai consolidato per usi militari ed è crescente per applicazioni civili, ad esempio in operazioni di prevenzione e intervento in emergenza incendi, per usi di sicurezza non militari, per sorveglianza di oleodotti, con finalità di telerilevamento o ricerca in generale. I costi economici sono decisamente minori rispetto ai mezzi aerei tradizionali. Queste apparecchiature volanti senza pilota sono noti anche con l'acronimo di UAV. [14]

Nelle reti di sensori wireless (WSN), l'utilizzo del veicolo aereo senza pilota (UAV) come collettore di dati mobile, provenienti dai sensori a terra, è una tecnica a basso consumo energetico utilizzata per prolungare la durata della rete. In particolare, poiché l'UAV può spostarsi sequenzialmente vicino a ciascuno dei sensori, si riduce la distanza di collegamento durante il processo di raccolta dati. Ne consegue un risparmio energetico di trasmissione del singolo sensore. [15]

2.2 Caratteristiche tecniche

Negli articoli [16] [17] [18] vengono analizzate caratteristiche un po' più tecniche delle reti WSN-UAV.

In particolare *"Mobile Networking with UAVs: Opportunities and Challen-*

ges" [16] descrive tre particolari tipologie di reti WSN, *"UAV-Based Data Communication in Wireless Sensor Networks: Models and Strategies"* [17] fa un elenco degli algoritmi utilizzabili per la raccolta dati analizzandone caratteristiche e tipologie di UAV in grado di supportare tali algoritmi, invece in *"Effects of UAV Mobility Patterns on Data Collection in Wireless Sensor Networks"* [18] analizza vari modelli di mobilità dell'UAV.

2.2.1 Tipologie di reti WSN

Secondo quanto analizzato nell'articolo [16] le reti WSN sono raggruppabili in tre tipologie:

1. MANET (Mobile Ad-Hoc Networking): insieme di nodi mobili collegati tramite reti wireless come IEEE 802.11.a/b/g/n, 802.16, ecc. i nodi non fungono solo da host, ma anche da router. I nodi sono liberi di muoversi e di conseguenza la topologia della rete cambia nel tempo. A causa della natura dei collegamenti wireless la connessione tra i nodi a volte si perde. Inoltre, i nodi che compongono una rete MANET, essendo mobili, sono di piccole capacità e aventi energia limitata. La mobilità dei nodi rende difficile mantenere questo tipo di rete, anche in termini di dispendio energetico e quindi della durata della vita della rete.

2. VANET (Veicular Ad-Hoc Network): stesso principio della rete MANET ma vengono utilizzati dei veri e propri veicoli come nodi mobili come ad esempio auto, ambulanze, autopompe e carri armati.

3. FANET (Flying Ad-Hoc Networks): stesso principio della rete MANET ma vengono utilizzati veicoli aerei come nodi mobili come gli UAV.

Utilizzando una rete FANET le operazioni, come la raccolta e il trasferimento dei dati, vengono completate più velocemente proporzionalmente al numero di UAV presenti nella rete. Infatti, maggiore è il numero degli UAV e minore è il tempo per portare a termine gli obiettivi. Un altro vantaggio nell'utilizzo

degli UAV è il risparmio economico. Nodi mobili senza pilota e di piccole dimensioni sono indubbiamente più economici e più mantenibili rispetto a veicoli di grosse dimensioni come carri armati o autopompe. L'utilizzo di UAV aumenta la durata della rete, non solo per il fatto che permette ad un singolo sensore di consumare meno energia, ma permette anche di mantenere la rete attiva più a lungo nonostante un nodo si scarichi, facendo esso stesso da arco di connessione con gli altri nodi della rete. [16]

"Mobile Networking with UAVs: Opportunities and Challenges" afferma quindi che i vantaggi della rete FANET sono molteplici, sia in termini di throughput[1], che di delay[2], economici ed energetici.

2.2.2 Algoritmi per la raccolta dati

"UAV-Based Data Communication in Wireless Sensor Networks: Models and Strategies" [17] presenta diversi modelli e strategie per la raccolta dati, al fine di ridurre in modo significativo il consumo energetico, le interferenze in radiofrequenza[3] e i problemi di routing[4].

[1]Nell'ambito delle telecomunicazioni, si intende per throughput di un canale di comunicazione la sua capacità di trasmissione "effettivamente utilizzata". [45]

[2]La latenza, in informatica e telecomunicazioni, è definita come l'intervallo di tempo che intercorre fra il momento in cui arriva l'input/segnale al sistema e il momento in cui è disponibile il suo output. In altre parole, la latenza non è altro che una misura della velocità di risposta di un sistema. [46]

[3]Una radiofrequenza, nota anche con la sigla RF, indica generalmente un segnale elettrico o un'onda elettromagnetica ad alta frequenza che si propaga nello spazio. [47]

[4]L'instradamento, nel campo delle reti di telecomunicazione, è la funzione di un router che decide su quale porta o interfaccia inviare un pacchetto ricevuto. [48]

ALGORITMO	DESCRIZIONE	CARATTERISTICHE	TIPI DI UAV ADATTI
Constant Speed UAV (CSU)	La velocità UAV viene mantenuta costante all'interno e all'esterno del range del nodo.	Questo è il modello più semplice. Tuttavia, può subire un ritardo nel trasferimento dei dati end-to-end più lungo.	UAV semplice a velocità costante.
Variable Speed UAV (VSU)	La velocità UAV è variabile, con una velocità tipicamente inferiore mentre si trova nell'intervallo del nodo e una maggiore velocità tra i nodi e l'intervallo del nodo esterno.	Modello leggermente più complesso. L'UAV si sposta a una velocità più veloce tra i nodi per ridurre il ritardo del trasferimento dati end-to-end.	Richiede un UAV con capacità di velocità variabile.
Adaptable Speed UAV (ASU)	La velocità UAV viene regolata una volta che l'UAV è compreso nell'intervallo del nodo per consentire il tempo per il trasferimento completo dei dati dal buffer del nodo al buffer UAV. Questo tempo di trasferimento dei dati dipende dalla dimensione del buffer dei dati e dalla velocità in bit del protocollo di comunicazione utilizzato.	Questo algoritmo offre maggiore flessibilità. UAV può andare più veloce con i buffer dei nodi più piccoli, il che si traduce in un ritardo più basso nel trasferimento dei dati end-to-end. Tuttavia, un nodo può avere un buffer molto grande, che può provocare la perdita di molto tempo.	Richiede un UAV con capacità di velocità variabile.
Hover with Unlimited Service Time (HUS)	L'UAV si posiziona sopra il nodo corrispondente per consentire il trasferimento completo dei dati dal buffer del nodo al buffer UAV.	Permette una maggiore flessibilità. Non richiede che l'UAV si soffermi sul nodo se la dimensione del buffer del nodo è sufficientemente piccola per essere trasferita mentre l'UAV si sta spostando nell'intervallo del nodo. Come in ASU, un nodo può avere un buffer molto grande, che può provocare la perdita di molto tempo. L'UAV può anche essere utilizzato per fornire il trasferimento di dati in tempo reale a reti infrastrutturali come satellite, cellulare o WiMAX, ecc.	Richiede un UAV sospeso. Adatto per il trasferimento di grandi quantità di dati come immagini ad alta risoluzione, audio e video.
Hover with maximum Service Time (HMS)	L'UAV si libra sopra il nodo corrispondente per consentire il trasferimento dei dati con un tempo di hover massimo al fine di fornire maggiore equità temporale agli altri nodi.	Consente una maggiore flessibilità, ma impedisce che l'UAV si soffermi troppo su un unico nodo. L'UAV può anche essere utilizzato per fornire il trasferimento di dati in tempo reale a reti infrastrutturali come satellite, cellulare o WiMAX, ecc. Tuttavia, questo viene fatto con una quantità massima di tempo di hovering.	Richiede un UAV sospeso. Adatto per il trasferimento di grandi quantità di dati come immagini ad alta risoluzione, audio e video.

Tabella 2.1: Algoritmi per la raccolta dati

2.2.3 Modelli di mobilità

L'Utilizzo dell'UAV, come detto in precedenza, è l'approccio più conveniente per coprire un'intera area di sensori e accedere a ciascuno di essi nel modo più veloce. In *"Effects of UAV Mobility Patterns on Data Collection in Wireless Sensor Networks"* vengono esplorati vari modelli di mobilità dell'UAV che seguono percorsi diversi al fine di cercare la migliore copertura dell'area con il numero massimo di nodi coperti nel minor tempo possibile.

a) Circular Mobility Pattern

In questo modello di mobilità, l'UAV segue un percorso circolare con un determinato angolo di rotazione per poter curvare la sua traiettoria. Quando il raggio di virata si riduce, l'area circolare che copre il percorso dell'UAV si riduce, lasciando maggiori buchi scoperti all'esterno dell'area. Aumentando invece il raggio aumentano l'area operativa di pertinenza del drone. Se aumenta la circonferenza dell'area aumenta la lunghezza del percorso e il tempo per completare la missione, diminuiscono le zone scoperte all'esterno del cerchio (Gap) ma aumentano quelle all'interno. Circonferenze più piccole invece diminuiscono la lunghezza del percorso, riducono il tempo per completare la missione, diminuisce la dimensione del Gap interno alla circonferenza ma aumenta quello esterno.

b) Square Mobility Pattern

L'UAV segue un percorso rettangolare. Rettangoli più grandi aumentano l'area di pertinenza dell'UAV, quindi la lunghezza del percorso, il tempo per completare la missione e i Gap all'interno del rettangolo aumentano a vantaggio di quelli ai lati. Rettangoli più piccoli, invece, accorciano il percorso che deve compiere l'UAV, si riducono quindi i tempi di volo, si riducono Gap all'interno del rettangolo, ma aumentano quelli all'esterno.

c) Angular Mobility Pattern

L'UAV segue un percorso piano e appena arriva al confine dell'area operativa compie una virata di 360°, variando anche l'angolazione della traiettoria, compiendo quindi un percorso angolare, non completamente rettilineo. Aumentando l'inclinazione di direzione l'UAV riesce a percorrere tutto il percorso in meno tempo, ma il numero di nodi scoperti potrebbe aumentare. Riducendo l'inclinazione, invece, la durata del percorso aumenta, in quanto saranno necessari più spostamenti per coprire tutta l'area ma diminuiscono i Gap lasciati dall'UAV.

d) Tractor Mobility Pattern

L'UAV segue un percorso lineare fino al confine dell'area operativa, poi si sposta sull'altra colonna di sensori e ripete la traiettoria lineare ma nella direzione opposta. Più ampio è lo spostamento tra una colonna di sensori e l'altra e minore è il tempo che impiega l'UAV a percorrere tutta l'area, ma maggiore è la zona che rimane scoperta.

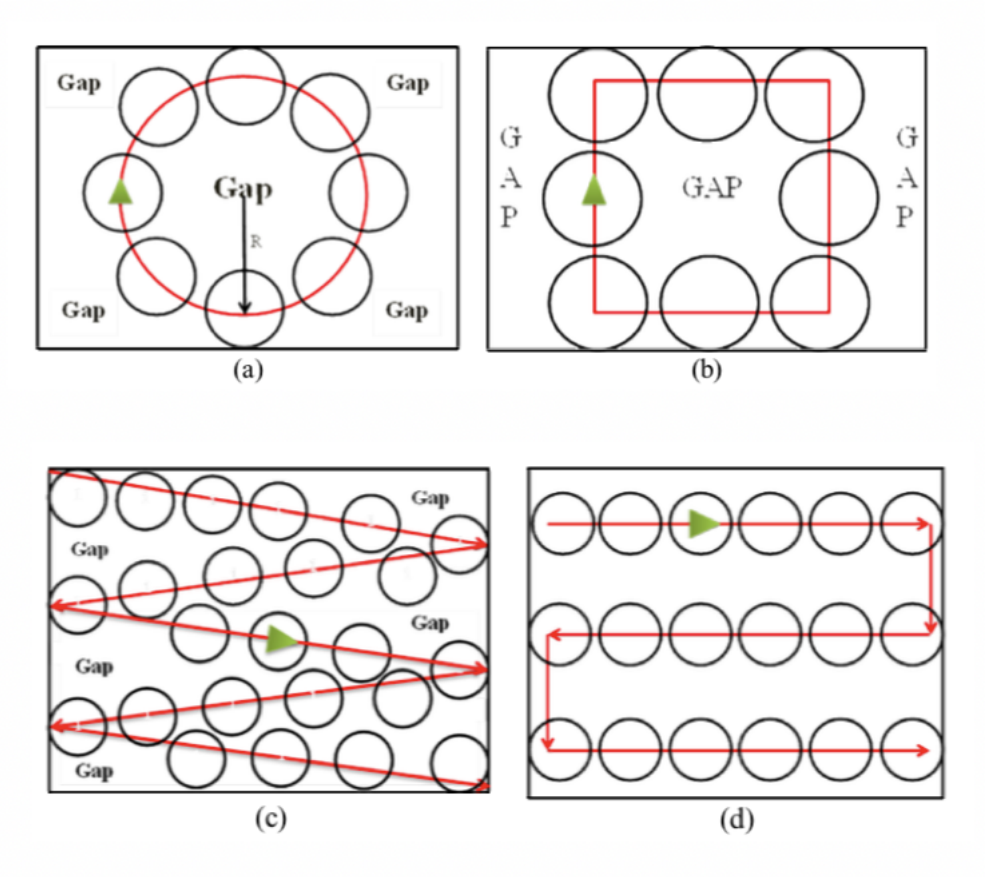

Figura 2.1: Modelli di mobilità

Sempre lo stesso articolo, [18], sostiene che tra tutti i modelli di mobilità, il modello di mobilità angolare (Angular Mobility Pattern) fornisce una copertura maggiore rispetto agli altri modelli, con un'inclinazione di 5 gradi per ogni cambio di direzione. Invece il modello di mobilità circolare offre un tempo di copertura inferiore a tutti gli altri modelli, e consiglia proprio questo modello con una grande quantità di sensori presenti nella rete. Il

principale problema del modello circolare, che emerge in *"Power-Efficient Communication in UAV-Aided Wireless Sensor Networks"* [19], è il grande consumo energetico.

2.3 Casi di studio

Come già accennato nel capitolo precedente le problematiche che insorgono nell'utilizzo di reti WSN sono molteplici come ad esempio il consumo energetico, la latenza, l'allocazione delle risorse e il calcolo della posizione del sensore. Questi problemi sono i principali oggetti di studio presenti in letteratura quando si parla di reti WSN. Oltre a queste problematiche vari articoli scientifici studiano anche l'utilizzo di diversi protocolli e la disposizione dei sensori all'interno nella rete.

2.3.1 Consumo energetico

Il più grande problema delle reti WSN è dovuto dal fatto che sono composte da sensori a batteria, i quali, una volta messi in funzione risulta difficile ricaricarne le batterie. Con l'utilizzo degli UAV per il recupero dei dati dagli SN (sensori a terra) vi è già una grande ottimizzazione dell'energia [19].

Oltre all'utilizzo degli UAV le tecniche utilizzate per minimizzare il consumo energetico e quindi per massimizzare la durata della vita di una rete WSN sono molteplici, una è quella dello sleep and wake up mechanism proposto in *"Energy-Efficient Data Collection in UAV Enabled Wireless Sensor Network"* [15]. Questo meccanismo consente al sensore di rimanere spento (stato di sleep) fin tanto che il beacon[5], situato sull'UAV, non gli manda un segnale

[5]Un beacon è un sistema di posizionamento indoor, descritto da Apple come "una nuova classe di trasmettitori, a bassa potenza e a basso costo, che può notificare la propria presenza a dispositivi vicini. La tecnologia consente ad uno smartphone o ad un altro dispositivo di effettuare delle azioni quando sono nelle vicinanze di un beacon. La tecnologia beacon sfrutta la tecnologia Bluetooth Low Energy (BLE). [49]

di waking up che gli dice che è presente un UAV nelle immediate vicinanze, a quel punto il sensore si veglia e manda i dati al UAV, finita la trasmissione dei dati il sensore torna a dormire. Oltre allo sleep and wake up mechanism del sensore, lo stesso articolo propone anche un algoritmo di volo dell'UAV permettendogli di volare sopra il sensore o addirittura di rimanere al di sopra di esso, soffermandosi un certo intervallo di tempo, per poter recuperare tutti i dati necessari. L'UAV può quindi aumentare o ridurre la velocità di volo in base alle esigenze. I risultati numerici mostrano significativi risparmi energetici rispetto ad altri schemi di riferimento.

"Power and Performance Tradeoff of MAC Protocol for Wireless Sensor Network Employing UAV" [20], sfrutta lo stesso meccanismo per far fronte al risparmio energetico ma si preoccupa anche del problema della perdita dei pacchetti. Infatti, a causa della mobilità dell'UAV alcuni sensori potrebbero soffrire della perdita di pacchetti o dell'effetto Doppler[6]. Quindi per ridurre il consumo energetico ma garantendo al tempo stesso un elevato throughput viene proposto un nuovo protocollo CDMA MAC[7]. Questo nuovo protocollo, Prioritized Frame Selection based CDMA MAC (PFSC-MAC) consente al sistema con UAV di ottenere una bassa percentuale di perdita di pacchetti dovuta ad una mobilità controllata dell'UAV e ad un'alta frequenza di aggiornamento delle informazioni di rilevamento. Questo grazie ad uno schema di prioritizzazione dei sensori: i sensori sono raggruppati in gruppi e ad ogni gruppo è stata attribuita una determinata priorità. La priorità serve all'UAV per stabilire l'ordine con cui visitare i vari nodi della rete. Questo protocollo sfrutta CDMA per il suo metodo di trasmissione fisica.

[6]L'effetto Doppler è un fenomeno fisico che consiste nel cambiamento apparente, rispetto al valore originario, della frequenza o della lunghezza d'onda percepita da un osservatore raggiunto da un'onda emessa da una sorgente che si trovi in movimento rispetto all'osservatore stesso.[50]

[7]Nell'ambito delle telecomunicazioni code division multiple access ("accesso multiplo a divisione di codice", nota anche con l'acronimo CDMA) è il protocollo di accesso multiplo a canale condiviso di comunicazione più diffuso nelle reti wireless. [51]

Un approccio completamente diverso viene descritto in *"Throughput per Pass for Data Aggregation from a Wireless Sensor Network via a UAV"* [21]. Nello scenario descritto in questo articolo i sensori vengono posizionati casualmente su una determinata area, ogni sensore viene utilizzato sia per la raccolta che per l'inoltro dei dati. Quindi in questo schema viene utilizzato un approccio multi-hop forwarding dei sensori oltre che all'UAV come sink[8]. Infatti, ogni nodo invia i dati o al sensore più vicino o all'UAV se si trova nelle immediate vicinanze. L'obbiettivo della rete è quello di far arrivare tutti i dati raccolti dai sensori ad un ricevitore lontano. Il contributo di questo articolo sta nel tentativo di ridurre il lavoro del singolo sensore, dando maggiori incarichi all'UAV. Ogni sensore invia un pacchetto di dati all'UAV indipendentemente dagli altri sensori. Questo vuol dire che l'UAV potrebbe ricevere lo stesso dato due volte. Un Ranke è stato quindi posizionato a bordo dell'UAV per la pulizia dei dati. L'introduzione del Ranke può provocare un ritardo dovuto al tempo impiegato per la pulizia dei dati, ma riduce il lavoro del singolo sensore, in quanto si dovrà solo preoccupare dell'invio dei pacchetti e non più della pulizia dei duplicati, e ne consegue una riduzione del consumo energetico e un aumento della vita della rete.

Un modo per definire il ciclo di vita (lifetime) di una rete di sensori wireless è impostare una soglia per il numero di disconnessioni tra i nodi. Le disconnessioni isolano sensori o gruppi di sensori che non possono fornire i loro dati acquisiti benché attivi. Al fine di avere percorsi di comunicazione alternativi in modo che i sensori siano sempre in grado di fornire i dati acquisiti, in letteratura sono presenti diverse soluzioni come: aggiunta di nodi relay (ricetrasmittenti), soluzioni Delay Tolerant Networks che consentono di controllare le connessioni del nodo in base alla necessità di comunicazione, infine un altra soluzione è avere un numero di nodi sink mobili che si muovo-

[8] I nodi sink hanno lo scopo di raccogliere i dati e trasmetterli tipicamente ad un server o ad un calcolatore. [13]

no in modo tale da coprire l'intera area, questo consente ai sensori di avere sempre un sink a cui trasmettere i dati. Per fornire una connessione affidabile alternativa *"Using Cooperative MIMO Techniques and UAV Relay Networks to Support Connectivity in Sparse Wireless Sensor Networks"* [22] propone l'utilizzo di tecniche multiple MIMO[9] (Multiple Input Multiple Output) per supportare comunicazione tra sensori statici in un WSN e una rete di UAV, mantenendo il WSN connesso e prolungandone la durata. Lo scenario proposto da questo articolo prevede un'area di 10km x 10km, 14 isole di sensori separate da una distanza pari a 3000 metri. Gli UAV vengono distribuiti casualmente e si muovono secondo il modello di mobilità casuale (RWP[10]), ad una velocità media di 85km/h. Il centro di raccolta dati è postato al centro dello scenario. I risultati ottenuti da questo scenario mostrano che per aumentare la connettività tra le isole basta aumentare il numero di UAV che cooperano per mezzo di MIMO. Quindi non basta aumentare il numero degli UAV per migliorare la connessione, ma è necessario che tutti i nodi utilizzino una cooperazione di tipo MIMO.

"Performance Evaluation of Cooperative Relay and Particle Swarm Optimization Path Planning for UAV and Wireless Sensor Network" [23] fornisce una soluzione al problema dell'efficienza energetica sfruttando Particle Swarm Optimization (PSO). PSO permette di ottimizzare il comportamento di uno sciame di UAV; in particolare, in questo documento, viene utilizzato per scegliere i migliori waypoint, intesi come punti da visitare, in termini di consumo energetico, BER[11] e tempo di viaggio dell'UAV. L'utilizzo di PSO

[9]Nella teoria dei sistemi dinamici, il termine Multiple-input and multiple-output, in sigla MIMO, indica un sistema dotato di svariati ingressi e uscite. Nelle telecomunicazioni, invece, indica l'uso di un sistema di antenne multiple sia sul lato emittente sia sul lato ricevente, allo scopo di migliorare le prestazioni del canale di comunicazione. [56]

[10]RWP è un algoritmo di controllo del movimento adattato che consente di beneficiare sia della gamma estesa di nodi MIMO cooperativi sia degli UAV. [22]

[11]In telecomunicazioni, in un sistema di trasmissione digitale il Bit Error Ratio (BER), è il rapporto tra i bit non ricevuti correttamente e i bit trasmessi. Il BER è un parametro molto importante perché fornisce una misura della qualità dell'intero sistema di

permette un consumo di energia significativamente basso, questo comporta un aumento del numero dei cicli di raccolta dei dati. Aumentando il numero dei cicli di raccolta, la perdita di dati diminuisce e di conseguenza diminuisce anche il BER.

Infine *"Effective Data Gathering and Energy Efficient Communication Protocol in Wireless Sensor Networks employing UAV"* [24] per aumentare la durata della rete propone un modello misto, che mette assieme approcci visti in precedenza e altri approcci studiati in letteratura. L'articolo propone un nuovo schema che è una combinazione tra Trasmission Priority e Circularly Optimized Frame Selection (COFS); non solo, in questo studio viene utilizzato l'algoritmo multi-hop forwarding e il sleep and wake up mechanism. In questo scenario prima di poter effettuare la trasmissione dei dati, viene suddivisa l'area di copertura del segnale in molte sezioni circolari attribuendo ad ogni sezione priorità diverse. All'interno di ogni gruppo di sensori vengono selezionati in modo del tutto causale dei nodi che fungono da Cluster Head (CH). I sensori cercano il CH più vicino per trasmettergli i dati. Se il nodo è il CH stesso, esso trasferisce i dati all'UAV. Se il nodo vicino non è un CH i dati vengono inoltrati da un nodo all'altro fino ad arrivare al CH. Dallo studio risulta che il numero ottimale di CH, per ottimizzare il dispendio energetico, per ogni frame deve essere minimo 2, ma questo può dipendere dal numero di nodi. Aumentando il numero di nodi è consigliabile anche un aumento del numero di CH. Utilizzando questo approccio "misto" vi è una grande riduzione del consumo energetico rispetto alle altre soluzioni proposte in letteratura.

2.3.2 Tempo di raccolta dati

Nel capitolo precedente viene spiegato come l'introduzione di un UAV all'interno di una rete WSN possa far risparmiare energia ai singoli sensori. Questo approccio d'altro canto potrebbe però causare un incremento

comunicazione. [52]

del tempo impiegato per la raccolta di tutti i dati. Lo studio proposto in *"Optimizing Energy-Latency Trade-off in Sensor Networks with Controlled Mobility"* [25] si preoccupa di minimizzare la latenza quando si usa un UAV per la raccolta dei dati. Come prima cosa utilizza un framework DMS per la pianificazione del percorso ottimale con una relativa velocità associata. La soluzione che utilizza per la raccolta dati invece è una soluzione intermedia tra i due approcci classici, ovvero l'utilizzo del solo UAV per raccogliere tutti i dati e il meccanismo di multi-hop forwarding. Questa soluzione fa si che la latenza diminuisca, in quanto il percorso compiuto dall'UAV è indubbiamente più breve rispetto ai casi precedentemente studiati. L'UAV non deve visitare tutti i nodi della rete ma solamente determinati nodi. Esistono quindi due tipologie di sensori: quelli che inoltrano i dati ai sensori vicini e quelli che inviano tutti i dati raccolti all'UAV, compresi i dati ricevuti dagli altri sensori. In questo modo gli autori dell'articolo sono riusciti a trovare una soluzione ottimale per la riduzione della latenza a discapito di un leggero consumo energetico.

In *"QoS Constraint with Prioritized Frame Selection CDMA MAC Protocol for WSN Employing UAV"* [26] viene fatto uno studio un po' più approfondito e cerca di trovare una relazione tra numero ottimale di sottogruppi di sensori, PER[12] e tempo medio per raccolta dei dati. Questo studio riesce inoltre a stabilire i fattori da cui dipende il tempo impiegato per la raccolta di tutti i dati. [26] afferma che il classico protocollo CSMA / CA, con accesso causale ai sensori non è utilizzabile in WSN-UAV perché provoca un ampio ritardo nella comunicazione tra i sensori e l'UAV, specialmente nelle reti di sensori ad alta densità, e perché provoca il così detto hidden terminal

[12]La Packet Error Rate (PER) viene utilizzata per testare le prestazioni del ricevitore di un terminale di accesso. PER è il rapporto, in percentuale, tra il numero di pacchetti non ricevuti correttamente dal terminale di accesso (AT) e il numero di pacchetti inviati all'AT dalla rete di sensori. [53]

effect[13]. Il nuovo protocollo utilizzato viene chiamato PFS-QCC MAC in quanto utilizza PFS-MAC per la scelta del sensore da cui recuperare i dati, CDMA come metodologia di trasmissione fisica e QoS come connessione. Il protocollo PFS-MAC (Prioritized Frame Selection) consente una scelta del sensore in base alla sua priorità associata e non più casualmente. Lo scenario prevede lo sleep and wake up mechanism per un maggior risparmio energetico e sempre per lo stesso motivo non prevede nessun tipo di multi-hop forwarding, ma solo trasferimento dei dati all'UAV. L'UAV per esplorare i gruppi di nodi con maggior priorità segue un percorso simile al Angular Mobility Pattern.

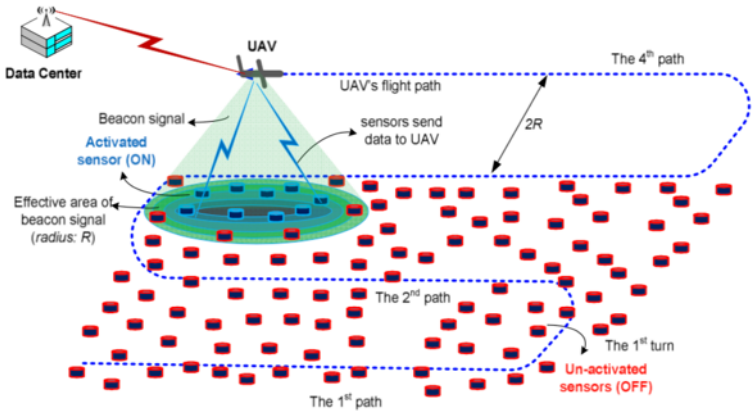

Figura 2.2: Angular Mobility Pattern

Con questa configurazione del sistema e dopo varie analisi sono giunti alla conclusione che il tempo necessario per portare a termine la missione dipende da tre fattori: il percorso dell'UAV, il raggio di sterzata dell'UAV che utilizza per curvare e il numero di sottogruppi di sensori. Infatti, utilizzando Angular Mobility Pattern si ottiene un buon tempo per visitare tutti i sensori, utilizzando altri percorsi invece le performance peggiorano. Aumentando il raggio di sterzata il tempo per portare a termine la missione si riduce perché il

[13]Effetto terminale nascosto: la distanza di comunicazione tra due sensori è piuttosto breve rispetto alla distanza tra il sensore e l'UAV. Questo potrebbe causare che l'algoritmo scelga sempre l'approccio multi-hop e mai l'invio dei dati all'UAV. [26]

percorso si accorcia. Minore è il numero di sottogruppi e maggiore è il tempo per raccogliere tutti i dati, in particolare si è calcolato che se l'UAV vola ad un'altezza di 200m si hanno ottimi risultati con 20 sottogruppi, se invece l'altezza si riduce a 50m bisogna diminuire il numero di sottogruppi. Questo perché maggiore è l'altezza dell'UAV e maggiore è il suo raggio d'azione. In generale, l'analisi ha mostrato che le prestazioni del sistema WSN-UAV basato sulla scelta per priorità (PFS-QCC MAC) sono migliori rispetto quelle mostrate fruttando un protocollo CSMA / CA, infatti garantisce un basso tasso di errore dei pacchetti, un breve tempo richiesto per la raccolta dei dati e un basso consumo energetico.

2.3.3 Calcolo della posizione

Un altro grande problema di una rete WSN è il riconoscimento della posizione dei sensori. Il sistema di localizzazione più utilizzato in letteratura è quello basato su ancore, un approccio ricorsivo (RPE), in cui un nodo stima la propria posizione in base alle informazioni relative alla posizione di tre o più nodi di riferimento (nodi vicini che conoscono la loro posizione). Una volta calcolata la sua posizione il nodo diventa anche lui un nodo di riferimento (ancora) per altri nodi che vogliono determinare la loro posizione. Questo approccio ha un grosso inconveniente: l'errore di propagazione. Ogni sensore, nel calcolo della propria posizione, si porta con sé l'errore del sensore "ancora", propagandolo per tutta la rete ed aumentandolo ad ogni iterazione. Altro inconveniente è che il sensore deve avere almeno tre ancore vicine per potere determinare la sua posizione. La soluzione più semplice sarebbe quella di fornire ad ogni nodo un sensore GPS. Questa soluzione individua una posizione con un errore bassissimo, ma presenta diversi inconvenienti come: 1) aumento delle dimensioni del sensore, 2) aumento del consumo energetico del sensore, 3) aumento del costo della rete (avere a disposizione un sensore GPS per ogni nodo risulta economicamente molto dispendioso). Quindi risulta una soluzione impraticabile per reti con molti sensori.

"3D Localization in Wireless Sensor Networks Using Unmanned Aerial Vehicle" [27] propone un sistema alternativo di localizzazione 3D. In questo modello viene sfruttato l'UAV per il calcolo della posizione del sensore. L'UAV sorvola l'area di monitoraggio in cui sono stati distribuiti i sensori. Durante il volo, l'UAV trasmette periodicamente la sua posizione geografica. Quando in nodo riceve quattro o più messaggi contenenti la posizione dell'UAV, il nodo è in grado di calcolare la sua posizione. Non basta però ricevere 4 messaggi da parte dell'UAV per calcolare la posizione esatta, si tratta di un algoritmo più complesso suddiviso in 2 fasi:

1. Stima della distanza: tramite RSSI. Quando un nodo riceve un messaggio, può misurare l'intensità del messaggio ricevuto e utilizzare un modello di propagazione del segnale per trovare la distanza dal nodo che ha inviato il messaggio.

2. Calcolo della posizione: necessari 4 punti di riferimento, ottenuti durante il volo degli UAV. Oltre ai 4 punti di riferimento è necessario sapere la distanza da ciascun punto (calcolata nel punto 1). Una volta ottenuti i 4 punti di riferimento e la distanza tramite RSSI viene utilizzato il metodo della multilaterazione (multilaterarion) per determinare la posizione.

Dai risultati si evince che l'algoritmo di stima della posizione ricorsiva (RPE) ha un errore di circa 3 volte superiore rispetto alla localizzazione 3D. Inoltre, l'errore di RPE è influenzato dal numero di sensori, aumentando il numero di sensori aumenta l'errore dovuto alla propagazione; invece l'algoritmo di localizzazione 3D è influenzato minimamente dal numero di sensori. Se si aumenta il numero di sensori della rete, basta aumentare il numero di UAV per correggere l'errore o i problemi causati dal sistema di localizzazione 3D. In ogni caso questo sistema permette sempre a tutti i nodi di ricevere abbastanza informazioni per determinare la propria posizione. Altro vantaggio di questo algoritmo è il risparmio energetico: lo scambio di messaggi per la stima della posizione avviene solo tra UAV e il sensore, e non tra i singoli

sensori; in questo modo i nodi risparmiano energia, non dovendo mandare messaggi extra con la loro posizione.

Lo stesso procedimento per il calcolo della posizione del sensore viene utilizzato anche in *"A Joint 3D Localization and Synchronization Solution for Wireless Sensor Networks Using UAV"* [28]. In questo studio però la tecnica viene leggermente raffinata, infatti l'UAV non solo manda informazioni relative alla posizione ma anche informazioni relative al tempo. In questo modo in un sistema dinamico il sensore riesce a determinare la posizione e in che tempo della simulazione si trovava in quella posizione.

2.3.4 Utilizzo di protocolli

Come già dimostrato precedentemente, ad esempio in [26], il protocollo più sconsigliato per le reti WSN è CSM / CA. Possiamo leggere le stesse conclusioni anche in *"Highly Reliable Communication Protocol for WSN-UAV System Employing TDMA and PFS Scheme"* [29] e in *"Effective Data Gathering Protocol in WSN-UAV employing Priority-based Contention Window Adjustment Scheme"* [24].

In particolare [29] afferma che CSMA / CA subisce un grande ritardo di comunicazione, soprattutto in reti con molti sensori, un elevato PER causato dal movimento dell'UAV e può provocare l'effetto terminale nascosto. L'effetto terminale nascosto avviene quando uno dei nodi della rete non riesce mai ad ottenere abbastanza priorità ad essere considerato dall'UAV, e in questo modo i suoi dati non vengono mai recuperati. Viene quindi proposto un nuovo protocollo MAC chiamato PFSC-MAC che sfrutta sia lo schema PFS sia TDMA. Questo sistema consente a WSN-UAV di ottenere un basso PER e un'alta frequenza di trasmissione dei dati da parte dei sensori. Anche i risultati ottenuti da questo esperimento sono molto simili a quelli ottenuti in [26]. Per diminuire il PER si consiglia di diminuire le dimensioni del pacchetto, il numero di sensori e l'altitudine dell'UAV. Se si aumenta il numero

di sensori è consigliato invece aumentare l'altitudine dell'UAV per ridurre il tempo impiegato per completare la missione, facendo attenzione a non aumentarla troppo se no vi è un aumento della perdita di pacchetti e quindi il PER si alza.

In [24] invece viene proposto un protocollo chiamato PCWAS che si basa sulla scelta dei sensori da visitare in base alla priorità, molto simile a quello proposto in [26], con la differenza che il volo dell'UAV segue un percorso circolare e non angolare. Con PCWAS si ha un throughput medio nettamente migliore rispetto a CSMA / CA indipendentemente dalla dimensione della rete. Inoltre, si ha una notevole riduzione della perdita dei pacchetti, un miglior rapporto di consegna e un minor ritardo medio rispetto a CSMA / CA.

2.3.5 Allocazione delle risorse

Nella maggior parte dei contributi proposti in letteratura nessuno si preoccupa di analizzare il problema dell'allocazione delle risorse per massimizzare la quantità totale di dati raccolti e la velocità con cui gli stessi vengono raccolti. Per problema di allocazione delle risorse si intende in particolar modo la quantità di banda da attribuire ad ogni singolo sensore per poter recuperare il maggior numero di dati.

Nell'articolo *"Resource allocation for data gathering in uav-aided wireless sensor networks"* [30] vengono confrontati due algoritmi per l'allocazione delle risorse: DPBA e ERAA. ERAA attribuisce una stessa larghezza di banda per tutti i sensori, invece DPBA attribuisce una specifica larghezza ad ogni specifico sensore. Aumentando la larghezza di banda aumenta la velocità di trasmissione, in quando sarà necessario un minor tempo per inviare tutto il pacchetto; tuttavia la larghezza di banda è limitata, quindi è importante assegnare ad ogni sensore una giusta quantità per evitare che finisca prima che l'UAV visiti tutti i sensori. L'algoritmo implementato in questo studio, riesce

ad attribuire una giusta larghezza di banda ad ogni sensore per ottimizzare anche la velocità di trasmissione.

2.4 Reti WSN-UAV in contesti reali

Gli studi descritti nel capitolo precedente, i cui risultati sono ottenuti tramite simulazioni, sono stati usati come punto di partenza per vari progetti concreti. Approcci per aumentare la vita della rete, algoritmi per abbassare il tempo di latenza, protocolli per ottimizzare throughput e PER sono stati utilizzati in reti reali per ottimizzare diversi tipi di processi in diversi campi come quelli militari, marittimi, agricoli o reti stradali.

Ad esempio, le principali attività di ricerca strategica degli autori di *"Performance Evaluation of Cooperative Relay and Particle Swarm Optimization Path Planning for UAV and Wireless Sensor Network"* [31] riguardano il monitoraggio di petrolio e ghiaccio, la sorveglianza del traffico navale, il monitoraggio e le operazioni marittime. La maggior parte di queste aree sono molto ampie e isolate a causa di una mancanza di infrastrutture di comunicazione. Viene quindi proposta una soluzione che sfrutta una rete di sensori, sul terreno e sulla superficie del mare, e un UAV per la raccolta dati. In [31] si fa affidamento a PSO (Particle Swarm Optimization) come metodologia di ottimizzazione per trovare i waypoint di un UAV, al fine di ridurre il consumo energetico e il bit error rate (BER) dei nodi e il tempo di viaggio.

Sempre rimanendo nel contesto marittimo in *"Distributed Task Allocation and Coordination Scheme for a Multi-UAV Sensor Network"* [32] viene proposto un algoritmo di allocazione delle attività distribuito per una rete di sensori multi-UAV per il rilevamento delle intensità di torio[14] nelle aree costiere dell'india. Lo scenario qui proposto prevede uno sciame di UAV cooperativi e una stazione di controllo che rileva efficacemente l'intensità radioattiva della

[14]Il torio è un metallo naturale, leggermente radioattivo. [54]

sabbia. La stazione di controllo è un computer portatile dotato di software specializzato. Il software dividerà l'area da scansionare in una sequenza di waypoint e ogni posizione avrà una coordinata GPS. La stazione di controllo comanda il volo degli UAV, i quali recuperano i dati e con essi la stazione centrale genera una mappa di intensità per mostrare dove si trova il torio. Ogni gruppo di UAV si dedicherà ad un preciso waypoint. Viene implementato un algoritmo di condivisione dinamica delle attività per un rilevamento più efficiente. La necessità di tale algoritmo deriva dal fatto che gli UAV hanno un diverso dispendio energetico dovuto dal waypont, dal tempo di volo, dall'ambiente, e dalla qualità delle misurazioni; quindi le batterie di alcuni UAV potrebbero scaricarsi prima di altri. Ogni UAV, dopo aver raccolto tutti i dati da un determinato waypoint, controllerà se il numero degli obiettivi rimasti (waypoint da visitare) è maggiore di 0 e se la batteria residua è minore della potenza richiesta per portare a termine il prossimo obiettivo e recuperare i dati necessari; ciò vuol dire che l'UAV non è in grado di portare a termine la prossima missione. In questo caso l'UAV trasmetterà una richiesta di condivisione ad un altro UAV e attiverà un timer di risposta. L'UAV che riceve la richiesta calcolerà a sua volta la possibilità di portare a termine l'obiettivo andando a determinare la potenza residua e la potenza necessaria per compiere gli obbiettivi che già aveva in carico. Nel caso in cui nemmeno lui riesca a portare a termine la richiesta risponderà con esito negativo, altrimenti con esito positivo. Allo scadere del timer l'UAV controlla la risposta e se ha esito negativo, inoltra la richiesta ad un altro UAV, questo processo viene ripetuto fin tanto che tutti gli obbiettivi non vengono portati a termine. Le analisi sono state fatte sia sfruttando l'algoritmo di condivisione, sia senza. Senza sfruttare l'algoritmo qui proposto, in uno scenario con 3 UAV e 100 waypoint per ogni UAV, il primo UAV riesce a portare a termine il 100% dei suoi compiti, il secondo l'80%, il terzo meno del 50%. In totale su 300 compiti totali solo 230 sono stati portati a termine (76,7%). Invece sfruttando l'algoritmo le attività totali portate a termine sono 250 (83,33%). I risultati dimostrano che, usufruendo di questo algoritmo, vi è

un miglioramento del 5% nel portare a termine i compiti da parte degli UAV.

Cambiando completamente settore, e passando a scenari agricoli, abbiamo due studi interessanti: *"The use of unmanned aerial vehicles and wireless sensor network in agricultural applications"* [33] e *"UAVs in WSNs for Agricultural Applications: an Analysis of the Two-Ray Radio Propagation Model"* [34].

In particolare [33] si basa sull'utilizzo di UAV in campo agricolo per pesticidi e fertilizzanti. L'utilizzo di UAV in questi contesti può risultare vantaggioso per quanto riguarda velocità ed efficacia, ma alcuni fattori potrebbero ridurre la resa o addirittura causare danni. Ad esempio, se il raggio d'azione dell'UAV è limitato, alcune zone potrebbero non venire irrigate, o se l'UAV non è ben programmato potrebbe spruzzare pesticidi in zone in cui non dovrebbe essere spruzzato. Inoltre le condizioni meteorologiche come piogge o forti venti potrebbero limitarne l'utilizzo. Per far fronte ad alcune di queste problematiche in [33] vengono utilizzati sensori wireless per fornire all'UAV dei feedback in modo tale da limitare la zona in cui l'UAV deve spruzzare le sostanze chimiche. Periodicamente l'UAV trasmette messaggi ai sensori posizionati a terra. Se il sensore riceve il messaggio, risponde con un messaggio che riporta la quantità di pesticida presente nel terreno e la posizione. Una volta ricevute queste informazioni, l'UAV è in grado di decidere se cambiare rotta o no; se la quantità di pesticida supera una certa soglia, l'UAV si preoccuperà di procedere verso il prossimo sensore, altrimenti rimarrà nella stessa zona a spruzzare altro pesticida. Questo algoritmo tiene conto anche della direzione e della potenza del vento, modificando al meglio velocità e quantità di pesticida da spruzzare durante il tragitto, per evitare che il pesticida cada al di fuori del perimetro d'azione.

Invece in [34] viene descritta una particolare configurazione della rete WSN utilizzata sempre in un contesto agricolo. Questa rete sfrutta come proto-

collo di comunicazione la tecnologia ZigBee[15], stessa tecnologia utilizzata in questo progetto di tesi.

L'articolo *"Multi-UAV-Aided Networks"* [35] sposta il suo studio in un contesto di rete stradale. Due UAV vengono schierati per cooperare con tre veicoli terrestri. Il primo UAV raccoglie le informazioni multimediali della situazione stradale e il secondo UAV trasferisce le informazioni al veicolo. I due UAV comunicano tra di loro tramite moduli Wi-Fi (802.11a) e ZigBee, in particolare il modulo Wi-Fi è destinato alla trasmissione delle immagini e il modulo ZigBee serve per l'invio dei messaggi di comando.

Anche *"Smooth Path Construction for Data Mule Tours in Wireless Sensor Networks"* [36] si concentra sul trasferimento di file multimediali. Questo studio sfrutta una rete WSN potenziata con sensori multimediali in grado di fornire un'immagine molto più chiara ed esplicita di ciò che sta accadendo nell'area monitorata, utilizzando sia dati audio che video. Questa nuova forma di WSN viene chiamata Wireless Multimedia Sensor Network (WMSN). L'enorme svantaggio di questa rete è l'inevitabile aumento della quantità di dati che devono essere raccolti. Utilizzando un tradizionale approccio multi-hop forwarding causerà un grande consumo di energia e collegamenti inaffidabili portando ad una diminuzione della durata della rete e bassi tassi di raccolta. Questo porterebbe grande svantaggio nelle applicazioni su larga scala come ad esempio il monitoraggio delle foreste. Per far fronte a queste problematiche si è ritenuto necessario utilizzare un UAV più potente e più veloce. All'UAV si è fornito un percorso veloce e fluido di lunghezza minima che passi per tutti i nodi che avessero dei dati da inviare. Per ottenere questo percorso si è utilizzato un algoritmo che sfrutta TSP (Travelling Salesman

[15]Nel mondo delle tecnologie wireless ZigBee rappresenta uno dei principali standard di comunicazione. Attraverso l'uso di piccole antenne digitali a bassa potenza e a basso consumo basate sullo standard IEEE 802.15.4 per wireless personal area networks (WPAN), lo standard specifica una serie di profili applicativi che permettono di realizzare una comunicazione specifica per le Wireless Sensor Networks. [55]

Problem) per il calcolo del percorso di lunghezza minima, e SPC (Smooth Path Constructions) per ottenere un percorso fluido e quindi percorribile da un UAV con determinate caratteristiche tecniche. Utilizzando questo algoritmo per la scelta del percorso e il protocollo AODV si ottengono significativi vantaggi come l'aumento della durata della rete e una riduzione del tempo di consegna del pacchetto.

Infine, gli UAV possono essere utilizzati anche per missioni di sicurezza nazionale o per operazioni militari. *"Leveraging Public Wireless Communication Infrastructures for UAV-Based Sensor Networks"* [38] afferma che per questo tipo di missioni è meglio non utilizzare reti 2G, in quanto potrebbero causare un ritardo di trasmissione dei dati molto elevato, o addirittura potrebbero esserci aree o situazioni in cui quel tipo di rete potrebbe subire guasti o interruzioni. Il contributo di questo articolo è appunto cercare di dare una soluzione a questo tipo di problema. Utilizzando collegamenti Air-to-Air [A2A] tramite l'utilizzo di droni è possibile compensare l'eventuale perdita di segnale nelle aree senza ricezione. Se tutte le reti pubbliche sono inattive a causa di un disastro o un'altra catastrofe, le WLAN in modalità infrastruttura o le stazioni mobili WiMAX ad hoc offrono soluzioni affidabili per le organizzazioni di soccorso. Per implementare collegamenti A2A sono disponibili diverse tecnologie e frequenze, ad esempio è possibile usare la tecnologia ZigBee che ha buone capacità di meshing ma ha una copertura bassa. Il problema di copertura viene però facilmente risolto dall'utilizzo di sciami di UAV cooperativi

2.5 Studi correlati

Altri esperimenti molto interessanti sono stati svolti sfruttando reti WSN-UAV per monitorare zone non accessibili o per esplorare ambienti sconosciuti o ostili, ad esempio aree in cui l'uomo non può accedere perché inquinate da gas nocivi o radiazioni provocate da incidenti.

Ad esempio, lo scenario previsto in *"Experimental Analysis of Coordination Strategies to Support Wireless Sensor Networks Composed by Static Ground Sensors and UAV-carried Sensors"* [39] prevede un'area contigua aperta in cui ogni elemento è localizzato con coordinate cartesiane. Non vi sono ostacoli e gli obbiettivi da sorvegliare sono veicoli o esseri umani, o gruppi di veicoli e esseri umani. Gli obbiettivi appaiono in modo non deterministico secondo la distribuzione di Poisson[16]. I bersagli (umani e veicoli) hanno una velocità costante, ma tra di loro possono avere velocità diverse. I bersagli possono cambiare in modo casuale la loro direzione. Il sistema di sorveglianza, invece, è composto da sensori a terra e sensori mobili. I sensori sul terreno sono distribuiti secondo una data distribuzione (casuale o uniforme), i sensori mobili invece sono UAV che volano sopra il perimetro da sorvegliare. Ovviamente, come in ogni rete WSN il numero dei sensori è maggiore del numero degli UAV. I sensori vicini comunicano tra di loro in modalità wireless. Se è necessario mandare un segnale di allarme all'UAV il segnale viene propagato tra tutti i sensori vicini, e quello più vicino all'UAV lo inoltra anche a lui. L'UAV si muove su traiettorie contigue a velocità costante e con angolo di sterzata limitato. Lo scopo dell'esperimento è trovare un algoritmo per la scelta dell'UAV più adatto in base a posizione, caratteristiche, disponibilità e sensori associati per portare a termine la missione. Le soluzioni possono essere due: scegliere l'UAV disponibile più vicino oppure scegliere l'UAV disponibile che più si addice alla richiesta, nonostante la distanza. Utilizzando la prima soluzione vi è un minor consumo energetico da parte del sensore, perché deve mandare meno messaggi di allarme, ma potrebbe capitare che l'UAV più vicino non sia in grado di portare a termine la missione, per mancanza di energia ad esempio. Il secondo approccio è leggermente più

[16]In teoria delle probabilità la distribuzione di Poisson (o poissoniana) è una distribuzione di probabilità discreta che esprime le probabilità per il numero di eventi che si verificano successivamente ed indipendentemente in un dato intervallo di tempo. Questa distribuzione è anche nota come legge degli eventi rari. Prende il nome dal matematico francese Siméon-Denis Poisson.

complesso da implementare ma sicuramente più performante. Questo secondo algoritmo prevede che l'allarme lanciato dal sensore si propaghi fin tanto che non trova un UAV che rispecchia le esigenze dell'allarme. L'UAV che riceve il messaggio controlla la presenza di altri UAV disponibili e nel caso contratta l'esecuzione del compito.

Nell'articolo *"Role-Based Connectivity Management with Realistic Air-to-Ground Channels for Cooperative UAVs"* [40] troviamo invece un elenco di quattro diversi scenari e per ognuno di essi una diversa strategia di controllo della mobilità, sempre per scopi esclusivamente esplorativi.

1. Primo scenario: se gli agenti si muovono a random o tramite SRW (self repelling walk), l'algoritmo Cooperative Area Exploration (CAE) riesce a raggiungere una massima copertura entro un dato periodo di tempo. CAE seleziona un obiettivo casuale tra tutte le celle della griglia non ancora visitate, lo sciame di UAV si dirige verso quell'obiettivo. Lo stesso procedimento si ripete finchè ogni cella del perimetro d'azione non viene visitato.

2. Secondo scenario: se invece gli agenti non si muovono in modo causale vengono utilizzati algoritmi come Clustering Breathing (CB) o Communication Aware Potential Files (CAPF).

3. Terzo scenario: se l'UAV lavora in maniera autonoma è stato utilizzato un Bounded Direct Mode (BDM

4. Quarto scenario: se l'UAV lavora in maniera cooperativa A2A (Air-to-Air) e A2G (Air-to-Ground), possono essere utilizzati due approcci: uno basato su BR (Bounded Relaying) e l'altro basato su RRS (Release and Return).

La seguente tabella riporta i risultati ottenuti da [40].

ALGORITMO	CONNESSIONE A2A	CONNESSIONE A2G	COPERTURA
CAE			Alta
CB, CAPF	X		Media
BDM		x	
BR		X	
RRS	X	X	Alta

Tabella 2.2: Risultati di [40]

Come si può notale dalla tabella l'algoritmo RRS è l'unico applicabile sia per connessioni A2A che per connessioni A2G e offre una buona copertura.

In *"Priority-based coverage path planning for Aerial Wireless Sensor Networks"* [41] a differenza dei due articoli precedenti considera anche la presenza di ostacoli all'interno dell'area d'azione. Inoltre assegna una diversa priorità alle diverse celle da esplorare. Per far si che l'UAV si concentri prima sulle zone prioritarie viene utilizzato POMDP (Partially Observabe Markov Decision Process), ottimale per passare dalla posizione di partenza a quella con priorità più alta al fine di massimizzare la performance della rete. L'obbiettivo dell'esperimento è quello di portare in salvo tutte le vittime. Le vittime sono situate all'interno delle diverse celle da esplorare. L'algoritmo utilizzato in questo esperimento è stato confrontato con altri due algoritmi: Greedy e Potential. POMDP riesce sempre ad ottenere una copertura del 100%. Greedy e Potential no. Anche nel caso peggiore, ovvero nel caso in cui le vittime siano situate nelle zone con meno priorità POMDP riesce a recuperarle tutte perché visita anche le zone con bassa priorità. In questo caso però impiegherà molto più tempo a concludere la missione perché l'algoritmo prevedere di lasciare per ultime le aree con minor priorità. POMDP in generale recupera tutte le vittime molto più velocemente rispetto agli altri algoritmi. L'articolo conclude affermando quindi che POMDP è l'unica soluzione che raggiunge il 100% di copertura e ottimizza il tempo per trovare tutte le vittime.

Infine, vi sono altri tre articoli di particolare rilevanza. Il primo, dal ti-

tolo *"Towards a new low cost, simple implementation using embedded system wireless Networking for UAVs"* [42] parla del progetto "SURYAAN", una rete di sensori wireless semplice, a basso costo, autonoma con sistemi UAV e UGV (Unamanned Ground Vehicle). Il secondo, *"UAVNet: A Mobile Wireless Mesh Network Using Unmanned Aerial Vehicles"* [43], descrive invece la rete UAVNet, una rete WMN altamente adattiva e mobile che utilizza UAV. Questa rete di comunicazione distribuita consente la connettività tra diversi sistemi come notebook, smartphone e tablet tramite wireless IEEE 802.11s. Per concludere, in *"Data Communication in Linear Wireless Sensor Networks Using Unmanned Aerial Vehicles"* [44] vi è uno studio approfondito di una LSN (Linear Sensor Network). Una rete di sensori WSN allineati al fine di ridurre significativamente il consumo energetico utilizzato nella trasmissione dei dati e prolungare la durata della rete.

Capitolo 3

Progettazione e implementazione

In questa sezione si parlerà per sommi capi del progetto realizzato. Partendo dalle specifiche, passando alla descrizione della piattaforma utilizzata per fare le simulazioni, fino ad arrivare alle tecnologie e alle configurazioni scelte per svolgere l'analisi.

3.1 Specifiche progetto

Il progetto consiste in un'analisi dell'utilizzo di sensori e droni in contesti generici, che posso svariare dall'agricoltura alla prevenzione di valanghe. Sensori posizionati sul terreno potrebbero essere utilizzati in agricoltura per recuperare informazioni sul sottosuolo, come ad esempio l'aridità del terreno, la quantità di pesticidi, fertilizzanti o concimi presenti nelle coltivazioni. Oppure potrebbero essere posizionati tra la neve delle piste da sci per prevenire valanghe lanciando eventuali messaggi di allerta se la temperatura o lo stato della neve possa essere considerato rischioso. Il drone potrebbe essere utilizzato per ottimizzare il processo di raccolta dati e velocizzare il trasferimento di essi alla stazione centrale, per procedere, se necessario, con l'irrigazione del campo se si parla di agricoltura.

Per compiere questa analisi si è utilizzato un ambiente simulato, quale OMNeT++. Per mezzo di questo ambiente sono state implementate due tipologie di scenari, il primo è caratterizzato da una rete di sensori a terra e un gateway centrale. Lo scambio di pacchetti, contenenti i dati raccolti, in questo contesto, avviene tramite metodologia multi-hop forwarding. Infatti, i pacchetti che partono da ogni sensore vengono inviati al sensore più vicino e poi inoltrati fino ad arrivare al gateway che si trova esattamente al centro della rete. Il secondo scenario prevede in aggiunta alla classica rete WSN anche un nodo mobile, un UAV. L'UAV in questo contesto funziona come "mulo" di raccolta dati, una sorta di gateway mobile che passa esso stesso sul sensore per recuperare i dati. Con l'aggiunta dell'UAV si vuole ottimizzare il processo di raccolta dati, minimizzando il dispendio energetico e il delay provocato dall'inoltro multi-hop del pacchetto.

Una volta configurato l'ambiente, sono state fatte svariate simulazioni allo scopo di determinare come variano parametri come delay, throughput, PDR e la durata della batteria dei sensori al variare dello scenario e delle sue configurazioni. L'analisi è stata fatta a supporto di altri studi, analizzati nel capitolo precedente, come ad esempio l'articolo [15] che afferma il prolungamento della lifetime della rete grazie all'introduzione dell'UAV. Per ogni scenario si è infatti analizzato il delay per capire quale fosse la strategia per ottenere un minor ritardo, calcolato come il tempo che impiega il pacchetto ad arrivare a destinazione. Oltre al delay si è cercato di ottimizzare la capacità di trasmissione e massimizzare il numero di pacchetti ricevuti dal gateway prima che la batteria dei sensori si scarichi.

L'obiettivo che si vuole raggiungere è quello di valutare l'efficacia di soluzioni basate su comunicazione aerea per il recupero di dati provenienti da reti WSN, confrontate con soluzioni "classiche" che prevedono solo comunicazione a terra in tipologie multi-hop.

3.2 Tecnologie utilizzate

Per l'implementazione del modello con cui eseguire le simulazioni è stato utilizzato l'ambiente OMNeT++, un simulatore di eventi discreti. Per facilitare la stesura del codice, soprattutto della parte grafica, si è sfruttato OMNeT++ IDE.

3.2.1 OMNeT++

OMNeT++ è un framework basato su librerie scritte in C++[1], estendibili e modulari, utilizzato principalmente per l'implementazione di simulatori di rete. Le reti di cui si parla possono includere reti di comunicazione cablate e wireless, reti su chip, reti di code e così via. Sono supportate anche reti di sensori, reti wireless ad-hoc, protocolli Internet, modellazione delle prestazioni e reti fotoniche.

Sono previsti modelli già sviluppati come progetti indipendenti, facili da importare e da integrare in base alle esigenze. Durante gli anni, infatti, sono stati scritti, da diversi ricercatori, innumerevoli modelli di simulazione e framework come: modelli di accodamento, modellazione delle risorse, protocolli Internet, reti wireless, LAN commutate, reti peer-to-peer, streaming multimediale, reti ad-hoc mobili, reti mesh, reti di sensori wireless, reti veicolari, reti ottiche, sistemi cloud computing, e tante altre reti ad hoc. La maggior parte di questi modelli sono open source, sviluppati come progetti indipendenti e puntualmente aggiornati. Uno di questi modelli è INET, sfruttato per la creazione della rete su cui si basano gli esperimenti di questa tesi.

La repository del codice sorgente di OMNeT ++ è ora disponibile su GitHub[2]. Ciò consente a chiunque di seguire e partecipare attivamente allo

[1] C++ è un linguaggio di programmazione a oggetti tra i più diffusi in svariati ambiti dell'informatica professionale. [60]

[2] GitHub è una piattaforma di sviluppo. Permette, in un formato del tutto open source, di ospitare e revisionare codice, gestire progetti e creare software. [61]

sviluppo di nuove versioni. Ora è anche attivo il tracker su GitHub per la segnalazione di problemi relativi a OMNeT ++, in modo tale da lasciare la possibilità ad ogni sviluppatore di evidenziarne eventuali bug.

Il kernel[3] di simulazione OMNeT ++ è implementato in C ++ e funziona fondamentalmente su tutte le piattaforme in cui è disponibile un compilatore C ++. L'IDE di simulazione richiede Windows, Linux o macOS. OMNeT ++ offre infatti un IDE basato su Eclipse[4], un ambiente di runtime Figura e una miriade di altri strumenti. Esistono estensioni per la simulazione in tempo reale, l'emulazione di rete, l'integrazione di database, l'integrazione SystemC e molte altre funzioni. OMNeT ++ è distribuito sotto la Licenza Pubblica Accademica. [57]

Ecco una breve panoramica sulla logica utilizza per programmare in OMNeT++:

Modello

Un modello OMNeT ++ è costituito da componenti (moduli) che comunicano scambiando messaggi. I moduli possono essere nidificati, cioè più moduli possono essere raggruppati per formare un modulo composto. Quando si crea il modello, è necessario mappare il sistema in una gerarchia di moduli comunicanti.

File NED

[3]Il kernel è la principale componente software di un sistema operativo. Il kernel consente al sistema operativo la gestione delle risorse hardware del sistema, della memoria e l'assegnazione della priorità tempo/macchina nel processore ai processi in corso di esecuzione (multitasking). Il kernel è quindi l'interfaccia di comunicazione tra il software e l'hardware. [62]

[4]Eclipse fornisce ad individui e organizzazioni un ambiente maturo, scalabile e commercialmente favorevole per la collaborazione e l'innovazione del software open source. Eclipse è famoso per l'ambiente di sviluppo integrato Java, ma anche per C / C ++ e PHP. [63]

La struttura del modello viene implementata per mezzo del linguaggio NED. È possibile modificare NED in un editor di testo o nell'editor Figura dell'IDE di simulazione OMNeT++ basato su Eclipse. I componenti attivi del modello (moduli semplici) devono essere programmati in C++, usando il kernel di simulazione e la libreria di classi.

File INI

La configurazione del modello e l'attribuzione di particolari parametri avviene in un file omnetpp.ini. Un file di configurazione, di questo tipo può descrivere diverse esecuzioni di simulazione con parametri diversi.

3.2.2 OMNeT++ IDE

Omnet Integrated Development Environment (IDE) si basa sulla piattaforma Eclipse e lo estende con nuovi editor, views, wizards e altre funzionalità aggiuntive. OMNeT++ IDE aggiunge funzionalità per la creazione e configurazione di modelli e consente la visualizzazione dei risultati di simulazione tramite appositi tool di analisi.

NED Editor

L'Editor NED può modificare i file NED sia graficamente che in modalità testo e l'utente può passare da una modalità all'altra in qualsiasi momento, utilizzando le schede nella parte inferiore della finestra dell'editor. In modalità grafica, è possibile creare moduli composti, canali e altri tipi di componenti.

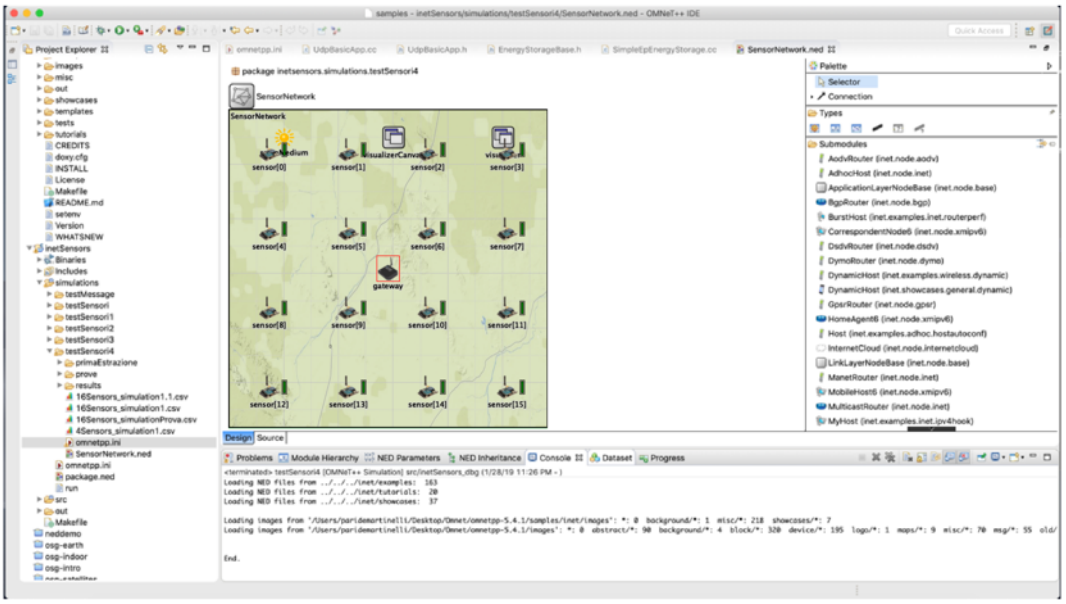

Figura 3.1: NED Editor - Design

L'editor offre molte funzionalità come ripristino, clonazione, spostamento, ridimensionamento, allineamento di oggetti e zoom. Le caratteristiche grafiche che possono essere modificate sono l'immagine e la griglia di sfondo,

le icone predefinite, l'intervallo di trasmissione e molto altro. La vista delle proprietà consente all'utente di modificare le proprietà grafiche e non grafiche degli oggetti. Speciali editor di celle facilitano la selezione di colori e icone. Sono supportate anche operazioni di "Annulla" e "Ripristina" per le modifiche delle proprietà. La modalità testo consente all'utente di lavorare direttamente con la sorgente NED.

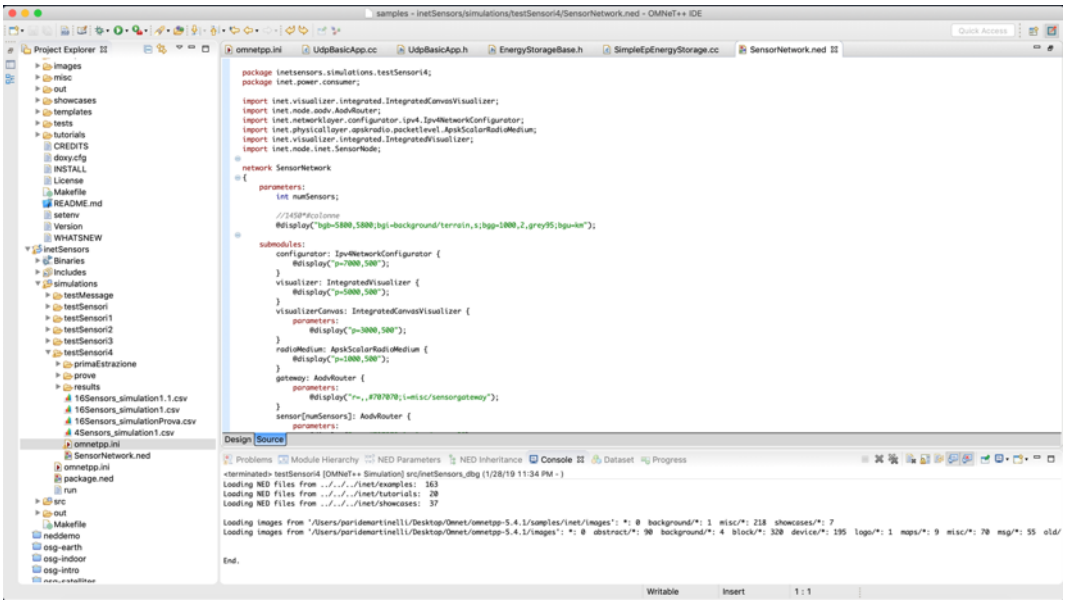

Figura 3.2: NED Editor - Source

Ini File Editor

Ini File Editor consente all'utente di configurare i modelli di simulazione per l'esecuzione. È dotato sia di un editor Figura che testuale.

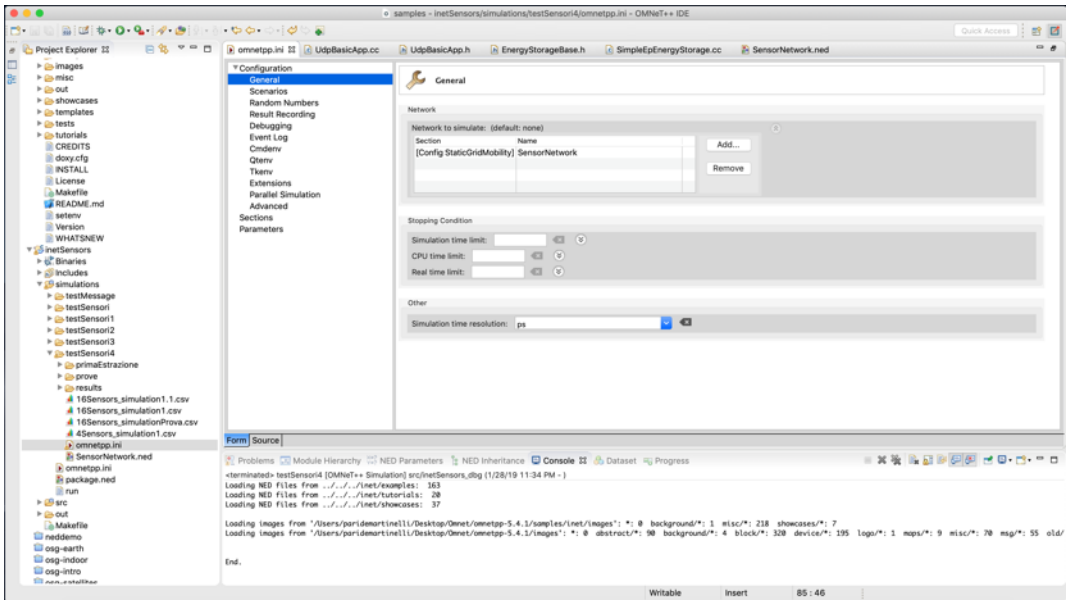

Figura 3.3: Ini File Editor - Form

L'editor Figura del file .ini considera tutte le opzioni di configurazione supportate e le espone in vari form editabili organizzati per argomenti. Descrizioni e valori predefiniti sono visualizzati nelle descrizioni dei comandi, che possono essere resi persistenti per una lettura più semplice. L'editor di testo consente all'utente di lavorare direttamente con il file .ini, che è più efficiente per gli utenti avanzati. L'editor di testo prende in considerazione tutte le dichiarazioni presenti nel file NED (moduli semplici, moduli composti, canali, ecc.) e può sfruttare a pieno queste informazioni per fornire diagnostica e assistenza all'utente.

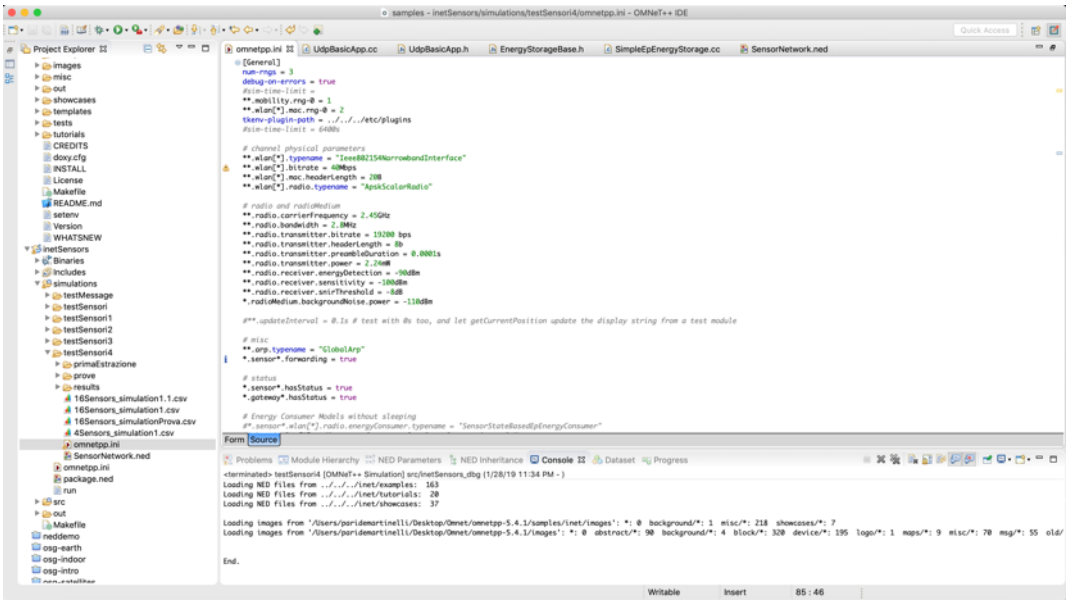

Figura 3.4: Ini File Editor - Code

Simulation Launcher

L'IDE di OMNeT ++ rende possibile eseguire simulazioni direttamente dall'ambiente integrato. È possibile eseguire una simulazione come una normale applicazione C / C ++ ed eseguire il debug del sorgente C ++. Per avviare un programma per la prima volta l'utente crea un'istanza di una configurazione di avvio, compila un modulo e fa clic sul pulsante "Esegui". La finestra di dialogo di avvio della simulazione aiuta l'utente a selezionare parametri che non ha dichiarato nel file .ini, ma obbligatori per la simulazione.

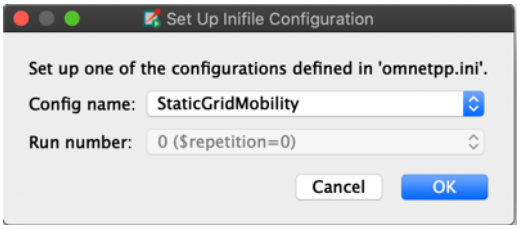

Figura 3.5: Simulation Launcher - Popup

È possibile annullare l'intera operazione batch con un solo clic, se necessario. Le simulazioni vengono eseguite in processi separati che non bloc-

cano l'IDE, quindi gli utenti possono continuare a lavorare mentre le loro simulazioni vengono eseguite in background.

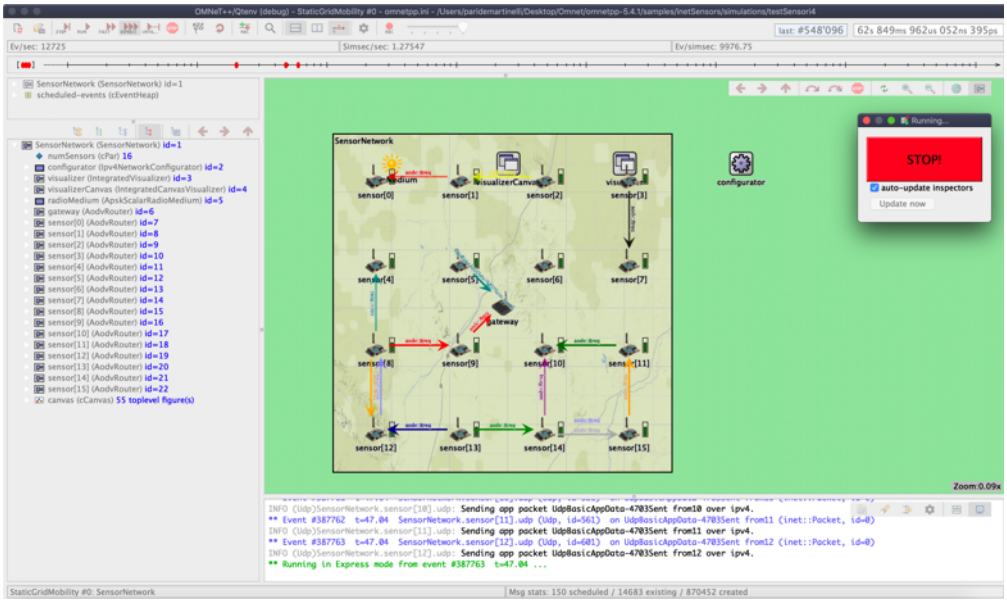

Figura 3.6: Simulation Launcher - IDE

Result Analysis

Scave è lo strumento di analisi dei risultati di OMNeT ++ e il suo compito è di aiutare l'utente a elaborare e visualizzare i risultati della simulazione, salvati in file vettoriali e scalari. Scave è progettato in modo che l'utente possa lavorare sia con l'output di una singola simulazione, sia con gli output di più simulazioni. Tramite una semplice sostituzione dei vecchi risultati con quelli nuovi permetterà una visualizzazione automatica dei grafici con i nuovi dati. Scave è implementato come editor multipagina. La prima pagina visualizza i file dei risultati che servono come input per l'analisi. La metà superiore specifica quali file selezionare, tramite nomi di file espliciti o caratteri jolly. La metà inferiore mostra quali file corrispondono effettivamente alle specifiche di input e quali sono le loro esecuzioni.

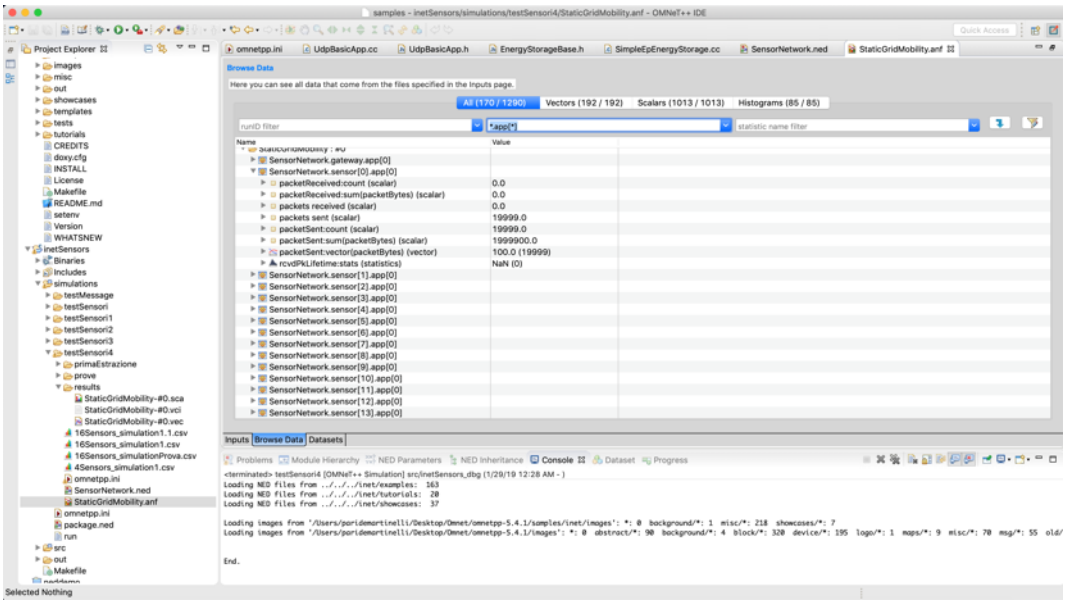

Figura 3.7: Result Analysis - prt. 1

Si noti che i file contenenti i risultati hanno un ID di esecuzione univoco e diverse annotazioni di metadati oltre ai dati registrati effettivi. [58]

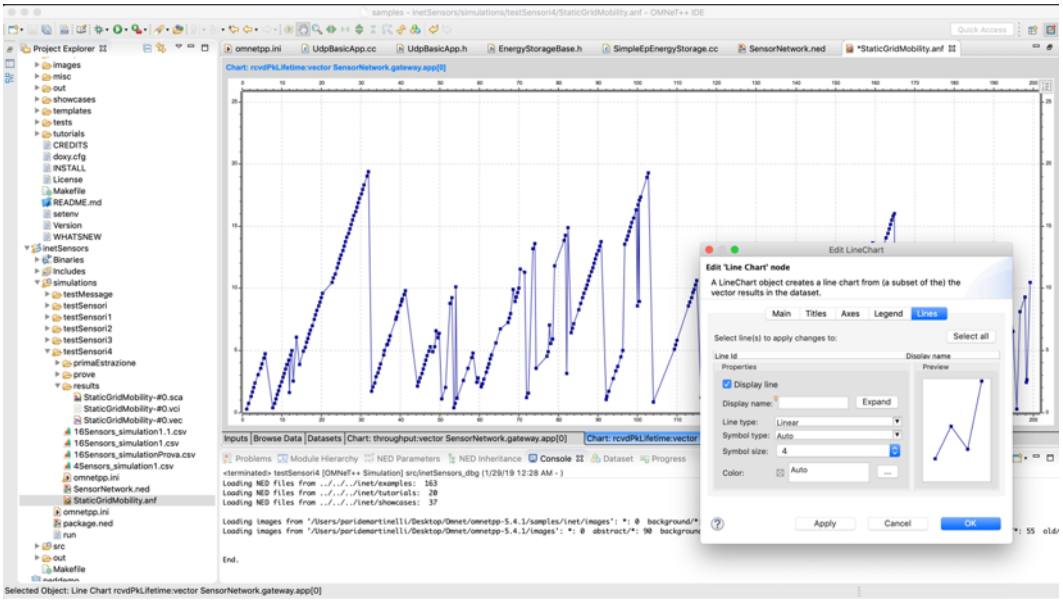

Figura 3.8: Result Analysis - prt. 2

3.3 INET

INET Framework è una libreria di modelli open-source per l'ambiente di simulazione OMNeT++. Fornisce protocolli e modelli per ricercatori e studenti che lavorano con reti di comunicazione. INET è particolarmente utile quando si progettano e si convalidano nuovi protocolli o si esplorano scenari nuovi o atipici. INET contiene modelli per lo stack Internet (TCP, UDP, IPv4, IPv6, OSPF, BGP, ecc.), protocolli di livello di collegamento cablato e wireless (Ethernet, PPP, IEEE 802.11, ecc.), supporto per la mobilità, protocolli MANET, DiffServ, MPLS con segnalazione LDP e RSVP-TE, diversi modelli di applicazione e molti altri protocolli e componenti. Diversi altri framework di simulazione prendono INET come base e lo estendono in direzioni specifiche, come reti veicolari, reti overlay / peer-to-peer o LTE.

INET è costruito attorno al concetto di moduli che comunicano attraverso il passaggio dei messaggi. Agenti e protocolli di rete sono rappresentati da componenti, che possono essere combinati liberamente per formare host, router, switch e altri dispositivi di rete. Nuovi componenti possono essere programmati dall'utente, e i componenti esistenti sono stati scritti in modo che siano facili da capire e modificare. I vantaggi di INET derivano dall'infrastruttura fornita da OMNeT++. Questo framework fa uso dei servizi forniti dal kernel e dalla libreria di simulazione OMNeT, e consente lo sviluppare nuovi modelli, assemblarli, parametrizzarli, eseguirli e valutarne i risultati tramite OMNet++ o tramite riga di comando. [59]

Nelle sezioni successive vediamo nel dettaglio alcune delle principali potenzialità messe a disposizione da INET.

3.3.1 Network Autoconfiguration

Una delle possibili modalità di configurazione di una rete implementate da INET è la rete IPv4. Una rete IPv4 è composta da diversi nodi come

host, router, switch, hub, bus Ethernet o punti di accesso wireless. Configurando la rete con IPv4 vengono assegnati in automatico gli indirizzi IP ai nodi e vengono riempite le loro tabelle di routing[5]. Se viene simulato l'inoltro multicast, vengono compilate anche le tabelle di routing multicast. La configurazione può avvenire anche manualmente, in questo caso ogni indirizzo e percorso viene specificato dall'utente.

Ipv4NetworkConfigurator, ad esempio, assegna gli indirizzi IP in modo del tutto automatico, e auto compila in maniera efficiente le tabelle di routing.

3.3.2 Ad Hoc Routing

Il problema delle reti ad hoc è che i nodi che la popolano non hanno nessuna informazione sugli altri componenti della rete. Questa difficoltà può essere aggravata se i nodi della rete sono mobili. INET mette a disposizione protocolli come AODV, DSDV, DYMO e GPSR che permettono ad ogni nodo di ottenere informazioni sui nodi vicini, quali sono, come raggiungerli e come fargli avere informazioni utili su come essere raggiunti.

AODV

AODV (Ad-hoc Distance Routing vettoriale ad-hoc) è un protocollo di routing per reti mobili ad hoc e altre reti wireless ad hoc. Offre un rapido adattamento alle condizioni di collegamento dinamico, un basso sovraccarico di elaborazione e memoria, un basso utilizzo della rete e determina percorsi ottimali per raggiungere il destinatario della rete.

[5]5. Una tabella di routing (tabella d'instradamento) è un database che viene memorizzato nel router e che tiene traccia dei percorsi dei dati inviati in una data rete. Quando un nodo di una rete deve inviare dati ad un altro nodo deve prima sapere dove inviarli, e se il nodo che deve inviare i dati non è connesso direttamente al nodo di destinazione, invierà il dato al gateway della rete di appartenenza, che poi stabilisce come indirizzare il "pacchetto" dati alla destinazione corretta. Ogni gateway ha quindi bisogno di tenere traccia delle rotte necessarie all'instradamento dei pacchetti di dati e, per questo motivo, si utilizza una tabella di routing. [64]

DSDV

DSDV (Destination-Sequenced Distance-Vector Routing) è uno schema di routing per reti mobili ad hoc basato sull'algoritmo di Bellman Ford per il calcolo del cammino minimo.

DYMO

Il protocollo di routing DYMO (Dynamic MANET On-demand) è il successore del protocollo di routing AODV. DYMO può funzionare sia come protocollo proattivo sia come protocollo di routing reattivo, cioè le rotte possono essere scoperte solo quando sono necessarie e non a prescindere come avviene per AODV.

GPSR

GPSR (Greedy Perimeter Stateless Routing) è un protocollo di routing per reti wireless mobili che utilizza le posizioni geografiche dei nodi per prendere decisioni di inoltro dei pacchetti.

3.3.3 Il modello 802.15.4

IEEE 802.15.4 è uno standard tecnico che definisce il funzionamento delle reti private wireless a bassa velocità (LR-WPAN). IEEE 802.15.4 è stato progettato per una velocità di trasmissione dati massima di 250 kbit, al fine di ottenere una lunga durata della batteria (mesi o anche anni) e una complessità molto bassa. Lo standard specifica il livello fisico e il controllo dell'accesso ai supporti. IEEE 802.15.4 è la base delle specifiche ZigBee, ISA100.11a, WirelessHART, MiWi, SNAP e Thread, ognuna delle quali estende ulteriormente lo standard sviluppando i livelli superiori che non sono definiti in IEEE 802.15.4.

Il framework INET contiene un'implementazione di base del protocollo IEEE 802.15.4 con due diversi tipi di interfacce di rete per il tipo di radio:

1. *Ieee802154NarrowbandInterface* è utilizzabile con radio a banda stretta.

2. *Ieee802154UwbIrInterface* viene utilizzata con la radio UWB-IR.

3.3.4 Livello fisico

Le interfacce di rete wireless contengono un componente del modello radio, responsabile della modellazione del livello fisico (PHY). Il modello radio descrive il dispositivo fisico che è in grado di trasmettere e ricevere segnali sul supporto.

Concettualmente, un modello radio si basa su diversi sotto-moduli (modelli):

- antenna;
- trasmettitore;
- ricevitore;
- errore (come parte del modello del ricevitore);
- consumo energetico.

Il modulo antenna è condiviso tra il modulo trasmettitore e il modulo ricevitore. La separazione tra il modulo del trasmettitore e quello del ricevitore consente configurazioni asimmetriche. Il modulo per il consumo energetico è facoltativo e viene utilizzato solo quando è necessaria la simulazione del consumo di energia.

In INET, i modelli radio implementano l'interfaccia del modulo *IRadio*. Un'implementazione generica e spesso utilizzata di *IRadio* è il tipo *Radio*. Radio contiene i suoi moduli di antenne, trasmettitori, ricevitori e di consumo energetico sotto forma di sotto moduli.

Alcuni dei principali modelli sono ad esempio il *Power Consumption Models*

e l'APSK Radio. Una parte sostanziale del consumo di energia dei dispositivi di comunicazione proviene dalla trasmissione e ricezione di segnali. Il modello di consumo energetico descrive come la radio consuma energia in base alla sua attività. Questo modello è facoltativo (se omesso, il consumo di energia viene ignorato). Esistono due tipi di modelli di consumo energetico:

1. *StateBasedEpEnergyConsumer*: il consumo di energia è determinato dallo stato radio (una combinazione di modalità radio, stato trasmettitore e stato ricevitore) e specificato in parametri come *receiverIdlePowerConsumption* e *receiverReceivingDataPowerConsumption*, in watt;

2. *StateBasedCcEnergyConsumer*: simile al precedente, ma il consumo è dato in ampere.

I modelli radio APSK forniscono un'ipotetica radio che simula una delle ben note modulazioni ASk[6], PSK[7] e QAM[8]. APSK sta per Amplitude e Phase-Shift Keying. La radio APSK può essere mono livello o stratificata. Le varianti mono livello, *ApskScalarRadio* e *ApskDimensionalRadio* trasmettono frame nello schema di modulazione selezionato ma senza utilizzare altre tecniche come forward error correction (FEC), interleaving, spreading, ecc. Le versioni a più livelli, *ApskLayeredScalarRadio* e *ApskLayeredDimensionalRadio* non solo possono modellare le fasi di elaborazione mancanti dalle loro controparti, ma offrono anche un livello di dettaglio configurabile: i moduli trasmettitore e ricevitore hanno parametri levelOfDetail che controllano quali domini sono effettivamente simulati.

[6]Nella modulazione ASK l'ampiezza della portante sinusoidale viene fatta variare in correlazione al segnale digitale modulante. Questo tipo di modulazione, di facile realizzazione, è molto sensibile al rumore, per cui oggi è quasi caduta in disuso. [65]

[7]Nella modulazione PSK ampiezza e frequenza della portante sinusoidale rimangono costanti, mentre è la fase che può subire dei cambiamenti. Questo metodo assicura un buon livello di immunità ai disturbi e consente delle velocità di trasmissione elevate, ma richiede un ricetrasmettitore più complesso di quello necessario per il metodo FSK. [65]

[8]La modulazione QAM si può definire come una modulazione combinata di fase e di ampiezza. La QAM è utilizzata in tutti quei casi in cui la velocità di trasmissione deve essere elevata perché essa permette una codifica multilivello molto spinta. [65]

3.3.5 Modelling Power Consumption

La modellizzazione del consumo di energia diventa sempre più importante con il crescente numero di dispositivi embedded[9] e l'imminente Internet of Things. Dispositivi medici personali mobili, dispositivi di monitoraggio dell'ambiente wireless su larga scala, veicoli elettrici, pannelli solari o sensori wireless a bassa potenza richiedono particolare attenzione al consumo energetico. Considerando il consumo energetico durante gli esprimenti di simulazione è possibile progettare protocolli di routing sensibili al consumo e protocolli MAC con funzionalità di gestione dell'alimentazione, che a loro volta si traducono in dispositivi più efficienti dal punto di vista energetico.

Per aiutare il processo di modellazione, l'INET power model è separato da altri modelli di simulazione. Questa separazione rende questo modello estendibile e consente anche una facile sperimentazione con implementazioni alternative. All'interno di INET questo modello è composto da tre sotto moduli:

1. Consumo energetico;
2. Generazione di energia;
3. Moduli temporanei di accumulo energetico.

Gli elementi del power model si dividono in due categorie abbreviati con Ep e *Cc* come parte del loro nome:

- I moduli Ep sono più semplici e si occupano di quantità di energia e potenza;
- I moduli Cc sono più realistici e si occupano di quantità di carica, corrente e tensione.

[9]Un sistema embedded, in italiano con sistema integrato, nell'informatica e nell'elettronica, identifica genericamente tutti quei sistemi elettronici di elaborazione digitale a microprocessore progettati appositamente per una determinata applicazione ovvero non riprogrammabili dall'utente per altri scopi. [66]

Le seguenti sezioni forniscono una breve panoramica del power model.

Energy Consumer Models
I modelli di consumo energetico descrivono il consumo energetico dei dispositivi nel tempo. Ad esempio, un ricetrasmettitore consuma energia quando trasmette o riceve un segnale, una CPU consuma energia quando il protocollo di rete inoltra un pacchetto e un display consuma energia quando viene acceso. In INET, un modello di consumo energetico è un modulo OMNeT++ semplice che implementa il consumo di energia dei processi software o dei dispositivi hardware nel tempo. Il suo scopo principale è fornire il consumo di energia durante la simulazione. INET fornisce solo alcuni modelli di consumo energetico integrati:

1. *lternatingEpEnergyGenerator* è un modello di consumo energetico semplice;

2. *StateBasedEpEnergyConsumer* è un modello di consumo energetico del ricetrasmettitore basato sulla modalità radio e gli stati di trasmissione / ricezione.

Per simulare il consumo energetico in una rete wireless, il tipo di modello di consumo energetico deve essere configurato per i ricetrasmettitori.

Energy Generator Models
Questi modelli descrivono la generazione di energia dei dispositivi nel tempo. Un pannello solare, ad esempio, produce energia in base al tempo, la posizione del pannello sul globo, l'orientamento verso il sole e le condizioni meteorologiche effettive. I generatori di energia si collegano a un accumulatore di energia che assorbe l'energia generata. In INET, un modello di generatore di energia è un modulo semplice OMNeT++ che implementa la generazione di energia di un dispositivo hardware utilizzando un fenomeno fisico nel tempo. Il suo scopo principale è appunto quello di generare energia durante la simulazione. INET fornisce solo un banale modello per generare

energia, chiamato *AlternatingEpEnergyGenerator*.

Energy Storage Models
I dispositivi elettronici che non sono collegati a una fonte di alimentazione esterna devono contenere alcuni componenti per immagazzinare energia. In INET, un modello di accumulo di energia è un modulo semplice OMNeT ++ che modella i fenomeni fisici utilizzati per immagazzinare l'energia prodotta dai generatori e fornire energia ai consumatori. Il suo scopo principale è calcolare la quantità di energia disponibile durante la simulazione corrente.

I modelli che INET predispone per l'accumulo di energia sono:

- *IdealEpEnergyStorage*: un modello ideale con capacità di energia infinita e flusso di potenza infinito;
- *SimpleEpEnergyStorage*: modello complesso che integra la differenza tra la potenza totale consumata e la potenza totale generata nel tempo;
- *SimpleCcBattery*: modello di batteria più realistico che utilizza una fonte di energia ideale indipendente dalla carica e una resistenza interna.

Energy Management Models
INET mette a disposizione anche un modulo per la gestione dell'energia, che si chiama *SimpleEpEnergyManagement*.

3.3.6 Node Mobility

Per simulare reti wireless ad-hoc, è importante modellare il posizionamento ed il movimento dei nodi. La potenza del segnale ricevuto, l'interferenza del segnale e l'occupazione del canale dipendono dalle distanze tra i nodi. I modelli di mobilità selezionati possono influenzare in modo significativo i risultati della simulazione. Un modello di mobilità descrive la posizione e l'orientamento nel tempo in un sistema di coordinate euclidee 3D. Il suo

scopo principale è quello di fornire dati di posizione, velocità e accelerazione durante il periodo di simulazione.

In INET, un modello di mobilità è un modulo semplice OMNeT ++ che implementa il movimento come un algoritmo C++. INET mette a disposizione i seguenti moduli di mobilità.

Stationary
I modelli stazionari definiscono solo la posizione (e l'orientamento), ma nessun movimento. Sono:

- *StationaryMobility*: fornisce un posizionamento deterministico e casuale;
- *StaticGridMobilit*: posiziona diversi modelli di mobilità in una griglia rettangolare;
- *StaticConcentricMobility*: posiziona diversi modelli in un insieme di cerchi concentrici.

Deterministic
I modelli deterministici di mobilità usano modelli matematici non casuali per descrivere il movimento. Sono:

- *LinearMobility*: si muove linearmente con una velocità costante o un'accelerazione costante;
- *CircleMobility*: si muove su un cerchio parallelo al piano XY a velocità costante;
- *RectangleMobility*: si muove attorno a un'area rettangolare parallela al piano XY a velocità costante;
- *TractorMobility* si muove in modo simile a un trattore su un campo;
- *VehicleMobility* si muove in modo simile a un veicolo generico lungo un percorso;

- *TurtleMobility* si muove secondo un percorso implementato all'interno di uno script XML.

Trace-Based
I modelli di mobilità basati su tracce riproducono un movimento osservato nella vita reale.

Stochastic
I modelli di mobilità stocastica o casuale utilizzano modelli matematici che producono numeri casuali.

- *RandomWaypointMobility* si sposta su una destinazione casuale con velocità casuale;
- *GaussMarkovMobility* utilizza un parametro per variare il grado di casualità del percorso e della velocità;
- *MassMobility* si muove in modo simile a una massa con inerzia e quantità di moto;
- *ChiangMobility* utilizza una matrice di transizione probabilistica per modificare lo stato del movimento.

Combining
Non è un vero e proprio modello di mobilità, ma prevede la combinazione di più modelli.

Applications

A livello applicazione, per gestire il traffico di rete, INET mette a disposizione sia moduli TCP che UDP. In particolare, i moduli UDP che possono essere utilizzati sono:

1. *UdpBasicApp* invia pacchetti UDP ad un dato indirizzo IP ad un dato intervallo temporale;

2. *UdpBasicBurs* invia pacchetti UDP agli indirizzi IP specificati in burs o agisce come un sink di pacchetti;

3. *UdpEchoApp* è simile a *UdpBasicApp*, ma restituisce il pacchetto dopo la ricezione;

4. *UdpSink* consuma e stampa i pacchetti ricevuti dal modulo Udp;

5. *UdpVideoStreamClient* e *UdpVideoStreamServer* simulano lo streaming video su UDP.

3.4 Configurazione rete

La rete che si è cercato di ottenere per compiere l'analisi è una rete che si avvicina il più possibile ad una rete wireless reale che sfrutta tutti gli standard IEEE 802.15.4.

Prima di ottenere la configurazione finale si sono fatte varie prove, utilizzando diversi moduli e modificando i parametri ad essi associati. Ad esempio, si è provato ad utilizzare *ApskScalarRadio* per definire il modulo della radio, e i livelli MAC e *radioMedium* associati alla wlan si sono configurati manualmente attribuendogli parametri ad hoc. Ottenendo però in questo modo dei valori del PDR troppo bassi, inferiori al 30%. Per questo si è deciso di utilizzare moduli INET di default per la wlan, sia a livello mac, che a livello radio, che radioMedium. In particolare, è stato attribuito il modulo *Ieee802154NarrowbandInterface* per definire la wlan, il modulo Ieee802154NarrowbandMac per il livello MAC,
Ieee802154NarrowbandScalarRadio al livello radio, e
Ieee802154NarrowbandScalarRadioMedium al livello radioMedium. Ottenendo in questo modo valori del PDR molto più elevati, addirittura maggiori del 90%.

Per quanto riguarda i moduli di consumo energetico invece si era pensato

inizialmente di utilizzare il modulo *StateBasedEpEnergyConsumer*. Un modulo generico di consumo energetico, provocando un dispendio energetico eccessivo per una rete di sensori ZigBee, abbassando esponenzialmente le prestazioni generali della rete. Si è deciso per questo motivo di sostituirlo con *SensorStateBasedEpEnergyConsumer*, un modulo che eredita le sue funzioanlità direttamente da *StateBasedEpEnergyConsumer*, ma pensato apposta per una rete di sensori.

Si sono riscontrate anche problematiche nel posizionamento dei sensori all'interno della griglia del modulo di mobilità *StaticGridMobility*. Inizialmente si era ipotizzato di posizionare i sensori ad una distanza l'uno dall'altro di 1.450m, questo per essere sicuri che i sensori scambiassero i messaggi solamente con i loro vicini più prossimi. Mettendo i sensori a questa distanza l'uno dall'altro, in uno scenario popolato da 100 sensori, si arriva ad una superficie quadrata di addirittura 14.500m di lato, ed un area complessiva di $210,25 km^2$; senza dubbio una superficie molto estesa per essere applicata ad un contesto reale come un campo agricolo o un percorso di montagna. Si è quindi ridotta la potenza del segnale per poter ridurre a sua volta l'area coperta dai sensori. Lo scenario risultante prevede quindi i sensori a soli 120m di distanza l'uno dall'altro. L'ultima problematica riscontrata è relativa alla randomicità dello scambio dei pacchetti. La prima soluzione non prevedeva randomicità, veniva infatti inviato 1 pacchetto al secondo a partire dal tempo 0. Per compiere simulazioni più precise e per avere una più ampia gamma di valori per l'analisi si è utilizzata la funzione *uniform(s1, s2)* per attribuire i valori a *sendInterval* e a *startTime*. *Uniform* restituisce un valore casuale compreso tra *s1* e *s2* sfruttando la distribuzione uniforme.

Nei capitoli successivi verrà fatta una panoramica più approfondita della rete, dei moduli utilizzati per l'implementazione e dei protocolli scelti.

3.4.1 SensorNetwork.ned

Viene utilizzato il file NED per definire i componenti e assemblarli in modo tale da creare una rete. In particolare, per questo progetto di ricerca viene creata un'unica rete, chiamata *SensorNetwork*, sia per la la rete statica che quella dinamica.

Rete statica
Questa rete viene utilizzata per effettuare la prima fase dell'esperimento, ovvero per analizzare la rete in cui sono presenti solo sensori posizionati sul terreno e un gateway in mezzo alla rete che raccoglie tutti i dati.

Per configurare la rete viene utilizzato il modulo *Ipv4NetworkConfigurator*. Questo modulo assegna ad ogni nodo un indirizzo IPv4 e popola la tabella di routing, ottimizzandola, cercando quindi di creare tabelle di routing più piccole possibili nonostante le grandi dimensioni della rete.
Ipv4NetworkConfigurator non assegna indirizzi o aggiunge percorsi direttamente, ma li memorizza nelle sue strutture dati interne. Assegnando questo modulo alla rete, automaticamente ogni nodo che la popola contiene un'istanza di Ipv4NodeConfigurator che effettivamente configura la tabella di interfaccia del nodo e la tabella di routing. Il configuratore supporta l'assegnazione manuale e automatica degli indirizzi e le loro combinazioni. Supporta sia percorsi manuali che percorsi automatici per l'invio dei pacchetti.

Invece, ogni nodo (sensori e gateway) viene creato attribuendogli il modulo *AodvRouter*. Questo modulo modella un *WirelessHost* esteso con il sottomodulo Aodv. Ogni nodo funziona come un "router": gestisce le proprie tabelle di routing, invia richieste di percorsi e consulta il livello IP per l'inoltro dei dati.

La *SensorNetwok* viene quindi rappresentata dal seguente codice.

```
network SensorNetwork
{
    parameters:
        int numSensors;
        @display("bgb=480,480;bgi=background/terrain,s;bgg=1000,2,grey95;bgu=km");

    submodules:
        configurator: Ipv4NetworkConfigurator {
            @display("p=7000,500");
        }
        visualizer: IntegratedVisualizer {
            @display("p=5000,500");
        }
        visualizerCanvas: IntegratedCanvasVisualizer {
            parameters:
                @display("p=3000,500");
        }
        radioMedium: <> like IRadioMedium {
            @display("p=1000,500");
        }
        gateway: AodvRouter {
            parameters:
                @display("r=,,#707070;i=misc/sensorgateway");
        }
        sensor[numSensors]: AodvRouter {
            parameters:
                @display("r=,,#707070;i=misc/sensor2");
        }
}
```

Rete dinamica

Questa rete viene utilizzata per effettuare il quarto ed ultimo esperimento, ovvero per analizzare la rete in cui sono presenti sensori posizionati sul terreno e il gateway è rappresentato da un nodo mobile, ovvero da un'UAV, e viene sfruttato come mulo di raccolta dati.

La configurazione della rete rimane esattamente la stessa rispetto al caso precedente, invece ogni nodo (sensori e UAV) viene creato attribuendogli il modulo *AdhocHost*. Un host wireless contenente routing, mobilità e componenti energetici. Questo è un tipico nodo mobile che può partecipare al routing ad hoc e può avere installato applicazioni TCP / UDP.

3.4.2 omnetpp.ini

Per poter eseguire la simulazione, è necessario creare un file omnetpp.ini. Questo file indica al programma di simulazione quale rete si desidera simulare e quali parametri o moduli assegnare alla rete.

Rete

La rete utilizzata in questo progetto, per lo scambio dei pacchetti, utilizza una connessione wireless che sfrutta IEEE 802.15.4. In particolare, viene utilizzato il modulo *Ieee802154NarrowbandInterface*.

```
**.wlan[*].typename = "Ieee802154NarrowbandInterface"
```

Data questa configurazione della wlan, vengono poi settati a livello MAC di default del modulo Ieee802154NarrowbandMac.

```
**.wlan[*].mac.typename = "Ieee802154NarrowbandMac"
```

Ovvero:

```
simple Ieee802154NarrowbandMac extends Ieee802154Mac
{
    parameters:
        useMACAcks = true;
        rxSetupTime = 0s;
        mtu = 127 Byte - 9 Byte;
```

```
7
8           // length of MAC header
9           headerLength = default(72 b);
10
11          // Exponential Backoff
12          backoffMethod = default("exponential");
13
14          // Maximum number of frame retransmission
15          macMaxFrameRetries = default(3);
16
17          // Maximum number of extra backoffs (excluding the ↩
                first unconditional one) before frame drop
18          macMaxCSMABackoffs = default(4);
19
20          // Minimum backoff exponent
21          macMinBE = default(3);
22
23          // Maximum backoff exponent
24          macMaxBE = default(5);
25  }
```

Questi moduli vengono implementati entrambi in due file .ned che e si trovano nella cartella al percorso inet/linklayer/ieee802154 di INET.

Livello fisico

Per definire il livello fisico viene utilizzato il modulo *Ieee802154NarrowbandScalarRadio* per definire la radio, e *Ieee802154NarrowbandScalarRadioMedium* per definire il radio medium. Due moduli radio implementati apposta per una rete Iee802154.

```
1  **.wlan[*].radio.typename = "Ieee802154NarrowbandScalarRadio"
2  *.radioMedium.typename = "Ieee802154NarrowbandScalarRadioMedium↩
       "
```

Anche in questo caso vengono lasciati i parametri di default, ovvero:

```
module Ieee802154NarrowbandScalarRadio extends FlatRadioBase
{
    parameters:
        antenna.typename = default("IsotropicAntenna");
        transmitter.typename = default("↩
            Ieee802154NarrowbandScalarTransmitter");
        receiver.typename = default("↩
            Ieee802154NarrowbandScalarReceiver");

        carrierFrequency = 2450 MHz;
        bandwidth = default(2.8 MHz);

        *.bitrate = default(250 kbps);

        // PHY Header (without preamble)
        *.headerLength = (1*8 + 7 + 1) * 1 b;

        // Preamble
        transmitter.preambleDuration = (4*8/4) * 16 us;

        // RSSI sensitivity
        receiver.energyDetection = default(-90dBm);

        // Receiver sensitivity (ATmega256RFR2, page 565)
        receiver.sensitivity = default(-100dBm);
        receiver.minInterferencePower = default(-120dBm);

        // Minimum SNIR
        receiver.snirThreshold = default(-8 dB);

        // TX Output power
        transmitter.power = default(2.24mW);

        @class(FlatRadioBase);
}
```

```
module Ieee802154NarrowbandScalarRadioMedium extends RadioMedium
{
    parameters:
        propagation.typename = default("ConstantSpeedPropagation");
        analogModel.typename = default("ScalarAnalogModel");
        backgroundNoise.typename = default("IsotropicScalarBackgroundNoise");

        mediumLimitCache.carrierFrequency = 2450 MHz;

        pathLoss.typename = default("BreakpointPathLoss");
        pathLoss.breakpointDistance = 8 m;
        pathLoss.l01 = 40.2;
        pathLoss.alpha1 = 2;
        pathLoss.l02 = 58.5;
        pathLoss.alpha2 = 3.3;

        backgroundNoise.power = default(-96.616dBm);
        backgroundNoise.dimensions = default("time");
}
```

Modello di consumo energetico

Ogni sensore, per renderlo più verosimile, viene equipaggiato con un modulo di energy storage. Ogni sensore infatti parte con 1J di energia.

```
*.sensor[*].energyStorage.typename = "SimpleEpEnergyStorage"
*.sensor[*].energyStorage.nominalCapacity = 1J
*.sensor[*].energyStorage.initialCapacity = 1J
```

INET implementa il modulo *SimpleEpEnergyStorage*, nel file SimpleEpEnergyStorage.ned all'interno della cartella inet/power/storage.

Per rendere ancora più reale la rete, ad ogni sensore viene assegnato un

modulo di consumo energetico. E per ridurre al minimo questo consumo, affinché si allunghi la vita della rete e per ottimizzare il processo viene utilizzato il modulo *SensorStateBasedgEpEnergyConsumer*.

```
*.sensor*.wlan[*].radio.energyConsumer.typename = "SensorStateBasedEpEnergyConsumer"
*.sensor[*].energyManagement.typename = "SimpleEpEnergyManagement"
*.sensor[*].energyManagement.nodeShutdownCapacity = 0J
*.sensor[*].energyManagement.nodeStartCapacity = 1J
```

Come per la wlan e come per i moduli della radio, anche i parametri del modulo utilizzato per il consumo energetico vengono lasciati quelli di default.

```
simple SensorStateBasedEpEnergyConsumer extends StateBasedEpEnergyConsumer
{
    parameters:
        offPowerConsumption = default(0mW);
        sleepPowerConsumption = default(0.001mW);
        switchingPowerConsumption = default(25mW);
        receiverIdlePowerConsumption = default(0.005mW);
        receiverBusyPowerConsumption = default(0.1mW);
        receiverReceivingPowerConsumption = default(50mW);
        transmitterIdlePowerConsumption = default(5mW);
        transmitterTransmittingPowerConsumption = default(75mW);
}
```

Livello app

Come protocollo è stato scelto UDP, infatti *UdpBasicApp* invia i pacchetti all'indirizzo IP specificato. *UdpBasicApp* è compatibile con Ipv4 e Ipv6 e viene implementato nel file UdpBasicApp.ned all'interno di inet/applications/udpapp. L'intervallo di invio dei pacchetti può essere un valore costante o casuale. Se il parametro *destAddresses* contiene più di un indirizzo, uno

di questi viene scelto casualmente per ciascun pacchetto. Ai sensori che popolano la rete è quindi stato attribuito questo modulo a livello app.

```
# sensor app
*.sensor[*].numApps = 1
*.sensor[*].app[0].typename = "UdpBasicApp"
*.sensor[*].app[0].destAddresses = "gateway"
*.sensor[*].app[0].destPort = 1000
*.sensor[*].app[0].sendInterval = uniform(0.5s, 1.5s)
*.sensor[*].app[0].startTime = uniform(0s, 1s)
*.sensor[*].app[0].messageLength = 50Byte
```

Dove *uniform()* è una funzione che restituisce un valore random compreso tra i due dati in input. Scelta presa per dare randomicità alla rete.

Anche al gateway è stato attribuito lo stesso modulo.

```
# gateway app
*.gateway.numApps = 1
*.gateway.app[0].typename = "UdpBasicApp"
*.gateway.app[0].localPort = 1000
*.gateway.app[0].destPort = 1000
*.gateway.app[0].sendInterval = 0.01s
*.gateway.app[0].messageLength = 50Byte
```

Mobilità dei nodi

Come descritto nel capitolo 3.1 sono stati utilizzati due moduli di mobilità. Un modulo statico, ovvero *StaticGridMobility*, e un modulo dinamico *TractorMobility*.

Il modulo *StaticGridMobility*, implementato in StaticGridMobility.ned, mette tutti i nodi della rete in una griglia rettangolare.

```
# sensors mobility
**.sensor*.mobility.typename = "StaticGridMobility"
**.sensor*.mobility.constraintAreaMinX = 0m
**.sensor*.mobility.constraintAreaMinY = 0m
**.sensor*.mobility.constraintAreaMaxX = 480m
**.sensor*.mobility.constraintAreaMaxY = 480m
**.sensor*.mobility.separationX = 120m
**.sensor*.mobility.separationY = 120m
**.sensor*.mobility.numHosts = 16
**.sensor*.mobility.columns = 4

# gateway mobility
**.gateway*.mobility.typename = "StaticGridMobility"
**.gateway*.mobility.constraintAreaMinX = 0m
**.gateway*.mobility.constraintAreaMinY = 0m
**.gateway*.mobility.constraintAreaMaxX = 480m
**.gateway*.mobility.constraintAreaMaxY = 480m
**.gateway*.mobility.numHosts = 1
**.gateway*.mobility.columns = 1
```

L'esempio di codice riportato viene utilizzato per una simulazione con 16 nodi, e per creare una griglia quadrata il numero di *columns* è stato quindi settato a 4. *separationX* e *separationY* viene impostato a 120m, in questo modo ogni sensore è posizionato in modo tale che possa inviare pacchetti solamente ai suoi più prossimi vicini e non ad un qualunque sensore della rete.

La *TractorMobility* invece simula quello che è il movimento di un trattore, un movimento per così dire a "S" come mostrato in figura 9.

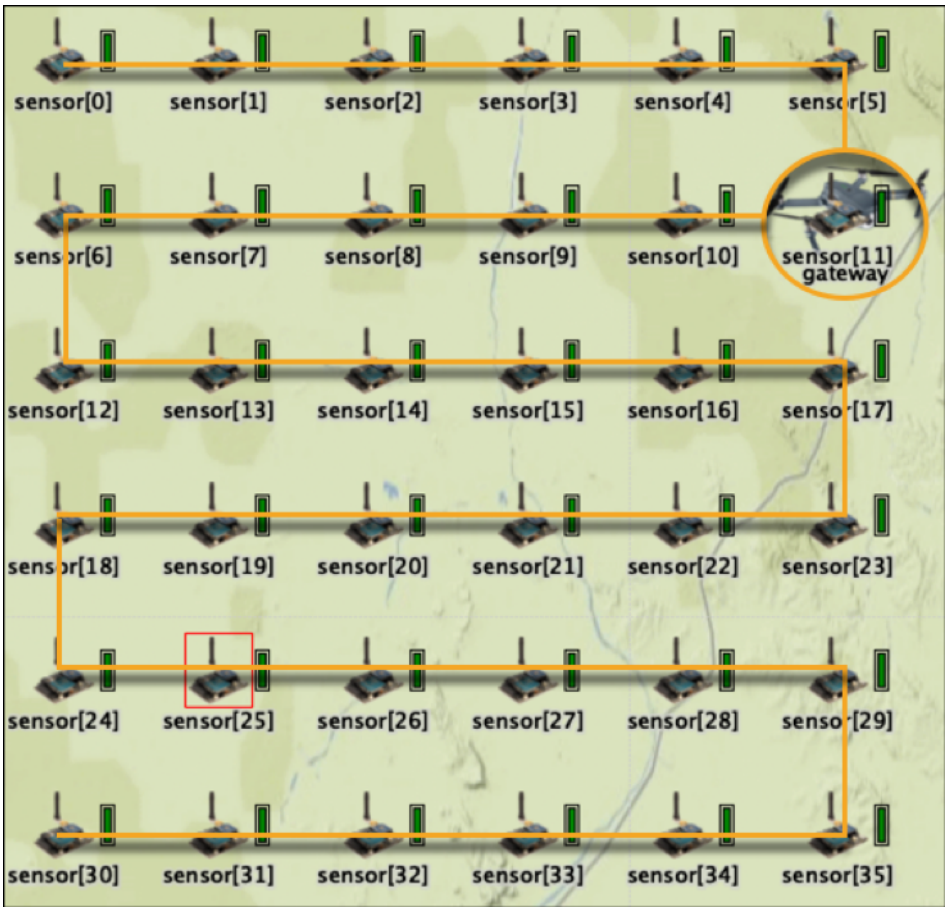

Figura 3.9: Tractor Mobility

Questo modulo di mobilità richiede, per essere inizializzato, la dimensione dell'area in cui si deve muovere il nodo mobile, il numero di righe su cui si deve muovere e la velocità con cui si muove il nodo.

```
1 **.UAV.mobility.typename = "TractorMobility"
2 **.UAV.mobility.constraintAreaMinX = 0m
3 **.UAV.mobility.constraintAreaMinY = 0m
4 **.UAV.mobility.constraintAreaMaxX = 720m
5 **.UAV.mobility.constraintAreaMaxY = 720m
6 **.UAV.mobility.x1 = 60m
7 **.UAV.mobility.y1 = 60m
8 **.UAV.mobility.x2 = 680m
```

```
**.UAV.mobility.y2 = 680m
**.UAV.mobility.rowCount = 5
**.UAV.mobility.speed = 5mps
```

Capitolo 4

Analisi e risultati

Nel quarto ed ultimo capitolo vengono analizzati i dati estratti. In particolare, verranno spiegate le metriche di analisi, la logica con cui vengono fatti i diversi esperimenti e per ogni esperimento vengono mostrati i dati ricavati dalle varie simulazioni effettuate per compiere l'analisi. Infine, i risultati verranno analizzati e commentati allo scopo di migliorare quanto più possibile le prestazioni della rete WSN, soprattutto in termini di packet delivery ratio e di durata complessiva della rete.

4.1 Metriche di analisi

Come appena accennato, lo scopo di questo progetto di tesi è quello di massimizzare il numero dei pacchetti che, partendo da un dato sensore, riescono ad arrivare al gateway di destinazione e contemporaneamente massimizzare la durata della rete, quindi cercare di ridurre il più possibile il dispendio energetico del singolo sensore. Per ottenere una configurazione di questo genere si sono ovviamente tenuti in considerazione anche valori come il delay e il throughput oltre che il packet delivery ratio e la lifetime.

4.1.1 Delay

Il primo parametro che viene studiato è il delay. Il delay, tempo di latenza, viene definito come l'intervallo di tempo che intercorre fra il momento in cui viene generato un pacchetto dal nodo e il momento in cui arriva a destinazione. In altre parole, il delay non è altro che la misura della velocità di un sistema. Minore è il tempo impiegato dal pacchetto ad arrivare al gateway, a partire dalla sua creazione all'interno del sensore, e maggiore è la velocità della rete WSN.

Nel sistema implementato il delay del singolo pacchetto viene calcolato sottraendo l'istante in cui viene creato il pacchetto al tempo in cui arriva a destinazione. L'oggetto Packet di INET mette a disposizione il metodo *getCreatinTime()* per recuperare l'istante della simulazione in cui viene creato il pacchetto, e il metodo *getArrivalTime()* per recuperare invece l'istante in cui arriva al gateway.

```
1 simtime_t creationTime = pk->getCreationTime();
2 simtime_t arrivalTime = pk->getArrivalTime();
3 simtime_t delay = arrivalTime - creationTime;
```

In questo studio vene analizzato il delay relativo al singolo sensore e all'intera rete. Per calcolare iI delay di un dato sensore si sono sommati i singoli delay relativi ai pacchetti da esso inviati e la risultante somma si è divisa per il numero totale di pacchetti arrivati a destinazione. Per calcolare il delay dell'intera rete invece si sono sommati i delay di tutti i sensori che popolano la rete per poi dividere il risultato per il numero di pacchetti ricevuti dal gateway.

4.1.2 Throughput

Il secondo valore che viene preso in considerazione per compiere l'analisi è il throughput. Questo parametro indica la capacità di trasmissione effetti-

vamente utilizzata da un canale di comunicazione. Il throughput è infatti la quantità di dati trasmessi in una unità di tempo.

Anche questo parametro, come il delay, viene calcolato sia a livello del singolo sensore, sia a livello dell'intera rete. Per calcolare il throughput dell'intera rete, ad esempio, si è moltiplicato il numero di pacchetti ricevuti per la dimensione del singolo pacchetto, e il risultato così ottenuto lo si è diviso per la durata della simulazione. In questo modo il throughput viene calcolato in bytes/secondi. INET mette a disposizione la funzione *simTime()* che ritorna la durata della simulazione, utilizzata appunto per definire il tempo di fine simulazione. Il numero di pacchetti ricevuti invece viene recuperato direttamente dal modulo UdpBasicApp del nodo di pertinenza.

```
1 double THR = (numReceived * 50) / simTime();
```

4.1.3 PDR

Il packet delivery ratio (PDR) di una rete di sensori è il rapporto tra il numero di pacchetti inviati e il numero di pacchetti che effettivamente arrivano a destinazione. Maggiore è il PDR e migliori sono le performance della rete in quanto vuol dire che un buon numero dei pacchetti inviati sono anche arrivati a destinazione. Più precisamente se il PDR tende al 100% vuol dire che tutti o quasi i pacchetti inviati dai vari sensori che popolano la rete sono arrivati al gateway.

Questo parametro, che identifica le performance dell'intera rete, viene per ovvie ragioni calcolato solo a livello di rete e non a livello del singolo sensore.

```
1 int DPR = (numReceived / sentTot) * 100;
```

4.1.4 Durata

L'ultimo parametro che viene studiato è la lifetime della rete. La lifetime è definita come la vita dell'intera rete. Uno dei principali scopi di questa tesi è cercare di massimizzare la durata della rete.

I sensori che popolano una rete WSN hanno purtroppo un grande limite, quello energetico. I sensori sono dotati di una batteria limitata e non è facile ricaricarla o sostituirla. Inoltre il continuo invio e inoltro di pacchetti contribuisce ad un rapido consumo energetico. Scaricandosi, i sensori si spengono e spegnendosi interrompono il flusso di scambio dei pacchetti. Interrompendo questo flusso il gateway cessa di ricevere i messaggi e la rete si interrompe. Quando il gateway non riceve più nessun pacchetto vuol dire il ciclo di vita della rete si è concluso.

Riuscire a minimizzare questo dispendio di energia e massimizzare la durata della rete è uno dei principali casi di studio presenti in letteratura. Nei successivi capitoli viene spiegato come questa tesi cerca di dare il suo contributo in merito a questa grande problematica.

4.2 Esperimenti

Prendendo in considerazione le metriche analizzate nei capitoli precedenti si è fatta una prima analisi variando il numero dei sensori che popolano la rete, per vedere come cambiano questi parametri al variare della densità della rete. Una volta trovata la configurazione con il numero ottimale di sensori si sono fatti diversi esperimenti variando il traffico generato. Infine, dopo aver capito come cambiano le metriche di analisi al variare dell'intervallo con cui vengono inviati i messaggi, si è cercato di massimizzare la durata della rete: prima aggiungendo il meccanismo di sleep and wake up e poi introducendo il nodo mobile.

4.2.1 Esperimento 1: Dimensione della rete

Il primo esperimento effettuato consiste nel variare il numero di nodi che compongono la rete per vedere come variano le metriche prese in considerazione, quali delay, throughput, PDR e lifetime al variare della densità dei sensori. I nodi che compongono la rete sono posizionati all'interno di una griglia quadrata; quindi per ogni simulazione effettuata sono state incrementate il numero delle righe e il numero delle colonne che formano la griglia. Partendo da una griglia 2x2, fino ad arrivare ad una griglia 12x12. Per questo esperimento sono state quindi utilizzate griglie quadrate di 4, 16, 36, 64, 100 nodi e 144 nodi. Ed essendo i nodi ad una distanza l'uno dall'altro di 120m, si arriva ad ottenere una griglia massima di 2.073.600m^2 ($2,07km^2$) con la configurazione di 144 sensori.

Numero sensori	Lato (m)	Area (m^2)
4	240	57.600
16	480	230.400
36	720	518.400
64	960	921.600
100	1.200	1.440.000
144	1.440	2.073.600

Tabella 4.1: Area di copertura della rete WSN

Per questo primo esperimento l'intervallo con cui ogni sensore invia pacchetti è stato definito secondo una funzione che restituisse un valore random compreso tra 0,5s e 1,5s; per una maggiore precisione, per ogni configurazione sono state effettuate circa trenta simulazioni. I valori medi ottenuti sono mostrati nella Tabella 4.2.

Numero sensori	Pacchetti inviati	Pacchetti ricevuti	Delay (s)	Throughput (byte/s)	PDR (%)	Tempo simulazione (secondi)	Durata reale rete (giorni)
4	3.992	3.991	0,01	200,25	100%	997	44,98
16	2.784	2.762	0,02	794,28	99%	174	7,85
36	2.706	2.519	0,21	1680,12	93%	75	3,38
64	3.347	1.955	1,03	1874,76	58%	52	2,35
100	4.963	1.915	1,31	1926,47	39%	50	2,24
144	6.429	1.700	1,41	1904,26	26%	45	2,01

Tabella 4.2: Analisi delle metriche al variare della densità

Come è possibile notare dalla tabella, delay e throughput aumentano all'aumentare nel numero dei nodi, al contrario del PDR e della durata che diminuiscono drasticamente all'aumentare dei nodi.

Il delay aumenta progressivamente in quando all'aumentare del numero dei nodi aumenta il numero di pacchetti inviati ed essendo sempre e solo uno il gateway di destinazione aumenta la congestione. Inoltre, aumenta la distanza tra i sensori, che stanno ai lati della griglia, e il gateway che rimane sempre posizionato al centro dell'area. Quindi, aumentando la distanza, si allunga il percorso che devono compiere i pacchetti e allungandosi il percorso anche il tempo che impiega il pacchetto ad arrivare a destinazione aumenta. In ogni caso, come possiamo vedere dai dati della Tabella 4.2 e dalla Figura 4.1, i valori del delay rimangono comunque abbastanza bassi, anche con 144 nodi.

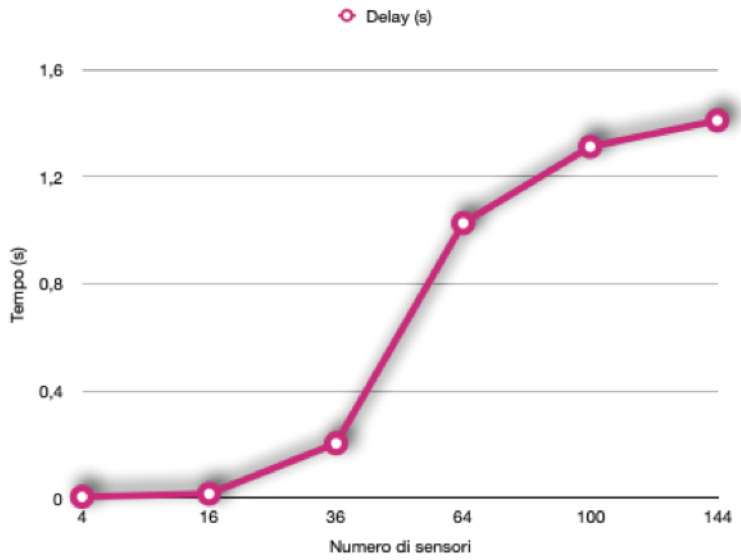

Figura 4.1: Delay al variare della densità dei sensori

Il throughput aumenta linearmente fino alla configurazione con 36 nodi, poi inizia ad aumentare sempre più lentamente fino a trovare un suo equilibrio. Questo vuol dire che la capacità del canale di comunicazione aumenta costantemente fino a 36 sensori. In ogni simulazione, visto che la curva, mostrata nella Figura 4.2, non decresce mai, il canale di comunicazione riesce sempre a sostenere la quantità di traffico generato dall'invio di circa 1 pacchetto al secondo.

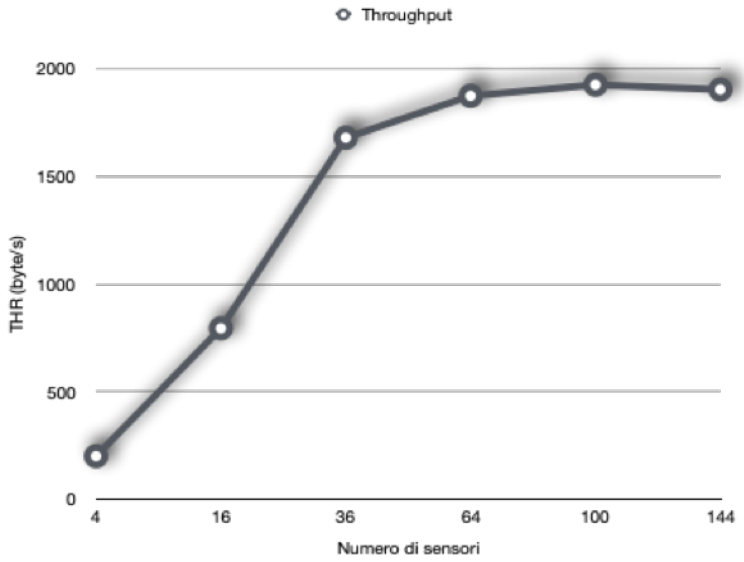

Figura 4.2: Throughput al variare della densità dei sensori

Le problematiche si verificano analizzando il PDR e la lifetime. Infatti, se mettiamo a confronto il numero dei pacchetti inviati con il numero dei pacchetti che effettivamente arrivano destinazione possiamo notare un leggero dislivello per gli scenari con un massimo di 16 e 36 nodi e un notevole dislivello per gli scenari più densi con più di 64 nodi.

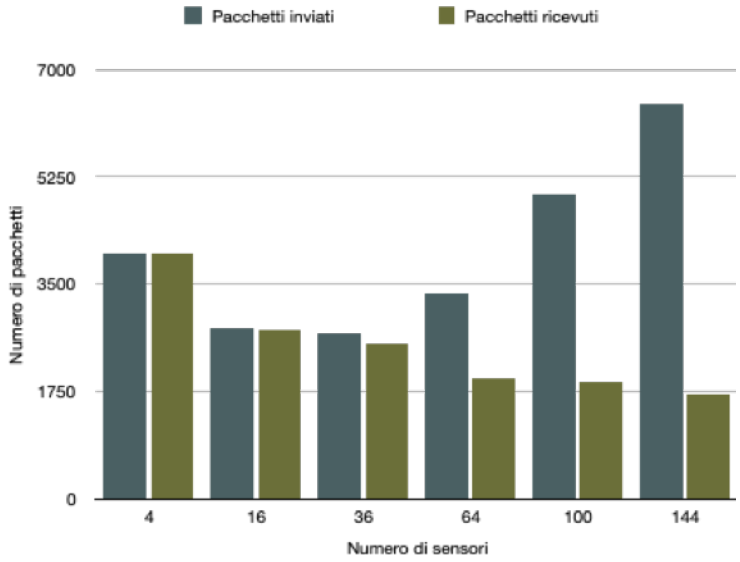

Figura 4.3: Confronto pacchetti inviati e ricevuti al variare della densità

Il numero di pacchetti inviati e ricevuti è calcolato per il tempo di simulazione. Per compiere ogni simulazione effettuata in fase di analisi, è stata assegnata ad ogni sensore una batteria di 1J. Una batteria molto limitata in modo da scaricare la rete più velocemente e riuscire a compiere un numero maggiore di simulazioni. La durata effettiva della rete è stata poi calcolata moltiplicando la durata della simulazione per 3900, in quanto una batteria di un sensore è di circa 3.900J. Nella Tabella 4.2 vengono infatti riportati, nella colonna "Tempo di simulazione" la durata della simulazione in secondi, e nella colonna "Durata" la durata effettiva della rete (in giorni) , quindi i valori di "Tempo di simulazione" moltiplicati per 3.900.

Da quanto detto in merito alla Figura 4.3 ne consegue che il PDR si abbassi progressivamente all'aumentare del numero dei sensori.

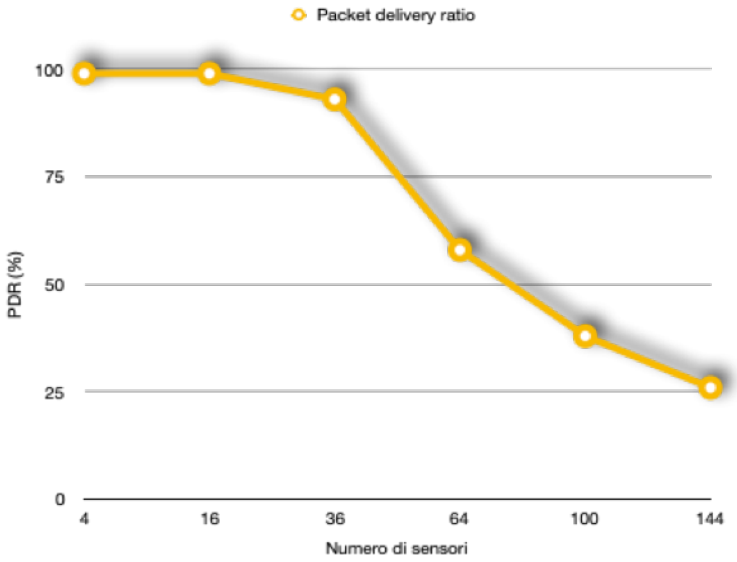

Figura 4.4: PDR al variare della densità dei sensori

L'abbassarsi del PDR è causato dalla progressiva diminuzione del tempo di vita della rete. Infatti, come si può notare dalla Figura 4.5 la durata della simulazione si abbassa all'aumentare del numero di sensori.

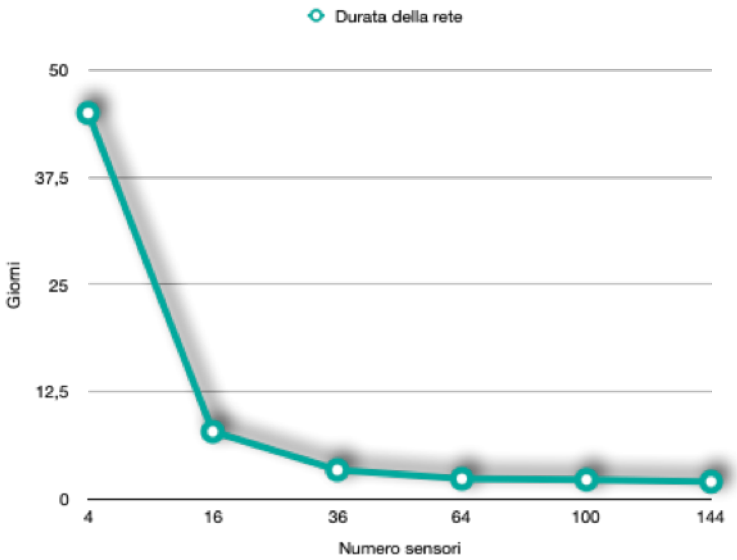

Figura 4.5: Durata al variare della densità dei sensori

Aumentando il numero di nodi e mantenendo costante l'intervallo con cui vengono generati i pacchetti, si ha un progressivo aumento del numero totale di pacchetti che vengono generati in un dato secondo. Quindi aumentando il numero di pacchetti che circola all'interno della rete in un certo intervallo di tempo aumenterà anche il dispendio energetico del singolo sensore che dovrà inoltrare sempre un maggior numero di pacchetti. Aumentando il consumo energetico, soprattutto per i nodi a stretto contatto con il gateway, si avrà una minor durata della batteria e quindi una durata della vita della rete che si riduce all'aumentare dei nodi. Accorciandosi, la vita della rete, si riduce la possibilità che i pacchetti riescano a compiere l'intero percorso, e quindi si riduce la possibilità che i pacchetti riescano ad arrivare a destinazione. Per questo il PDR si riduce all'aumentare della densità di nodi che popolano la rete.

Prendendo come riferimento la Tabella 4.2, che ci da una visione completa di tutti i risultati ottenuti, possiamo affermare che la configurazione con 36 nodi è quella che porta i risultati migliori: una buona densità di nodi, un PDR al 93%, un delay inferiore ad 1s e una buona capacità di trasmissione. Il valore che si potrebbe migliore è la durata, infatti rispetto alle configurazioni meno dense di sensori, la rete composta da 36 nodi presenta una durata complessiva ridotta, una durata di soli 3 giorni, a differenza della rete composta da soli 4 sensori che dura più di 44 giorni. Nei capitoli successivi infatti utilizzeremo delle strategie per cercare di prolungare la lifetime della rete. Oltre al tentativo di migliorare il delay cercheremo anche di incrementare il PDR.

Prima di passare agli esperimenti successivi diamo uno sguardo in maniera più approfondita alla rete composta da 36 sensori.

Sensore	Pacchetti inviati	Pacchetti consegnati	Delay (s)	Throughput (byte/s)	Distanza (m)
0	77	72	1,04	1,03	424,26
1	78	75	0,08	1,04	349,86
2	76	73	0,06	1,01	305,94
3	71	67	0,31	0,95	305,94
4	78	74	0,09	1,04	349,86
5	76	71	0,11	1,01	424,26
6	80	74	0,08	1,07	349,86
7	77	74	0,07	1,03	254,56
8	73	68	0,02	0,97	189,74
9	72	70	0,04	0,96	189,74
10	75	69	0,09	1,00	254,56
11	75	70	0,09	1,00	349,86
12	78	72	0,18	1,04	305,94
13	76	73	0,02	1,01	189,74
14	73	71	0,01	0,97	84,85
15	74	72	0,01	0,99	84,85
16	73	68	0,09	0,97	189,74
17	75	73	0,10	1,00	305,94
18	80	73	0,60	1,07	305,94
19	78	71	0,08	1,04	189,74
20	78	78	0,01	1,04	84,85
21	70	69	0,01	0,93	84,85
22	79	76	0,041	1,05	189,73
23	72	66	0,10	0,96	305,94
24	75	64	0,17	1,00	349,86
25	76	66	0,43	1,01	254,56
26	78	72	0,24	1,04	189,74
27	72	68	0,02	0,96	189,74
28	76	70	0,30	1,01	254,56

Tabella 4.3: Analisi delle metriche sulla rete con 36 nodi - prt. 1

29	75	70	0,40	1,00	349,86
30	75	57	0,54	1,00	424,26
31	71	64	0,54	0,94	349,86
32	73	67	0,40	0,97	305,94
33	70	64	0,14	0,93	305,94
34	75	70	0,44	1,00	349,86
35	76	68	0,59	1,01	424,26

Tabella 4.4: Analisi delle metriche sulla rete con 36 nodi - prt. 2

Il primo dato che ci forniscono le Tabelle 4.3 e 4.4 è il confronto tra il numero dei pacchetti inviati dal singolo sensore e il numero dei pacchetti che effettivamente arrivano a destinazione.

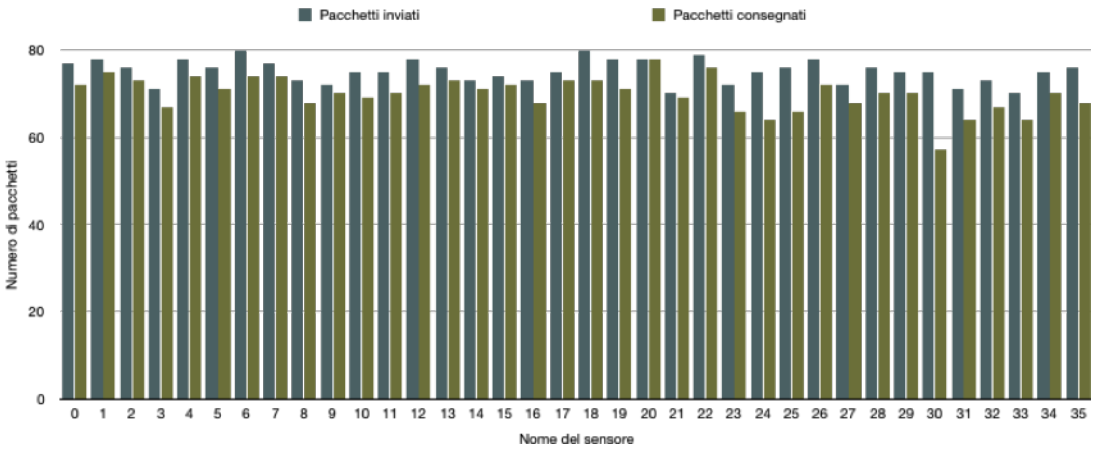

Figura 4.6: Confronto pacchetti inviati e ricevuti della rete con 36 nodi

Come possiamo vedere con questa configurazione vi è una piccolissima perdita di pacchetti. Quasi tutti i pacchetti generati riescono anche ad arrivare a destinazione. Questo dimostra il PDR al 93% citato precedentemente.

La seconda cosa che possiamo notare è che i ritardi di consegna non sono lineari. Il sensore 0, e il sensore 35 hanno un delay medio maggiore di 1 secondo. A differenza del sensore 21 ad esempio che ha un ritardo di 0,9

secondi. Questa differenza è dovuta dalla distanza tra il sensore e il gateway. Infatti, il sensore 21, trovandosi a soli 84,8m dal gateway ci impiegherà meno tempo a spedire i suoi pacchetti. In ogni caso queste differenze sono poco percettibili in quanto la densità della rete è comunque limitata e i tragitti sono relativamente brevi. Questa differenza la si nota meglio in reti più dense, ad esempio con 100 nodi, in cui la distanza dei nodi che si trovano alle estremità della griglia dal gateway è molto elevata, fino ad arrivare a 700m di distanza con un ritardo di quasi 3 secondi. Un ritardo tre volte più elevato rispetto ai nodi che si trovano a stretto contatto con il gateway.

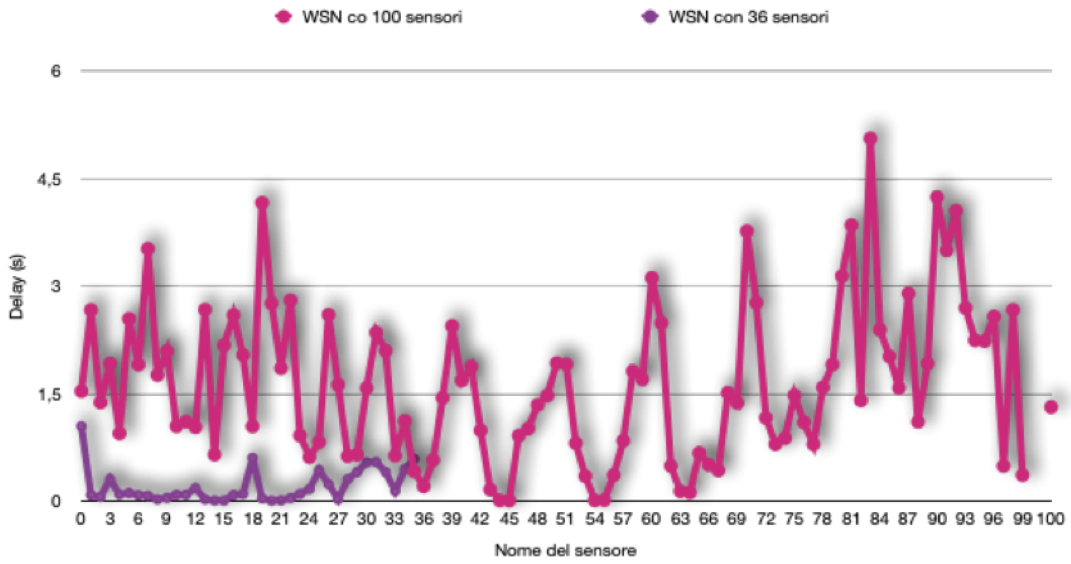

Figura 4.7: Confronto delay con 36 e 100 nodi

4.2.2 Esperimento 2: Generazione di traffico

Dalle analisi fatte nel primo esperimento si è dimostrato che la configurazione con 36 nodi risulta essere quella ottimale. Quindi è stato preso questo scenario per effettuare delle analisi al modificare della quantità di traffico generato in un dato secondo. Per compiere questo esperimento si è modificato il valore assegnato al *sendInterval* del modulo UdpBasicApp. Il *sendInterval* stabilisce quale è l'intervallo di frequenza con cui vengono inviati i pacchetti.

La Tabella 4.5 descrive gli intervalli di tempo utilizzati e i valori con essi ottenuti.

Numero sensori	sendInterval (s)	Pacchetti inviati	Pacchetti ricevuti	Delay (s)	Throughput (byte/s)	PDR (%)	Tempo simulazione (secondi)	Durata reale rete (giorni)
36	0,1	11.382	1.766	1,59	2.748,34	16%	32,13	1,45
36	0,2	6.815	1.987	2,06	2.592,45	29%	38,32	1,73
36	0,3	4.704	1.873	1,70	2.364,36	40%	39,61	1,79
36	0,4	4.278	2.006	1,53	2.093,75	47%	47,90	2,16
36	0,5	3.516	2.218	0,92	2.258,80	63%	49,10	2,22
36	0,6	3.043	2.082	0,83	2.041,74	68%	50,99	2,30
36	0,7	2.860	2.377	0,35	2.128,38	83%	55,84	2,52
36	0,8	2.741	2.437	0,28	1.994,37	89%	61,10	2,76
36	1	2.690	2.482	0,20	1.659,17	92%	74,80	3,38
36	1,1	2.647	2.433	0,23	1.503,96	92%	80,89	3,65
36	1,2	2.986	2.929	0,05	1.474,25	98%	99,34	4,48
36	1,3	3.065	2.972	0,07	1.341,39	97%	110,78	5,00
36	1,4	2.893	2.789	0,06	1.240,11	96%	112,45	5,08
36	1,5	3.186	3.126	0,03	1.179,07	98%	132,56	5,98
36	1,6	2.931	2.856	0,03	1.097,57	97%	130,11	5,87
36	1,7	2.991	2.917	0,04	1.033,11	98%	141,18	6,37
36	1,8	2.885	2.801	0,04	970,66	97%	144,28	6,51
36	1,9	3.013	2.934	0,05	925,94	97%	158,43	7,15
36	2	3.098	3.021	0,04	877,91	98%	172,06	7,77
36	2,3	2.956	2.873	0,04	761,24	97%	188,71	8,52
36	2,5	2.953	2.858	0,05	696,96	97%	205,03	9,25

Tabella 4.5: Variazione del *sendInterval*

Da questa tabella è possibile notare fondamentalmente due cose: la prima è che, al variare della generazione di traffico, il delay diminuisce costantemen-

te fino a che il *sendInterval* non supera il valore di 1s, dopodiché raggiunge il suo equilibrio e rimane costante sempre attorno ai 0,1 e 0,2 secondi.

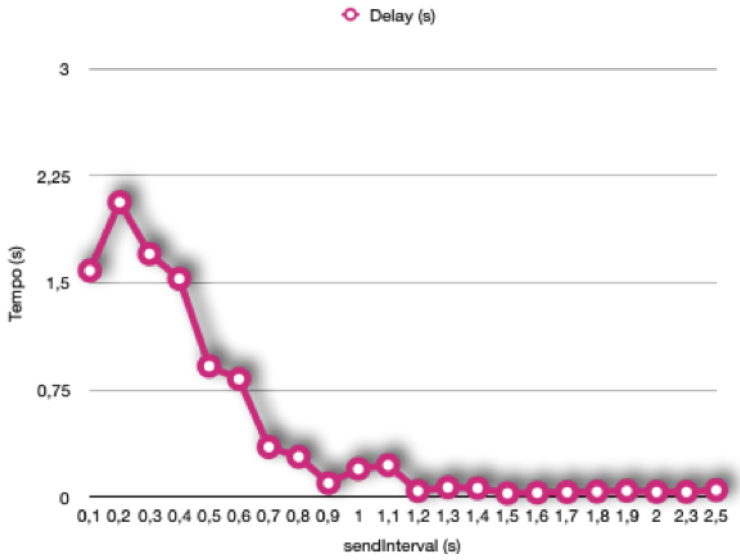

Figura 4.8: Delay al variare del *sendInterval*

La seconda cosa che si nota dalla tabella è che il PDR aumenta all'aumentare del *sendInterval* fino a raggiungere l'equilibrio a partire dal *sendInterval* settato a 1 secondo e mezzo.

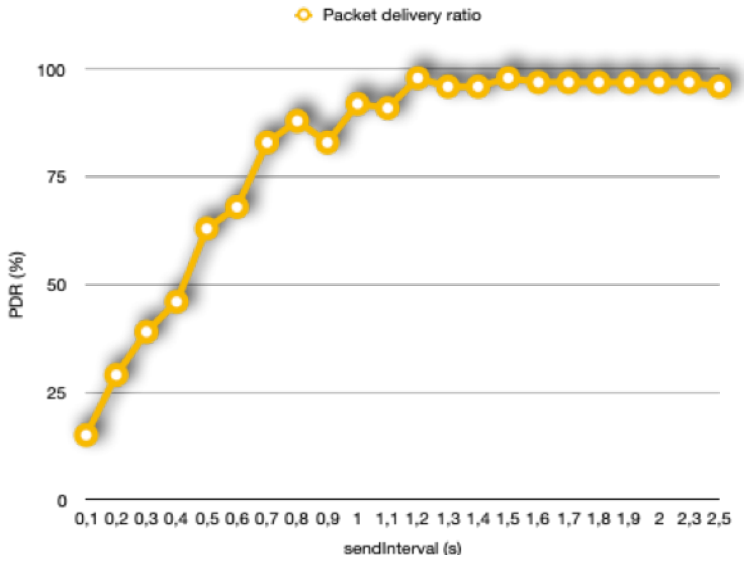

Figura 4.9: PDR al variare del *sendInterval*

La Figura 4.9, in cui sull'asse delle ascisse sono presenti i secondi attribuiti al sendInterval e sull'asse delle ordinate sono presenti le percentuali di PDR, possiamo notare che ad intervalli di frequenza di invio pacchetti troppo elevati abbiamo un PDR molto più basso. Questo perché il numero di pacchetti inviati aumenta enormemente, producendo un dispendio energetico elevatissimo che provoca una durata molto limitata della rete. Infatti, utilizzando un sendInterval di 0,5 secondi, abbiamo che ogni sensore invia un pacchetto ogni mezzo secondo, per un totale di 3516 pacchetti generati in soli 49 secondi. Un tempo troppo limitato per permettere al gateway di recuperare tutti i pacchetti. Riducendo la generazione di traffico, abbiamo un progressivo aumento del PDR, arrivando fino ad un PDR del 98% con un intervallo di invio pacchetti assegnato a 1,5 secondi.

Tornando alla lifetime, a testimoniare che all'aumentare del *sendInterval* vi è un minor consumo energetico e un conseguente prolungamento della durata della rete vi è la Figura 4.10. La Figura mostra che con il *sendInterval* = 0,5s la rete dura solo circa 2 giorni e aumenta progressivamente fino ad arrivare ad una durata complessiva di circa 10 giorni con *sendInterval* = 2,5s.

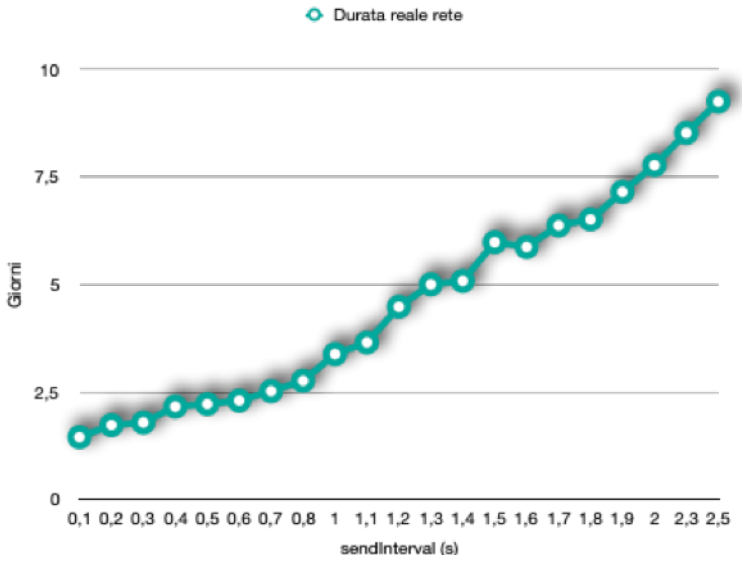

Figura 4.10: Durata al variare del *sendInterva*

Prendendo quindi la configurazione di 36 nodi di partenza, in cui il sendInterval medio era circa 1 secondo, e quella in cui l'intervallo di invio pacchetti è 1,5 secondi possiamo notare dei grossi miglioramenti anche a livello sensore, non solo a livello di rete.

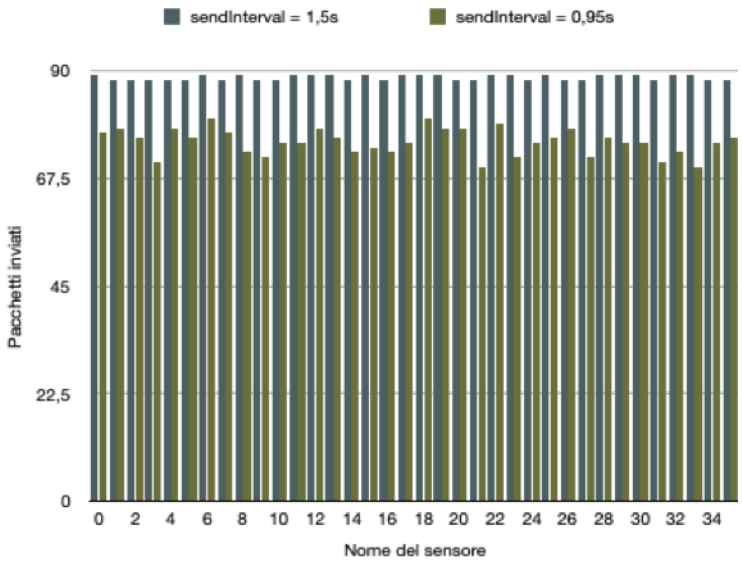

Figura 4.11: Confronto pacchetti inviati cambiando il *sendInterval*

La Figura 4.11 mostra in blu il numero di pacchetti inviati dal singolo sensore con *sendInterval* = 1,5 secondi, e in verde il numero di pacchetti inviati dal singolo sensore con *sendInteval* a circa 1 secondo. Come possiamo notare, con la nuova configurazione ogni sensore riesce ad inviare un maggior numero di pacchetti. Ogni sensore riesce ad inviare fino a 89 messaggi totali, a differenza della vecchia configurazione in cui solamente un sensore riusciva a raggiungere un totale di 80 messaggi inviati. Questo è dovuto sempre al fatto che ogni sensore produce un minor consumo energetico nell'unità di tempo, quindi la vita di ogni sensore è maggiore rispetto a prima. In questo modo ogni sensore riesce ad inviare una quantità maggiore di pacchetti. Non solo, ma anche il numero di pacchetti che riescono ad arrivare a destinazione aumenta per ogni singolo sensore. Questo miglioramento lo possiamo notare nella Figura 4.12, in cui abbiamo il numero di pacchetti consegnati dal singolo sensore con la nuova configurazione in blu e in verde quelli ottenuti dalla vecchia configurazione. Infatti, come abbiamo detto in precedenza in PDR migliora.

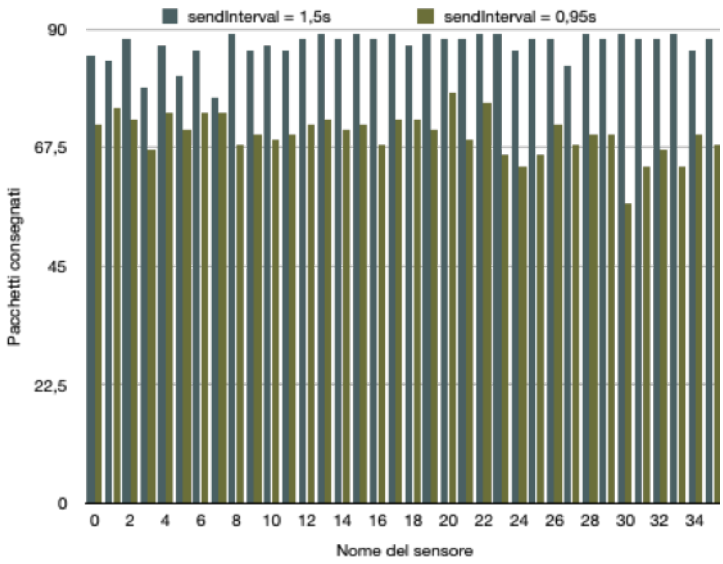

Figura 4.12: Confronto pacchetti consegnati cambiando il *sendInterval*

Oltre ad un miglioramento del PDR, abbiamo un netto miglioramento

anche del delay, come mostra la Figura 4.13. In rosa viene mostrato il ritardo medio del singolo sensore calcolato con la configurazione in cui il *sendInterval* è settato a 1,5 secondi e in viola il ritardo medio del singolo sensore recuperato dalla configurazione in cui venivano inviati circa un pacchetto al secondo.

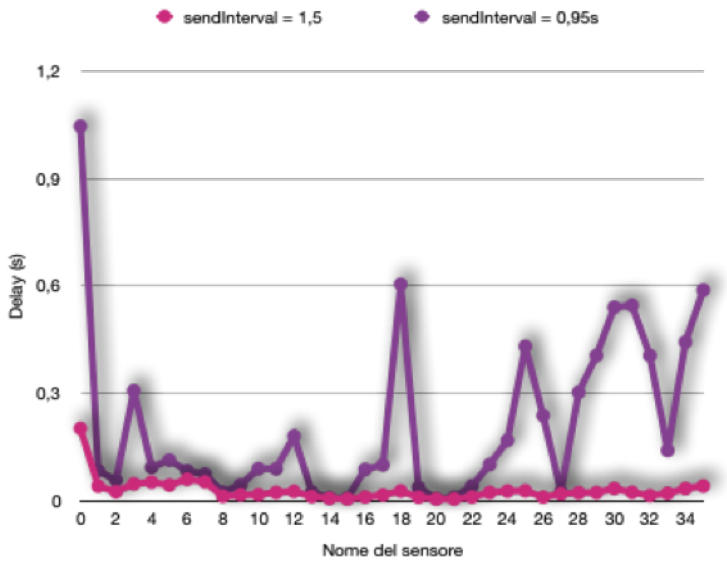

Figura 4.13: Confronto delay cambiando il *sendInterval*

4.2.3 Esperimento 3: Durata della rete

Nell'esperimento precedente siamo riusciti, modificando la generazione di traffico a migliorare il PDR e a prolungare la durata della lifetime. Il terzo esperimento consiste nell'introdurre il meccanismo di sleep and wake up, proposto in diversi articoli descritti nel Capitolo 2. Questo meccanismo prevede di introdurre un sistema grazie al quale ogni sensore ha un periodo di tempo in cui si riposa, ovvero un intervallo di tempo di inattività, oltre ad un periodo in cui invia e inoltra pacchetti, per fare in modo che ogni sensore consumi meno energia possibile. In questo modo si dovrebbe prolungare ulteriormente la lifetime della rete.

Per poter implementare questo meccanismo si è fatta una piccola modifica

al modulo del MAC di INET. All'interno del modulo *Ieee802154Mac*, utilizzato appunto per definire il livello MAC della wlan del progetto, sono state aggiunte due variabili: *dcTime* e *dcAlpha*. Il *dcTime* definisce il periodo del duty-cycle, e viene inizializzato con lo stesso valore del *sendInterval*, in questo modo un sensore appena ha un dato da inviare accende la radio, invia il pacchetto e poi si rimette a dormire. *dcAlpha*, invece, definisce il rapporto di tempo in cui la radio è attiva.

Per effettuare questo esperimento si è partiti tenendo il sendInterval, e quindi il *dcTime*, al valore che dall'esperimento precedente si è ritenuto quello ottimale, ovvero 1,5 secondi, per vedere se effettivamente introducendo questo meccanismo si possa veramente aumentare la durata della rete. L'analisi è stata fatta al variare del *dcTime*, ovvero al variare del periodo di attività della radio.

Numero sensori	sendInterval (s)	dcAlpha	Pacchetti inviati	Pacchetti ricevuti	Delay (s)	Throughput (byte/s)	PDR (%)	Tempo simulazione (giorni)	Durata reale rete (giorni)
36	1,5	0,1	9.039	1.932	1,204	256,525	21%	376,57	17,00
36	1,5	0,2	5.077	1.642	1,679	388,110	32%	211,54	9,55
36	1,5	0,3	3.639	1.653	1,289	545,286	45%	151,57	6,84
36	1,5	0,5	2.611	1.977	0,576	910,297	76%	108,59	4,90
36	1,5	0,6	2.703	2.497	0,189	1.108,999	92%	112,58	5,08
36	1,5	0,8	3.051	2.947	0,062	1.161,935	97%	126,81	5,72
36	1,5	0,9	2.887	2.769	0,066	1.150,360	96%	120,35	5,43
36	1,5	1,0	3.186	3.126	0,028	1.179,071	98%	132,56	5,98

Tabella 4.6: Duty cycle con *sendInterval* = 1,5s

Con *dcAlpha* = 0,1 abbiamo che la radio è praticamente sempre nello stato di sleep, aumentando il valore di *dcAlpha*, aumenta il periodo di attività della radio, fino ad arrivare a dcAlpha = 1 in cui abbiamo la radio sempre attiva (situazione di partenza).

Con $dcAlpha = 0{,}1$ la durata della rete è 1.468.629s, ovvero 17 giorni, quindi quasi quattro volte tanto rispetto alla configurazione di partenza. Ma come si può notare dalla tabella il PDR è bassissimo, questo perché appunto la radio passa praticamente tutto il tempo in stato di sleep, quindi la maggior parte dei pacchetti generati dai sensori vengono persi. Con questa configurazione siamo riusciti a migliorare la lifetime, ma a discapito del PDR. Se proviamo a diminuire il traffico di dati nell'unità di tempo, aumentando il sendInterval, abbiamo un notevole miglioramento delle prestazioni della rete, sia a livello di PDR, sia a livello di durata complessiva.

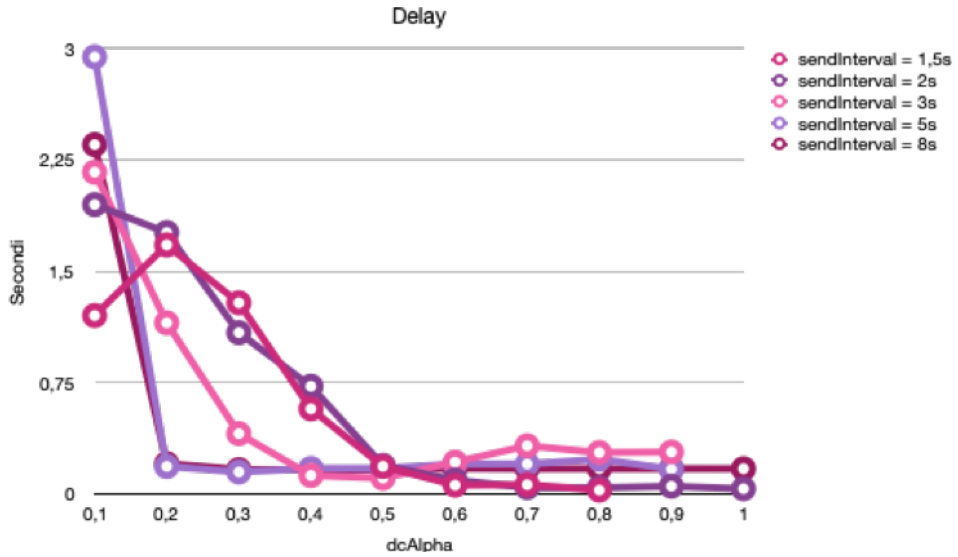

Figura 4.14: Duty cycle - Delay

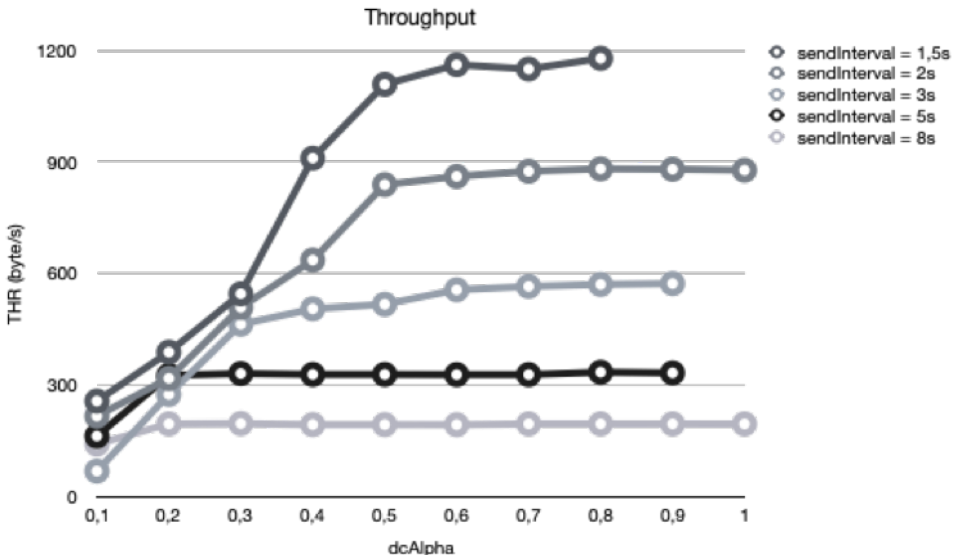

Figura 4.15: Duty cycle - Throughput

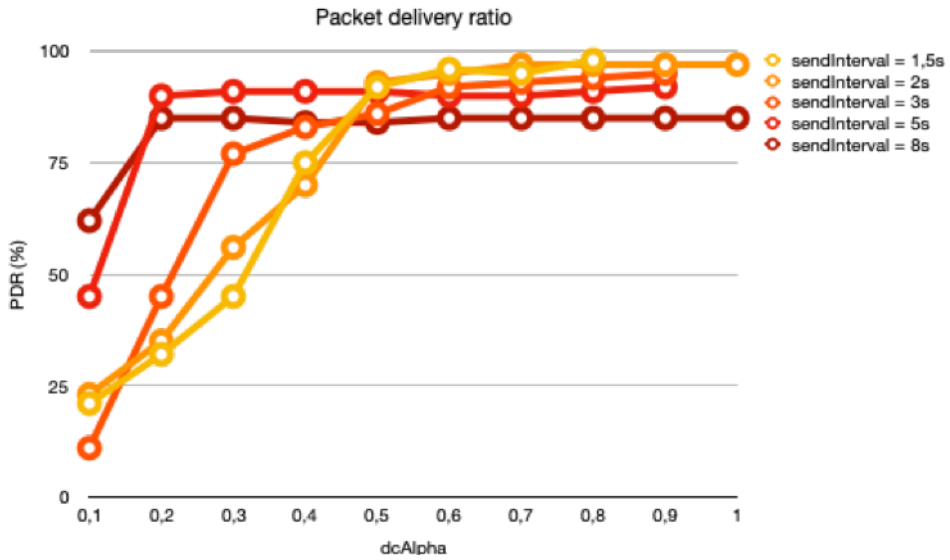

Figura 4.16: Duty cycle - PDR

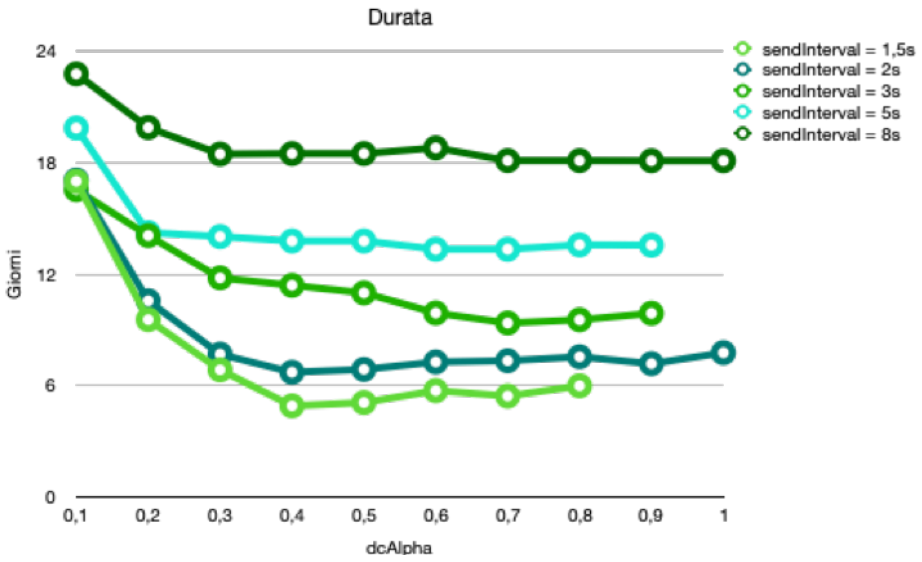

Figura 4.17: Duty cycle - lifetime

I grafici riportati sopra mostrano un recap dell'analisi del duty cycle al variare del sendInterval. I grafici viola, i primi a partire da sinistra, mostrano l'andamento del delay, i grafici in nero rappresentano il throughput, calcolato sempre in byte/s, in giallo viene mostrato il PDR e in verde la durata totale della rete. Ogni Figura presenta sull'asse delle ascisse il valore di *dcAlpha*.

In generale, quello che notiamo da questi grafici è che in ogni caso, all'aumentare del dcAlpha il delay diminuisce, il throughput raggiunge il suo equilibrio, il PDR migliora e la durata della rete diminuisce. D'altro canto, diminuendo il traffico generato il delay si riduce anche all'aumentare del tempo di inattività della radio, il throughput raggiunge prima il suo equilibrio, e la durata della rete aumenta ulteriormente, senza andare ad influenzare negativamente sul PDR.

Se con il *sendInterval* = 1,5s la rete migliorava solamente con valori del dcAlpha inferiori a 0,3, a discapito del PDR che con *dcAlpha* = 0,3 rimaneva inferiore del 50%, già con *sendInterval* = 3s, iniziamo a vedere i primi miglioramenti. Inviando pacchetti ogni 3 secondi, e mettendo *dcAlpha* = 0,5,

la durata della rete aumenta di più di un giorno e il PDR raggiunge quasi il 90%. Questi miglioramenti si notano ancora di più con il sendInterval = 8s, in cui abbiamo una durata totale della rete di quasi 19 giorni, con *dcAlpha* = 0,6, e nonostante la radio si metta a riposo per un certo periodo di tempo, il PDR rimane comunque alto, 85%. Se pensiamo che siamo partiti da una durata massima di circa 6 giorni con un *sendInterval* di 1,5, e siamo arrivati ad una durata di quasi 20 giorni, possiamo dire con certezza che utilizzando questo meccanismo di sleep and wake up abbiamo un grande risparmio energetico.

I grafici relativi alle Figure 4.14, 4.15, 4.16 e 4.17 è possibile trovarli per esteso nell'Appendice A, in modo da poter avere una visione più chiara sul loro andamento.

4.2.4 Esperimento 4: Nodo mobile

L'ultimo esperimento trattato in questo progetto di tesi consiste nel sostituire il gateway fisso con un nodo mobile. Il nodo mobile sarà un UAV che passerà al di sopra di ogni sensore per raccoglierne i dati. Per questo tipo di esperimento vengono fatte due tipologie di analisi, la prima sfruttando l'UAV solo ed esclusivamente come mulo di raccolta dati, la seconda utilizzando l'UAV anche come sveglia per i sensori che si trovano nello stato di sleep e sono ad una distanza minore di 10m. L'algoritmo implementato in questa seconda fase dell'Esperimento 4 viene descritto nell'articolo *Dual-Mode Wake-Up Nodes for IoT Monitoring Applications: Measurements and Algorithms* [67]

I dati in Tabella 4.7 mostrano i risultati ottenuti dal primo esperimento, ovvero utilizzando il drone solo come mulo, messi a confronto con i risultati ottenuti dall'esperimento precedente.

Esperimento	Numero sensori	sendInterval (s)	Pacchetti inviati	Pacchetti ricevuti	Delay (s)	Throughput (byte/s)	PDR (%)	Tempo simulazione (secondi)	Durata reale rete (giorni)
ESPERIMENTO 2 (Senza sleep and wake up)	36	5	2.166	1.995	0,17	332,17	92%	300	14
ESPERIMENTO 3 (Con sleep and wake up)	36	5	2.201	2.003	0,17	328,05	91%	305	14
ESPERIMENTO 4 (Con UAV come mulo)	36	5	44.136	12.128	1,42	98,97	27%	6127	277

Tabella 4.7: Nodo mobile - confronto con esperimenti precedenti

La prima riga della tabella mostra i valori ottenuti mettendo il *sendInterval* = 5 s nella configurazione della rete senza il meccanismo di sleep and wake up e senza UAV. La seconda riga mostra i risultati ottenuti dalla stessa configurazione, ma aggiungendo il meccanismo di sleep and wake up. Infine l'ultima riga mostra i risultati ottenuti sostituendo il gateway fisso con l'UAV utilizzato solo come mulo. Come si può notare siamo riusciti ad aumentare di gran lunga la lifetime: da circa 13 giorni e mezzo nella configurazione senza sleep and wake up, fino a 9 mesi con lo scenario con il mulo mobile. Per quanto riguarda le altre metriche invece, possiamo notare un notevole peggioramento. Il PDR si abbassa enormemente in quanto in quest'ultimo scenario i sensori continuano ad inviare continuamente pacchetti nonostante l'UAV non si trovi sopra di loro, quindi tutti questi dati verranno poi persi.

Aumentando l'intervallo con cui vengono inviati i pacchetti, le prestazioni della rete iniziano a migliorare (Tabella 4.8). Il PDR aumenta, infatti con una generazione di 1 pacchetto al minuto il PDR arriva a 57%, e la rete dura fino a 6 anni.

Numero sensori	sendInterval (s)	Pacchetti inviati	Pacchetti ricevuti	Delay (s)	Throughput (byte/s)	PDR (%)	Tempo simulazione (s)	Durata reale rete (giorni)
36	5	44.136	12.128	1,42	98,97	27%	6.127	277
36	10	26.964	12.256	1,16	81,86	45%	7.486	338
36	20	21.924	12.413	1,16	51,01	57%	12.168	549
36	50	20.268	11.497	1,09	20,44	57%	28.125	1.270
36	100	17.604	10.091	0,60	10,33	57%	48.829	2.204

Tabella 4.8: Nodo mobile - variazione del *sendInterval* (UAV come mulo)

La tabella 4.9 invece mostra i risultati ottenuti dallo scenario che utilizza il drone come sveglia oltre che come mulo.

Numero sensori	sendInterval (s)	Pacchetti inviati	Pacchetti ricevuti	Delay (s)	Throughput (byte/s)	PDR (%)	Tempo simulazione (s)	Durata reale rete (giorni)
36	5	222.876	62.305	0,29	100,07	28%	30.950	1.397
36	10	148.932	70.154	0,21	84,80	47%	41.366	1.867
36	20	149.436	86.246	0,14	51,95	58%	83.007	3.747
36	50	148.932	86.633	0,14	20,95	58%	206.806	9.335
36	100	148.932	86.742	0,13	10,49	58%	413.613	18.670

Tabella 4.9: Nodo mobile - variazione del *sendInterval* - UAV come sveglia

Ancora una volta siamo riusciti ad aumentare la durata della rete, fino a farla arrivare a circa 50 anni con la configurazione in cui viene settato il *sendInterval* = 100s. Anche con questa configurazione il PDR non è altissimo, ma comunque accettabile (58%). I valori bassi dipendono dalla velocità del veicolo mobile, se vola ad una velocità troppo elevata o troppo lenta non fa in tempo a raccogliere tutti i pacchetti generati ed inviati dal sensore.

I seguenti grafici mostrano un miglioramento delle prestazioni della rete utilizzando l'UAV anche come meccanismo per attivare i sensori che si trovano

in stato di sleep. In particolare la Figura 4.18 mostra l'andamento del ritardo, la Figura 4.19 l'andamento del PDR e la Figura 4.20 la durata della rete.

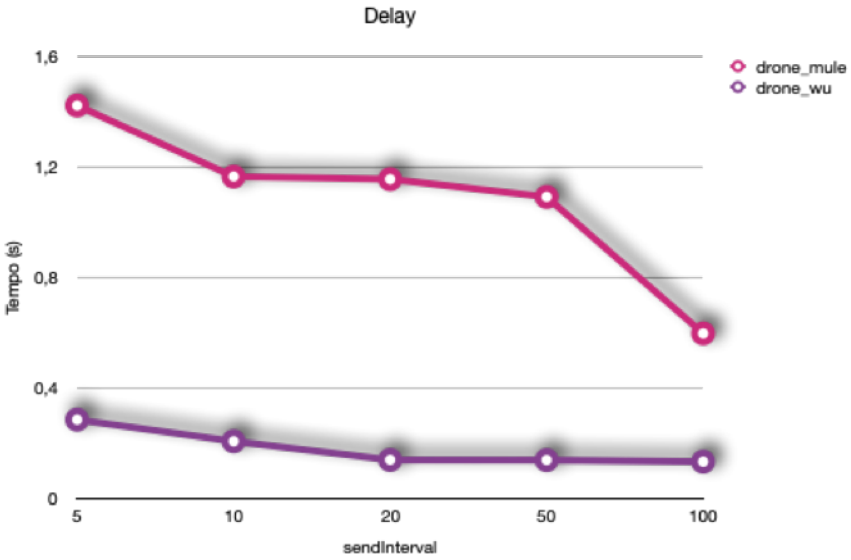

Figura 4.18: Scenario con UAV: Delay

Molto evidenti sono i miglioramenti sul delay, che parte da 1,4 secondi nella configurazione con sendInterval = 5 e UAV come mulo, fino ad abbassarsi a 0,1 secondo nello scenario con *sendInterval* = 100 e drone utilizzato come sveglia.

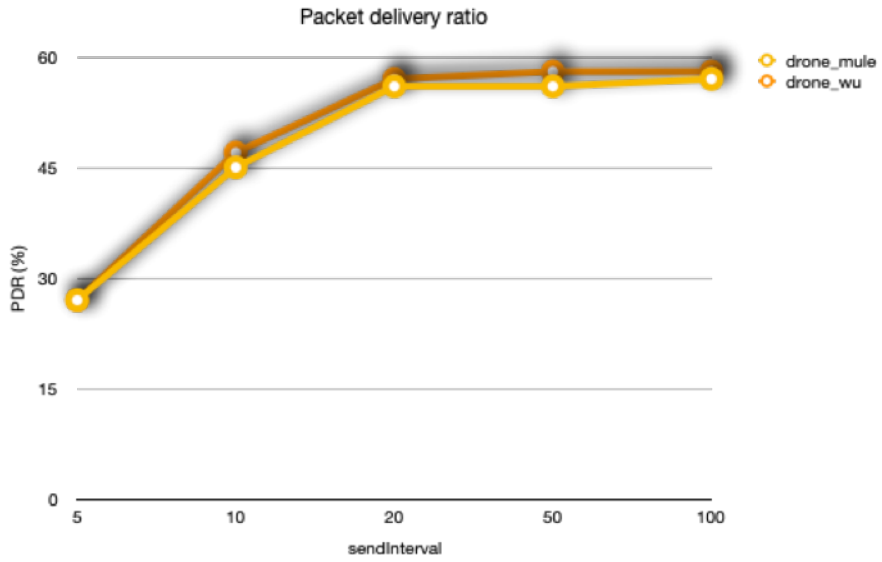

Figura 4.19: Scenario con UAV: PDR

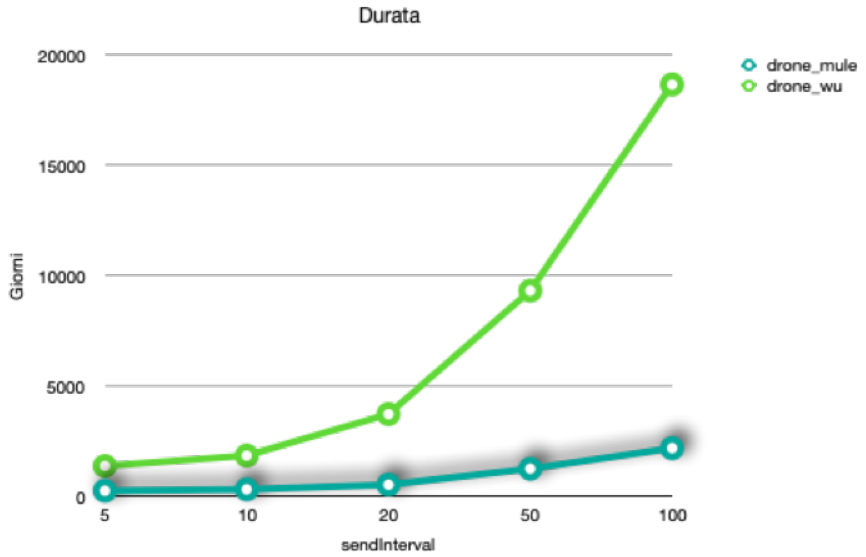

Figura 4.20: Scenario con UAV: Durata

Come dimostrato dai dati presenti nell'appendice B e C, l'utilizzo dell'UAV consente un prolungamente significativo della durata complessiva della rete anche con densità diverse. Infatti anche con un numero maggiore o inferiore di sensori abbiamo una lifetime di ordine di grandezza superio-

re rispetto allo scenario senza drone, pur mantenendo il livello degli altri indicatori analogo o accettabile.

Conclusioni

Il contributo che vuole dare questa tesi è quello di analizzare una rete wireless di sensori al fine di massimizzarne la durata (lifetime). Oltre alla durata vengono studiate altre metriche: il ritardo di trasmissione (delay), la capacità effettiva della rete (throughput) e il packet delivery ratio (PDR).

La rete studiata utilizza lo standard IEEE 802.15.4 per definire il livello fisico e il livello MAC della radio e i sensori che la popolano. In particolare a livello MAC la header length è di 72bit, una queue length di massimo 100 pacchetti per buffer e un bit rate di 250000bps. I sensori che popolano la rete hanno una batteria di 3900J e sono posizionati su una griglia quadrata ad una distanza di 120m l'uno dall'altro. In mezzo alla griglia è posizionato il gateway, che non deve fare altro che ricevere i pacchetti inviati dai vari sensori. Ogni pacchetto ha una dimensione di 50Byte.

Il primo obbiettivo della ricerca è stato studiare come cambiano le metriche prese in analisi al variare della densità di sensori che popolano la rete. Dall'esperimento è emerso che all'aumentare del numero di sensori le prestazioni della rete si riducono ad eccezione del throughput. Quest'ultimo aumenta all'aumentare del numero dei nodi fino alla configurazione con 64 sensori, dopodiché trova il suo equilibrio e non aumenta più; questo vuol dire che la rete mantiene un'alta capacità di trasmissione effettiva anche con 144 sensori. La lifetime, invece, parte molto alta per abbassarsi progressivamente; la rete rimane attiva infatti per più di 44 giorni se composta da 4 sensori

e solo 2 giorni se composta da 144 sensori. L' abbassarsi della lifetime è dovuto al meccanismo di inoltro multi-hop. La principale conseguenza dell'abbassarsi della lifetime è che anche il PDR si abbassa. Lo scenario meno denso ha un PDR del 99% e si abbassa fino ad arrivare al 26% per lo scenario più denso. Questo perché il numero di pacchetti inviati al secondo rimane sempre più o meno costante, e all'aumentare del numero di sensori aumenta, per uno stesso intervallo di tempo, il numero di pacchetti scambiati all'interno della rete; riducendosi la durata della rete, il gateway fa sempre più fatica a riceverli tutti. Oltre alla lifetime e al PDR, anche il delay peggiora : questo è dovuto principalmente al fatto che aumenta il numero di nodi ad una distanza maggiore dal gateway, che si trova sempre al centro della rete, quindi il percorso multi-hop che deve compiere il pacchetto prima di arrivare a destinazione aumenta progressivamente.

Da questa prima analisi è risultato che la configurazione con 36 sensori è quella ottimale, in quanto copre un'area abbastanza elevata, di $518.400m^2$ e mantiene un PDR del 93%. Ma ha il problema di una limitata lifetime a soli 3 giorni.

Con il secondo esperimento si riesce a prolungare la lifetime della rete modificando la quantità di traffico generato. Inviando un pacchetto ogni 2,5 secondi la rete dura più di 9 giorni. Aumentando l'intervallo di invio dei pacchetti si riesce anche a migliorare il PDR che passa dal 93% al 96%.

Per prolungare ulteriormente la durata della rete, nell'esperimento tre si è introdotto un meccanismo chiamato sleep and wake up. Grazie a questo meccanismo siamo riusciti a configurare una rete con una durata totale di addirittura 20 giorni. Questa configurazione, ottenuta inviando pacchetti ogni 8 secondi, e tenendo a riposo la radio della rete dei sensori per un certo intervallo di tempo, mantiene anche un PDR abbastanza elevato (85%).

L'ultimo esperimento, infine, si concentra sull'analisi della rete sfruttando un UAV per la raccolta dati al posto del gateway fisso. In una prima fase l'UAV viene utilizzato solo esclusivamente come mulo per la raccolta dati, i sensori sono sempre attivi e continuano ad inviare pacchetti a prescindere che l'UAV si trovi sopra di loro o meno. I dati ottenuti mostrano che la lifetime aumenta enormemente rispetto agli esperimenti precedenti, raggiungendo addirittura 6 anni di vita. Il PDR (57%) è inferiore rispetto agli esperimenti, ma comunque accettabile in parecchi contesti, ad esempio quello agricolo; questo perché i sensori inviano pacchetti anche se l'UAV non è pronto per la loro raccolta. In questo scenario molti pacchetti vengono persi perché l'unico destinatario dei pacchetti è l'UAV, e non essendoci il meccanismo di multi-hop forwarding, i pacchetti inviati dai sensori che non hanno l'UAV sopra di loro vengono persi.

La seconda fase di analisi, di questo ultimo esperimento, permette di migliorare il PDR e di aumentare ulteriormente la lifetime della rete. Questo avviene grazie all'introduzione di un meccanismo che consente all'UAV di funzionare non solo come mulo ma anche come sveglia per i sensori, i quali sono in perenne stato di sleep. Solo quando l'UAV si trova a meno di 10m di distanza dai sensori, li accende e consente loro, di conseguenza, l'invio dei pacchetti. Con questo meccanismo si arriva ad ottenere una durata totale della rete di più di 50 anni.

Siamo quindi riusciti a dimostrare gli obbiettivi proposti e dare un ulteriore contributo allo stato dell'arte. Abbiamo confermato quanto ci eravamo proposti, ovvero cercare di ottimizzare la lifetime della rete mantenendo un PDR abbastanza elevato.

Come dimostrato dai dati presenti nell'appendice B e C, l'utilizzo dell'UAV consente un prolungamente significativo della durata complessiva della rete anche con densità diverse. Infatti anche con un numero maggiore o inferiore

di sensori abbiamo una lifetime di ordine di grandezza superiore rispetto allo scenario senza drone, pur mantenendo il livello degli altri indicatori analogo o accettabile.

Il limite di questa analisi risiede nel fatto che ogni simulazione viene effettuata con una batteria limitata, di solo 1J, questo per poter rendere le simulazioni fattibili a livello temporale e livello di potenza hardware. Questo limita molto l'analisi, soprattutto nell'esperimento quattro, che magari equipaggiando i sensori con una batteria più adeguata avremmo potuto ottenere risultati migliori anche a livello di PDR.

Il secondo limite riguarda il modulo utilizzato per la mobilità dell'UAV. Il modulo utilizzato, *TractorMobility*, sfrutta solamente due dimensioni, e non tre, quindi non è possibile variare l'altezza a cui si trova l'UAV. L'altezza non è stata considerata in quanto gli UAV volano comunque ad un'altezza limitata, che rientra sempre nel raggio d'azione dei sensori, quindi alzando l'UAV rispetto al livello del terreno i risultati non dovrebbero cambiare.

L'utimo limite risiede nel fatto che non viene cosiderata la batteria del drone. In un contesto reale anche il drone ha una batteria limitata. Aggiungendo un modulo di consumo energetico anche al nodo che rappresenta il drone la durata della rete sarebbe rimasta pressochè invariata. Quello che potrebbe cambiare è il delay, in quanto il drone dovrebbe regolarmente cambiare la sua traiettoria per dirigersi verso la stazione base, in cui potrà ricaricare le batterie. Quindi molti pacchetti inpiegeranno più tempo ad arrivare a destinazione in quanto dovranno aspettare che il drone si ricarichi.

Oltre al tentativo di trovare una soluzione a questi limiti, l'analisi potrebbe essere integrata cambiando alcuni parametri nella configurazione della rete. In primis si potrebbe aumentare il numero di UAV all'interno della rete, per cercare di far fronte al problema del basso PDR nello scenario con l'UAV uti-

lizzato solo come mulo di raccolta dati. Si potrebbe inoltre variare la velocità del drone e svegliare i sensori che si trovano anche ad una distanza superiore ai 10m: per verificare se, riducendo la sua velocità, sia possibile migliorare il PDR ed in generale migliorare le condizioni della WSN.

Un'altra analisi che si potrebbe effettuare è cambiare il percorso che compie l'UAV per la raccolta dati. In questa tesi viene utilizzato un movimento chiamato TractorMobility che permette al drone di spostarsi ad "S" all'interno dell'area in cui si trovano i sensori. Cambiando il tipo di mobilità, utilizzandone uno circolare ad esempio, i vari parametri presi in considerazione potrebbero cambiare e forse addirittura migliorare.

Infine, si potrebbe modificare la queue length, che adesso è fissa a 100 per vedere come cambia il throughput e il PDR.

Appendice A

Grafici di dettaglio

In questa appendice vengono mostrati i grafici relativi all'Esperimento 3 che raffigurano nel dettaglio le metriche analizzate al variare del *dcAlpha* e del *sendInterval*.

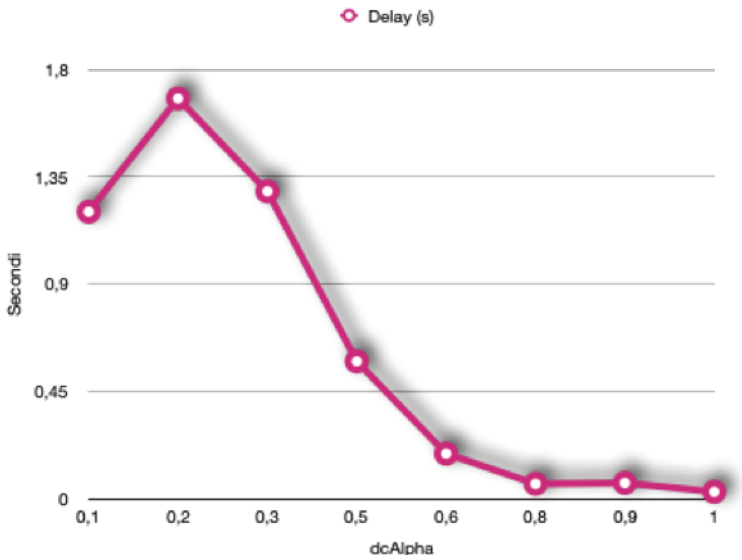

Figura A.1: Delay con *sendInterval* = 1,5

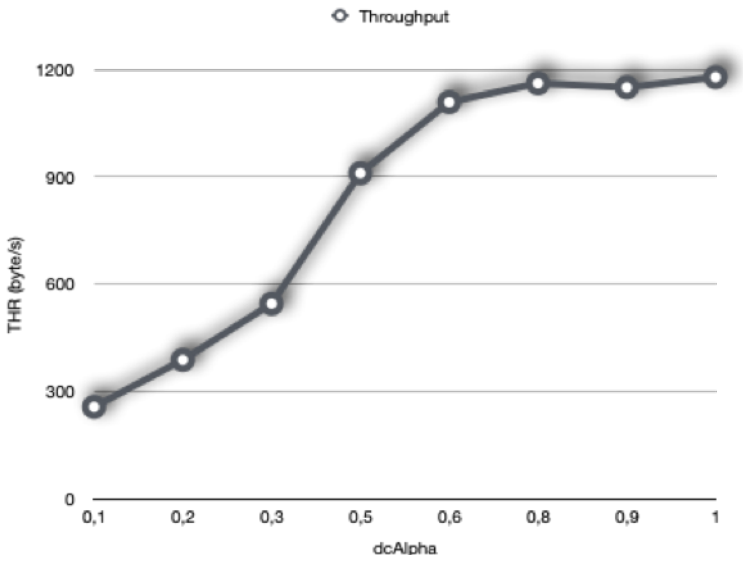

Figura A.2: Throughput con *sendInterval* = 1,5

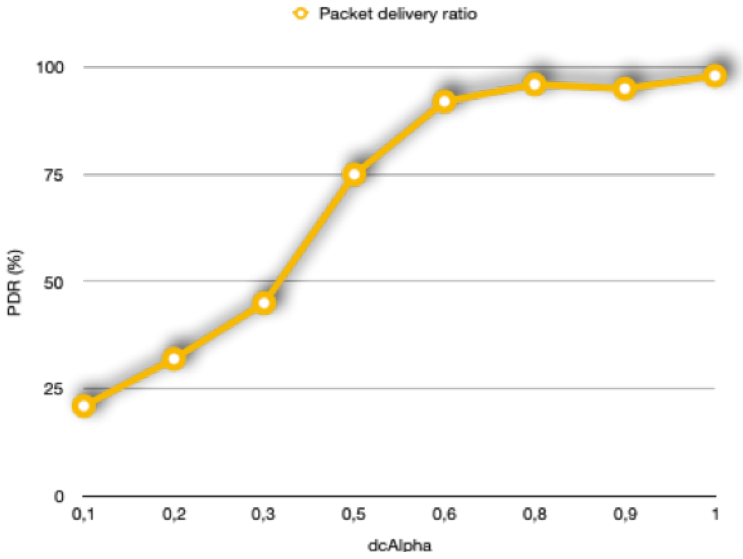

Figura A.3: PDR con *sendInterval* = 1,5

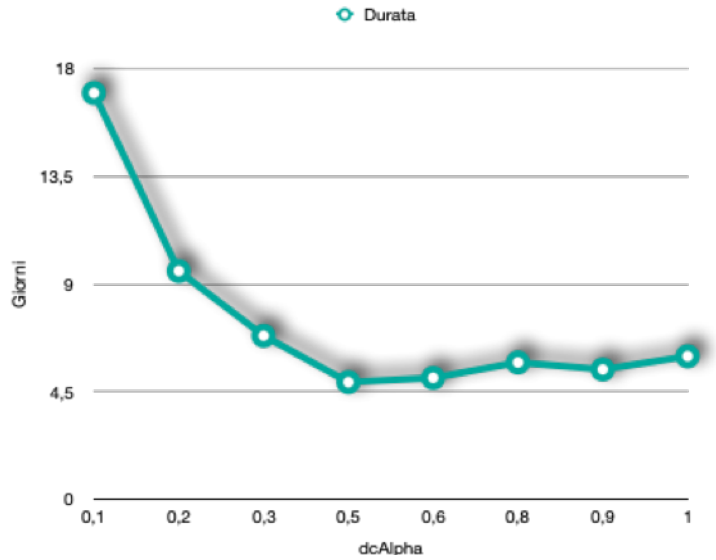

Figura A.4: Lifetime con *sendInterval* = 1,5

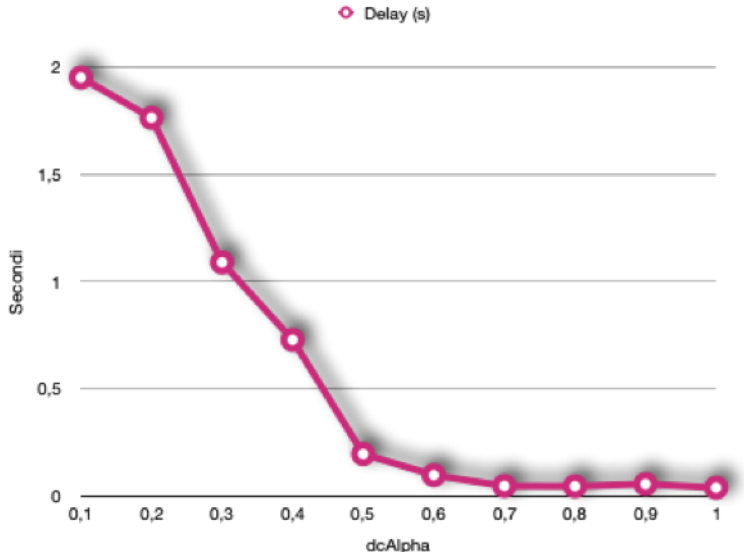

Figura A.5: Delay con *sendInterval* = 2

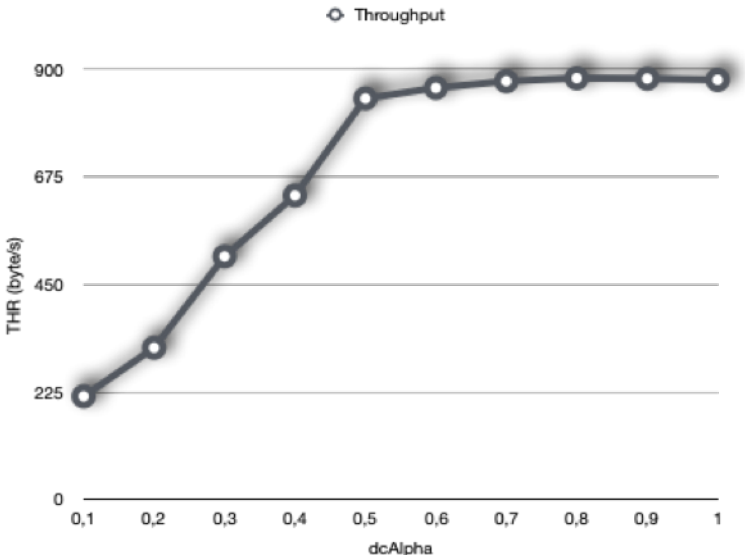

Figura A.6: Throughput con *sendInterval* = 2

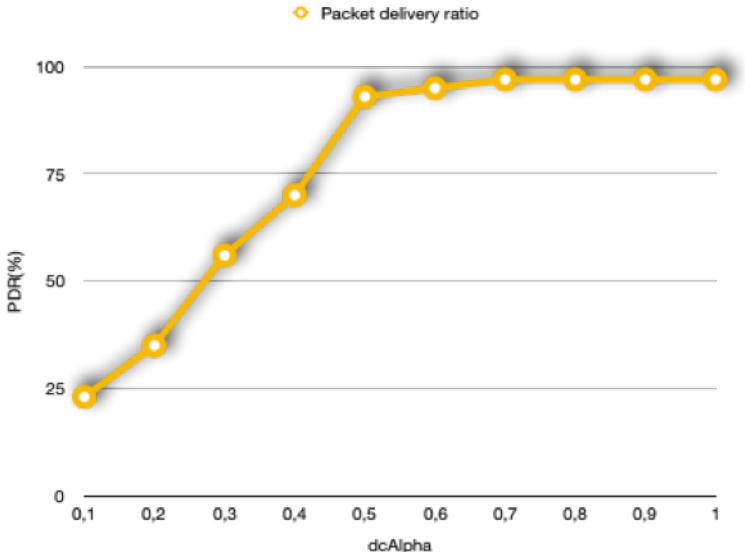

Figura A.7: PDR con *sendInterval* = 2

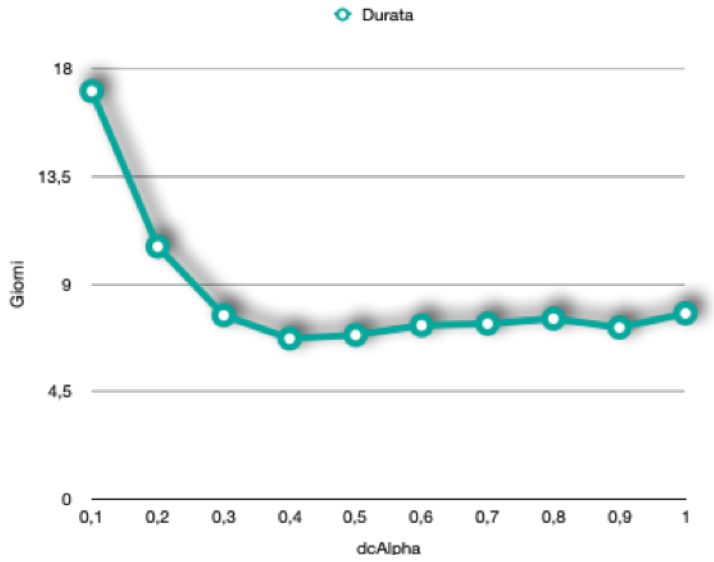

Figura A.8: Lifetime con *sendInterval* = 2

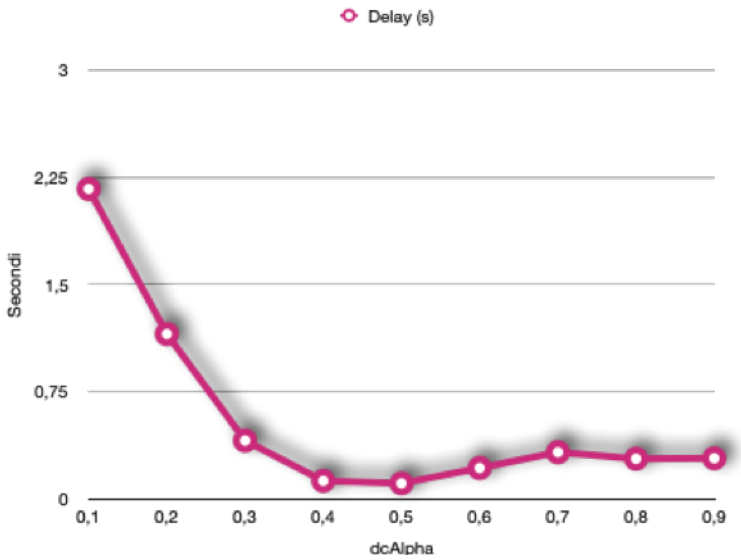

Figura A.9: Delay con *sendInterval* = 3

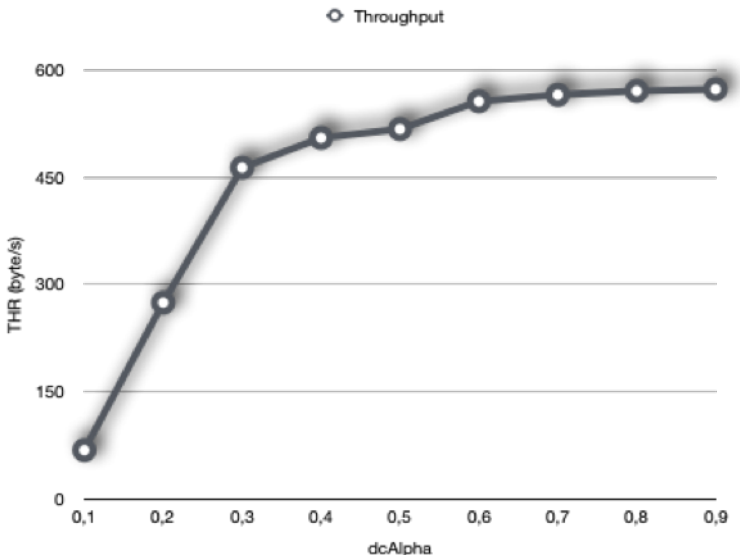

Figura A.10: Throughput con *sendInterval* = 3

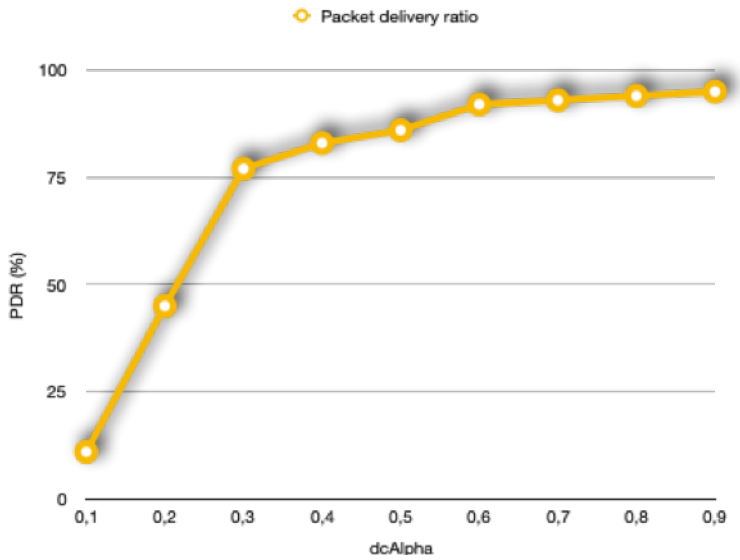

Figura A.11: PDR con *sendInterval* = 3

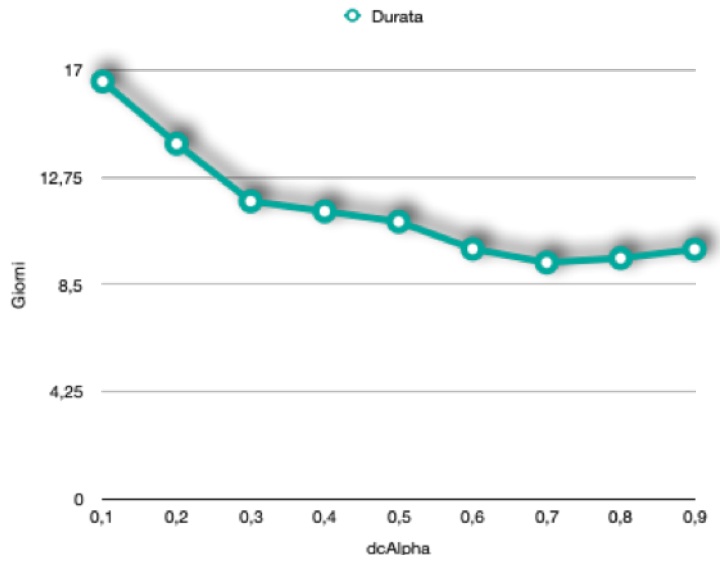

Figura A.12: Lifetime con *sendInterval* = 3

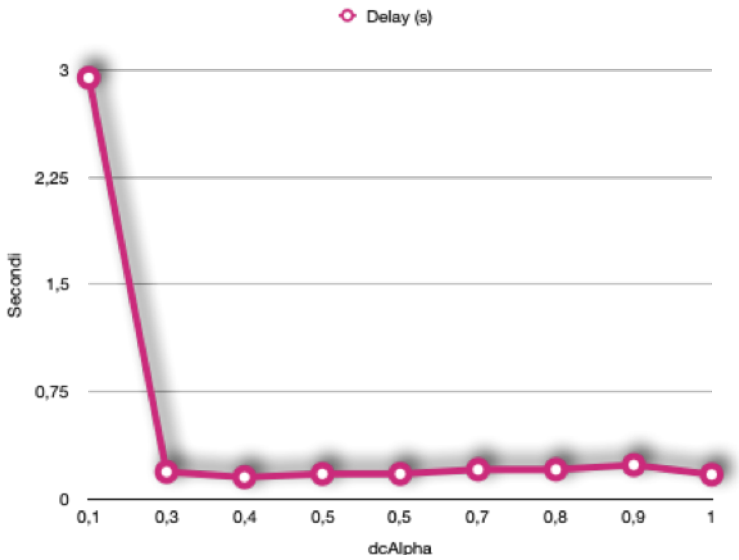

Figura A.13: Delay con *sendInterval* = 5

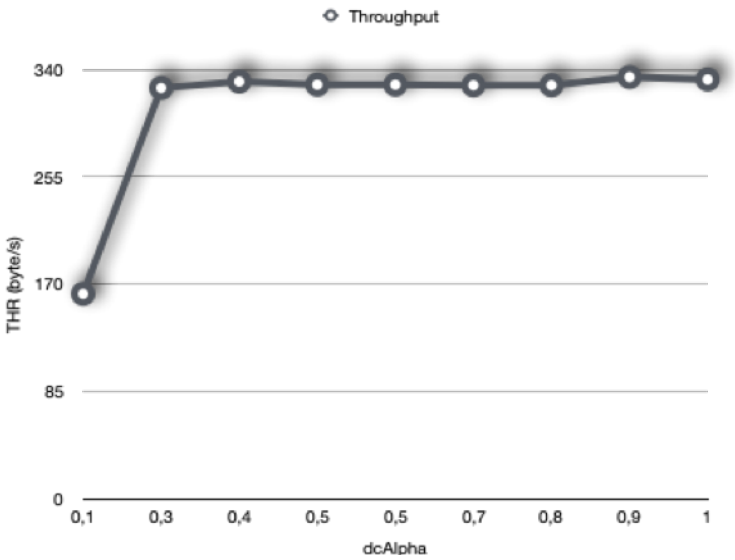

Figura A.14: Throughput con *sendInterval* = 5

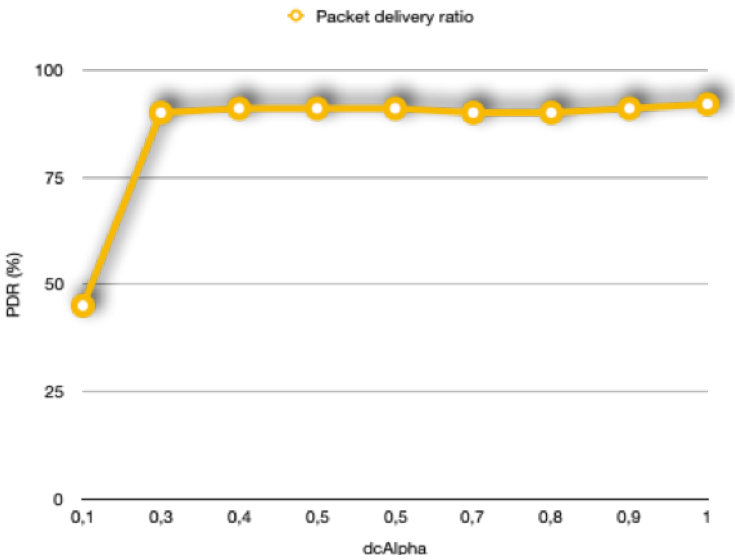

Figura A.15: PDR con *sendInterval* = 5

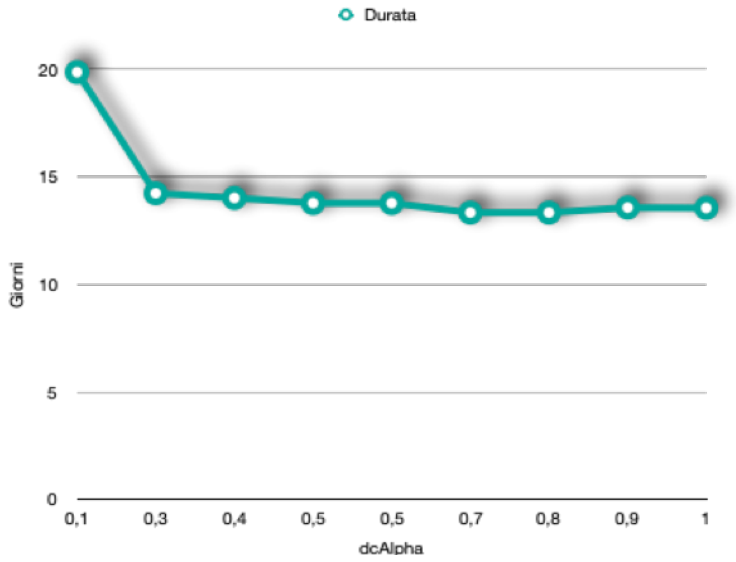

Figura A.16: Lifetime con *sendInterval* = 5

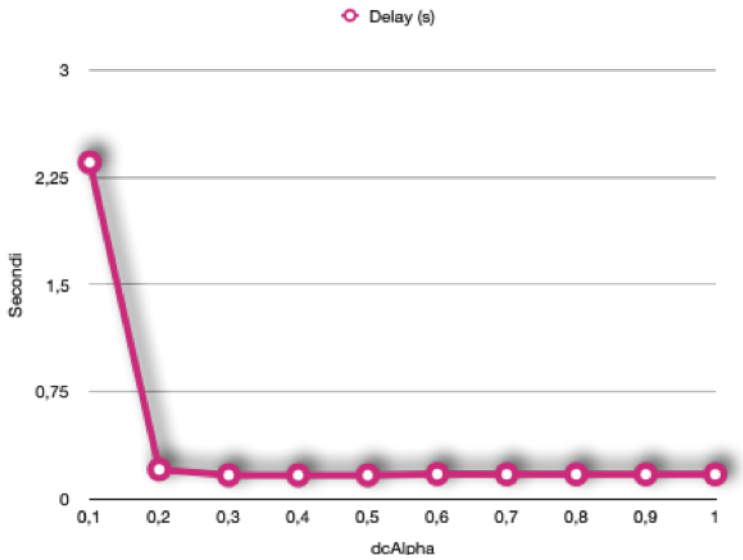

Figura A.17: Delay con *sendInterval* = 8

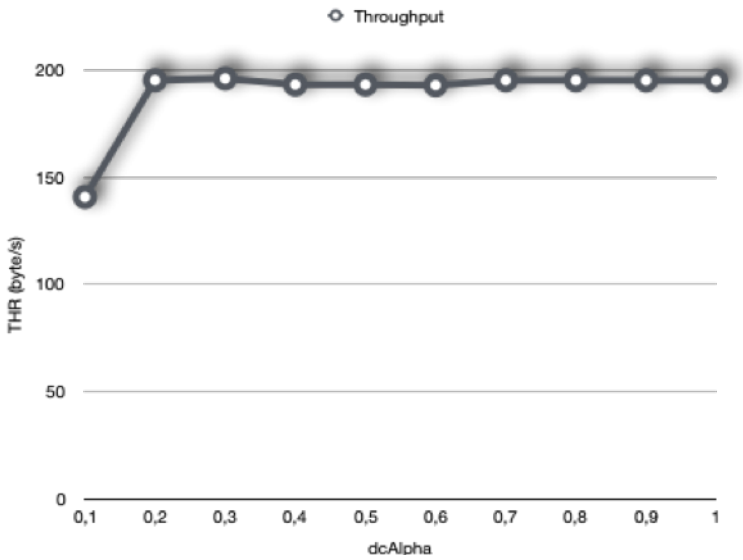

Figura A.18: Throughput con *sendInterval* = 8

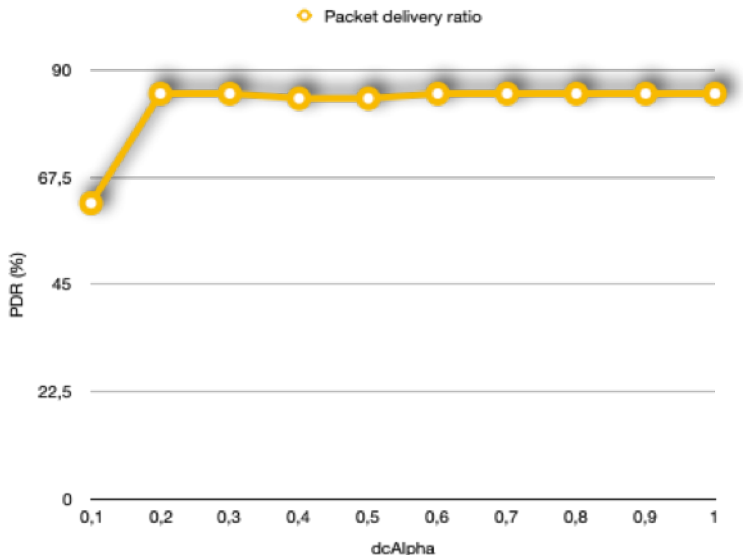

Figura A.19: PDR con *sendInterval* = 8

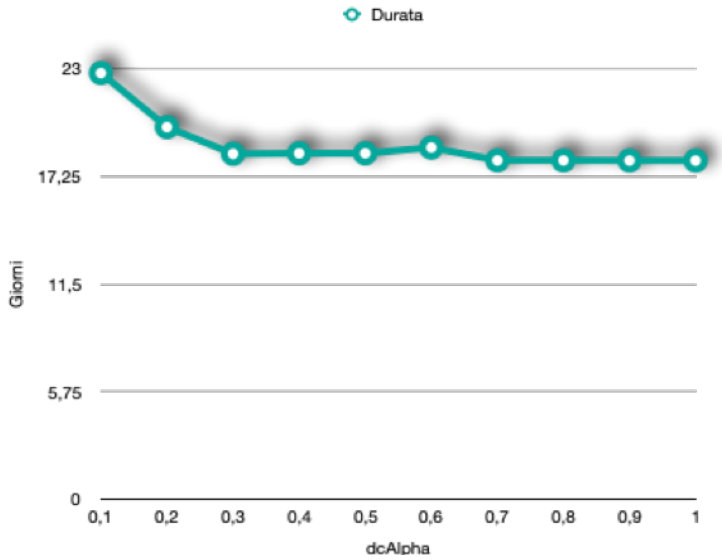

Figura A.20: Lifetime con *sendInterval* = 8

Appendice B

Tabelle di confronto

In questa appendice vengono mostrati i dati relativi all'Esperimento 2, 3 e 4 tenendo il *sendInterval* = 20s e variando la densità della rete, nella Tabella B.1. In particolare per l'Esperimento 3 viene settato il $dcAlpha = 0{,}5$, e l'Esperimento 4 viene diviso in 4.1 e 4.2. Il 4.1 si riferisce alle simulazioni effettuato utilizzando l'UAV come mulo, e il 4.2 allo scenario in cui l'UAV viene utilizzato anche come sveglia per i sensori in stato di sleep.

Invece la Tabella B.2 mostra i dati ottenti, mantenendo la configurazione da 36 nodi e variano il *sendInterval*. Partendo con un *sendInterval* pari a 5s fino a 20s (valore utilizzato per il confronto della tabella precedente).

Numero sensori	sendInterval (s)	Pacchetti inviati	Pacchetti ricevuti	Delay (s)	Throughput (byte/s)	PDR (%)	Tempo simulazione (s)	Durata reale rete (giorni)	Esperimento
4	20	1236	1193	0,166	9,683	96	6.160	278,071	2
36	20	1476	964	1,815	59,696	65	807	36,446	2
64	20	1984	1095	2,465	90,562	55	604	27,289	2
4	20	1252	1208	0,167	9,679	96	6.240	281,681	3
36	20	1512	966	1,819	58,900	63	820	37,015	3
64	20	1984	1098	2,470	90,692	55	605	27,324	3
4	20	5940	4445	0,730	7,487	74	9.686	437,215	4.1
36	20	21.924	12.413	1,16	51,01	57	12.168	549	4.1
64	20	104000	21361	1,8	32,868	20	32.494	1.466,772	4.1
4	20	16548	12394	0,026	7,491	74	82.726	3.734,171	4.2
36	20	149.436	86.246	0,14	51,95	58	83.007	3.747	4.2
64	20	785920	169846	0,75	34,579	21	245.593	11.085,812	4.2

Tabella B.1: Confronto esperimenti al variare della densità della rete

Numero sensori	sendInterval (s)	Delay (s)	Throughput (byte/s)	PDR (%)	Durata reale rete (giorni)	Esperimento
36	5	0,17	332	92	14	2
36	20	1,80	59	65	36	2
36	5	0,17	328	91	14	3
36	20	1,80	59	63	37	3
36	5	1,42	99	27	277	4.01
36	20	1,16	51	57	549	4.01
36	5	0,29	100	28	1397	4.02
36	20	0,14	52	58	3747	4.01

Tabella B.2: Confronto esperimenti al variare del *sendInterval*

Appendice C

Grafici di confronto

In questa appendice vengono mostrati i grafici relativi alla Tabella B.1 presente nell'appendice B.

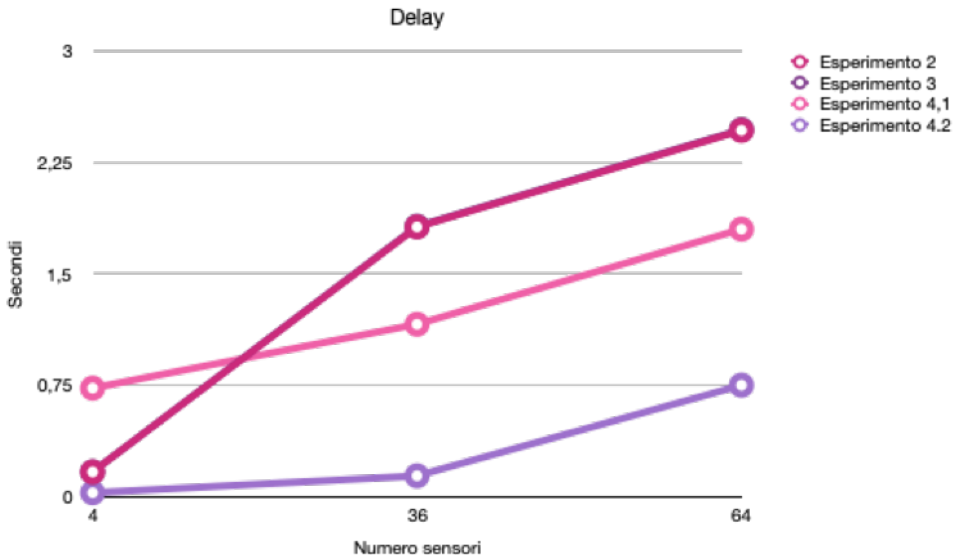

Figura C.1: Confronto Delay

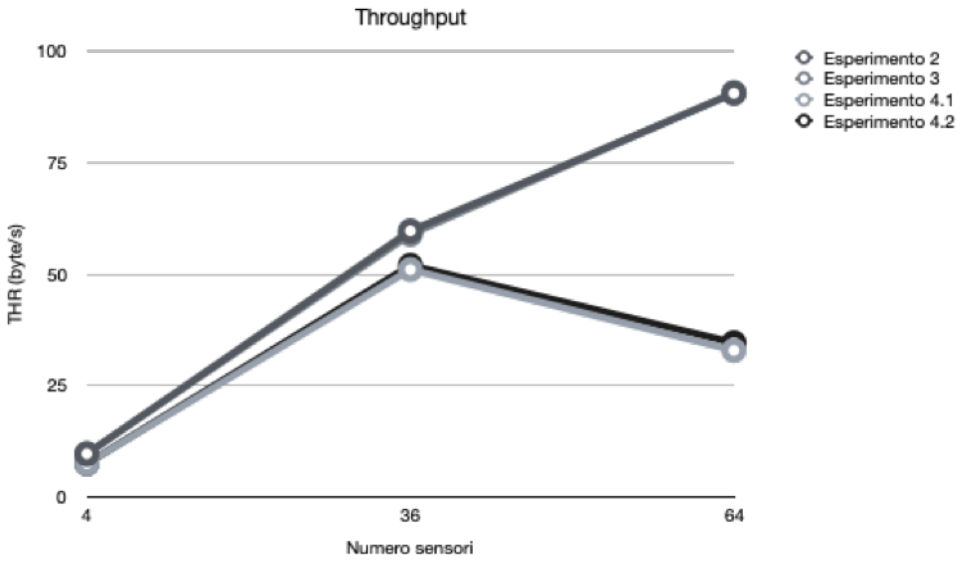

Figura C.2: Confronto THR

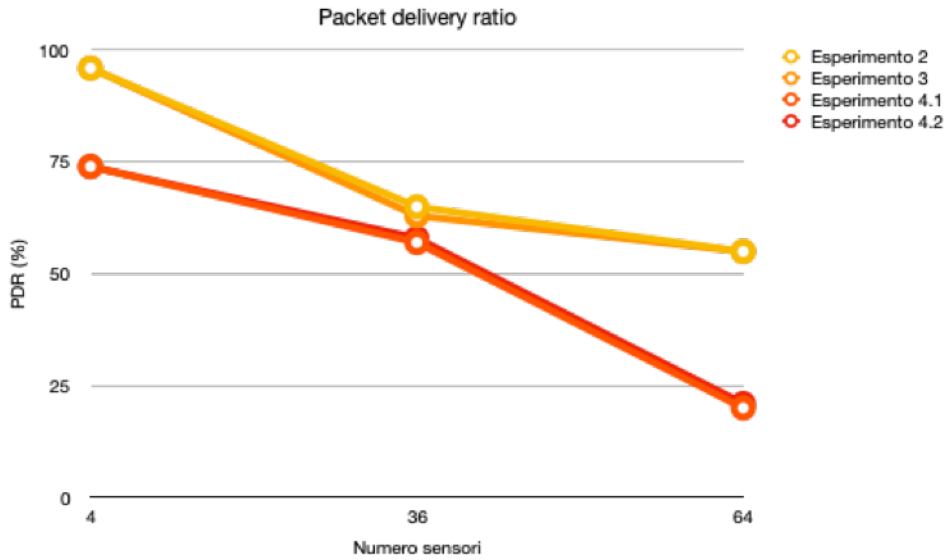

Figura C.3: Confronto PDR

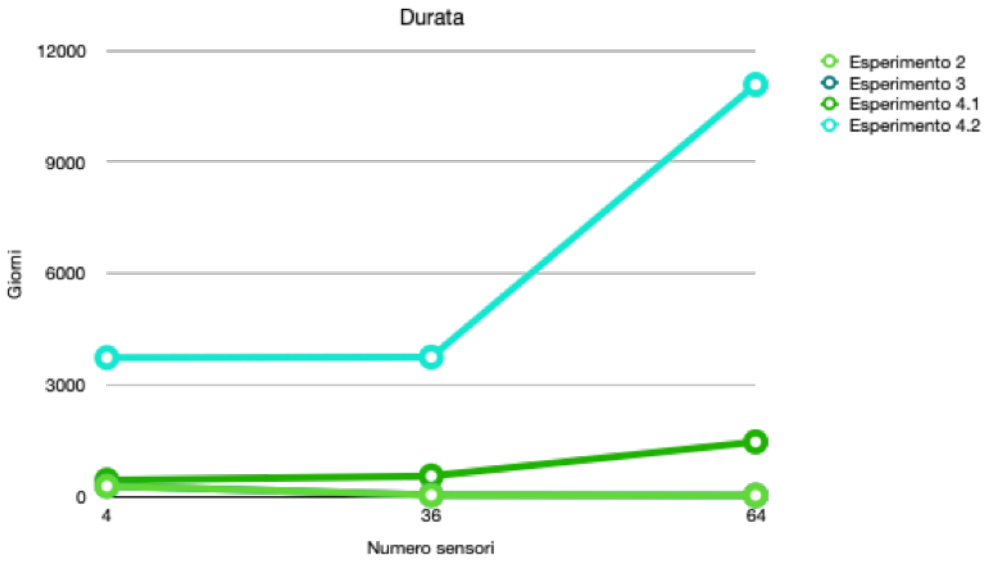

Figura C.4: Confronto lifetime

Bibliografia

[1] Electronics notes – Incorporating Radio-Electronics.com. IEEE 805.15.4 Standard: a tutorial / primer. URL https://www.electronics-notes.com/articles/connectivity/ieee-802-15-4-wireless/basics-tutorial-primer.php

[2] Agricoltura Smart – La storia di Soonapse, intervista a Marco Ciarletti. URL https://www.antoniosavarese.it/2017/10/24/agricoltura-smart-la-storia-di-soonapse-intervista-a-marco-ciarletti/

[3] House of Switzerland. Agricoltura 4.0 – la rivoluzione dello smart farming in Svizzera. URL https://houseofswitzerland.org/it/swissstories/economia/agricoltura-40-la-rivoluzione-dello-smart-farming-svizzera

[4] Hardwire. IoT e Smart Agriculture per un uso intelligente delle risorse. URL https://www.hardwire.io/it/iot/blog/iot-e-smart-agriculture-per-uso-intelligente-delle-risorse

[5] The Internet Of Things – Disi department, University Of Bologna. URL http://www.cs.unibo.it/projects/iot/

[6] Semtech – Beyond Remarkable. URL https://www.semtech.com/

[7] LoRa Alliance. URL https://lora-alliance.org/

[8] Wikipedia. Personal Area Network - Wikipedia, l'enciclopedia libera. URL https://it.wikipedia.org/wiki/Personal_Area_Network

[9] Wikipedia. Banda Radio - Wikipedia, l'enciclopedia libera. URL https://it.wikipedia.org/wiki/Banda_radio

[10] Wikipedia. CSMA/CA - Wikipedia, l'enciclopedia libera. URL https://it.wikipedia.org/wiki/CSMA/CA

[11] Agroscope – Ricerca svizzera per l'agricoltura, l'alimentazione e l'ambiente. URL https://www.wbf.admin.ch/wbf/it/home/themen/landwirtschaft/agroscope.html

[12] Soonapse. Tecnologia a supporto delle telecomunicazioni. URL https://www.soonapse.com/it/

[13] Wikipedia. Wireless Sensor Network - Wikipedia, l'enciclopedia libera. URL https://it.wikipedia.org/wiki/Wireless_sensor_network

[14] Wikipedia. Aeromobile a pilotaggio remoto - Wikipedia, l'enciclopedia libera. URL https://it.wikipedia.org/wiki/Aeromobile_a_pilotaggio_remoto

[15] Cheng Zhan, Yong Zeng and Rui Zhang, "Energy-Efficient Data Collection in UAV Enabled Wireless Sensor Network", IEEE, Aug. 2017

[16] Ozgur Koray, "Mobile Networking with UAVs: Opportunities and Challenges", IEEE, May 2013

[17] Imad Jawhar, Nader Mohamed and Jameela Al-Jaroodi, "UAV-Based Data Communication in Wireless Sensor Networks: Models and Strategies", IEEE, June 2015

[18] Sarmad Rashed and Mujdat Soyturk, "Effects of UAV Mobility Patterns on Data Collection in Wireless Sensor Networks", IEEE, June 2015

[19] Meng Hua, Yi Wang, Zhengming Zhang, Chunguo Li, Yongming Huang, and Luxi Yang, "Power-Efficient Communication in UAV-Aided Wireless Sensor Networks", IEEE, June 2015

[20] Tu Dac Ho, Jingyu Park and Shigeru Shimamoto, "Power and Performance Tradeoff of MAC Protocol for Wireless Sensor Network Employing UAV", IEEE, 2010

[21] Andrea Giorgetti, Matteo Lucchi, Marco Chiani and Moe Z. Win, "Throughput per Pass for Data Aggregation from a Wireless Sensor Network via a UAV", IEEE, 2011

[22] Marco A. M. Marinho, Edison Pignaton de Freitas, Joao Paulo C. Lustosa da Costa, Andre Lima F. de Almeida and Rafael Timoteo de Sousa Junior, "Using Cooperative MIMO Techniques and UAV Relay Networks to Support Connectivity in Sparse Wireless Sensor Networks", IEEE, 2013

[23] Dac-Tu Ho, Esten Ingar Grøtli, P. B. Sujit, Tor Arne Johansen, Joa o Borges De Sousa, "Performance Evaluation of Cooperative Relay and Particle Swarm Optimization Path Planning for UAV and Wireless Sensor Network", IEEE, 2013

[24] Say Sotheara, Kento Aso, Naoto Aomi, and Shigeru Shimamoto, "Effective Data Gathering and Energy Efficient Communication Protocol in Wireless Sensor Networks employing UAV", IEEE, 2014

[25] Ryo Sugihara and Rajesh K. Gupta, "Optimizing Energy-Latency Trade-off in Sensor Networks with Controlled Mobility", IEEE, 2009.

[26] Tu Dac Ho, Jingyu Park and Shigeru Shimamoto, "QoS Constraint with Prioritized Frame Selection CDMA MAC Protocol for WSN Employing UAV", IEEE, 2010

[27] Leandro A Villas, Daniel L. Guidoni and Jo Ueyama, "3D Localization in Wireless Sensor Networks Using Unmanned Aerial Vehicle", IEEE, 2013

[28] Leandro A. Villas, Azzedine Boukerche, Daniel L. Guidoni, Guilherme Maia, Antonio A.F. Loureiro, "A Joint 3D Localization and Synchronization Solution for Wireless Sensor Networks Using UAV", IEEE, 2013

[29] Dac-Tu Ho and Shigeru Shimamoto, "Highly Reliable Communication Protocol for WSN-UAV System Employing TDMA and PFS Scheme", IEEE, 2011

[30] Huang Zanjie, Nishiyama Hiroki, Kato Nei, Ono Fumie, Miura Ryu, Zhao Baohua, "Resource allocation for data gathering in uav-aided wireless sensor networks", IEEE, 2014

[31] Dac-Tu Ho, Esten Ingar Grøtli, P. B. Sujit, Tor Arne Johansen, Joa o Borges De Sousa, "Performance Evaluation of Cooperative Relay and Particle Swarm Optimization Path Planning for UAV and Wireless Sensor Network", IEEE, 2013

[32] Simi S, Rakesh Kurup, Sethuraman Rao, "Distributed Task Allocation and Coordination Scheme for a Multi-UAV Sensor Network", IEEE, 2013

[33] Fausto G. Costa, Jo Ueyama, Torsten Braun, Gustavo Pessin, Fernando S. Oso rio and Patr ıcia A. Vargas, "The use of unmanned aerial vehicles and wireless sensor network in agricultural applications", IEEE, 2012

[34] Manlio Bacco, Erina Ferro, Alberto Gotta, "UAVs in WSNs for Agricultural Applications: an Analysis of the Two-Ray Radio Propagation Model", IEEE, 2014

[35] Yi zhou, nan cheng, ning lu, and Xuemin (Sherman) Shen, "Multi-UAV-Aided Networks", IEEE, December 2015

[36] Andrew Wichmann, Justin Chester, Turgay Korkmaz, "Smooth Path Construction for Data Mule Tours in Wireless Sensor Networks", IEEE, 2012

[37] Huang Zanjie, Nishiyama Hiroki, Kato Nei, Ono Fumie, Miura Ryu, Zhao Baohua, "Resource allocation for data gathering in uav-aided wireless sensor networks", IEEE, 2014

[38] Kai Daniel, Sebastian Rohde and Christian Wietfeld, "Leveraging Public Wireless Communication Infrastructures for UAV-Based Sensor Networks", IEEE.

[39] Edison Pignaton de Freitas, Tales Heimfarth, Carlos Eduardo Pereira, Armando Morado Ferreira, Flavio Rech Wagner and Tony Larsson, "Experimental Analysis of Coordination Strategies to Support Wireless Sensor Networks Composed by Static Ground Sensors and UAV-carried Sensors", IEEE.

[40] Niklas Goddemeier, Kai Daniel, and Christian Wietfeld, "Role-Based Connectivity Management with Realistic Air-to-Ground Channels for Cooperative UAVs", IEEE, June 2012

[41] Ghulam Murtaza, Salil Kanhere, Sanjay Jha, "Priority-based coverage path planning for Aerial Wireless Sensor Networks", IEEE, 2013

[42] Shailendra Singh, Priya Ranjan, "Towards a new low cost, simple implementation using embedded system wireless Networking for UAVs", IEEE

[43] Simon Morgenthaler, Torsten Braun, Zhongliang Zhao, Thomas Staub, Markus Anwander, "UAVNet: A Mobile Wireless Mesh Network Using Unmanned Aerial Vehicles", IEEE, 2012

[44] Imad Jawhar, Nader Mohamed, Jameela Al-Jaroodi and Sheng Zhang, "Data Communication in Linear Wireless Sensor Networks Using Unmanned Aerial Vehicles", IEEE, May 2013

[45] Wikipedia. Throughput - Wikipedia, l'enciclopedia libera. URL https://it.wikipedia.org/wiki/Throughput

[46] Wikipedia. Latenza - Wikipedia, l'enciclopedia libera. URL https://it.wikipedia.org/wiki/Latenza

[47] Wikipedia. Radiofrequenza - Wikipedia, l'enciclopedia libera. URL https://it.wikipedia.org/wiki/Radiofrequenza

[48] Wikipedia. Instradamento - Wikipedia, l'enciclopedia libera. URL https://it.wikipedia.org/wiki/Instradamento

[49] Wikipedia. iBeacon - Wikipedia, l'enciclopedia libera. URL https://it.wikipedia.org/wiki/IBeacon

[50] Wikipedia. Effetto Doppler - Wikipedia, l'enciclopedia libera. URL https://it.wikipedia.org/wiki/Effetto_Doppler

[51] Wikipedia. Code division multiple access - Wikipedia, l'enciclopedia libera. URL https://it.wikipedia.org/wiki/Code_division_multiple_access

[52] Wikipedia. Bit Error Ratio - Wikipedia, l'enciclopedia libera. URL https://it.wikipedia.org/wiki/Bit_Error_Ratio

[53] Packet Error Rate (PER) Measurement Description. URL http://rfmw.em.keysight.com/rfcomms/refdocs/1xevdo/1xevdo_meas_cperror_desc.html

[54] World Nuclear Association. URL http://www.world-nuclear.org/information-library/current-and-future-generation/thorium.aspx

[55] Wikipedia. ZigBee - Wikipedia, l'enciclopedia libera. URL https://it.wikipedia.org/wiki/ZigBee

[56] Wikipedia. Multiple-input and multiple-output- Wikipedia, l'enciclopedia libera. URL https://it.wikipedia.org/wiki/Multiple-input_and_multiple-output

[57] Omnet++, Discrete Event Simulator. URL https://omnetpp.org/

[58] A quick overview of the Omnet++ IDE. URL https://doc.omnetpp.org/omnetpp/IDE-Overview.pdf

[59] INET Framework. URL https://inet.omnetpp.org/

[60] HTML.it, Guida C++. URL https://www.html.it/guide/guida-c2/

[61] GitHub, Built for developers. URL https://github.com/

[62] Okpedia, Kernel. URL https://www.okpedia.it/kernel

[63] Eclipse Foundation. URL https://www.eclipse.org/

[64] Telecommunication, Tabella di routing. URL http://www.cvsperoni.it/index.php/tabella-di-routing/

[65] ELEMANIA, TLC- Mod. digitali con portante sinusoidale. URL http://www.elemania.altervista.org/telecom/telecom14.html

[66] Wikipedia. Sistema embedded - Wikipedia, l'enciclopedia libera. URL https://it.wikipedia.org/wiki/Sistema_embedded

[67] Luca Bedogni, Luciano Bononi, Roberto Canegallo, Fabio Carbone, Marco Di Felice, Eleonora Franchi Scarselli, Federico Montori, Luca Perilli, Tullio Salmon Cinotti, Angelo Trotta, "Dual-Mode Wake-Up Nodes for IoT Monitoring Applications: Measurements and Algorithms", IEEE, 2018. URL https://ieeexplore.ieee.org/document/8422173