Università degli Studi di Padova

Dipartimento di Ingegneria dell'Informazione
Corso di Laurea in Ingegneria delle Telecomunicazioni

Studio di tecniche per il recupero di dati da remoto in reti acustiche sottomarine: il protocollo UFETCH

Laureando:
Loris Brolo

Relatore:
Ch.mo Prof. Michele ZORZI
Correlatore:
Dott. Ing. Paolo CASARI

Anno accademico 2012/2013

Studio di tecniche per il recupero di dati da remoto in reti acustiche sottomarine: il protocollo UFETCH.

RELATORE: Ch.mo Prof. Michele ZORZI
CORRELATORE: Dott. Ing. Paolo CASARI
LAUREANDO: Loris BROLO

15-07-2013

alla mia Famiglia,
e a Silvia...

Indice

Ringraziamenti

Un grazie va rivolto al professore e relatore di questa tesi Michele Zorzi, che mi ha dato la possibilità di lavorare su un argomento molto interessante, affascinante e poco conosciuto qual é il mondo delle reti acustiche sottomarine.
Un grazie va dato inoltre a Paolo Casari, correlatore di questa tesi, che mi ha seguito pazientemente dal primo all'ultimo giorno dandomi molti aiuti.
Non dimentico di ringraziare tutti i membri del gruppo SIGNET LAB, Federico, Giovanni, Ivano che mi hanno dato un grosso aiuto nella realizzazione di questo progetto, in particolar modo Federico che nella fase di programmazione del protocollo è stato davvero di grande sostegno (all'inizio non sapevo proprio dove andare a parare, ma alla fine con la sua disponibilità e pazienza sono riuscito a fare ciò che mi ero e ci eravamo prefissati).

Un sentito grazie lo devo ai miei genitori, Renzo e Teresa, a mio fratello Stefano e famiglia, che in questi anni mi hanno sempre incentivato e dato le giuste motivazioni per arrivare fino a questo traguardo stupendo.
A loro aggiungo una persona speciale, Silvia, che è sempre stata presente con parole di supporto anche nei momenti di scoraggiamento.

Non voglio però dimenticare i tre moschettieri e amici d'infanzia Stefano, Nicola e Nicolò, tre persone con cui nei momenti di divertimento sono di ottima compagnia e si riesce a fare molta festa, ma persone con cui è anche possibile parlare e chiedere consigli.

Spero di non aver dimenticato nessuno. Grazie a tutti!!

Acronimi

ARQ Automatic Retransmission reQuest

AUV Acoustic Underwater Vehicles

BPSK Binary Phase Shifting Keys

CBR Constant Bit Rate

CDMA Code Division Multiple Access

CSMA Carrier Sense Multiple Access

CTS Clear To Send

DACAP Distance Aware Collision Avoidance Protocol

DFE Decision Feedback Equalizer

D-MAC Dual-Media Access Control protocol

FAMA Floor Acquisition Multiple Access

FDMA Frequency Division Multiple Access

MACA Multiple Access with Collision Avoidance

NAV Network Allocation Vector

ns-2 Network Simulator-2

ns-Miracle Network Simulator-Multi Interface Cross Layer Extension

PDR Packet Delivery Ratio

p.s.d. Power Spectral Density

PSK Phase Shifting Keys

ROV Remotely Operated Vehicles

RTS Request To Send

RTT Round Trip Time

S2C Sweep Spread Carrier

SINR Signal Interference to Noise Ratio

TDMA Time Diviosn Multiple Access

WFCTS Wait First-CTS

WFDATA Wait First-DATA

UWAN UnderWater Acoustic Networks

UW-LAN UnderWater-Local Area Network

Sommario

Negli ultimi anni la comunità scientifica che opera nell'ambito delle Telecomunicazioni ha dimostrato grande interesse verso le comunicazioni acustiche sottomarine. La continua ricerca nell'arco degli anni ha portato ad oggi ad avere da tali tipi di reti miglioramenti in termini di performance e affidabilità molto elevate se confrontate con i primi sistemi di comunicazione sviluppati. Molti articoli forniscono una eccellente review a riguardo dello sviluppo di tale settore: per ulteriori e più approfondite informazioni si rimanda alle seguenti letture: [1]-[2]-[3].

Le reti acustiche sottomarine sono di notevole importanza dato che sono di fondamentale supporto alle continue missioni di monitoraggio, di sorveglianza e pattugliamento degli oceani, e sono da considerarsi profondamente diverse dalle reti di comunicazoni radio terrestri. Tali differenze si possono riscontrare sia a livello fisico, sia a livello tecnologico e infine anche a livello economico.
Il fattore chiave che differenzia le reti sottomarine da qualsiasi altra tipologia di rete è l'utilizzo di un canale acustico. Questo diventa un problema nel caso in cui si volessero applicare le tecnologie radio utilizzate per le reti terrestri. La propagazione in ambito subacqueo del segnale avviene ad una velocità di 1500 m/s, nel caso di reti radio terrestri invece, con l'utilizzo delle onde radio, la velocità di propagazione è pari alla velocità delle luce, cioé 3×10^8 m/s, cinque ordini di grandezza superiore. La conseguenza di tutto ciò è, nelle reti UWAN, l'elevato ritardo di propagazione, che come sarà osservato nei prossimi capitoli, è il principale elemento che implica la revisione di tutti i protocolli e tecnologie utilizzate in reti radio, affinché possano essere applicate in ambito sottomarino.
Sempre a livello fisico, il canale acustico sottomarino introduce una forte interferenza multipath causando un canale con fading selettivo in frequenza. Questo comporta che per avere una elevata efficienza di banda, sia richiesto l'utilizzo intensivo e

congiunto di sistemi avanzati di signal processing.

Gli elementi che costituiscono una rete acustica sottomarina pongono forti limitazioni tecnologiche, un esempio è dato dal fatto che i normali nodi sensori non possono simultaneamente trasmettere e ricevere, infatti la comunicazione è quasi sempre half-duplex. Un ulteriore problema è dato dalle basse velocità di trasmissione per reti UWAN installate in acque poco profonde, infatti si è nell'ordine di grandezza dei 5 kbps, che sono ben distanti da quelle utilizzate per reti radio terrestri.

Il seguente elaborato si prefissa l'obiettivo di lavorare ad un livello leggermente più alto rispetto a quello fisico, ovvero a livello MAC, considerando lo stack protocollare ISO/OSI.
Uno dei principali problemi, su cui è da molti anni che ricercatori stanno lavorando, è la modalità di accesso al mezzo, essendo questo condiviso fra un insieme di nodi sensori sottomarini. L'accesso al canale deve essere esclusivo: ciò significa che entro un'area prestabilita un solo nodo alla volta vi può accedere; se così non fosse, le informazioni trasmesse sarebbero corrotte e quindi sarebbe necessaria una ritrasmissione con conseguente diminuzione del throughput di sistema.
Questo è ancora oggi un problema aperto tra la comunità scientifica che lavora in questo ambito, poiché a causa di tutte le problematiche di livello fisico descritte appena sopra, la maggior parte dei protocolli di livello MAC che sono stati realizzati per reti radio terrestri non è adatta per reti UWAN.
Quindi il fine della relazione è quello di presentare un nuovo protocollo (UFETCH) di accesso al canale, che consenta di ottimizzare il throughput di sistema e che permetta il recupero dei dati raccolti dai nodi sensori da parte di un elemento posto in superficie, con un end-to-end delivery time confrontabile con i protocolli già esistenti, ed una packet error rate che sia la più bassa possibile.

Nella fase di progettazione di un protocollo di livello MAC, un ultimo problema di cui tenere conto, ma non ultimo per importanza, è l'efficienza energetica. Questo è molto importante per una rete acustica sottomarina perché tutti gli elementi che la costituiscono sono alimentati a batteria, e com'é noto la capacità di queste non è infinita. Inoltre a differenza delle reti radio terrestri, dove la potenza richiesta per la trasmissione e ricezione è pressoché la stessa, in ambito subacqueo è richiesta una maggiore potenza per la trasmissione che a volte può superare quella di ricezione

di un fattore pari a 100. In conseguenza di ciò, particolare attenzione deve essere riservata al problema del consumo, poiché la sostituzione di una batteria di un nodo sensore posto magari ad una profondità di 100 m non è immediata, dovranno essere impiegati macchinari che potrebbero far aumentare i costi della rete, oltre ai già elevati costi degli elementi base quali nodi sensori o AUV che la costituiscono.

L'elaborato è quindi così organizzato: si parte dal capitolo 1 il quale fornisce una breve descrizione del canale sottomarino presentando in maniera abbastanza dettagliata i problemi che questo introduce ed eventuali soluzioni attualmente disponibili per mitigarli.
Il capitolo 2 effettua una breve panoramica dei protocolli per l'accesso al canale sottomarino. Per ognuno di questi sarà indicato il principio di funzionamento e alcuni risultati in termini di throughput e consumo energetico utili per indicare la bontà del protocollo rispetto ad altri.
Il capitolo 3 presenta un'analisi preliminare realizzata in ambiente Matlab. Tale analisi è stata effettuata con lo scopo di verificare quali potrebbero essere i margini di miglioramento utilizzando una tecnica ibrida per la raccolta dei dati, ampiamente descritta in questo capitolo. Tale tecnica deriva dalle altre due, Polling e Multihop routing, che sono alla base poi dei protocolli uwPOLLING e MSUN.
Il capitolo 4 è centrale in questa relazione, e presenta nei minimi dettagli il nuovo protocollo chiamato UFETCH. In questo capitolo saranno descritte le entità che costituiscono il protocollo, indicando anche qual è la tecnologia attualmente presente nel mercato, successivamente saranno descritte le macchine a stati di ogni singolo elemento, cioé quali sono i compiti che ogni entità esegue durante il suo funzionamento descrivendo anche in maniera dettagliata tutta la segnalazione che circola nella rete affinché i dati possano essere recuperati dalla stazione base.
L'elaborato si conclude con il capitolo 5, dove verranno descritte le simulazioni eseguite, esposti i grafici ottenuti ed infine effettuati i commenti riguardo le performance del protocollo UFETCH rispetto ad altri due protocolli quali MSUN ed uw-POLLING.

Capitolo 1

Reti acustiche sottomarine

Nel seguente capitolo sarà effettuata una breve panoramica dello stato attuale relativo alle reti acustiche sottomarine, denominate d'ora in poi anche con l'acronimo UWAN.
Il capitolo inizierà con la presentazione di una classica topologia di rete sottomarina descrivendo in maniera abbastanza dettagliata gli elementi di cui essa è costituita. Successivamente sarà presentato il modello di canale sottomarino utilizzato in seguito per tutte le simulazioni effettuate. La descrizione è seguita da una trattazione matematica in modo da mostrare come avviene la propagazione del segnale in un tale scenario. Particolare attenzione è posta nella descrizione del path loss e del rumore introdotto dall'ambiente di propagazione subacqueo.
Nell'ultima parte del capitolo sarà inoltre presentata la relazione che sussiste tra la larghezza di banda e la capacità di canale, questo è molto importante dato che in ambito sottomarino uno dei principali problemi è appunto la ridotta disponibilità di banda.

1.1 La topologia di rete

Una rete acustica sottomarina può essere sintetizzata come mostrato in figura (1.1). La rete consta di un insieme di reti locali UW-LAN, note anche come cluster o celle, costituite da una collezione di nodi sensori posti ad una certa profondità e collegati tra di essi mediante un canale acustico. L'intera rete è poi connessa

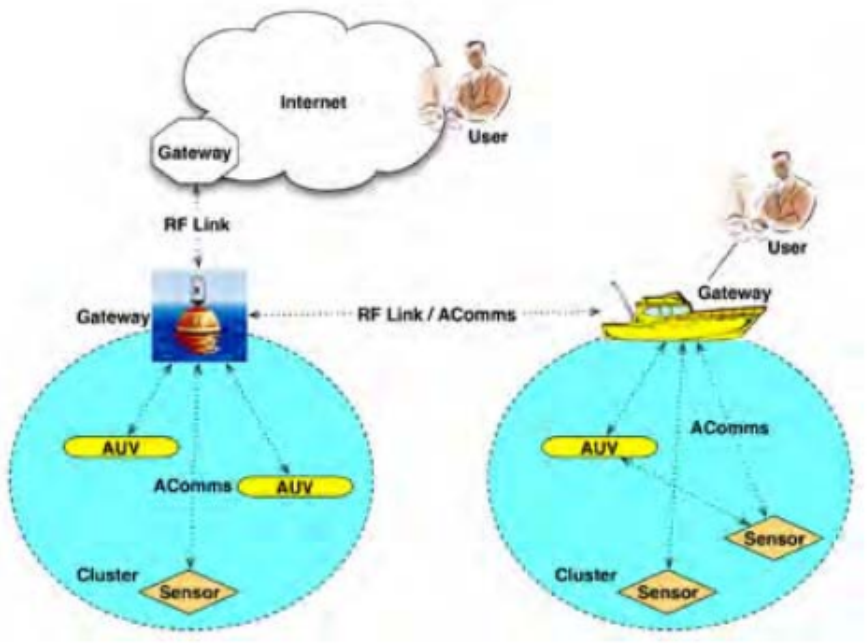


Figura 1.1: Topologia di una rete acustica sottomarina.

ad una stazione posta in superficie, gateway, che a sua volta può essere allacciata mediante un link RF alla rete Internet o a altre reti wireless e/o cablate. Il gateway fornisce funzioni di amministrazione, sicurezza e routing tra le varie reti UW-LAN.

Due sono le tipologie di reti acustiche sottomarine e la distinzione è effettuata sulla base della profondità alla quale vengono disposti i nodi sensori:

- deep-water networks,
- shallow-water networks.

Le deep-water networks introducono meno problemi sotto il punto di vista della propagazione del segnale rispetto alle shallow-water network, tanto che, negli ultimi anni, negli oceani, in alcuni punti di elevata profondità, si sono riuscite ad installare reti che comunicano con bit-rate abbastanza alte.
Le shallow-water networks invece aggiungono tutta una serie di problematiche che complicano la realizzazione di una rete di comunicazione sottomarina. Di seguito, e per il resto dell'elaborato, si farà riferimento esclusivamente a questa seconda categoria dato che sono quest'ultime ad essere maggiormente utilizzate

nella realtà. Instaurare una rete ad una profondità elevata richiede sia uno sforzo economico dovuto all'utilizzo di strumenti più sofisticati in termini di manodopera e di elementi (nodi sensori) che costituiscono la rete, sia una difficoltà di installazione e mantenimento delle risorse.

Entrando più nel dettaglio, una shallow-water network tipica è costituita da un elevato numero di nodi sensori che operano ad una profondità compresa tra i [50÷200] m e separati tra di essi da una distanza inferiore ai 10 km. Questi possono essere distribuiti nell'area marina di interesse senza alcuna particolare geometria, solamente se richiesto vengono disposti secondo un certo ordine. L'insieme di nodi costituisce la rete UW-LAN di cui sopra specificato e rappresentata in figura (1.1).

I nodi riposti sul fondale sono equipaggiati di sensori capaci di monitorare un'area circostante che può raggiungere anche dimensioni di 2 km^2. Il controllo continuo riguarda eventuali variazioni della salinità dell'acqua, della torbidità, la presenza di eventuali elementi, quali possono essere navi o pesci, che cambiano l'ambiente di propagazione. Tutte le informazioni raccolte saranno successivamente trasmesse dal nodo stesso ad una stazione base, solitamente riposta nel fondale. Quest'ultima unità operativa ha sotto il proprio controllo circa una decina di nodi sensori, e sarà a sua volta collegata ad una boa di superficie, il gateway di figura (1.1), su cui è implementato un collegamento radio affinché la rete subacquea possa essere collegata ad altre reti.

Un vincolo molto importante da rispettare in ambito sottomarino è il consumo energetico da parte degli elementi che costituiscono la rete. I modem acustici sono alimentati da una batteria e il ricambio quotidiano, ma anche settimanale, non è un'opzione preventivabile dal momento che il modem, installato nel nodo sensore, è posizionato nel fondale marino ad una profondità fino a 200 m. Oltre alla difficoltà di sostituzione, il problema è anche relativo ai costi, infatti la batteria di per sé ha un costo, come anche le strumentazioni utilizzate per la sostituzione.
Al fine di ridurre il più possibile il problema di consumo di energia due sono le soluzioni implementate:

1. ai nodi sono assegnate due modalità di lavoro, *sleep* e *active*. Il nodo si trova in modalità *sleep* quando non deve effettuare la comunicazione con il gateway, ma solamente monitorare l'area di sua competenza. In questo stato l'energia

consumata è molto bassa.
La presenza del nodo nello stato *active*, significa invece che deve avvenire la comunicazione tra nodo e gateway, in questo caso il risveglio è a carico del gateway il quale manda un segnale acustico per informare il nodo della volontà di voler comunicare.

2. si realizza una rete multihop, dove solamente i nodi più vicini al gateway hanno la possibilità di comunicare con esso, mentre gli altri dovranno propagare le proprie informazioni a nodi ad esso vicini finché queste non arriveranno ai nodi in prossimità del gateway. Una tale soluzione consente di utilizzare una potenza di trasmissione minore, dato che lo spazio da percorrere per il segnale è più piccolo rispetto al caso in cui ogni nodo, posto anche a distanze elevate dal gateway, debba comunicare direttamente con quest'ultimo.
In sostanza, i nodi che non sono visibili al gateway passeranno le loro informazioni ai nodi ad essi più vicini, la trasmissione sarà con potenza ridotta, ottenendo così un elevato risparmio energetico.
La tecnica multihop, come sarà descritto nel capitolo 3, se considerata una situazione ideale in cui i nodi non prediligono sempre la medesima rotta per la consegna dei pacchetti, non è la migliore soluzione implementabile, infatti in tal caso lo svantaggio principale introdotto è l'elevato ritardo nella consegna dell'informazione e la presenza di alcuni colli di bottiglia soprattutto in prossimità di quei nodi vicini al gateway. Nel capitolo 5 invece, in cui saranno mostrati i risultati relativi ad MSUN, protocollo che si basa su multihop routing, si vedrà che il ritardo di consegna migliorerà notevolmente, dato che i nodi implementano dapprima un algoritmo di routing in cui individuano il percorso migliore da utilizzare per inviare i pacchetti, e mediante tale tragitto poi spediranno un flusso informativo. In tal modo, la non ricerca continua di nuove rotte consente un guadagno in termini temporali nel ritardo di consegna end-to-end.

In conclusione, durante la fase di progettazzione di una rete UWAN, si dovrà cercare un tradeoff tra la qualità del servizio offerto dalla rete, cioé massimo throughput con il minimo ritardo di consegna delle informazioni e il consumo energetico dei vari componenti.

1.2 Il canale underwater

La comunicazione per mezzo di un canale sottomarino non può essere paragonata a quella di un canale radio terrestre, in conseguenza di ciò un modello di canale valido per queste ultime reti non sarà applicabile per le comunicazioni sottomarine. Soprattutto nel caso di reti implementate in acque poco profonde, come già espresso nel sottoparagrafo 1.1, il canale esibisce un lungo ritardo di propagazione a causa delle numerose componenti multipath che incidono al ricevitore.

Il problema del multipath si presenta nelle comunicazioni non cablate e deriva dal fatto che un segnale emesso dal trasmettitore con una certa potenza P_{in}, durante la sua propagazione verso il ricevitore può incontrare una serie di ostacoli: mura di edifici e/o automobili per un ambiente urbano, alberi e montagne nel caso di ambiente non urbano, mentre in ambito sottomarino il fondale, ma anche le stesse onde superficiali che riflettono il segnale, creando così delle componenti che seguono un percorso diverso dalla componente principale, la quale trasporta la maggior quantità di potenza. Le diverse traiettorie eseguite fanno si che al ricevitore arrivino diverse repliche del segnale trasmesso in istanti temporali distinti, con ritardi, fasi e attenuazioni differenti rispetto alla componente diretta (line-of-sight) qualora essa esista.
Ipotizzando che il trasmettitore emetta il segnale $s(t)$ dato da:

$$s(t) = u(t)e^{j2\pi f_0 t} \tag{1.1}$$

con $u(t) = a(t) + jb(t)$ l'inviluppo complesso espresso come somma tra la componente in fase $a(t)$ e in quadratura $b(t)$, f_0 la frequenza portante utilizzata per la trasmissione, allora il segnale ricevuto è esprimibile come:

$$y(t) = r(t)e^{j2\pi f_0 t} \tag{1.2}$$

con $r(t)$ che tiene conto degli N cammini multipath generati dalla riflessione con gli elementi che il segnale trova lungo il suo percorso di propagazione:

$$r(t) = \sum_{i=1}^{N} \alpha_n(t)u(t - \tau_n(t))e^{-j2\pi((f_0+f_{D,n(t)})\tau_n(t)-f_{D,n(t)}t)} \tag{1.3}$$

Definendo la fase dell'n-esimo cammino come:

$$\phi_n(t) = 2\pi((f_0 + f_{D,n(t)})\tau_n(t) - f_{D,n(t)}t) \tag{1.4}$$

allora la relazione (1.3) diventa:

$$r(t) = \sum_{i=1}^{N} \alpha_n(t)u(t - \tau_n(t))e^{-j\phi_n(t)} \tag{1.5}$$

Le relazioni (1.2), (1.4) e (1.5) affermano quindi che il problema dei cammini multipath genera:

- diverse repliche del segnale trasmesso, ognuna delle quali ha una propria attenuazione che dipende dal percorso eseguito, una propria fase e frequenza di Doppler $f_{D,n(t)}$;
- variazioni repentine della fase dell'inviluppo $\phi_{n(t)}$, queste generate dal fatto che $f_0 + f_{D,n(t)}$ è un valore elevato e quindi una piccola variazione del ritardo di propagazione $\tau_n(t)$ provoca una variazione della fase dell'inviluppo, il tutto creando interferenza distruttiva e costruttiva.

L'ultimo punto è quello maggiormente delicato da considerare, poiché se il ritardo di fase è pari a 180° allora l'interferenza è tale da cancellare il segnale diretto causandone la perdita.

La descrizione fin qui effettuata è valida anche per canali acustici sottomarini, soprattutto nel caso di canali shallow-water.
I rimedi a riguardo il problema del mutlipath possono essere alleviati utilizzando schemi di modulazione coerenti quali BPSK o PSK combinati assieme a equalizzatori DFE e tecniche di spatial-diversity combining. Le tecniche appena citate e ulteriori metodi per la riduzione del multipath non sono qui trattate in quanto ciò esula dallo scopo dell'elaborato, ma per maggiori dettagli si rimanda alle letture [4], [5].

1.2.1 Path Loss e rumore

I canali acustici sottomarini sono caratterizzati da path loss che dipende principalmente da due fattori:

1. dalla distanza tra trasmettitore e ricevitore come avviene nel caso di comunicazioni radio terrrestri;

2. dalla frequenza con cui viene trasmesso il segnale.

La frequenza di trasmissione influenza la perdita per assorbimento a causa della trasformazione dell'energia acustica in calore, che a sua volta, implica la dipendenza tra la larghezza di banda utilizzabile e la distanza tra trasmettitore e ricevitore [7]. La perdita per assorbimento è proporzionale alla frequenza di trasmissione e alla distanza tra i due terminali che intendono comunicare.

L'attenuazione, espressa in scala logaritmica e citata in [6], nel caso di un canale acustico sottomarino è data dalla seguente relazione:

$$10 \log A(l, f) = 10k \log(1000l) + 10l \log a(f) \tag{1.6}$$

con l m la distanza tra trasmettitore e ricevitore, f kHz la frequenza di trasmissione, k lo spreading factor, che può essere assunto pari a k=1.5, e $a(f)$ il coefficiente di assorbimento.
Nell'equazione (1.6) il primo addendo esprime lo *spreading loss*, mentre il secondo l'*absorption loss*.
L'absorption loss può essere espressa in maniera empirica utilizzando la formula di Thorp, che nel caso di frequenze superiori alle centinaia di Hz vale:

$$a(f) = 0.11 \frac{f^2}{1+f^2} + 44 \frac{f^2}{4100+f^2} + 2.751\dot{0}^{-4} f^2 + 0.003 \tag{1.7}$$

mentre per frequenze più basse è:

$$10 \log a(f) = 0.002 + 0.11 \frac{f^2}{1+f^2} + 0.011 f^2 \tag{1.8}$$

Dalle relazioni (1.7) e (1.8) si può vedere come l'absorption loss sia fortemente dipendente dalla frequenza e questo è il fattore che maggiormente influenza il limite superiore delle frequenze utilizzabili per un canale acustico data una certa lunghezza l del link.

Il path loss, dato dalla relazione (1.6), descrive l'attenuazione su un singolo path. Ipotizzando che un solo tono venga trasmesso con la frequenza f e con potenza P su tale path, allora il segnale ricevuto avrà potenza pari a $P/A(l, f)$. Assumendo che ci siano più path ognuno di lunghezza l_p e su ognuno di essi la trasmissione avvenga con potenza P_p, allora la funzione caratteristica del canale è data dalla sommatoria:

$$H(l, f) = \sum_{p=0}^{P-1} \frac{\Gamma_p}{\sqrt{A(l_p, f)}} e^{-j2\pi f \tau_p} \tag{1.9}$$

con Γ_p il modello di perdita associato al path p-esimo, p=0, 1, ..., $P-1$, $\tau_p = l_p/c$ il ritardo di propagazione, mentre c=1500 m/s indica la velocità di propagazione, equivalente alla velocità del suono nell'acqua.

Allo stesso modo dell'attenuazione è possibile ricavare un modello per il rumore introdotto dal canale acustico sottomarino. Quattro sono le principali fonti di rumore in un ambiente subacqueo:

1. turbolenze;
2. navigazione;
3. onde;
4. rumore termico.

Tali sorgenti di rumore possono essere sintetizzate mediante statistiche Gaussiane e le relative p.s.d. Grazie a queste si ottengono le relazioni associate ad ogni fonte di rumore e qui presentate:

$$10 \log N_t(f) = 17 - 30 \log f \tag{1.10}$$

$$10 \log N_s(f) = 40 + 20(s - 0.5) + 26 \log f - 60 \log(f + 0.03) \tag{1.11}$$

$$10 \log N_w(f) = 50 + 7.5\sqrt{w} + 20 \log f - 40 \log(f + 0.4) \tag{1.12}$$

$$10 \log N_{th}(f) = -15 + 20 log f \tag{1.13}$$

Il rumore dovuto alle turbolenze e descritto dalla relazione (1.10) è dominante soprattutto alle basse frequenze, cioé valori inferiori a 10 Hz. Il rumore introdotto dalla navigazione delle navi, descritto dalla formula (1.11), con s fattore di attività

di navigazione, valore compreso tra 0 (bassa attività) e 1 (elevata attività), è prevalente a frequenze comprese nell'intervallo [10÷100] Hz. Per quanto riguarda le ultime due sorgenti di rumore, per range di frequenze comprese tra [0.1÷100] kHz, il fattore principale che genera rumore é il movimento delle onde superficiali, descritto da (1.12), mentre l'ultimo fattore, (1.13), indicante il rumore termico, è rilevante per frequenze superiori ai 100 kHz.
La somma tra le relazioni (1.10), (1.11), (1.12) e (1.13) consente di ottenere la p.s.d.:

$$N(f) = N_t(f) + N_s(f) + N_w(f) + N_{th(f)} \tag{1.14}$$

raffigurata in figura (1.2).
Dal grafico si può osservare come il rumore decada all'aumentare della frequenza limitando però di fatto il lower bound della banda acustica utilizzabile. Solitamente, la frequenza utilizzata per comunicazioni underwater è nell'intorno dei 25 kHz, frequenza che sarà impiegata per tutte le simulazioni che saranno di seguito presentate.

Effettuando il rapporto tra l'attenuazione $A(l, f)$ data da (1.6) e la p.s.d. $N(f)$ descritta da 1.14, si ottiene il rapporto segnale rumore SNR:

$$SNR(l, f) = \frac{P}{A(l, f)N(f)\Delta f} \tag{1.15}$$

con P che identifica la potenza di trasmissione mentre Δf coincide con la larghezza di banda del rumore al ricevitore.
Dalla relazione (1.15) si può notare come il rapporto SNR dipenda dalla distanza tra trasmettitore-ricevitore e dalla frequenza portante utilizzata per la comunicazione. Considerando la quantità $1/(A(l, f)N(f))$, il cui andamento è rappresentato in figura (1.3(a)) al variare della lunghezza del link di comunicazione, è possibile notare come minore é la distanza tra trasmettitore e ricevitore e maggiore sarà la frequenza utilizzabile ottenendo così valori di SNR ottimali.
In maniera più dettagliata è possibile mostrare la relazione che sussiste tra la distanza e la frequenza utilizzata per la comunicazione, in particolar modo come mostrato dalla figura (1.3(b)), è visibile l'esistenza di una frequenza portante ottima per cui si abbia il valore massimo di SNR.

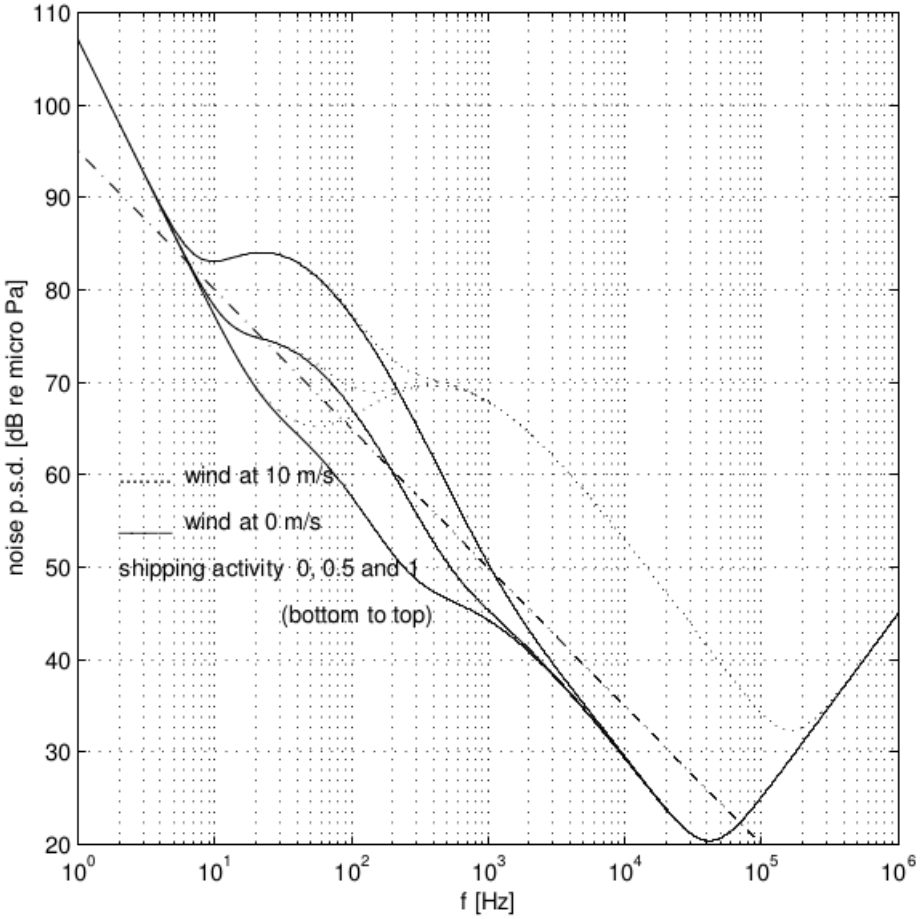
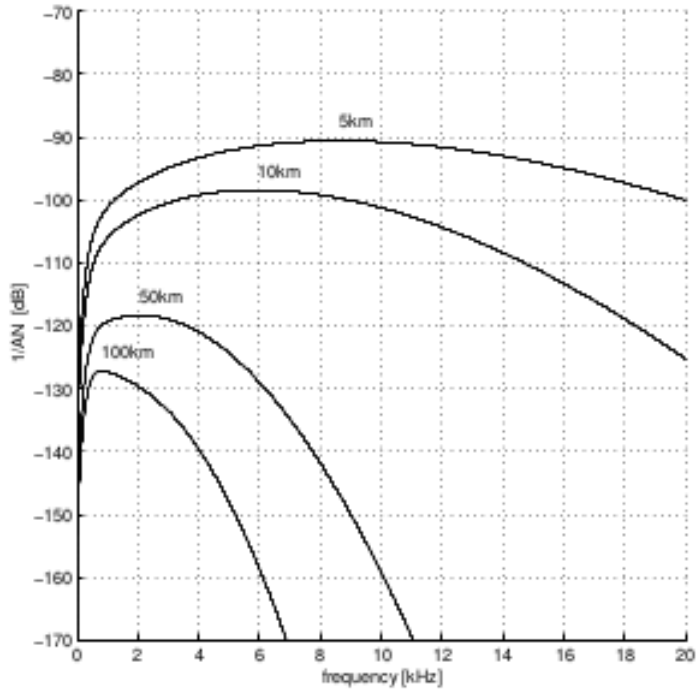


Figura 1.2: Power spectral density del rumore in funzione della frequenza utilizzando varie situzioni di vento ed attività navali.

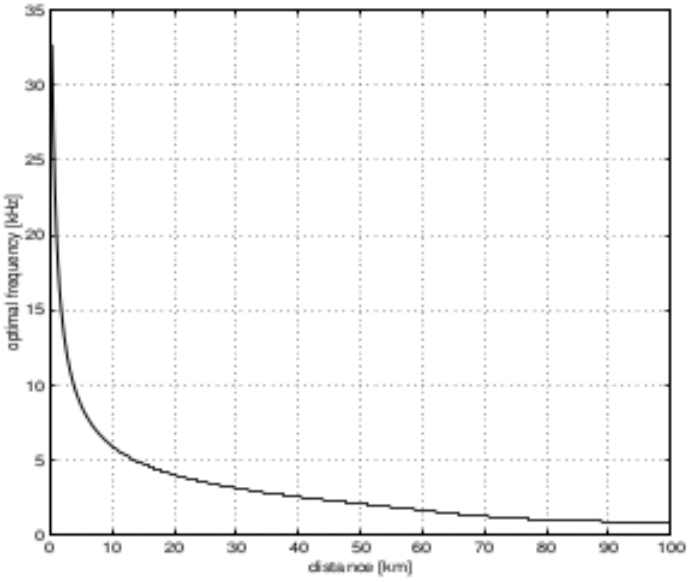
1.2.2 Relazione tra larghezza di banda e capacità in canali underwater

Uno dei problemi maggiori per le comunicazioni acustiche sottomarine, come già accennato nei sottoparagrafi precedenti, è la ridotta disponibilità di larghezza di banda. Un eventuale utilizzo di bande molto elevate aumenterebbe il problema del path loss con una conseguente scarsa qualità del segnale. Inoltre un fattore determinante per la scelta della larghezza di banda è la lunghezza del link di comunicazione, più è elevata quest'ultima e minore sarà la banda a disposizione.

Di seguito verrà presentato, in maniera non fortemente dettagliata, il legame



(a) Dipendenza dell'SNR dalla frequenza utilizzando diverse lunghezze del canale di comunicazione, spreading factor k=1.5 per il calcolo dell'attenuazione, un'attività navale media pari a s=0.5 e in assenza di vento w=0 .



(b) Frequenza portante ottima che consente di ottenere un valore di SNR ottimale sulla base della distanza tra trasmettitore e ricevitore scelta.

che sussiste tra la capacità di canale e la larghezza di banda, la descrizione completa è consultabile in [7].
Si ipotizzi di avere una banda a disposizione B_w che viene suddivisa in piccole sottobande, tutte della medesima dimensione pari a Δf a cui ad ognuna è assegnata una frequenza f_i con i=1, 2, ...,N ed N il numero di sottobande create. A seguito di ciò la capacità totale di canale può essere definita come la somma delle singole capacità di ogni sottobanda creata.
In termini matematici la capacità $C(f)$, assumendo che $C_i(f)$ sia la capacità associata alla singola sottobanda i-esima con $i = 1, \ldots, N$ è pari a:

$$C(l) = \sum_{i=1}^{N} C_i(f) \tag{1.16}$$

Approssimando un rumore bianco AWGN con p.s.d. $N(f_i)$ ad ogni singola sottobanda e indicando con $S_l(f)$ la p.s.d. del segnale trasmesso, allora la relazione (1.16) può essere così riscritta:

$$C(l) = \sum_{i=1}^{N} \Delta f \log_2 \left(1 + \frac{S_i(f)}{A(l, f_i)N(f_i)}\right) \tag{1.17}$$

Massimizzando la p.s.d. del segnale e sottostando al vincolo sulla potenza totale utilizzata per la trasmissione del segnale che dovrà essere finita, applicando l'algoritmo di Water-Filling [5] è possibile ottenere la distribuzione di energia ottima.
La potenza totale di trasmissione, anch'essa dipendente della distanza tra trasmettitore e ricevitore, è data da:

$$P(l) = \int_{B(l)} S_l(f) df \tag{1.18}$$

Si osservi che $P(l)$ dovrà essere scelta in modo tale da poter avere un rapporto SNR al ricevitore al di sopra di una data soglia stabilita a priori.
Il rapporto SNR sfruttando la definizione (1.15), e sostituendo in essa la relazione

(1.18), vale:

$$\begin{aligned} SNR(l, B(l)) &= \frac{\int_{B(l)} S_l(f) A^{-1}(l, f) df}{\int_{B(l)} N(f) df} \\ &= K_l \frac{\int_{B(l)} A^{-1}(l, f) df}{\int_{B(l)} N(f) df} \end{aligned} \tag{1.19}$$

con K_l valore costante scelto in modo tale da soddisfare l'uguaglianza a seguito dell'algoritmo di Water-Filling:

$$S_l(f) + A(l, f)N(f) = K_l \tag{1.20}$$

In conclusione la capacità di canale in presenza di una distanza l tra trasmettitore-ricevitore e una data soglia di rumore SNR_0, è ottenibile sfruttando le definizioni (1.18), (1.19), (1.20). Il risultato finale è dato da:

$$C(l) = \int_{B(l)} \log_2 \left(\frac{K_l}{A(l, f)N(f)} df \right) \tag{1.21}$$

A seguito delle descrizioni effettuate si può osservare come la larghezza di banda per un canale acustico sottomarino sia limitata a causa della dipendenza del path loss dalla frequenza. Dalle relazioni viste si può affermare inoltre che minore è la lunghezza del link di comunicazione e maggiore sarà la banda utilizzabile. Tutto ciò inevitabilmente va ad influire nella progettazione della rete, infatti per ottimizzare tutti i parametri in gioco, la scelta del posizionamento dei nodi, il throughput, l'allocazione ottimale delle risorse e la capacità di rete dovranno essere quindi tutte espresse in funzione della distanza, come appunto è stato presentato nei sottoparagrafi di questo capitolo.

Capitolo 2

Protocolli di medium access control

Nelle reti costituite da un numero elevato di nodi che fanno utilizzo di un canale condiviso per la comunicazione, qual é una rete acustica sottomarina, uno dei principali problemi che sorge è la gestione dell'accesso al mezzo di comunicazione in modo che non ci siano collisioni. Con il termine collisione ci si riferisce al fatto che parte di un pacchetto trasmesso da un nodo si sovrappone parzialmente ad un pacchetto di un secondo nodo anch'esso già presente nel canale. Un tale evento crea tipicamente la perdita di tutte le informazioni immesse nel mezzo di comunicazione da entrambe le entità.
In questi casi di competizione, si necessita di un meccanismo all'interno della rete che stabilisca tra tutti i nodi presenti chi abbia precedenza, in che modo e per quanto tempo abbiano l'accesso esclusivo al mezzo. Tale strategia è implementata al sottostrato MAC appartenente al livello data link dello stack protocollare ISO/OSI. Si osservi, come si vedrà nel dettaglio in seguito, che non ci deve essere per forza di cose un elemento fisico che coordina tutte le operazioni, ma tale compito può essere redistribuito tra tutti i nodi, in tal caso si ha un'auto-gestione da parte dei nodi in modo da non creare collisioni.

Negli ultimi anni, e ancora oggi, molta ricerca è stata effettuata in questo ambito portando alla realizzazione di un elevato numero di protocolli, alcuni dei quali saranno descritti nel seguente capitolo.

La maggior parte di questa continua innovazione è stata effettuata soprattutto per reti di sensori terrestri, ma negli ultimi tempi, con la nascita e il diffondersi delle reti acustiche sottomarine, il problema si è riversato anche su quest'ultima categoria.
La complicazione sta nel fatto che il canale acustico sottomarino ha delle proprietà di propagazione totalmente differenti dal canale wireless terrestre, si pensi che la distanza di 1.5 Km è percorsa dal segnale in 1 s nel primo caso, mentre in appena 5 μs nel secondo caso; in conseguenza di ciò la maggior parte dei protocolli progettati per quest'ultime reti non è applicabile direttamente in ambito subacqueo. Tutta la ricerca effettuata fino ad ora in ambito terrestre è però una base di partenza per la progettazione di protocolli specifici per reti acustiche sottomarine, infatti molti derivano da un adattamento di protocolli per reti radio terrestri.

Il capitolo seguente si pone quindi come obiettivo quello di fornire una panoramica dei protocolli attualmente utilizzati a livello MAC per reti UWAN, il tutto fatto a partire dai primi protocolli sviluppati fino ad arrivare a quelli più recenti ed attualmente utilizzati.

Prima di immergersi nella descrizione dei vari protocolli, alcune parole devono essere spese per indicare quali sono gli obiettivi che un determinato protocollo deve raggiungere.
Come un qualsiasi protocollo di livello MAC progettato per reti radio terrestri, anche in ambito sottomarino l'obiettivo principale è quello di realizzare un metodo per accedere al canale che eviti collisioni tra i nodi. Il tutto deve essere effettuato in modo da massimizzare il più possibile il throughput e minimizzare il ritardo end-to-end di consegna dei frame.
Soprattutto nelle reti acustiche sottomarine, oltre ai due parametri sopra indicati, un terzo elemento deve essere considerato: il consumo energetico. In queste reti la potenza utilizzata per la trasmissione predomina rispetto a quella di ricezione arrivando la prima anche a valori di 100 volte superiori. Solitamente un nodo sensore ha consumi di potenza che si aggirano nell'intorno di 2.5 mW durante la fase di inattività (idle), da 5 mW fino a 1.3 W per le fasi di ascolto e decodifica del pacchetto, mentre per quanto concerne la fase di trasmissione si hanno valori che variano da 2.8 W fino anche a 50 W, il tutto dipende dalla distanza tra i nodi che devono comunicare.

Infine può essere effettuata una classificazione dei protocolli sulla base della modalità di allocazione del canale. Le due tipologie di allocazione sono:

- *statica*, la banda a disposizione viene suddivisa in parti eguali tra i vari utenti che costituiscono la rete. Questa tipologia può portare ad uno spreco di banda perché nel caso in cui un nodo non abbia dati da trasmettere, la frazione di banda ad esso associata è inutilizzata, mentre potrebbe essere assegnata ad altri nodi che in quel momento necessiterebbero di una banda maggiore.
  Fanno parte di questa categoria protocolli come TDMA, FDMA, CDMA.

- *dinamica*, alloca il canale in base alla richiesta dell'utente. A sua volta questa si suddivide in *centralizzata* e *distribuita*, nel primo caso esiste un'unità centrale che ha il compito di stabilire a chi deve essere assegnato il canale, nel secondo caso ogni nodo deve decidere in autonomia se trasmettere o meno le proprie informazioni, il tutto può essere effettuato in maniera casuale, cioé appena l'utente ha dati da trasmettere richiede l'utilizzo del canale (ALOHA), oppure utilizza una tecnica di sensing della portante prima di trasmettere l'informazione (CSMA).

2.1 Protocolli ad allocazione statica

2.1.1 FDMA

La tecnica FDMA suddivide l'intera banda di frequenze in N sottobande ognuna delle quali è assegnata ad un singolo utente (nodo). Poiché ciascun utente utilizza una propria banda di frequenza, la trasmissione non sarà sottoposta ad interferenze da parte di nodi adiacenti.
Se il numero di utenti è piccolo e ognuno di essi genera un traffico elevato, allora lo schema FDMA raggiunge risultati buoni. Nel caso di reti UWAN, l'utilizzo dello schema FDMA non è consigliato, dato che i nodi solitamente generano un traffico tale da non occupare l'intera banda assegnata ad ognuno di essi con un'efficienza prossima al 100%.

Le scarse prestazioni del protocollo FDMA possono essere mostrate mediante un semplice esercizio di teoria delle code, infatti considerando: T il ritardo medio del

canale, C bps la capacità del canale, λ la frequenza degli arrivi e ipotizzando inoltre che ogni pacchetto abbia una lunghezza ricavata da una funzione di densità di probabilità esponenziale con media $(1/\mu)$bit/frame generati dal nodo, dalla teoria delle code si ha che per tempi di servizio e arrivo modellabili con la statistica di Poisson, il ritardo medio del canale vale:

$$T = \frac{1}{\mu C - \lambda} \tag{2.1}$$

con μC frame/sec la frequenza di servizio.
Assumendo ora il traffico complessivo generato dai nodi della rete costante, dividendo la banda di frequenza in N sottocanali indipendenti, ognuno di capacità C/N bps, si ha che la frequenza degli arrivi diventa λ/N e quindi il ritardo medio è pari a:

$$T = \frac{N}{\mu C - \lambda} \tag{2.2}$$

cioé nel caso di utilizzo di FDMA il ritardo medio è N volte peggiore del ritardo che si otterrebbe se tutti i frame fossero ordinati ed inseriti in un'unica coda centrale.

Tutti questi problemi e le non buone prestazioni fanno si che un tale protocollo sia poco utilizzabile per reti acustiche sottomarine.

2.1.2 TDMA

Il protocollo TDMA, anziché suddividere la banda a disposizione in sottobande come accade nel caso FDMA, segmenta l'intervallo temporale in slot tutti della medesima durata. Ognuno di questi slot è assegnato ad un nodo, il quale a sua volta avrà l'accesso esclusivo al canale per l'intera durata dell'intervallo.

Un tale protocollo non può essere applicato in reti acustiche sottomarine per due motivi principali: perché richiede una esatta sincronizzazione tra tutti i nodi della rete, cosa difficilmente attuabile in reti UWAN, e perché essendo il ritardo di propagazione alto in reti subacque rispetto al tempo di trasmissione delle informazioni, non é in grado di raggiungere la massima efficienza di canale.

Un adattamento del protocollo TDMA per reti acustiche sottomarine è presentato in [8]. Nell'articolo gli autori hanno osservato che per applicare l'originale TDMA in reti UWAN, la lunghezza dello slot temporale dovrebbe essere almeno

pari alla somma tra il tempo di trasmissione T_{tx} del pacchetto informativo e il massimo ritardo di propagazione τ_{max} presente nella rete, solamente in tal caso le collisioni potrebbero essere evitate.
Il problema però è il ritardo di propagazione che, in una rete UWAN, è molto elevato e quindi non consente di arrivare a risultati ottimali alla pari di quelli ottenuti per reti di sensori terrestri.
L'innovazione quindi introdotta in [8] prevede, anziché di suddividere l'intervallo temporale in slot da assegnare ad ogni nodo, di introdurre il concetto di *defer time*, $T_{defer}(i)$, cioé il tempo trascorso da quando il nodo i-esimo ha ricevuto la fine del super frame a quando il nodo aveva concluso la trasmissione del suo frame.

Il nodo sink in prima battuta calcola la distanza da tutti i suoi nodi vicini e successivamente effettua un ordinamento ascendente di questi sulla base del valore calcolato. L'indice i in $T_{defer}(i)$ indica l'i-esimo nodo che si trova nella posizione i di tutta la sequenza di nodi che il sink ha rilevato essere suoi vicini. Definendo inoltre: T_{tx} il tempo di trasmissione del pacchetto, L_i la distanza tra il nodo i-esimo e il sink e c la velocità di propagazione del segnale, si ha che il nodo sink calcola il *defer time* da ogni suo vicino come:

$$T_{defer}(i) = (i-1)T_{tx} - 2\left(\frac{L_i - L_1}{c}\right) \tag{2.3}$$

Se ogni nodo della rete utilizza la relazione (2.3), allora il nodo sink sarà in grado di ricevere tutti pacchetti senza alcuna collisione. Un'approfondita dimostrazione di tale tesi è presente in [8].

Un'aggiunta al protocollo appena descritto è l'inserimento di un intervallo di guardia dopo ogni trasmissione di pacchetto effettuata, il tutto per diminuire ulteriormente la probabilità di collisione.
Con questa feature la relazione 2.3 dovrà essere modificata nel seguente modo:

$$T_{defer}(i) = (i-1)(T_{tx} + T_{guard}) - 2\left(\frac{L_i - L_1}{c}\right) \tag{2.4}$$

con T_{guard} che indica il valore di intervallo di guardia. Il valore T_{guard} dovrà essere tale da consentire l'ottimizzazione del throughput di sistema, infatti più elevato è T_{guard} e minore è la probabilità che avvenga una collisione, ma allo stesso tempo

aumenta il ritardo di consegna end-to-end del pacchetto, diminuendo di fatto il throughput. Quindi ci deve essere un trade-off tra il valore di T_{guard} scelto e la percentuale di collisioni ammesse dal sistema.
La relazione che consente di ricavare il valore ottimale del tempo di guardia esula dallo scopo dell'elaborato, pertanto per maggiori dettagli e per maggiori spiegazioni sulla procedura si rimanda a [8].

L'attuazione delle modifiche presentate consentono di ottenere delle prestazioni soddisfacenti del protocollo TDMA anche in ambito sottomarino, infatti confrontandolo con PCAP[11] e Aloha-CA (2.2.1), ulteriori due protocolli specifici per reti UWAN, si ha che all'aumentare del traffico offerto dalla rete il throughput per TDMA è decisamente migliore rispetto ai due protocolli citati, com'é visibile anche dal grafico (2.1).

Le simulazioni sono state ottenute considerando una rete acustica sottomarina a singolo hop in cui la distanza tra i nodi e il sink è un valore aleatorio compreso tra i 1000 m e 2000 m, mentre il numero di nodi è stato fatto aumentare da un minimo di 2 fino ad un massimo di 15. Inoltre si è considerata la versione del protocollo TDMA con l'aggiunta dell'intervallo di guardia T_{guard}. Questo perché

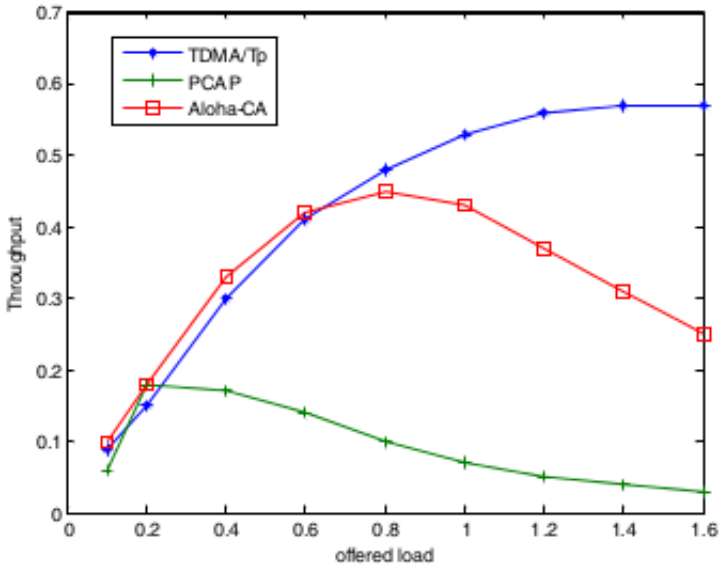


Figura 2.1: Confronto del throughput tra i protocolli TDMA, PCAP, ALOHA-CA.

da un primo confronto tra: il protocollo TDMA originale, TDMA con il solo defer time e TDMA con defer time più intervallo di guardia, si era giunti alla conclusione che la seconda versione introduce una presenza elevata di collisioni, portando alla perdita di quasi il 75% dei frame.

In conclusione il protocollo TDMA con le aggiunte citate può essere considerato come un'estensione molto valida del protocollo originale e utilizzabile soprattutto per reti UWAN, obiettivo che ci si era posti inizialmente.

2.1.3 CDMA

La tecnica CDMA è venuta in soccorso alle due tecniche appena sopra descritte, FDMA e TDMA, infatti essa consente ai nodi di trasmettere con la medesima frequenza portante e simultaneamente.
L'invio delle informazioni da parte di nodi distinti con la stessa frequenza, implica che al ricevitore ci sia un metodo tale per cui si possa prelevare la parte di informazione trasmessa da un nodo piuttosto che da un altro. La tecnica utilizzata va sotto il nome di *spread spectrum*. Con tale metodo viene assegnato a ciascun utente un codice binario, denominato sequenza di spreading o chip, che consente la codifica e decodifica in maniera univoca dell'informazione.

Matematicamente, il segnale trasmesso con la tecnica CDMA, indicando con $s_i(t)$ e $c_i(t)$ rispettivamente il segnale da trasmettere dal nodo i-esimo e la sequenza di spreading assegnata ad ogni singolo utente, sarà pari a:

$$s_{TOT}(t) = \sum_{i=0}^{N} s_i(t)c_i(t) \tag{2.5}$$

La sequenza di spreading ha la particolarità di avere una velocità di trasmissione molto maggiore rispetto alla bit rate utilizzata dal nodo per l'invio dei propri dati, questo fa sì che lo spettro occupato dal segnale $s_i(t)c_i(t)$ sia molto maggiore rispetto a quello occupato dal semplice segnale $s_i(t)$. Questa potrebbe sembrare un'inefficienza di sistema, ma è quello che consente a più utenti di trasmettere la propria informazione sullo stesso spettro. Inoltre il sistema risulterà tanto più robusto quanto più il rapporto tra la frequenza trasmissiva della sequenza di chip e quella del segnale originale è elevata.

A lato ricevitore il problema che nasce è quello di recuperare l'informazione del nodo i-esimo dato che sarà sovrapposta a quella del nodo j-esimo perché entrambe trasmesse con la stessa frequenza. A tal proposito si effettuerà un'operazione di de-spreading, ovverò il segnale ricevuto $s_{TOT,rx}$ verrà moltiplicato per la sequenza di chip associata all'utente i-esimo seguita poi da un filtraggio mediante filtro passa banda. Il risultato ottenuto coinciderà con il semplice segnale $s_i(t)$ trasmesso in precedenza dal nodo i-esimo. Il tutto è valido se e solo se le sequenze di chip $c_i(t)$ sono ortogonali tra esse e un semplice metodo per generarle è quello di applicare il metodo di Walsh-Hadamard.

Si osservi che la tecnica di de-spreading come sopra descritta consente di ottenere il segnale uguale a quello trasmesso solo nel caso di condizioni ideali, dovrà poi essere aggiunta la componente di rumore introdotta dal canale che andrà a minare l'ortogonalità tra i segnali creando problemi in fase di decodifica.

Il problema principale del protocollo CDMA è la vulnerabilità al *near-far problem*, problema che si verifica nel caso in cui due nodi a distanze diverse dal nodo ricevente trasmettano con la medesima potenza. Questo fatto comporta che il nodo più distante dal ricevitore sarà sopraffatto dal nodo più vicino poiché il suo segnale arriverà con una potenza decisamente minore rispetto al secondo considerandolo così come interferenza.
La soluzione è ottenuta inserendo un algoritmo di controllo della potenza, *power control*, che consenta di ridurre la potenza in uscita da ogni nodo in modo da stabilire un trasferimento affidabile del pacchetto senza creare un'eccessiva interferenza. Questa minimizzazione della potenza è essenziale soprattutto in reti sottomarine per ridurre anche il consumo energetico della batteria.

In definitiva la tecnica CDMA accoppiata alla tecnica di spread spectrum è una soluzione valida per reti SHALLOW-water poiché riduce il consumo energetico dei componenti che costituiscono la rete e al tempo stesso permette di ottenere valori ottimali di throughput.

2.2 Protocolli ad accesso casuale

2.2.1 ALOHA e SLOTTED ALOHA

Il funzionamento del protocollo ALOHA è molto semplice: quando un nodo ha dati (pacchetti) da trasmettere, incurante del fatto che il canale sia già occupato o meno da altri nodi, effettua la trasmissione. Al termine della trasmissione il nodo si colloca in attesa di un pacchetto di acknowledgement, il quale indicherà se la trasmissione è andata a buon fine o meno. Nel caso di una risposta negativa il nodo ritenta la trasmissione dopo un tempo casuale scelto in un intervallo temporale stabilito in fase di progetto. La casualità nella scelta del tempo di ritrasmissione è obbligatoria altrimenti i nodi che in precedenza hanno colliso lo continueranno a fare all'infinito.

Un tale metodo di accesso al mezzo introduce un'elevata percentuale di collisioni con conseguenti ritrasmissioni dei pacchetti. Il tempo medio affinché il pacchetto sia correttamente trasmesso è quindi maggiore rispetto al tempo minimo richiesto per la trasmissione del singolo pacchetto. A tal proposito è possibile definire throughput del sistema il rapporto tra il tempo necessario per la trasmissione di un pacchetto dati, senza considerarne eventuali ritrasmissioni, e il tempo medio totale affinché questo venga correttamente recapitato a destinazione.

Per il protocollo ALOHA il massimo throughput raggiungibile è pari al 18%, valore ricavabile osservando che la probabilità di collisione è pari a:

$$P_c(k) = \frac{G^k e^{-G}}{k!} \tag{2.6}$$

dove si è supposto che i nodi generino pacchetti secondo un processo di Poisson di media G=$1/\lambda$ pck/s, con λ tasso di generazione dei pacchetti di un singolo nodo. Dalla relazione (2.6) si ha che la probabilità di successo per un pacchetto è ottenuta per k=0 e vale P_{succ}=e^{-G}.
Facendo riferimento alla figura (2.2), si può osservare come esista un intervallo temporale in cui nessun nodo debba immettere le proprie informazioni nel canale affinché possano essere evitate le collisioni, un tale intervallo è definito periodo di vulnerabilità. Per il protocollo ALOHA la lunghezza di tale periodo è pari a due tempi di frame, $2t$, e, da quanto detto sopra, il numero medio di frame generati in

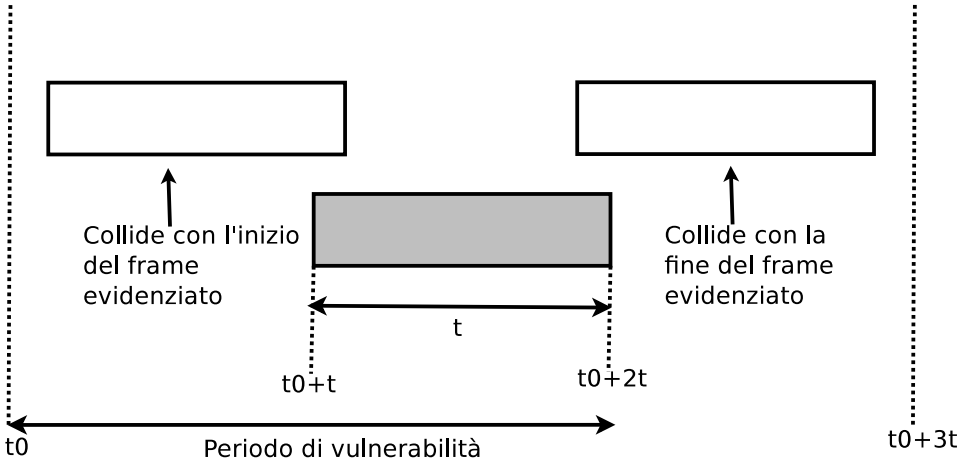


Figura 2.2: Periodo di vulnerabilità di un pacchetto nel caso di utilizzo del protocollo ALOHA.

questo lasso temporale dal nodo equivale a 2G.
La probabilità quindi che non vengano immessi nel canale frame durante il periodo di vulnerabilità, cioé la probabilità di non collisione, vale: $P_0 = e^{-2G}$, da cui è ottenibile poi il throughput di sistema:

$$S = Ge^{-2G} \tag{2.7}$$

Il valore massimo di throughput lo si ottiene per un tasso di generazione medio di G=0.5 ed equivale ad un utilizzo massimo del canale pari al 18% come inizialmente detto.

A seguito di questi risultati, le prestazioni offerte dal protocollo ALOHA non sono soddisfacenti, un aumento di queste è ottenibile semplicemente slottizzando l'asse temporale e consentendo ad un nodo di immettere le proprie informazioni nel canale solo all'inizio di un nuovo slot, solitamente quello successivo alla generazione del pacchetto. Inserendo questa variazione si ottiene il protocollo denominato SLOTTED ALOHA.

Lo SLOTTED ALOHA consente di ridurre l'intervallo di vulnerabilità da $2t$, valido per l'ALOHA, a t, e di conseguenza il throughput vale:

$$S = Ge^{-G} \tag{2.8}$$

Massimizzando la funzione (2.8) si ha che, slottizzando l'asse temporale, si raddoppia il throughput di sistema, portando l'utilizzo del canale da un valore pari al 18% valido per l'ALOHA ad un 36% qual é quello per lo SLOTTED ALOHA.
La figura (2.3) mostra l'efficienza di canale per i due protocolli ALOHA e SLOTTED ALOHA al variare della capacità di carico della rete.

I protocolli ALOHA e SLOTTED ALOHA potrebbero essere applicati in reti UWAN, ma il loro basso throughput offerto ha portato all'introduzione di ottimizzazioni che consentono di aumentare le prestazioni. Inoltre per lo SLOTTED ALOHA si richiede una sincronizzazione tra i nodi, fattore complesso da ottenere per tale categoria di reti.

Si è cercato quindi di adattare questi protocolli alle reti UWAN, ottenendo anche risultati positivi e alcuni esempi sono dati da *ALOHA with HALF DUPLEX (ALOHA-HD)* oppure *ALOHA with CARRIER SENSING (ALOHA-CS).*

Un adattamento molto interessante è fornito dal protocollo *ALOHA with COLLISION AVOIDANCE (ALOHA-CA)* presentato in [9]. Questo protocollo sfrutta la presenza di un elevato ritardo di propagazione che è introdotto nelle reti UWAN, facendo in modo che un nodo riesca a ricavare delle informazioni utilizzabili poi per la trasmissione dei dati ascoltando comunicazioni di terzi, quali: informazioni come l'identificatore del nodo trasmettitore (id_{tx}) e l'identificatore del nodo a cui è destinato il pacchetto (id_{rx}).
L'applicabilità di ALOHA-CA richiede però la conoscenza dei ritardi di propagazione

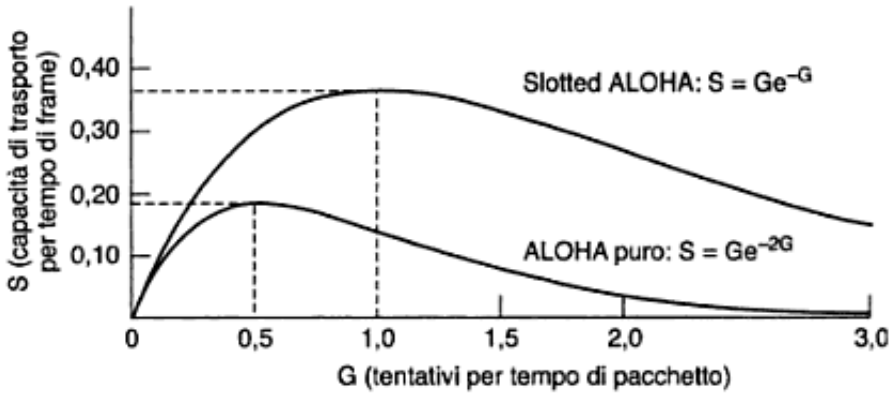


Figura 2.3: Throughput per i protocolli di livello MAC ALOHA e SLOTTED ALOHA

da un nodo a qualsiasi altro nodo della rete, cosa semplice da ricavare nel caso di reti statiche, basta una fase pre-iniziale al protocollo in cui si calcolano tali valori, mentre non banale nel caso di reti dinamiche, in questo caso si necessita di un continuo aggiornamento dei ritardi.
Dalla conoscenza del ritardo di propagazione e di id_{tx} e id_{rx} delle estremità comunicanti, un nodo potrà stabilire quando iniziare la propria trasmissione, cioé quando potrà immettere le informazioni nel canale senza avere collisioni. Si osservi che con tale metodo nessun tipo di sincronizzazione è richiesto tra i nodi, tutte le informazioni sono contenute in tabelle memorizzate all'interno di ognuno di essi.
In sostanza se un nodo ha la necessità di occupare il canale, prima di immettervi i pacchetti dati controlla le proprie tabelle interne verificando che il canale non sia già occupato, se risulta esserlo allora rimanda la propria comunicazione di un tempo aleatorio, scelto all'interno di un range prestabilito.

Dalle simulazioni effettuate e discusse in dettaglio in [9], si ha che ALOHA-CA presenta un'efficienza di canale sempre superiore al protocollo ALOHA-CS qualsiasi sia la dimensione dei pacchetti, valori quindi superiori anche ai protocolli originali ALOHA e SLOTTED ALOHA presentati in precedenza.

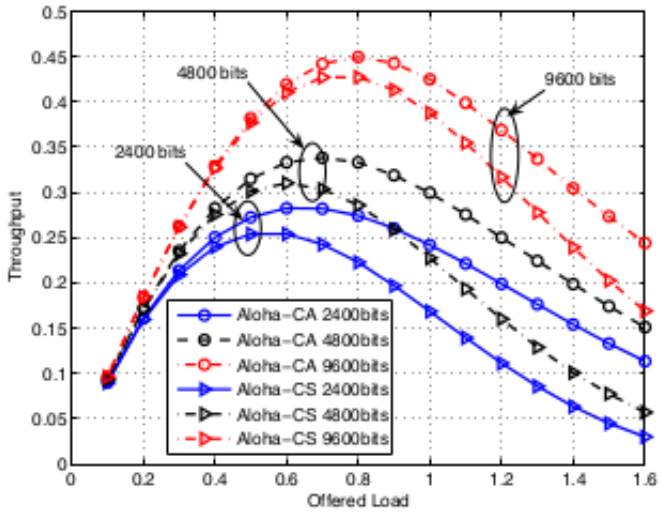


Figura 2.4: Efficienza di canale al variare del traffico offerto dalla rete per i protocolli ALOHA-CS e ALOHA-CA. Esperimenti effettuati per diverse dimensioni dei pacchetti dati.

La figura (2.4) mette in risalto quanto detto. Tutte le simulazioni sono state eseguite considerando una rete di 4 nodi predisposti in maniera aleatoria su un'area di 3000 m^2 e fissati nella posizione assegnatagli. Ogni nodo è dotato di un modem acustico che trasmette dati ad una velocità di 2400 bps e il tasso di generazione dei pacchetti dati è regolato secondo la distribuzione di Poisson.

Il protocollo ALOHA-CA oltre ad offrire un'efficienza di canale maggiore, riduce la percentuale di collisioni di circa il 6-9% rispetto al protocollo ALOHA-CS comportando anche una marcata riduzione dell'energia consumata dai nodi, fattore da tenere molto in considerazione per protocolli di livello MAC adibiti a reti acustiche sottomarine.

2.2.2 CSMA

Nei protocolli ALOHA e SLOTTED ALOHA gli utenti non tengono conto dello stato del canale prima di immettervi le proprie informazioni, conseguenza di ciò si ha un'elevata probabilità di collisione.

Un metodo per diminuire la quantità delle collisioni è quello descritto dai protocolli che hanno alla base CSMA, metodo tra le altre cose già introdotto anche in una versione estesa del protocollo ALOHA, ovvero ALOHA-CS. In questo caso un nodo prima di iniziare la trasmissione controlla la portante verificando così se il canale è libero o occupato, nel primo caso verrà immesso il frame nel mezzo trasmissivo, nel secondo caso la comunicazione sarà rinviata dopo un tempo aleatorio di backoff.

Il protocollo CSMA inizialmente come per la maggior parte dei protocolli è stato progettato per reti radio wireless, quindi una versione per reti UWAN non é così efficiente come da aspettativa.

Uno dei maggiori problemi riscontrati in CSMA è la vulnerabilità al problema del terminale nascosto ed esposto. Tale problema deriva principalmente dal fatto che il ritardo di propagazione $\tau_{MAX} \ll T_D$ con T_D il tempo di trasmissione del pacchetto.
Considerando solamente i primi tre nodi ($A-B-C$) di figura (2.5), in cui solamente B è entro il raggio di copertura di entrambi i nodi A e C, il problema del terminale nascosto si verifica quando il terminale A, dopo aver ascoltato la portante ed

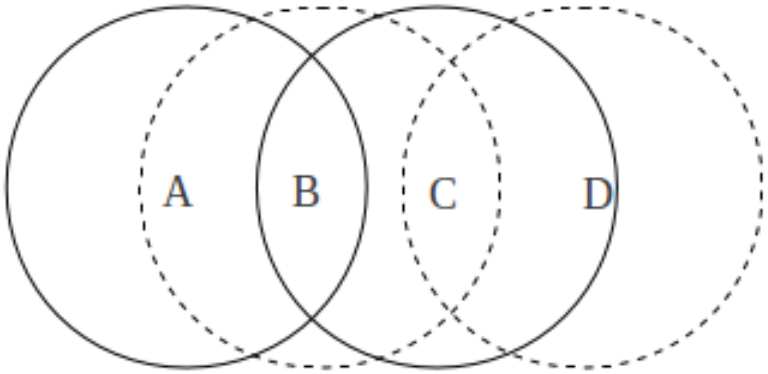


Figura 2.5: Schema per descrivere i problemi di terminale nascosto ed esposto all'interno di una rete radio-wireless o UWAN.

aver asserito che il canale è libero, inizia la propria trasmissione verso B, ma allo stesso momento anche C, dopo aver eseguito la stessa procedura di A e dopo aver anch'esso trovato il canale libero, inizia la comunicazione con B. I nodi A e C non essendo l'uno entro il raggio di copertura dell'altro non riescono a comunicare tra di essi e quindi non sono in grado di capire quando uno o l'altro hanno occupato il canale. Il risultato è una collisione dei frame che avverrà al nodo B.

Il problema del terminale esposto, che si verifica solamente nel caso di utilizzo del protocollo CSMA-CA, è invece l'inverso rispetto a quello del terminale nascosto, ovvero un nodo è inibito a comunicare con un qualsiasi altro nodo a causa di una trasmissione che non lo riguarda. Considerando tutti e quattro i nodi di figura (2.5), assumendo che il terminale A stia comunicando con B, allora C sarà inibito dalla comunicazione con B, ma al tempo stesso gli dovrebbe essere consentito la comunicazione con D, poiché un'eventuale comunicazione tra C e D non andrebbe ad interferire con quella tra A e B, fatto che però non avviene. La conseguenza di tale problema è la diminuzione del throughput di sistema.

I problemi di terminale nascosto ed esposto bloccano l'applicabilità del protocollo CSMA sia in reti radio terrestri che UWAN, anche se alcune soluzioni sono state trovate, come ad esempio l'aggiunta di un intervallo di guardia tra due trasmissioni successive proporzionale al massimo tempo di propagazione della rete. Quest'ultimo fatto rende meno gravosi i due problemi in reti radio, ma ne riduce ulteriormente

l'utilizzo in reti UWAN.

2.2.3 MACA e MACA-U

Il protocollo MACA è nato con l'intento di eliminare il problema del terminale nascosto introdotto nel protocollo CSMA e descritto nel sottoparagrafo 2.2.2.
A tal proposito sono stati introdotti due nuovi pacchetti di segnalazione RTS e CTS che hanno il compito di informare tutti i nodi vicini al trasmettitore e ricevitore della presenza di una comunicazione attiva e quindi inibire questi dalla volontà di instaurare una connessione che comporterebbe interferenza.

In sostanza, con il protocollo MACA non è più eseguito l'ascolto della portante prima dell'immissione dei frame nel canale, ma la procedura di occupazione è la seguente: il nodo mittente inizialmente invia un pacchetto RTS al nodo destinatario, il quale una volta ricevuto tale frame, se non è occupato in una qualsiasi altra comunicazione, risponde con un pacchetto CTS. Solamente alla ricezione del CTS, la sorgente inizierà la trasmissione dei dati verso il destinatario essendo sicura che la sua comunicazione non verrà disturbata da nessun altro nodo adiacente. In questo modo non si crea interferenza perché tutti i nodi che sono entro il raggio di copertura del nodo destinatario hanno ricevuto anch'essi il CTS, interpretandolo come segno di una comunicazione attiva e rimandando di fatto la propria occupazione del canale. Sia RTS che CTS contengono al loro interno un campo detto NAV che identifica la durata temporale in cui il canale è mantenuto occupato. Questo è un parametro molto importante, dato che consente a tutti i nodi vicini di sapere dopo quanto tempo eventualmente possono tentare l'accesso al canale.

Il protocollo MACA, realizzato in principio per reti di sensori wireless, è diventato la base di molti protocolli per reti acustiche sottomarine in quanto riduce di molto le collisioni, alcuni esempi sono dati da: *MACA-MN (Multiple Access Collision Avoidance with packet train for Multiple Neighbors)* descritto in [10] oppure *MACA-U (Multiple Access Collision Avoidance protocol for Underwater)* presentato in [12]. L'introduzione però dei pacchetti di segnalazione RTS-CTS crea un degrado delle prestazioni sia in termini di throughput (poiché RTS e CTS non trasportano informazione utile), sia in termini di ritardo end-to-end in quanto

prima della consegna vera e propria dei dati deve essere completata una fase di scambio appunto di RTS e CTS.

La nostra attenzione si concentrerà sul protocollo MACA-U. Questo è un adattamento per reti UWAN del protocollo originale MACA, che in sostanza gli autori di [12] hanno riprogettato modificando gli stati in cui un nodo può trovarsi durante la sua fase di vita, le transizioni tra di essi, le strategie di trasmissione dei pacchetti dati ed infine l'algoritmo di backoff.

Partendo dagli stati, un nodo su cui è implementato il protocollo MACA-U può essere in uno dei seguenti cinque stati: *IDLE-CONTEND-WFCTS-WFDATA-QUIET*.
Dallo stato di IDLE, in cui inizialmente il nodo si trova, passerà nello stato di CONTEND quando ha un pacchetto dati pronto da trasmettere. Una volta entratovi e dopo un tempo aleatorio di attesa, il nodo trasmetterà un pacchetto RTS verso il destinatario e successivamente si porrà nello stato di WFCTS, all'interno del quale attenderà l'arrivo del primo pacchetto CTS. L'attesa durerà al massimo $(2\tau_{MAX}+T_{CTS})$, con τ_{MAX} tempo massimo di propagazione e T_{CTS} tempo necessario per la trasmissione del pacchetto CTS.
Durante la permanenza in WFCTS il nodo ignorerà qualsiasi RTS sia esso il destinatario o meno, mentre alla ricezione di un CTS che non sia ad egli rivolto, abbandonerà immediatamente lo stato WFCTS per entrare nello stato QUIET. Questa strategia consente di aumentare il throughput del sistema e diminuire le collisioni tra i pacchetti. Solamente alla ricezione del CTS proveniente dal nodo a cui aveva in precedenza trasmesso l'RTS, il nodo uscirà dallo stato WFCTS per iniziare la trasmissione vera e propria dei dati.

Dalla parte del ricevitore, quando questo riceverà il pacchetto di RTS risponderà immediatamente con un pacchetto CTS se in quel momento è libero da qualsiasi comunicazione, ed entrerà poi nello stato WFDATA dove vi rimarrà per un tempo pari a $(2\tau_{MAX} + T_{DATA})$, con T_{DATA} tempo di trasmissione di un pacchetto dati. Dall'istante di ingresso in WFDATA il nodo ricevitore ignorerà qualsiasi RTS o CTS siano destinati o meno a lui.
Quando invece il nodo sorgente o ricevitore si trova nello stato di QUIET, vi rimane finché continuerà a ricevere pacchetti RTS o CTS che non siano a lui destinati, ma che sono segno della presenza di una comunicazione già in atto tra altri nodi.

La seconda modifica applicata al protocollo originale MACA riguarda la strategia di trasmissione. Infatti, un nodo che implementa MACA-U assegna ai pacchetti da trasmettere delle diverse priorità: in particolare, due sono i livelli utilizzati. Ogni nodo al suo interno gestisce due code, nella prima sono inseriti i pacchetti che il nodo stesso genera, mentre nella seconda i pacchetti che il nodo riceve e che deve solo inoltrare (reti multi-hop). La priorità maggiore è assegnata alla seconda categoria, così facendo sono ridotte le volte in cui scade il timeout scelto dal nodo quando era entrato nello stato WFCTS, diminuendo di conseguenza le circostanze in cui l'algoritmo di acquisizione del canale dovrà essere fatto partire.

La terza innovazione introdotta in MACA-U è l'algoritmo di backoff utilizzato a seguito di una collisione da parte del nodo che l'ha subita. Quando il timeout di attesa del CTS scade, il nodo che in principio aveva trasmesso il pacchetto RTS ne ritenta la trasmissione, raddoppiando però il tempo di attesa del CTS rispetto a quello precedente.
La regola è data dalla seguente relazione:

$$B_{wait} = \begin{cases} min(2B_{wait}, B_{max}), & \text{se c'é collisione} \\ B_{min}, & \text{se non c'é collisione.} \end{cases} \tag{2.9}$$

da cui si ricava il tempo di backoff che sarà scelto in maniera uniforme:

$$T_{bck} = uniform(0, B_{wait})(T_{RTS} + \tau_{MAX}) \tag{2.10}$$

Si osservi che se non viene tenuto in considerazione l'elevato ritardo di propagazione che è introdotto dalle reti UWAN, si incapperebbe in un aumento smisurato delle collisioni con una conseguente diminuzione del throughput.

Per verificare l'efficienza del protocollo MACA-U le simulazioni sono state realizzate su una rete costituita da 36 nodi distanziati uno dall'altro di 1300 m circa. Inoltre la rete multihop dà la possibilità ad ogni nodo di inoltrare il proprio pacchetto ad 8 nodi ad esso adiacenti e quindi a 16 nodi considerando due hop. La velocità di trasmissione è 2400 bps mentre la velocità di propagazione è sempre pari a quella del suono in ambiente subacqueo (1500 m/s).
I nodi generano traffico secondo una distribuzione di Poisson e il canale è assunto

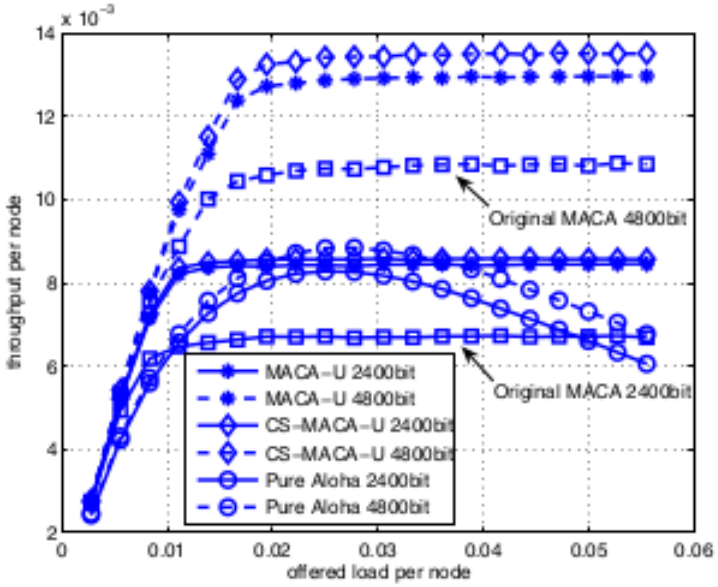


Figura 2.6: Confronto del throughput del protocollo MACA-U con i protocolli PURE-ALOHA, MACA originale, MACA-CS.

essere ideale, cioé non introduce errori sui pacchetti. Sotto questa condizione le ritrasmissioni dei pacchetti saranno dovute solo a collisioni e non ad errori presenti all'interno del pacchetto.
Tutte le simulazioni sono inoltre effettuate con tre distinte dimensioni dei pacchetti dati: (1200-2400-4800) bits, mentre i pacchetti di segnalazione RTS-CTS hanno dimensioni pari a 100 bits.

La figura (2.6) confronta il throughput del protocollo MACA-U con altri protocolli di livello MAC quali PURE-ALOHA, CSMA-CS e MACA originale implementato per reti radio wireless.
Come si può osservare, se confrontato con il PURE-ALOHA, si ha che per quest'ultimo il throughput decade all'aumentare del traffico offerto, mentre MACA-U lo mantiene costante, fatto dovuto alle buone strategie impiegate dal protocollo per cercare di risolvere le collisioni (algoritmo di backoff).

Confrontando invece MACA-U con l'originale MACA, si può vedere come il primo abbia performance sempre superiori rispetto al secondo, ottenendo anche miglioramenti di circa il 20%, il che dimostra come il semplice protocollo MACA

senza l'introduzione di alcune varianti non possa essere utilizzato per reti acustiche sottomarine.

2.2.4 DACAP

Un ulteriore protocollo di livello MAC che per instaurare una comunicazione sicura e priva di collisioni tra i nodi si basa su pacchetti di segnalazione, quali RTS e CTS visti nel sottoparagrafo precedente 2.2.3, è il protocollo DACAP descritto nel dettaglio in [13].

Ipotizzando che un nodo i debba trasmettere i propri dati ad un nodo j della rete, con $i \neq j$, allora la catena di messaggi scambiati dal protocollo DACAP affinché la trasmissione dati possa avvenire è la stessa che si è vista per il protocollo MACA-U, cioé inizialmente il nodo i trasmetterà un pacchetto RTS in unicast a j, il quale alla ricezione di questo invierà come risposta un pacchetto CTS che darà il via libera al nodo i per la trasmissione dei frame dati.

DACAP si differenzia dal protocollo MACA-U perché introduce un nuovo pacchetto definito $WARNING$. La causa dell'invio del WARNING è facilmente spiegabile mediante un esempio.
Si assuma sempre il caso di comunicazione tra i nodi i e j citati in precedenza e si ipotizzi che il nodo j abbia trasmesso il pacchetto CTS ad i. Un intervallo temporale, abbastanza piccolo, intercorrerà dal momento in cui è stato inviato il CTS all'istante in cui il nodo i completerà la ricezione di tale frame. Se in questo lasso di tempo il nodo j, che nel frattempo è rimasto attento ai pacchetti che viaggiano nel canale, si accorge della presenza di altre comunicazioni attive, allora invierà il messaggio di WARNING verso i in modo tale da bloccarlo dalla trasmissione dei dati. Questo perché una eventuale immissione dei dati nel canale da parte di i creerebbe delle collisioni con la trasmissione individuata da j.
Si osservi che essendo molto breve l'intervallo temporale tra CTS-DATI, il pacchetto di WARNING dovrà avere dimensioni ridotte, di pochi bit, in modo da velocizzarne la trasmissione e tale per cui arrivi prima che il nodo i inizi la spedizione dei pacchetti dati.
Per agevolare la ricezione del WARNING, il nodo i ritarderà l'invio dei frame dati dopo aver ricevuto il pacchetto CTS, in questo modo la probabilità che il

messaggio di WARNING arrivi a destinazione in tempo aumenterà e la percentuale di collisioni potrebbe sensibilmente diminuire. Solitamente il periodo di attesa dipende dalla distanza tra i nodi che il trasmettitore i ha entro il proprio raggio di copertura, valore calcolato dividendo il tempo trascorso da quando viene trasmesso il pacchetto RTS e ricevuto il messaggio CTS con la velocità di propagazione del suono nell'acqua c=1500 m/s. Scegliendo quindi come tempo di attesa il massimo ritardo di propagazione, con probabilità elevata si é in grado di ricevere il messaggio di WARNING prima di iniziare la trasmissione dei frame dati. La probabilità di non collisione non é pari al 100%, infatti potrebbe accadere che il pacchetto di WARNING venga perso, caso che si verifica quando la sua ricezione avviene mentre il nodo ha già iniziato la trasmissione dei pacchetti data. In tal caso essendo il canale half-duplex non può avvenire la contemporanea ricezione e trasmissione da parte del nodo, la precedenza è data alla trasmissione, dato che è già iniziata, e quindi il WARNING è considerato come perso.

In [13] sono stati presentati dagli autori due versioni del protocollo DACAP: con e senza l'utilizzo di pacchetti di acknowledgement (ACK).
Si ricorda che gli ACK sono pacchetti particolari e di piccole dimensioni che hanno il solo compito di indicare se un pacchetto dati è ricevuto correttamente o meno. Sulla base della risposta contenuta nell'ACK il nodo sorgente può capire se il pacchetto è arrivato a destinazione positivamente, oppure in caso contrario, se è implementato uno schema ARQ, tentare una ritrasmissione del pacchetto errato.

Nel caso di non utilizzo di ACK, l'invio del WARNING avviene solamente quando il ricevitore sente nel canale la presenza di RTS e/o CTS nell'intervallo temporale ($2T$-t_{min}) s dopo l'invio del suo CTS.
Allo stesso tempo, il trasmettitore alla ricezione del WARNING rinvierà la propria spedizione di dati. In un altro caso la trasmissione sarà posticipata dalla sorgente, ovvero se entro l'intervallo temporale t_{min} dopo l'avvenuta trasmissione del pacchetto RTS riceve un CTS non ad essa indirizzato. Con t_{min} è indicato il tempo minimo necessario per scambiare i messaggi RTS-CTS da un nodo a qualsiasi altro nodo della rete.
Da quanto detto, si ha che il tempo di attesa del pacchetto WARNING da parte

del nodo che aveva in principio trasmesso il messaggio RTS è pari a:

$$T_w(d/c) = \begin{cases} t_{min} - 2d/c, & \text{se d/c<}t_1 \\ 2(d + \Delta d)/c - t_{min}, & \text{se d/c>}t_1 \end{cases} \tag{2.11}$$

dove (d+Δd) è la distanza oltre il quale il nodo non è più considerato interferente perché il messaggio arriverebbe talmente attenuato dal canale che non interferirebbe con la comunicazione, c è la velocità di propagazione del suono nell'acqua, mentre t_1 vale:

$$t_1 = \frac{t_{min} - min(\Delta d/c, t_{data}, 2T - t_{min})}{2} \tag{2.12}$$

definendo t_{data} come il tempo necessario per trasmettere il pacchetto dati.

Nella seconda versione del protocollo DACAP, ad ogni pacchetto dati correttamente ricevuto, il destinatario invia un messaggio di ACK alla sorgente. In tale situazione, il tempo di attesa prima di iniziare la trasmissione dei pacchetti dati è pari a:

$$T_w(d/c) = \begin{cases} 2(d + \Delta d)/c - t_{min}, & \text{se d/c } \in(t_1,t_2) \\ 2(d + \Delta d)/c - T_{W_{min}}, & \text{se d/c>max}(t_2,t_3) \\ t_{min} - 2d/c, & \text{altrimenti} \end{cases} \tag{2.13}$$

dove $T_{W_{min}}$ è il valore calcolato al passo precedente di T_w. Definendo Δt_{data} come la massima differenza tra la durata di trasmissione di due pacchetti dati, i valori temporali t_1, t_2 e t_3 valgono:

$$t_1 = \frac{t_{min} - min(\Delta d/c, t_{data}, 2T - t_{min})}{2} \tag{2.14a}$$

$$t_2 = \frac{t_{min} - \Delta t_{data}}{2} \tag{2.14b}$$

$$t_3 = min\left(t_1, \frac{t_{min} + T_{W_{min}} - 2\Delta d/c}{4}\right) \tag{2.14c}$$

Per verificare l'efficacia del protocollo DACAP, gli autori in [13] hanno effettuato alcuni confronti in termini di throughput rispetto ad altri protocolli già esistenti e utilizzati per reti UWAN: SLOTTED-FAMA e ALOHA-CS.
Gli scenari di simulazione sono descritti in maniera dettagliata all'interno del-

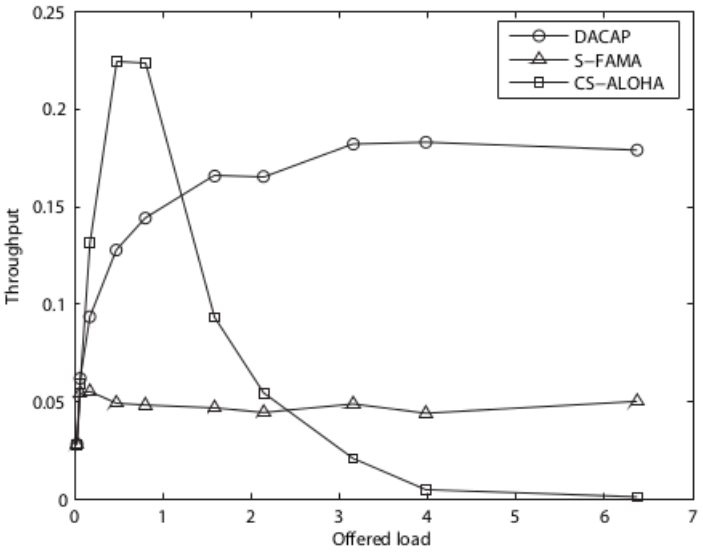


Figura 2.7: Throughput per i protocolli DACAP versione senza ACK, CS-ALOHA e S-FAMA considerando un range di trasmissione dei nodi di 7000 m e tolleranza all'interferenza di Δd=3500 m.

l'articolo [13], qui di seguito saranno solamente riportati e commentati alcuni risultati.

Considerando la versione del protocollo DACAP senza ACK ed un range di trasmissione per i nodi di 7000 m rendendo la rete completamente connessa dato che i nodi sono distribuiti su un'area di 5000 m, si ha che sia il protocollo DACAP sia S-FAMA, rispetto ad ALOHA-CS hanno una percentuale di collisioni maggiore, circa il 2% in più, ma allo stesso tempo raggiungono valori di throughput ben più elevati a parità di traffico offerto dalla rete. Il tutto è espresso anche in figura (2.7).

Considerando invece la versione con ACK si ha che, nonostante l'introduzione di tali pacchetti, il throughput per DACAP raggiunge risultati migliori rispetto S-FAMA e ALOHA-CS nel caso di link di breve lunghezza. ALOHA-CS risulta essere stabile per elevati valori di traffico offerto, ma contemporaneamente consuma maggior potenza.
Il throughput inoltre per DACAP è maggiore rispetto a quello ottenuto con il protocollo S-FAMA, perché considerando la durata dello scambio dei messaggi RTS-CTS uguale per entrambi, DACAP trae vantaggio dal fatto che la trasmissione

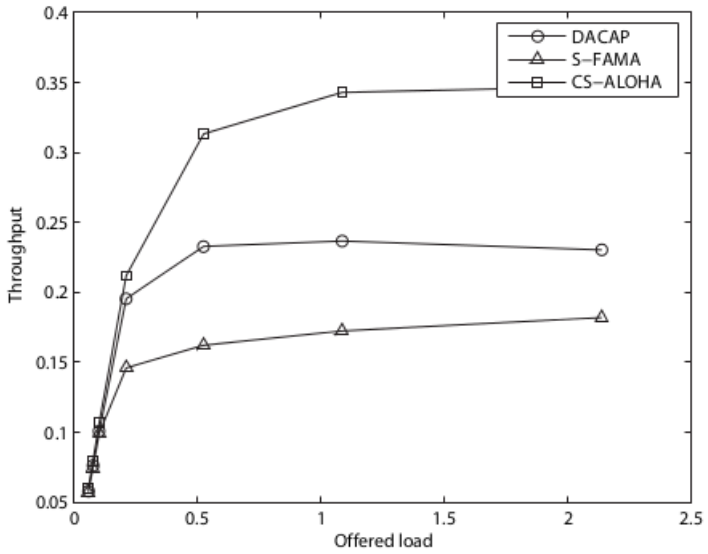


Figura 2.8: Throughput per i protocolli DACAP versione con ACK, CS-ALOHA e S-FAMA considerando un range di trasmissione dei nodi di 2500 m e tolleranza all'interferenza di Δd=0 m.

inizia appena si ha il via libera del destinatario (ricezione del CTS), senza attendere l'inizio dello slot successivo come avviene nel protocollo S-FAMA. Le considerazioni qui effettuate sono visibili dalla figura (2.8).

2.2.5 FAMA e SLOTTED-FAMA

I protocolli FAMA e S-FAMA oltre ad utilizzare i messaggi RTS e CTS nello stesso modo che sono stati presentati per i protocolli MACA, MACA-U e DACAP, recuperano un elemento che quest'ultimi hanno abbandonato, ovvero l'ascolto della portante prima di effettuare l'immissione del pacchetto dati nel canale.

I progettisti del protocollo FAMA, [14], hanno osservato che le collisioni, utilizzando un protocollo che verifichi lo stato della portante prima della trasmissione del pacchetto, con probabilità molto elevata non si presentano se le seguenti due condizioni sono verificate:

1. la lunghezza del pacchetto RTS è maggiore del ritardo massimo di propagazione;

2. il tempo di trasmissione del pacchetto CTS (T_{CTS}) é maggiore della quantità ($T_{CTS} + 2\tau_{MAX} + RTT$) con τ_{MAX} il massimo ritardo di propagazione e RTT il round trip time.

Queste condizioni, che sono alla base del protocollo FAMA, consentono di eliminare le collisioni tra i frame, ma sono troppo stringenti per una sua applicazione in reti UWAN. Infatti, l'elevato ritardo di propagazione presente in questa categoria di reti, costringerebbe ad un'eccessiva lunghezza dei pacchetti RTS-CTS e di conseguenza ottimi risultati non sarebbero ottenibili, dato che verrebbe aumentata: la quantità di informazione non utile immessa nella rete e il ritardo di consegna end-to end.

Di seguito è presentata in maniera riassuntiva un'evoluzione del protocollo FAMA, adattabile anche per reti acustiche sottomarine, che porta il nome di S-FAMA. Per una più dettagliata descrizione del protocollo si rimanda a [14].

In sostanza il protocollo S-FAMA suddivide l'asse temporale in slot aventi una durata tale per cui si possano evitare le collisioni tra frame di nodi distinti. Solitamente la lunghezza dello slot è pari a ($\tau_{MAX} + T_{CTS}$), con τ_{MAX} il massimo ritardo di propagazione e T_{CTS} il tempo di trasmissione dell'omonimo pacchetto. Ogni nodo è inoltre vincolato a trasmettere qualsiasi frame, sia esso RTS-CTS-DATA o ACK, solamente all'inizio dello slot temporale.
Si osservi che, con una tale slottizzazione del tempo, si introduce un certo grado di sincronizzazione tra i vari nodi sensori, passando quindi da un protocollo non sincronizzato (FAMA) ad uno sincronizzato (S-FAMA).

La sequenza di scambio di messaggi utilizzata da S-FAMA, affinché la trasmissione dei dati da parte del nodo possa avvenire, è sempre data da RTS-CTS-DATA come succede per i protocolli MACA-U, sezione (2.2.3), e DACAP, sezione (2.2.4), con l'unica eccezione che S-FAMA obbliga i nodi a trasmettere tutti i pacchetti all'inizio dello slot successivo a cui essi sono stati creati e solamente se non ci sono già comunicazioni di terzi nello slot che il nodo dovrà occupare.

Si osservi ora il comportamento del protocollo S-FAMA nel momento in cui un nodo qualsiasi riceve pacchetti RTS-CTS o DATA che non sono indirizzati ad esso. Assumendo inizialmente che il nodo si trovi nello stato *idle*, nel momento in cui rileva dal mezzo di comunicazione la presenza di una portante, entra nello stato di *Receiving* all'interno del quale inizierà la ricezione del pacchetto. Sulla base del

tipo e del contenuto del pacchetto il nodo effettua alcune operazioni. Se è ricevuto:

- un RTS, di cui però esso non è il destinatario, allora il nodo attende per due ulteriori slot temporali in modo da dare la possibilità ai nodi che devono comunicare di portare a termine la trasmissione dati. Se al termine di questa attesa altre portanti non sono state rilevate, allora il nodo ritorna in stato di *idle* da cui magari potrà successivamente iniziare una nuova trasmissione;

- un CTS che è destinato ad altri nodi, allora il nodo in questione attenderà per un intervallo temporale abbastanza lungo tale da consentire l'invio dei pacchetti dati da parte del nodo a cui è effettivamente destinato il CTS. Il tempo di attesa si conclude quando il nodo intercetta l'ACK relativo che equivale alla prova che la trasmissione dati si è conclusa;

- un pacchetto DATA, allora il nodo si mette in attesa del pacchetto ACK o NACK relativo che darà la conferma di corretta o meno ricezione. Se non è in grado, per un qualche motivo, di rilevare la presenza di uno dei due pacchetti, allora il nodo attende uno slot temporale in più con l'obiettivo di vedere se il pacchetto dati è stato ritrasmesso;

- un ACK relativo ad un pacchetto dati trasmesso da un altro nodo, allora il terminale prima di iniziare una sua eventuale trasmissione attende la fine dello slot, la quale è indice di una corretta fine della trasmissione;

- un NACK non destinato ad esso, allora il nodo attende un tempo abbastanza lungo in modo che il terzo nodo, che attualmente sta occupando il canale, ritrasmetta il pacchetto dati e riceva il relativo ACK.

Dai casi sopra descritti, il nodo prima di occupare il canale si assicura sempre che il mezzo sia libero da altre comunicazioni, in caso negativo attende per una certa quantità di slot in modo tale che la trasmissione rilevata si concluda con successo evitando così delle collisioni.

Il protocollo S-FAMA, abbinato ad una qualsiasi tecnica ARQ (Stop & wait, Go-Back-N, Selective Repeat) ripresenta uno dei problemi analizzati già nel protocollo MACA, ovvero il problema del terminale esposto. In [14] si è osservato che tale incoveniente non può essere del tutto superato, l'unica soluzione è cercare di trovare

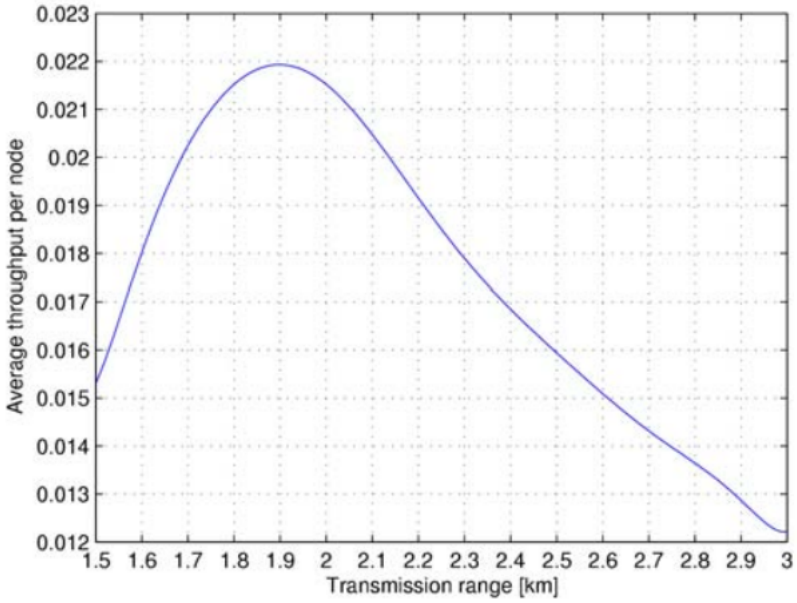


Figura 2.9: Throughput relativo ai nodi che utilizzano il protocollo S-FAMA al variare del range di trasmissione utilizzato per la trasmissione.

il giusto compromesso tra il throughput desiderato e l'utilizzo di tecniche ARQ, infatti in alcuni casi è possibile rimandare controlli sulla correttezza dei pacchetti ai livelli superiori, escludendo di fatto l'utilizzo di ARQ a livello data link.

Le performance del protocollo sono discusse dettagliatamente nell'articolo [14], di seguito sono riportati solo alcuni grafici che indicano come il protocollo S-FAMA raggiunga valori di throughput molto elevati per particolari range di trasmissione da parte dei nodi. In dettaglio il valore di throughput massimo per nodo è ottenuto con range di 1900 m, andamento raffigurato in figura (2.9).

Il grafico (2.10) invece mostra il ritardo di consegna end-to-end, ritardo che indica il tempo necessario affinché il pacchetto dati venga correttamente consegnato a destinazione da quando questo è stato generato. Dall'andamento si può vedere come bassi valori di ritardi di consegna siano raggiunti dal protocollo, soprattutto nel caso in cui il range di trasmissione utilizzato sia di circa 2000 m, cui tra le altre cose corrisponde il massimo throughput.

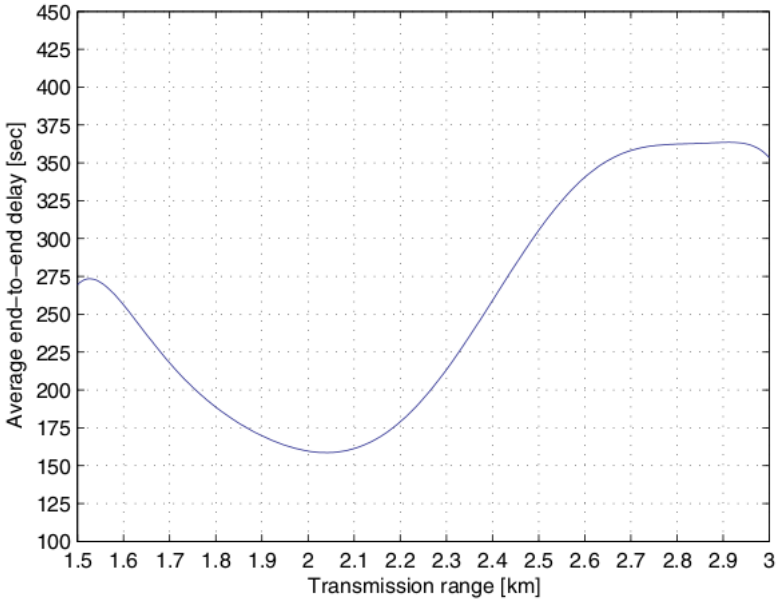


Figura 2.10: Ritardo di consegna end-to-end di un pacchetto dati al variare del range di trasmissione utilizzato dai nodi.

2.3 Protocolli di confronto con UFETCH

2.3.1 uw-UPOLLING

UW-POLLING ([15]) è un protocollo di livello MAC sviluppato appositamente per reti acustiche sottomarine che ha come scopo quello di regolamentare l'accesso al mezzo e raccogliere dati da remoto. La tecnica che sta alla base del protocollo affinché possano essere raccolte le informazioni precedentemente rilevate ed elaborate dai nodi sensori è quella di polling, da cui deriva il nome del protocollo.
UW-POLLING assume la presenza nella rete di un nodo speciale, denominato AUV, che governa l'accesso al canale da parte di tutti i nodi sensori e richiede le informazioni da essi raccolte. La particolarità del protocollo sta nel fatto che l'AUV è mobile, cioé segue una prestabilita traiettoria, tale da ricoprire periodicamente i vari punti della rete installata in mare, potendo così interrogare ciclicamente i nodi sensori. Solitamente la velocità di viaggio è compresa tra i 2 e 4 nodi; l'AUV svolge le operazioni di guida in completa autonomia senza la necessità di un pilota umano che coordini le azioni.

Grazie a questa organizzazione per la raccolta dei dati, è possibile mitigare l'interferenza tra i vari nodi sensori, infatti è l'AUV che decide a quali nodi sensori, quando e per quanto tempo dare l'accesso esclusivo del canale. Eventuali collisioni potranno avvenire solamente durante la fase di instaurazione della connessione tra AUV e nodi, fase regolamentata sempre dai pacchetti di segnalazione visti per MACA, MACA-U e DACAP: RTS-CTS, oltre ad un pacchetto aggiuntivo denominato TRIGGER che sarà descritto successivamente.
Le collisioni, durante questa fase iniziale, non saranno deleterie in termini di throughput, dato che esse non vanno ad interferire con i pacchetti DATA. Il throughput diminuisce nel momento in cui si ha collisione tra pacchetti di segnalazione, infatti la perdita di uno di questi potrebbe inibire la trasmissione di pacchetti DATA da parte di un nodo che sarebbe pronto alla trasmissione.

Il protocollo uw-POLLING in altre parole crea una rete in cui i nodi appartengono a due livelli di priorità: quello più basso costituito dai nodi sensori, mentre quello più alto composto dall'AUV.
I nodi sensori, che sono redistribuiti in maniera aleatoria tale da occupare una certa porzione di mare o oceano, hanno il solo compito di monitorare l'area circostante ad essi e quando interrogati dall'AUV inviare le informazioni raccolte e da loro stessi elaborate.
L'AUV, come detto, è un nodo mobile che eseguendo una certa traiettoria passa in rassegna l'intera rete chiedendo periodicamente a tutti i nodi sensori che la costituiscono le proprie informazioni.

La comunicazione consta nella seguente sequenza di messaggi: TRIGGER-PROBE-POLL-DATI.
All'avvio dell'AUV ad intervalli regolari questo trasmette in broadcast un pacchetto denominato TRIGGER che sarà ricevuto da tutti i nodi sensori posizionati entro il raggio di copertura dell'AUV. Alla ricezione di tale pacchetto da parte del nodo sensore, dopo un certo tempo di backoff aleatorio e scelto all'interno di un intervallo temporale prestabilito in fase di progettazione, se il sensore ha dati da inviare, trasmette un pacchetto in risposta al TRIGGER, chiamato PROBE, che è l'equivalente all'RTS trasmesso dai nodi nel caso di utilizzo di MACA, MACA-U e DACAP. Il tempo di backoff è aleatorio in modo da evitare collisioni da parte dei nodi sensori che hanno ricevuto contemporaneamente il pacchetto di TRIGGER.

In caso contrario, il nodo continua con le operazioni che stava eseguendo prima della ricezione del TRIGGER.

Dopo la trasmissione del TRIGGER, l'AUV si pone in attesa di eventuali pacchetti PROBE. Rimane in tale stato per un intervallo abbastanza lungo da ricevere eventuali PROBE da quei nodi collocati nelle estremità del suo raggio di copertura. La ricezione del PROBE è indice del fatto che il nodo ha effettivamente pacchetti dati da trasmettere.
Allo scadere dell'intervallo di attesa, l'AUV possiede una lista di nodi sensori da cui poter ricevere i pacchetti dati, quindi, secondo una regola descritta in [15], sceglierà a chi concedere l'accesso al mezzo affinché il nodo possa poi trasmettere le proprie informazioni, senza la preoccupazione di eventuali collisioni dovute ad altre comunicazioni. Il pacchetto utilizzato per far capire al nodo sensore quando iniziare la trasmissione dati è chiamato POLL.
Alla ricezione del POLL, il nodo sensore verifica se é egli stesso il destinatario di tale pacchetto, in caso affermativo inizia l'invio dei propri pacchetti dati all'AUV, contrariamente lo ignora e continua l'esecuzione delle operazioni che stava effettuando prima della ricezione, cioé raccolta di informazioni o attesa di un altro POLL.
Al termine della trasmissione dei pacchetti dati il nodo sensore ritorna nello stato di idle, dove continuerà la sua fase di monitoraggio e recupero delle informazioni dell'area ad esso circostante, mentre per quanto concerne il nodo AUV, dopo la ricezione di tutti i pacchetti ad esso si aprono due possibilità: trasmettere un nuovo POLL nel caso in cui abbia ulteriori nodi sensori da cui farsi spedire informazioni, altrimenti iniziare un nuovo ciclo inviando un ulteriore TRIGGER in broadcast.

Per testare l'efficacia del protocollo uw-POLLING, gli autori in [15] hanno effettuato test confrontandolo con il protocollo DACAP, descritto nel sottoparagrafo 2.2.4, ed una versione modificata del protocollo CSMA-ALOHA denominata CSMA-ALOHA-TRIG in cui i nodi sono abilitati alla trasmissione solo dopo aver ricevuto un pacchetto di TRIGGER dal nodo AUV.

I risultati sono mostrati nelle figure (2.11), (2.12) e (2.13). Tutti sono stati ottenuti considerando una rete costituita da 25 nodi disposti su di una griglia quadrata 5x5, ognuno distanziato dai sui vicini di 2000 m e ad una profondità di 100 m.

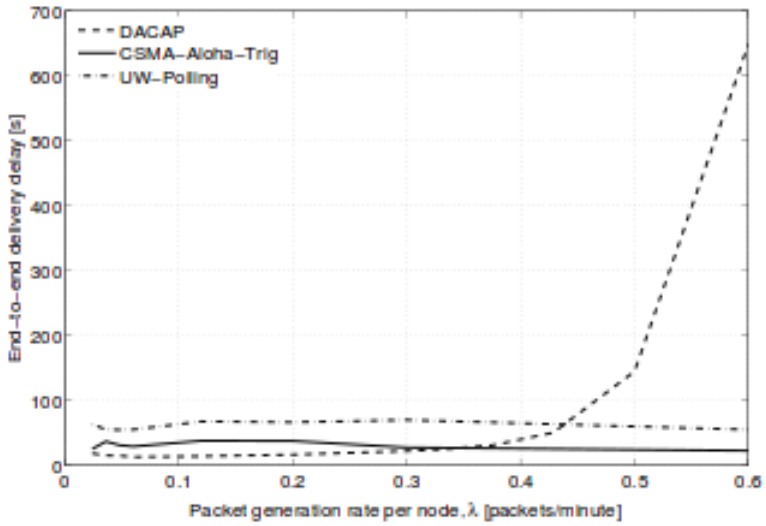


Figura 2.11: Ritardo di consegna end-to-end al variare del rate di generazione dei pacchetti per nodo.

Il nodo AUV viaggia ad una velocità di 4 nodi, ad una profondità di 60 m e segue una traiettoria prestabilita in modo da passare almeno una volta entro il raggio di copertura di tutti i nodi sensori.
La comunicazione avviene con la portante 25 kHz, la potenza di trasmissione è di 150 dB re μPa e la dimensione dei pacchetti è di 125 byte.

La figura (2.11) rappresenta al variare del tasso di generazione di ogni singolo nodo, quindi del traffico offerto dalla rete, il ritardo di consenga end-to-end. Come si può osservare il protocollo CSMA-ALOHA-TRIG presenta il più basso ritardo di consegna, questo perché implementa una semplice politica di trasmissione rispetto uw-POLLING e DACAP. Viceversa il protocollo uw-POLLING ha un ritardo più elevato dovuto al fatto che il nodo sensore debba aspettare che l'AUV sia nelle proprie vicinanze, e questo intervallo di attesa potrebbe essere elevato se l'AUV è in posizione opposta a quella del nodo. D'altro canto però si può vedere come uw-POLLING sia migliore rispetto DACAP per valori di traffico molto elevati, dove appunto quest'ultimo presenta un ritardo che aumenta in maniera esponenziale per valori di traffico generato dai nodi superiore a 0.4 pck/min.

Come per tutti i protocolli per reti UWAN, un'occhio di riguardo deve essere dato al consumo energetico di ogni nodo. A tal proposito la figura (2.12) effettua

un confronto tra i tre protocolli. Assumendo che 100 W siano utilizzati per la trasmissione, 0.8 W per la ricezione e 0.008 W nel caso in cui il nodo sia nello stato di idle, si ha che il maggior consumo energetico è dato dal protocollo DACAP, il quale presenta un incremento lineare dell'energia spesa all'aumentare del traffico generato dai singoli nodi sensori. Presentano invece all'incirca lo stesso consumo energetico i protocolli uw-POLLING e CSMA-ALOHA TRIG, leggermente superiore è quello di uw-POLLING perché l'energia è utilizzata per la trasmissione dei messaggi di segnalazione.

Un ultimo confronto è effettuato sulla base della PDR, grafico (2.13). In questo caso le prestazioni peggiori sono date dal protocollo CSMA-ALOHA-TRIG, il quale presenta un decremento della PDR lineare all'aumentare del traffico generato da ogni singolo nodo. Le prestazioni migliori in questo confronto sono offerte dal protocollo uw-POLLING, traendo vantaggio dal fatto che i pacchetti dati sono trasmessi su un canale privo di collisioni, reso tale grazie al precedente scambio di segnalazione tra nodo sensore e AUV (TRIGGER-PROBE-POLL).

In conclusione il protocollo uw-POLLING è un protocollo che offre performance buone in termini di *packet error rate*. Uno svantaggio è il tempo di consegna

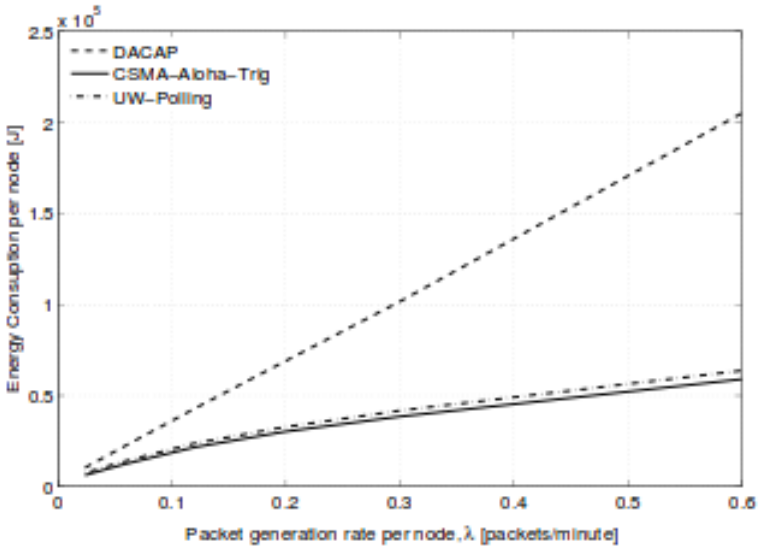


Figura 2.12: Consumo di energia del nodo al variare del tasso di generazione dei pacchetti da parte dei singoli nodi sensori.

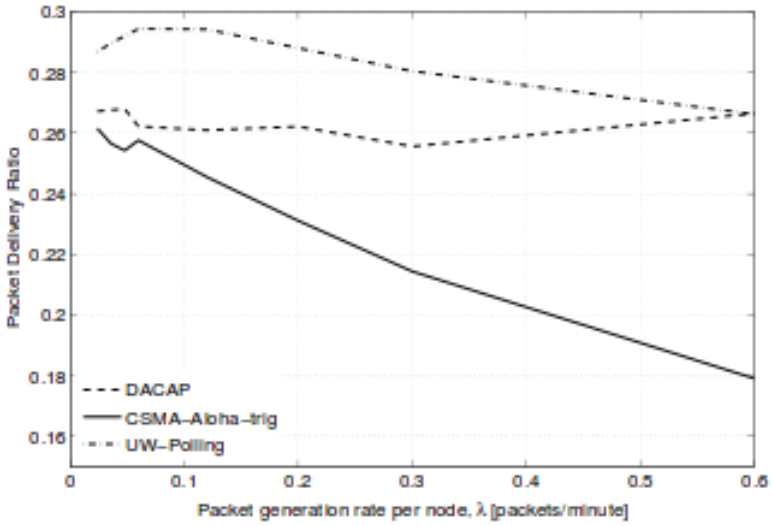


Figura 2.13: Packet Delivery Ratio al variare del tasso di generazione dei pacchetti da parte dei nodi singoli sensori.

end-to-end che può essere però leggermente diminuito ottimizzando la traiettoria dell'AUV. Inoltre un fattore bloccante per uw-POLLING potrebbe essere quello che, in una rete di elevate dimensioni, l'AUV debba passare in rassegna tutti i nodi per raccogliere i dati, ma questo richiede un tempo abbastanza grande a causa delle ridotte velocità di movimento di tale nodo.

2.3.2 MSUN

Altro protocollo utilizzato per il confronto delle prestazioni nei prossimi capitoli dell'elaborato è quello definito MSUN. Questo a differenza di uw-POLLING e UFETCH, si pone ad un livello appena superiore dello stack protocollare ISO/OSI rispetto ai due protocolli appena citati, ovvero al livello di rete.

Come tutti i protocolli di routing, anche MSUN ha come obiettivo la ricerca di un percorso tra i nodi della rete affinché un pacchetto generato da un nodo definito sorgente possa essere propagato ad un nodo specifico definito destinatario.
MSUN appartiene alla categoria dei protocolli source routing, quindi il nodo sorgente anziché utilizzare le routing table dei nodi intermedi per creare il percorso verso la destinazione, ad ogni pacchetto dati aggiunge all'interno del suo header

l'intera lista di nodi da attraversare, lista ottenuta a seguito di un algoritmo di *discovery* applicato nel momento in cui un nodo che deve inviare un pacchetto ad una destinazione non conosce il percorso che lo colleghi ad esso.
La tecnica di source routing introduce uno svantaggio che non deve essere sottovalutato, ovvero essendo inserita l'intera lista di nodi all'interno di ogni pacchetto dati trasmesso, più il numero di hop aumenta per raggiungere la destinazione e più il pacchetto aumenta la sua dimensione, aggiungendo di fatto overhead nella trasmissione.

MSUN è inoltre un protocollo reattivo, ovvero l'algoritmo di discovery per la ricerca di un percorso da sorgente a destinatario viene avviato solamente nel momento in cui deve avvenire la comunicazione tra i due nodi. Questo allegerisce la quantità di traffico presente nella rete, poiché non è presente una continua segnalazione che viaggia all'interno di essa occupando banda solamente per aggiornare le rotte. D'altro canto il costo da pagare è un maggiore ritardo di consegna, infatti con un protocollo proattivo nel momento in cui un nodo deve inviare un pacchetto dati, non deve avviare prima l'algoritmo di discovery dato che esso conosce già il percorso che lo può portare alla destinazione. Con un protocollo reattivo invece una prima fase è necessaria per la ricerca della rotta che collega sorgente-destinazione, aumentando così il ritardo medio di consegna dei pacchetti.

Il protocollo MSUN utilizza tre particolari pacchetti:

- *path request*: utilizzato dall'algoritmo di discovery per la ricerca del percorso verso uno specifico nodo.
  L'algoritmo di discovery funziona nel seguente modo: il nodo sorgente che vuole ricercare il percorso per raggiungere il nodo destinatario, invia un path request in broadcast a tutti i nodi entro il proprio raggio di copertura, aggiungendo nell'header di tale pacchetto il proprio identificatore (*id*) e quello del destinatario. Ogni nodo che riceve tale messaggio, se non ha già elaborato quella particolare richiesta, propaga a sua volta il pacchetto a tutti i nodi entro il proprio raggio di copertura, inserendo a sua volta il proprio *id* nell'header. Al contrario se il pacchetto è già stato elaborato, il messaggio sarà semplicemente ignorato dal nodo. Il procedimento continua finché il messaggio di path request non raggiunge il nodo destinatario, il

quale si impegna a ricezione avvenuta, di rispondere al nodo sorgente con un pacchetto di tipo *path answer* specificato al punto successivo.
Si osservi che ogni nodo, per ogni pacchetto ricevuto, invia al mittente un pacchetto *ACK* in modo da specificare la corretta ricezione di esso.
Alla ricezione del pacchetto path answer il nodo sorgente termina l'algoritmo di discovery, e utilizzando la sequenza di *id* contenuta nell'header di tale messaggio, effettua l'invio dei pacchetti dati verso il nodo destinatario.

- *path answer*: pacchetto inviato dal nodo che è il destinatario del pacchetto path request. All'interno dell'header di tale messaggio, il nodo destinatario specifica il percorso completo effettuato dal path request per raggiungerlo, sequenza di identifcatori che sarà poi utilizzata dalla sorgente per trasmettere i pacchetti dati.

- *path error*: un caso che potrebbe accadere durante la trasmissione di un pacchetto dati è quello in cui uno dei link intermedi sia corrotto e il nodo non è in grado quindi di propagare i pacchetti da esso ricevuti verso la destinazione finale. In questo caso tale nodo invia un pacchetto tipo path error verso la sorgente che ha immesso il pacchetto dati nella rete con l'obiettivo di notificare che quel particolare link non è più utilizzabile. Alla ricezione del path error il nodo sorgente o utilizza un percorso alternativo che esso ha già in memoria per comunicare con il destinatario, oppure avvia una nuova fase di discovery per ricercare altre rotte.

Una particolarità di MSUN riguarda la modalità con cui un percorso tra sorgente e destinazione viene scelto nel momento in cui più di uno sia disponibile. Infatti il nodo destinatario durante la fase di routing discovery risponde con un path answer ad ogni messaggio di path request ricevuto, ciò implica che più rotte alternative sono disponibili alla sorgente per poter comunicare con esso. Due sono le possibilità per scegliere il tragitto da far effettuare ai pacchetti data, la prima definita *lowest hop count* in cui la sorgente sceglie il percorso con il minore numero di hop che consente di raggiungere la destinazione, mentre la seconda possibilità è definita *maxmin SNR*, in cui per ogni rotta scoperta durante la fase di discovery la sorgente ricava il link con il rapporto SNR minore e di tutte queste considera poi il massimo

di tutti gli SNR estrapolati. Quello corrispondente al massimo rapporto SNR è il percorso scelto per comunicare con la destinazione.

In conclusione il protocollo MSUN essendo un algoritmo reattivo consente di mantenere le rotte solamente tra nodi che effettivamente devono comunicare, questo riduce l'overhead per il mantenimento dei percorsi, ma al tempo stesso può far aumentare il tempo medio di consegna dei pacchetti dati a causa della ricerca del percorso che consente di collegare i due nodi sorgente-destinazione.

Capitolo 3

Analisi preliminare: Polling, Multihop routing o schema Ibrido?

Data una rete di nodi, sia essa una rete acustica di sensori sottomarini o una rete radio terrestre, nel momento in cui il canale è condiviso tra tutte le entità sorge il problema già accennato nel capitolo 2, cioé quello dell'occupazione del canale in modo che un solo nodo alla volta ne abbia l'accesso esclusivo, se così non fosse si avrebbero delle collisioni tra i pacchetti dati e quindi la completa perdita delle informazioni. Il tutto è gestito dai protocolli di livello MAC che hanno appunto come obiettivo quello di risolvere questo problema.

Ipotizzando ora che un nodo (sorgente) abbia dei dati da trasmettere ad un altro nodo (destinatario) della rete e il canale sia condiviso, cioé una sola connessione disponibile, due possono essere le tecniche utilizzabili per la comunicazione: la prima denominata *multihop routing*, la seconda *polling*. Entrambe le modalità presentano vantaggi e svantaggi: a tal proposito il seguente capitolo inizialmente darà una breve descrizione di queste due metodologie, indicandone pregi e difetti. L'obiettivo principale di tale paragrafo è però quello di presentare una nuova tecnica che faccia un utilizzo combinato delle due precedenti sopra citate cercando di unirne i vantaggi. Tale modalità di comunicazione è chiamata *schema ibrido* come ovvia conseguenza del fatto che è l'unione delle due tecniche multihop routing

e polling. Questa nuova modalità sarà poi alla base del protocollo UFETCH che verrà presentato ed ampiamente discusso nel capitolo successivo 4.

Il capitolo è quindi organizzato come segue: una prima parte è dedicata alla presentazione delle tecniche multihop routing, polling e ibrida, nella seconda viene descritto lo schema e i parametri utilizzati nelle simulazioni, mentre nella terza ed ultima sezione si mostrano e commentano i risultati ottenuti a seguito di alcune simulazioni.

3.1 Multihop routing, Polling e Ibrido

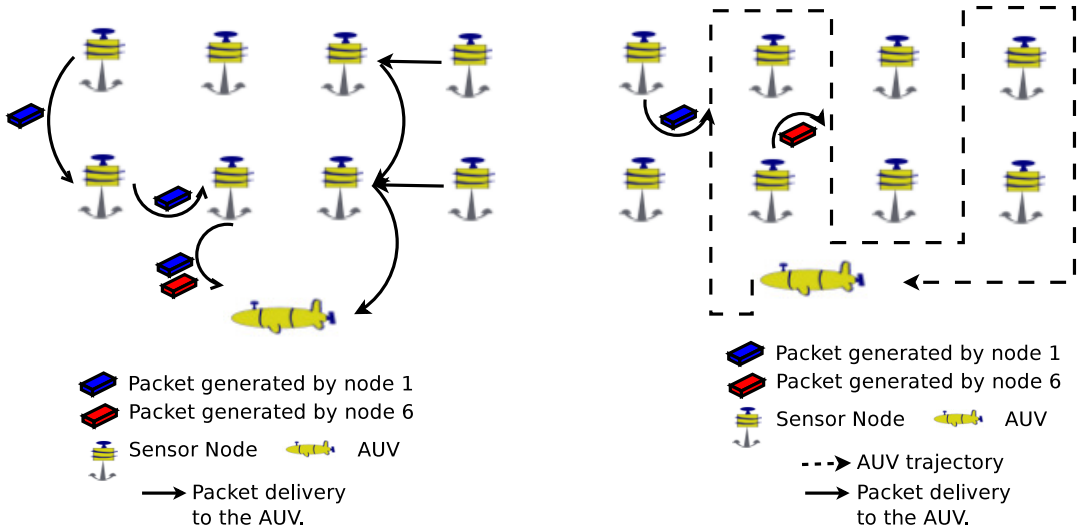
Svariati metodi esistono per la raccolta dei dati da parte dell'AUV e due sono quelli più importanti: *multihop routing* e *polling*.

Con la tecnica *multihop routing* la rete può essere suddivisa in maniera gerarchica considerando due livelli: un primo livello, quello di minore priorità, a cui appartengono i nodi sensori che raccolgono le informazioni dall'area monitorata, mentre un secondo livello a cui appartiene solamente il nodo AUV.
Utilizzando multihop routing il nodo AUV ha solamente la funzione di raccolta dei dati, senza interpellare direttamente i nodi affinché questi gli trasmettano le proprie informazioni. In questo caso l'AUV non è un'entità in movimento, ma fissa in una data posizione individuabile da coordinate geografiche (latitudine, longitudine, profondità).
Il fatto che l'AUV non sia in movimento implica che solamente un ristretto gruppo di nodi sensori lo abbia sotto la propria visibilità e quindi unicamente questi possano comunicare con esso. Da ciò deriva appunto il termine multihop, poiché un nodo che non sia entro il raggio di copertura dell'AUV, affinché vi possa trasmettere i propri frame, deve propagare i pacchetti dati ai nodi ad esso adiacenti, che a loro volta, se anche costoro non sono visibili all'AUV, dovranno inoltrare i frame propri e quelli ricevuti dai nodi adiacenti ai nodi interni al proprio raggio di copertura. L'informazione di un nodo quindi prima di arrivare all'AUV deve eseguire un certo numero di hop passando da nodi intermedi fino ad arrivare a quelli maggiormente vicini all'AUV che sono gli unici a poter inviare i dati all'AUV con una trasmissione diretta. Una rappresentazione grafica del procedimento è illustrata in figura (3.1(a)).

Un primo problema immediatamente osservabile è il collo di bottiglia che potrebbe verificarsi in prossimità dei nodi sensori più vicini all'AUV. Infatti tali nodi potrebbero ricevere una quantità di dati maggiore rispetto a quella che sono in grado di smaltire.
Una possibile soluzione, come sarà possibile vedere nella sottosezione successiva, è quella di aumentare la velocità di trasferimento dei dati verso il nodo AUV.

Un secondo problema riguarda gli algoritmi di routing, cioé la scelta del percorso migliore che il pacchetto deve effettuare affinché questo arrivi nel tempo più basso possibile a destinazione. Questo è un problema che appartiene al livello *network* dello stack protocollare ISO/OSI, ma è importante tanto quanto le tecniche di accesso al mezzo di cui si occupa il livello MAC, infatti un'ottimizzazione della rotta eseguita dal pacchetto è fondamentale per diminuire il ritardo di propagazione end-to-end.

La tecnica multihop routing a sua volta porta un vantaggio molto importante: ogni nodo può trasmettere con una potenza molto inferiore rispetto al caso in cui il pacchetto debba essere trasmesso direttamente al nodo AUV, cioé nel caso in cui si abbia una rete *fully connected*. Questo è molto importante sia per evitare il



(a) Multihop routing.

(b) Polling.

Figura 3.1: Esempio di reti in cui sono impiegate le tecniche di multihop routing e polling per il recupero dei dati dal nodo AUV.

near-far problem, già descritto in 2.1.3, sia per diminuire l'energia consumata dai nodi dato che è una risorsa molto preziosa in reti UWAN.

Una seconda tecnica utilizzabile per la raccolta dati in un rete acustica sottomarina è quella denominata *polling*. Con questa modalità la rete è ancora suddivisa in maniera gerarchica come descritto per il caso multihop routing, ma qui la sostanziale differenza sta nel fatto che è il nodo AUV stesso ad interpellare i nodi sensori affinché questi direttamente, senza alcun hop, gli inviino le proprie informazioni. Un esempio di comunicazione è mostrato in figura (3.1(b)).
In una rete di sensori, con l'utilizzo di una tale tecnica, il problema che potrebbe nascere riguarda l'elevata potenza utilizzata per la trasmissione dei pacchetti, dato che i nodi potrebbero essere molto distanti dall'AUV. La questione è risolvibile rendendo mobile l'AUV, in questo modo all'AUV gli viene impostata una traiettoria tale per cui consenta a tutti i nodi sensori periodicamente di averlo entro il proprio raggio di copertura per un certo intervallo temporale grande abbastanza affinché vi possano trasmettere le proprie informazioni.

L'introduzione della mobilità dell'AUV introduce a sua volta un problema: l'ottimizzazione della traiettoria. Infatti se la rete da monitorare è di vaste dimensioni e l'AUV viaggia ad una velocità molto bassa, solitamente 4 nodi, si può ben capire che il periodo con cui l'AUV si presenta entro il raggio di copertura del nodo sensore è molto alto.
L'elevato periodo tra due comunicazioni successive tra sensore e AUV è causa anche di un ulteriore problema, ovvero l'impossibilità da parte del nodo sensore di immagazzinare nuove informazioni raccolte durante tale intervallo temporale. Infatti non essendo in grado il nodo sensore di svuotare l'intera memoria durante la comunicazione con l'AUV, e non essendo la memoria di dimensioni illimitate, lo spazio disponibile per la memorizzazione di nuovi dati da parte del nodo diminuirà sempre più fino al completo esaurimento, con conseguente impossibilità di memorizzazione e quindi la perdita di nuovi dati raccolti.

Il vantaggio principale della tecnica polling è dato invece dal basso numero di collisioni, perché una volta che il nodo ha l'accesso per poter comunicare con l'AUV, e quest'ultimo è effettivamente ad una distanza dal nodo tale per cui possa avvenire la comunicazione, il canale sarà a sua completa disposizione ed altri interferenti non saranno presenti.

A seguito delle due tecniche multihop routing e polling sopra descritte, un miglioramento è possibile ottenerlo ipotizzando di unire le potenzialità delle due dando così vita alla tecnica *ibrida* e il cui nome è ovvia conseguenza.

La modalità ibrida in sostanza effettua una suddivisione della rete in tre livelli gerarchici, dove: al livello più basso si trovano i nodi sensori, al livello intermedio sono presenti i nodi sink mentre al livello più alto si posiziona l'AUV. Nodi sensori e AUV hanno i medesimi compiti descritti per le tecniche multihop routing e polling, la novità invece è costituita dai nodi sink che suddividono la rete in cluster, tanti quanti sono i nodi sink. Ad ogni gruppo appartengono un certo numero di nodi sensori che saranno gestiti da un nodo sink.
La raccolta delle informazioni recuperate ed elaborate dai nodi sensori all'AUV avviene nel seguente modo: periodicamente, e a seguito della richiesta da parte del nodo sink a cui fa riferimento, il nodo sensore gli trasmette i pacchetti dati, successivamente questi saranno inviati dai nodi sink al nodo AUV quando quest'ultimo si troverà nelle loro vicinanze. In questo caso, come per la tecnica polling, l'AUV è in movimento e segue una prestabilita traiettoria e sarà lui a stabilire quando è il momento di farsi recapitare i dati dai nodi sink.

Una maggiore delucidazione del procedimento è data dalla figura (3.2), infatti si può vedere come due siano i nodi sink nella rete e ognuno di essi abbia sotto il proprio controllo quattro nodi sensori. Le informazioni generate da quest'ultimi nodi vengono dapprima passate ai sink (freccia continua) e successivamente inoltrate all'AUV (freccia punteggiata).

Dalla descrizione del procedimento si intuisce da dove deriva il nome *ibrido*, perché sfruttando una topologia di rete particolare il pacchetto per arrivare a destinazione esegue due hop (Nodo sensore-Sink e Sink-AUV), ma al tempo stesso la comunicazione Sink-AUV avviene secondo la tecnica polling, essendo l'AUV ad interrogare il nodo sink per indicargli il momento in cui dovrà iniziare la comunicazione con esso.

Con la tecnica Ibrida si sono raggruppati i vantaggi delle due precedenti tecniche, multihop routing e polling. Essendo la distanza di tutti i nodi sensori al nodo sink relativamente breve, circa 2000 m, la potenza utilizzata per la trasmissione dei pacchetti è bassa, allo stesso tempo la comunicazione Sink-AUV avviene solamente quando l'AUV è nelle vicinanze del sink e quindi anche in tal caso la potenza di

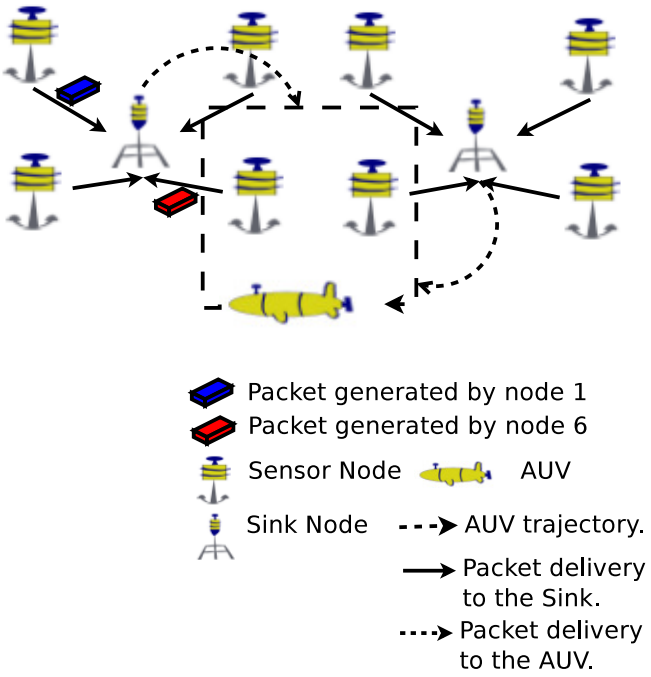


Figura 3.2: Esempio di rete in cui viene impiegata la tecnica ibrida per la consegna dei pacchetti al nodo AUV.

tramissione è limitata.
In seconda battuta le collisioni sono in percentuale ridotta come avveniva nel caso di polling, poiché il nodo sink funge da controllore per i nodi sensori, questi trasmettono solamente quando il sink gli darà il consenso, e simultaneamente anche il nodo AUV funge da controllore essendo colui che darà il via libera per la trasmissione dei dati da parte dei nodi sink verso di esso.

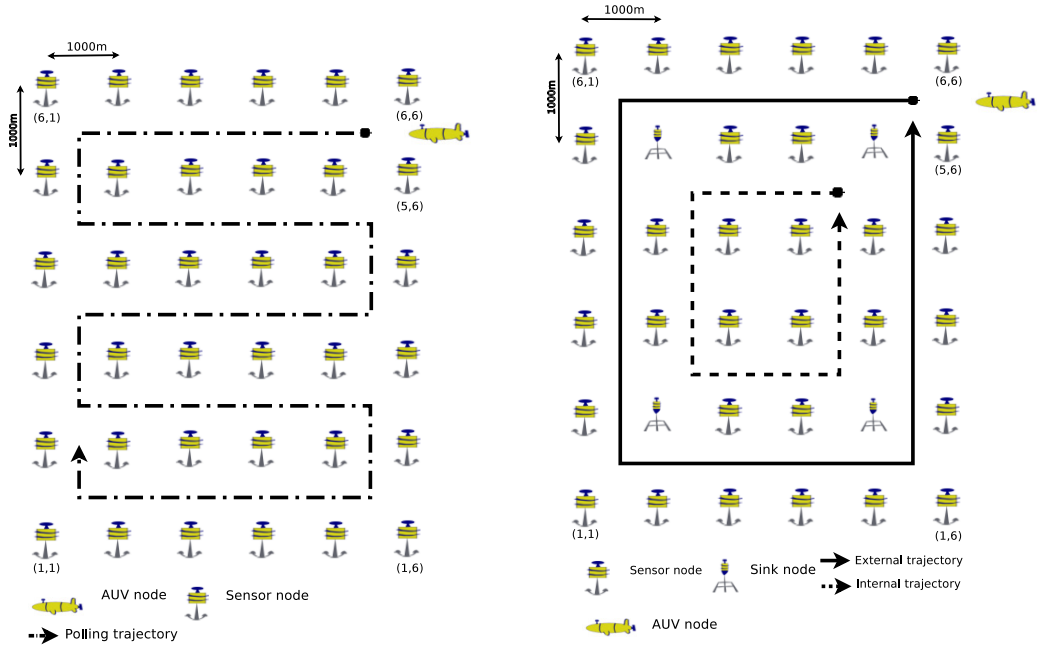
Uno svantaggio introdotto da questa tecnica potrebbe essere però l'eccessivo ritardo di consegna dei pacchetti end-to-end a causa della segnalazione che circola all'interno della rete.
Inoltre potrebbero formarsi dei colli di bottiglia in prossimità dei nodi sink, ma sarà possibile evitare tutto ciò considerando due distinte bit rate, una per la comunicazione nodo sensore-sink, l'altra per la trasmissione sink-AUV.

3.2 Schema e parametri di simulazione

Nel sottoparagrafo 3.1, sono state descritte tre tecniche per la raccolta dati in remoto utilizzabili per reti acustiche sottomarine. Di seguito sono valutate le prestazioni di tali tecniche tramite simulazioni effettuate in ambiente MATLAB.

Lo scenario utilizzato per le simulazioni è rappresentato in figura (3.3), dove sono presenti due distinte topologie, la prima, figura (3.3(a)), utilizzata per le tecniche multihop routing e polling, mentre la seconda, figura (3.3(b)), per la tecnica ibrida.
Entrambe le reti UWAN sono costituite da 36 nodi sensori predisposti ognuno sui vertici di una griglia 6x6. La distanza tra nodi adiacenti che si trovano nella medesima riga o colonna della griglia è di 1000 m, in questo modo l'area marina sorvegliata è di 6000 m^2.

Come già specificato le tipologie di nodi sono principalmente tre: i nodi sensori,



(a) Traiettoria AUV percorsa utilizzando uw-POLLING.

(b) Traiettoria AUV percorsa utilizzando UFETCH.

Figura 3.3: Topologie di reti in cui vengono impiegate le tecniche multihop routing, polling e ibrida per la consegna dei pacchetti dati al nodo AUV.

l'AUV e i nodi sink.
In questa sezione, per semplicità di simulazione, tutte le entità vengono considerate come degli elementi basilari che raccolgono, trasmettono e ricevono pacchetti dati. Una dettagliata descrizione dell'hardware che li costituiscono è presente nel sottoparagrafo 4.1.1 del capitolo successivo 4.

Considerando lo schema Multihop routing, si ha che gli elementi da esso utilizzati sono due: i nodi sensori e l'AUV. I nodi sensori avranno il compito di monitorare un'area circostante ad essi pari ad una circonferenza di 1500 m e trasferire tali dati, prima elaborati e organizzati in pacchetti, al nodo AUV. Considerando lo schema di comunicazione presentato in 3.1, tutti i nodi sensori che non sono entro il raggio di copertura dell'AUV, comunicheranno con altri nodi sensori, mentre solamente quei nodi nelle vicinanze dell'AUV potranno inviare messaggi a quest'ultimo, ed eventualmente anche ad altri nodi sensori.
L'altra entità presente nello schema Multihop routing è l'AUV, il quale è considerato fisso in una posizione, si veda la figura (3.3(a)). Esso avrà il compito di recuperare le informazioni raccolte da tutti i nodi sensori della rete e inviarle eventualmente ad una centrale di elaborazione dati presente in superficie.

Allo stesso modo di Multihop routing, le medesime entità sono utilizzate dallo schema Polling. I nodi sensori avranno il solito compito di monitorare l'area e successivamente trasmettere l'informazione all'AUV, mentre quest'ultimo di raccogliere i dati da tutti i nodi sensori. Ciò che differenzia le due tecniche è la mobilità dell'AUV, il quale non è più fisso in una posizione prestabilita, ma segue un percorso con una velocità di circa 4 nodi, e la trasmissione dati all'AUV da parte del sensore, che può avvenire solo a seguito di un'esplicita richiesta da parte dell'AUV stesso.
La traiettoria percorsa in questo caso è quella indicata in figura (3.3(a)), percorso scelto in modo tale che periodicamente l'AUV entri nel raggio di copertura di tutti i nodi sensori della rete.

Il terzo schema, quello Ibrido, introduce una nuova entità rispetto le due tecniche appena sopra descritte, ovvero i nodi sink. Questi hanno gli stessi compiti di un nodo sensore, ma sono gli unici nello schema ibrido ad avere la possibilità di trasmettere i dati all'AUV. Questo implica che i sink dovranno recuperare tutte le informazioni dai nodi sensori presenti entro un certo raggio dalla loro posizione,

raggio solitamente di 1500 m. Sono inoltre in un numero molto minore rispetto ai nodi sensori, come si può notare dalla figura (3.3(b)), per una rete di 36 nodi solamente quattro sono i nodi sink.
Anche per lo schema Ibrido il nodo AUV è considerato in movimento, ma la traiettoria percorsa cambia rispetto a quella implementata in Polling, dato che tale nodo deve entrare nel raggio di copertura dei soli nodi sink.

Il modello di canale utilizzato per le simulazioni coincide con quello ideale, errori sui bit a causa di attenuazione o interferenze quindi non saranno presenti. Tutti i nodi trasmettono con la medesima potenza, sia il nodo al limite del raggio di copertura entro il quale è visibile dal nodo sink o AUV, sia esso nelle immediate vicinanze di questo. L'assunzione fatta non è sicuramente valida ai fini pratici, perché provocherebbe sicuramente l'effetto near-far problem, ma l'obiettivo di queste simulazioni è solamente osservare se alcuni vantaggi sono ottenibili dalla tecnica Ibrida in termini di ritardo di consegna end-to-end.

Si assume inoltre che ogni nodo sensore, a seguito di una richiesta di comunicazione, abbia sempre almeno un pacchetto dati da trasmettere al nodo AUV nel caso di multihop routing, o al nodo sink nel caso di polling e ibrido. Un singolo pacchetto ha dimensioni L_D=4096 bits e nella rete ci possono essere fino ad un massimo di 5000 pacchetti redistribuiti in modo aleatorio tra tutti i nodi sensori. Anche il numero di pacchetti totali presenti nella rete è un parametro variabile in alcune simulazioni con lo scopo di poter effettuare l'analisi in diverse situazioni. In tal caso il range di variabilità è (36÷5000) pacchetti. Tuttavia il caso standard presuppone la presenza di 5000 pacchetti redistribuiti fra tutti i nodi sensori.

Come già accennato in precedenza, per evitare che si vengano a creare colli di bottiglia in prossimità dei nodi sensori vicini all'AUV nel caso di utilizzo Multihop routing, oppure ai nodi sink nel caso di impiego della tecnica Ibrida, le velocità di trasmissione dei dati sono due: la prima riguarda tutte le comunicazioni che si hanno verso il nodo AUV, in tal caso la velocità è pari a $B_{n,AUV} = 10$ kbps, la seconda riguarda tutte le comunicazioni da nodo sensore verso nodo sensore e da nodo sensore verso nodo sink, in tal caso la velocità è molto inferiore e vale $B_{n,n}$=1 kbps.
In alcune simulazioni effettuate la velocità $B_{n,n}$ sarà fatta variare all'interno di un certo range con estremi [4÷10] kbps, quando non espressamente indicato la velocità

sarà fissa a $B_{n,n}$=1 kbps.

Dalle assunzioni fatte precedentemente riguardo alla dimensione del pacchetto e la bit rate utilizzata per trasmetterlo, si ha che il tempo necessario affinché questo venga trasmesso ad un qualsiasi altro nodo è pari a:

$$t_{tx} = \frac{pck_L}{B_x} + \tau_p \tag{3.1}$$

dove nella (3.1) B_x varia, come spiegato poco sopra, a seconda che la trasmissione avvenga tra due nodi sensori ($B_{n,n}$), tra nodo sensore ed sink ($B_{n,AUV}$) o da sink verso AUV ($B_{n,AUV}$). Il parametro τ_p coincide con il tempo di propagazione del segnale affinché questo arrivi al nodo ricevitore. C'é da notare che a differenza delle trasmissioni wireless terrestri in cui il segnale si propaga con una velocità pari alla velocità della luce, $c \cong 3 \times 10^8 m/s$, in ambiente sottomarino la velocità di propagazione è molto inferiore e pari alla velocità del suono, quindi $c = 1500 m/s$. Nella relazione (3.1) il parametro τ_p vale:

$$\tau_p = \frac{d}{c} \tag{3.2}$$

intendendo con d la distanza tra il nodo trasmettitore e ricevitore.

Dalle simulazioni il principale parametro che si vuole ricavare, e utilizzato poi per il confronto delle tre tecniche, è il tempo necessario per la raccolta di tutti i pacchetti dati che i nodi hanno in memoria. In sostanza si calcola il tempo necessario affinché tutti i nodi svuotino la propria coda contenente i pacchetti dati, cioé la memoria del nodo sia completamente libera e tutti i pacchetti siano consegnati all'AUV.

Per quanto concerne la tecnica Multihop routing, il modello utilizzato per evitare le collisioni è il seguente: un nodo che propaga il pacchetto ad uno dei propri nodi vicini, ovvero ad un nodo entro il raggio di copertura pari a 1500 m, oltre ad inibire alla ricezione e trasmissione tutti i nodi all'interno di tale raggio, inibisce anche alla trasmissione tutti quei nodi entro il raggio di copertura del nodo ricevitore.
Infatti questo simula il fatto che il nodo ricevitore invii come risposta al nodo sorgente che vuole intraprendere la comunicazione, un pacchetto che, a sua volta,

sarà ricevuto anche da tutti i propri nodi adiacenti, i quali intuiscono che per un certo intervallo temporale essi non potranno comunicare con tale nodo, perché occupato in un'altra comunicazione. Il tempo di attesa è pari al tempo necessario affinché, tutti i pacchetti inviati dal nodo mittente, possano essere ricevuti dal nodo destinatario, valore contenuto all'interno dell'header del pacchetto inviato in risposta al mittente con cui ha allacciato la comunicazione.

3.3 Risultati

Le simulazioni hanno principalmente come scopo quello di osservare il tempo di raccolta dei dati da parte del nodo AUV sulla base delle tre tecniche: multihop routing, polling e ibrida descritte nella sottosezione 3.1.
Tre sono i blocchi di analisi effettuati che hanno portato ai grafici qui di seguito riportati e descritti.

Il primo set di prove è stato eseguito con l'obiettivo di verificare qual è il tempo medio di raccolta dati da parte del nodo AUV al variare della quantità di pacchetti dati presenti nella rete.
La tabella 3.1 contiene nel dettaglio tutti i valori dei parametri che sono mantenuti costanti durante l'intera simulazione. A questi deve essere aggiunta la quantità di pacchetti totali di cui è costituita la rete che viene fatta variare nell'intervallo seguente (36÷5000). Si osservi che il valore minimo coincide con il numero di nodi di cui è costituita la rete, questo perché si assume che dal momento in cui l'AUV inizia la raccolta dei dati, ogni singolo nodo sensore abbia almeno un pacchetto

Parametro	Valore assegnato
$B_{n,n}$	1 kbps
$B_{n,AUV}$	10 kbps
velocità AUV	2 m/s
raggio di copertura nodi sensori	1500 m
raggio di copertura AUV	1500 m
dimensione singolo pacchetto dati	4096 bits

Tabella 3.1: Valori dei parametri mantenuti fissi durante la simulazione che consentono di ottenere il primo set di grafici (3.4) e (3.5)

da trasmettergli. Inoltre all'aumentare del numero di pacchetti, questi vengono redistribuiti in maniera aleatoria tra tutti i nodi sensori.

Il grafico (3.4) mostra i risultati ottenuti: in ascissa si ha il numero di pacchetti redistribuiti tra i nodi sensori, in ordinata il valore temporale espresso in minuti affinché l'AUV raccolga tutte le informazioni dalla rete.
Le simulazioni sono state effettuate per tutte e tre le tecniche: multihop routing, polling e ibrida e i valori calcolati sono valori medi ottenuti a seguito di una serie di simulazioni. Il numero di esperimenti su cui si è effettuata la media aumentano al diminuire del numero di pacchetti totali presenti nel sistema, in modo da avere una maggiore precisione dei risultati.

Dalla figura 3.4, il tempo medio di recupero dei dati utilizzando la tecnica multihop routing cresce linearmente all'aumentare del numero di pacchetti. Questo é dovuto al fatto che essendo il nodo AUV fisso in una posizione, ed essendoci solamente due nodi sotto il raggio di copertura dell'AUV che hanno la possibilità di consegnare i pacchetti di tutti i nodi sensori che costituiscono la rete, comporta un collo di bottiglia in prossimità di questi, nonostante essi possano comunicare con una velocità molto più elevata e 10 volte superiore alla velocità di comunicazione che si ha tra due semplici nodi.

Nel caso di utilizzo del protocollo di Polling, si può osservare come il tempo medio di raccolta dei dati sia pressoché costante e pari al tempo necessario affinché l'AUV completi l'intera traiettoria e sia cosí in grado di interrogare tutti i nodi sensori. Dalla figura (3.4) si nota come sia presente, sia per la tecnica di polling, sia per la tecnica ibrida, un tempo minimo al di sotto del quale completare la raccolta dei dati è impossibile. Tale tempo dipende dalla traiettoria seguita dall'AUV e questo fattore permette già di osservare una cosa molto importante: la tecnica ibrida consente al nodo AUV di eseguire una traiettoria di lunghezza minore, quindi, a parità di dati presenti nella rete, il tempo di raccolta per la modalità ibrida è di molto inferiore rispetto alla tecnica polling. La minore lunghezza della traiettoria per la modalità ibrida è dovuta al fatto che l'AUV non dovrà passare in rassegna ogni centimetro dell'area marina monitorata, ma dovrà fare in modo di avvicinarsi solamente ai nodi sink che hanno recuperato i dati dai nodi sensori ad essi associati e che saranno posizionati in maniera strategica.
Sempre osservando la figura (3.4), si nota come la quantità di dati generata dai

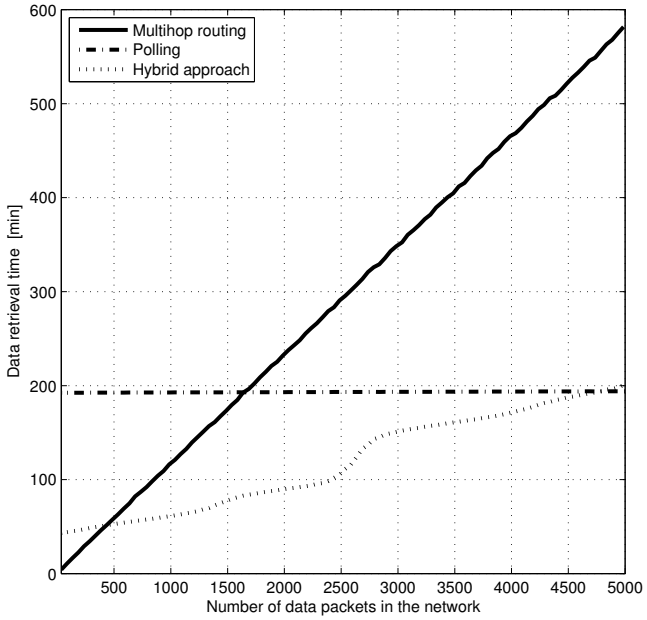


Figura 3.4: Rappresentazione del tempo medio totale per la raccolta di tutti i frame dati generati dai nodi sensori da parte del nodo AUV al variare della quantità di informazioni presenti nella rete utilizzando le tre tecniche: Multihop routing, Polling, Ibrido.

nodi sensori sia tale per cui l'AUV eseguendo una sola volta la traiettoria è in grado di recuperare l'intero set di dati. La quantità generata non è bassa, si pensi che avere 5000 pacchetti generati equivale circa a 20 Gbit di informazione, valori che in una rete UWAN attualmente difficilmente si raggiungono in un tempo di 200 minuti.

La figura (3.5) mostra nel dettaglio invece l'andamento della curva per la tecnica ibrida, già presentata nel grafico (3.4).
Da questa figura è possibile osservare un fatto importante, ovvero il cambio di pendenza della curva in prossimità dei valori in ascissa compresi nell'intervallo [2500÷3000]. Questo è dovuto alla minore lunghezza della traiettoria rispetto a quella utilizzata per la tecnica Polling. Infatti con una tale lunghezza della rotta, il nodo AUV non è in grado di recuperare con un singolo giro tutte le informazioni, per questo motivo il tragitto deve essere effettuato una seconda volta.

In conclusione da questo primo insieme di simulazioni si può affermare come inizialmente la modalità multihop routing abbia prestazioni decisamente migliori

rispetto alla tecnica Polling, questo perché l'AUV, nel secondo caso, deve attendere almeno un tempo minimo pari al completamento dell'intera traiettoria per visitare tutti i nodi sensori. Tuttavia, all'aumentare del traffico di rete, polling diventa migliore per il fatto che la bit rate tra nodo sensore e AUV è notevolmente superiore rispetto a quella tra due nodi sensori.
Per una bassa quantità di dati generata dai nodi sensori, la tecnica ibrida ha prestazioni peggiori rispetto a multihop routing, ma migliora decisamente non appena la quantità di pacchetti totali nella rete supera la soglia dei 500 pacchetti, sempre a causa del fatto che la bit rate tra nodo e AUV è molto maggiore rispetto alla velocità tra due nodi sensori.
Infine a parità di dati nella rete, la modalità di raccolta ibrida è molto più efficiente di quella polling come conseguenza dell'ottimizzazione della traiettoria eseguita dall'AUV.

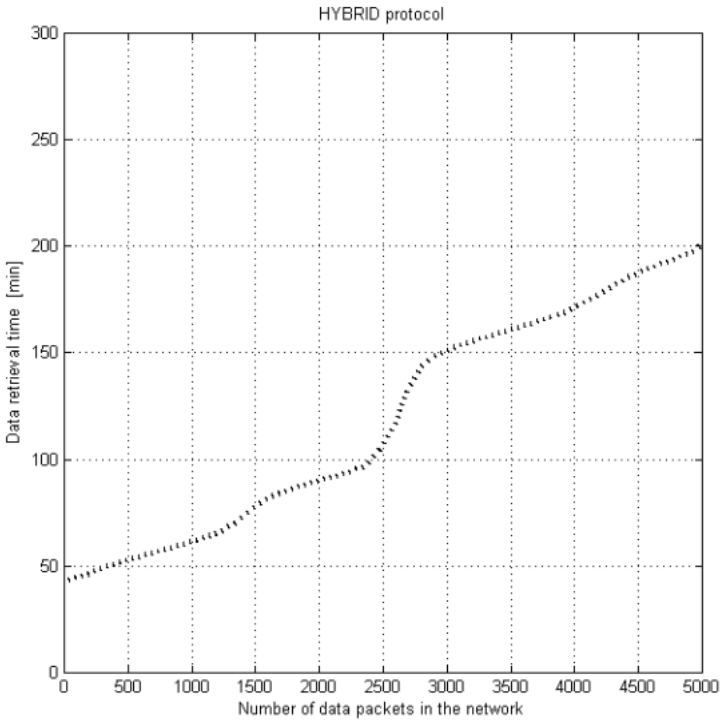


Figura 3.5: Rappresentazione del tempo medio totale per la raccolta di tutti i frame dati generati dai nodi sensori da parte del nodo AUV al variare della quantità di informazioni presenti nella rete utilizzando la sola tecnica Ibrida.

Il secondo set di simulazioni ha come obiettivo quello di presentare la relazione che sussiste tra la bit rate utilizzata per il trasferimento dei dati verso il nodo AUV e il tempo totale di cui esso necessita per la raccolta di tutti i frame dati generati dai nodi sensori.
I parametri utilizzati sono elencati in tabella 3.2. Oltre all'elenco presentato, l'ulteriore parametro considerato, che sarà poi quello variabile nelle simulazioni, è la bit rate di trasmissione che si ha dai nodi sensori verso l'AUV nel caso di multihop routing e ibrido, oppure da sink verso AUV nel caso di tecnica Ibrida, cioé quello precedentemente indicato con $B_{n,AUV}$. Il range di variabilità è $[2 \div 10]$ kbps.

La figura (3.6) mostra i risultati ottenuti applicando i parametri di tabella 3.2. Come nel caso del primo set di simulazioni, tutti i valori indicano valori medi ottenuti a seguito di una serie consecutive di prove.

Nel caso di utilizzo della tecnica Multihop routing non si trae molto beneficio dall'aumento della bit rate $B_{n,AUV}$, questo perché sono le trasmissioni tra nodi sensori, che sono la maggioranza, ad influenzare il tempo di consegna dei dati all'AUV. Il collo di bottiglia è sempre presente in prossimità degli unici due nodi che sono entro il raggio di copertura dell'AUV.

La bit rate a differenza del Multihop routing influenza, e di molto, il tempo di raccolta dei dati nel caso di Polling e Ibrido.
Dal grafico (3.6) si puó vedere come nel caso di Polling il tempo di consegna dei dati diminuisce di molto all'aumentare della bit rate, fino a quando non si raggiunge un punto critico, che utilizzando i parametri di simulazione impostati e descritti in tabella 3.2, si attesta nell'intorno dei 3.5 kbps. Oltre tale valore un ulteriore

Parametro	Valore assegnato
$B_{n,n}$	1 kbps
velocità AUV	2 m/s
raggio di copertura nodi sensori	1500 m
raggio di copertura AUV	1500 m
dimensione singolo pacchetto dati	4096 bits
numero totale di pacchetti dati nella rete	5000 pcks

Tabella 3.2: Valori dei parametri mantenuti fissi durante la simulazione che consentono di ottenere il secondo set di grafici (3.6).

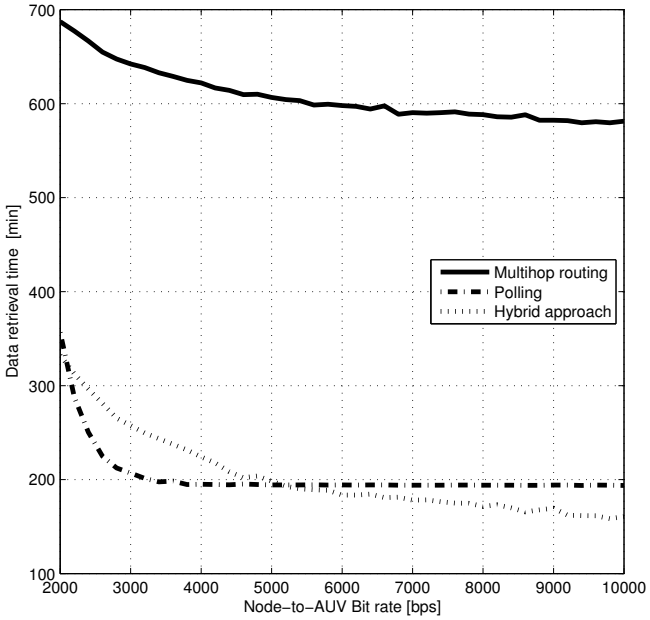


Figura 3.6: Rappresentazione del tempo medio totale per la raccolta di tutti i frame dati generati dai nodi sensori da parte del nodo AUV al variare della bit rate verso il nodo AUV utilizzando le tre tecniche: Multihop routing, Polling, Ibrido.

aumento della bit rate sarebbe inutile ai fini del miglioramento del tempo di raccolta dei dati perché il limite minimo è sempre imposto dalla velocità con cui l'AUV viaggia, che determina appunto il tempo necessario per interrogare tutti i nodi che monitorano l'area. Per bit rate basse il tempo è elevato perché l'AUV, rimanendo per poco tempo entro il raggio di copertura dello specifico nodo sensore, non è in grado di raccogliere tutti i dati che un nodo sensore ha bufferizzato, e quindi necessita di percorrere più volte il tracciato.

Nel caso della tecnica Ibrida, avendo i sink maggiori dati da spedire all'AUV, questi richiedono che l'AUV rimanga per più tempo entro il loro raggio di copertura, ma ciò non può accadere perché l'AUV ha una velocità di viaggio prefissata e costante, capita quindi che esso debba ripercorrere più volte la traiettoria prima di recuperare tutti i dati. Conseguenza affinché il numero di giri del percorso necessari per la raccolta totale dei dati diminuisca, la bit rate tra sink e AUV dovrà essere molto elevata rispetto al caso di Polling prima che si raggiunga il punto critico oltre il quale un ulteriore aumento non comporterebbe alcun beneficio.

In conclusione, a parità di velocità di viaggio dell'AUV, la tecnica Multihop

routing non è la migliore da considerare a causa dell'elevato tempo di raccolta, mentre la scelta tra Polling e Ibrido dipende dal carico della rete, infatti per basse bit rate e poca capacità di rete, la prima è la scelta migliore, però già per valori nell'intorno di 3.5 kbps la seconda tecnica presenta prestazioni decisamente migliori.

Il terzo set di simulazioni è valido solamente per le tecniche di Polling e Ibrido in quanto si assume che il nodo AUV sia in movimento e non fissato su coordinate predefinite a priori.
L'obiettivo di queste prove è quello di verificare come varia il tempo di raccolta totale dei dati al variare della velocità del nodo AUV.

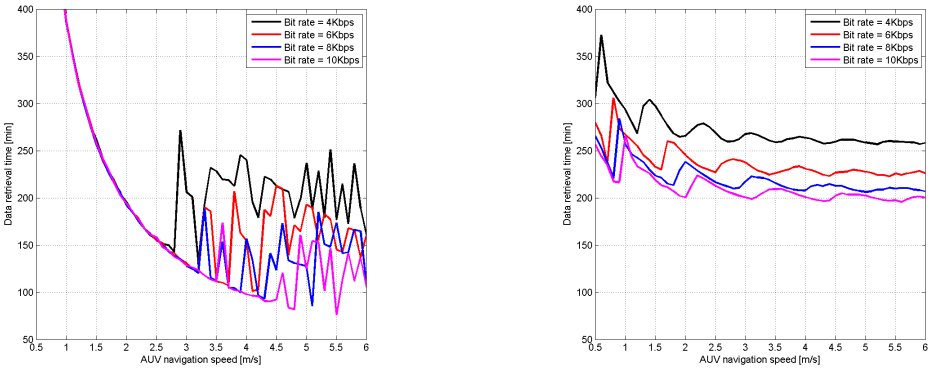
I parametri utilizzati sono presentati in tabella 3.3. Oltre a tali parametri mantenuti costanti durante tutta la fase di simulazione si aggiungono due misure che sono: la velocità di movimento dell'AUV e la bit rate utilizzata per trasmettere i frame dati verso di esso. Tutte le simulazioni sono state effettuate per quattro bit rate distinte, $B_{n,AUV}$=4,6,8,10 kbps e per ognuna di queste è stata fatta variare la velocità di movimento del nodo AUV che è compresa nell'intervallo [0.5÷6] m/s. Tutti i valori ottenuti coincido con valori medi dato che sono ricavati a seguito di serie distinte di run.

Dalla figura (3.7) si può vedere come in entrambi i casi, cioé sia per la tecnica Polling, sia per la tecnica Ibrida, all'aumentare della velocità di viaggio dell'AUV si ha una diminuzione del tempo di raccolta totale dei dati. Questo è un risultato ovvio per il semplice motivo che, a parità di dati presenti nella rete e a parità della lunghezza del percorso eseguito dall'AUV, tale percorso sarà effettuato in un tempo sempre più basso dato che la velocità aumenta.

Un fattore che influenza il tempo medio di raccolta dei dati è comunque la bit

Parametro	Valore assegnato
$B_{n,n}$	1 kbps
raggio di copertura nodi sensori	1500 m
raggio di copertura AUV	1500 m
dimensione singolo pacchetto dati	4096 bits
numero totale di pacchetti dati nella rete	5000 pcks

Tabella 3.3: Valori dei parametri mantenuti fissi durante la simulazione che consentono di ottenere il terzo set di grafici (3.7).



(a) Polling.

(b) Ibrido.

Figura 3.7: Tempo totale di recupero dei dati per le tecniche Polling e Ibrida al variare della velocità di movimento dell'AUV e al variare della bit rate verso il nodo AUV.

rate utilizzata per comunicare verso l'AUV, infatti più questa è bassa e peggiori saranno le prestazioni, questo perché la traiettoria potrebbe essere percorsa più di una volta non essendo in grado il nodo di trasmettere tutti i propri dati in un unico intervallo temporale pari al tempo in cui l'AUV rimane all'interno del proprio raggio di copertura. Lo si può notare dalla figura (3.7(a)) per il caso Polling e (3.7(b)) per la tecnica Ibrida, dove utilizzando una bit rate di 4kbps la curva è al di sopra di tutte le altre curve ottenute con bit rate più elevate.
Una particolarità è data dalle continue oscillazioni presentate per valori di velocità elevate. Queste oscillazioni, che sono maggiormente marcate nella figura (3.7(a)), dipendono dal modo in cui i pacchetti sono redistribuiti tra i vari nodi sensori. La redistribuzione aleatoria utilizzata potrebbe causare la memorizzazione dei pacchetti nel nodo presente nella parte opposta rispetto al punto di partenza dell'AUV, facendo eseguire quindi un giro della traiettoria aggiuntivo solamente per recuperare tali dati.

Dalle due figure (3.7) è osservabile come nel caso di tecnica Ibrida, a parità di dati presenti nella rete, il tempo totale necessario per la raccolta dei dati sia inferiore, soprattutto per basse velocità dell'AUV, rispetto al caso di Polling.

Dall'analisi preliminare sopra presentata si evince come la tecnica Multihop routing abbia prestazioni peggiori rispetto alle modalità di Polling e Ibrida. Solo in

alcuni casi la tecnica Multihop routing presenta prestazioni migliori, ovvero quando la rete è poco carica di dati, cioé i nodi hanno pochi pacchetti da consegnare all'AUV.

L'introduzione del fatto che il nodo AUV possa muoversi in modo tale da esplorare la rete comporta un beneficio nel tempo di raccolta dati da parte di esso. Ottimizzando la traiettoria da percorrere, quindi organizzando la rete in cluster come appunto avviene nel caso della tecnica Ibrida, si ottengono ulteriori benefici in termini temporali. In entrambi i casi però un lower bound è dato dalla velocità con cui il nodo AUV si muove e questo, come mostrato dai grafici comporta che un eventuale aumento della bit rate verso l'AUV non comporterebbe alcun vantaggio in termini prestazionali.

A seguito di questo studio che ha portato ai risultati preposti all'inizio del lavoro, il passo successivo e già accennato in precedenza, è ora testare questo nuovo protocollo in un ambito più realistico, utilizzando il simulatore di rete ns2/NS-Miracle e le librerie di DESERT, lavoro presentato nei capitoli successivi 4 e 5. In questo modo è possibile effettuare simulazioni più approfondite aggiungendo una certa modulazione di canale, inserendo la presenza di errori nel canale e attenuazioni dovute alla propagazione del segnale in ambito sottomarino.

Capitolo 4

Il protocollo UFETCH

Fino ad ora nei capitoli precedenti si è fatta una lunga panoramica di tutte le problematiche che una rete acustica sottomarina può presentare, concentrandosi soprattutto sul livello MAC, ovvero quello strato dello stack protocollare ISO/OSI che si pone come obiettivo l'ottimizzazione dell'accesso al mezzo di comunicazione condiviso tra una serie di entità che, nel caso di reti UWAN, sono i nodi acustici. A tal proposito sono stati presentati molti protocolli, alcuni derivanti da reti radio terrestri e adattati all'ambito sottomarino, altri specifici per reti subacque.
Sapendo inoltre che uno degli obiettivi fondamentali di una rete UWAN è quello della raccolta dati da remoto, da inviare poi ad una centrale di elaborazione dati posta in superficie, il precedente capitolo 3 ha presentato due tecniche utili a tale scopo, multihop routing e polling, aggiungendone poi una terza che costituisce un'innovazione in tale ambito, la modalità ibrida.
A partire da tutto ciò, il fine del seguente capitolo è quello di presentare un nuovo protocollo denominato UFETCH e specifico per reti acustiche sottomarine che va ad affiancarsi a tutti quelli già descritti in 2 utilizzando come base per la raccolta delle informazioni la nuova tecnica ibrida, esposta in 3.

Il capitolo è quindi così organizzato: una prima sezione, in cui saranno esposte tutte le entità che costituiscono il protocollo UFETCH dandone una descrizione dettagliata anche in termini di componentistica e presentando le migliori soluzioni ad oggi presenti nel mercato. Successivamente una seconda sezione è dedicata all'esposizione del protocollo D-MAC che sta alla base di UFETCH. Infine il capitolo

si conclude con una terza parte, dedicata alla presentazione dei compiti associati ad ogni elemento che costituisce la rete, mostrando le macchine a stati di ognuno di essi e evidenziando le varie fasi in cui il singolo elemento potrebbe trovarsi durante la sua attività di monitoraggio e comunicazione verso altre entità.

4.1 La struttura del protocollo

4.1.1 Le entità in gioco

Il protocollo UFETCH è costituito principalmente da tre entità che possono essere anche raggruppate in tre livelli gerarchici:

- nodi sensori;
- sink;
- AUV.

Al più basso livello di priorità appartengono i nodi sensori. Questi costituiscono il cuore di una rete acustica sottomarina dato che sono quelli che hanno il compito di monitorare e recuperare le informazioni dall'area in cui sono installati. Solitamente la quantità presente all'interno della rete si aggira nell'intorno di qualche decina o al massimo centinaia di nodi, predisposti casualmente e ad una distanza uno dall'altro che può variare dai 2000 m ai 5000 m, potendo così ricoprire aree marine anche di 50000 m^2. La rete considerata per le successive simulazioni presentate, a riguardo del protocollo UFETCH, impiega ad esempio 36 nodi che ricoprono un'area di 6000 m^2.

Un nodo sensore generalmente al suo interno monta un processore interfacciato con uno o più sensori, i quali, ognuno di questi ha il compito di rilevare caratteristiche e qualità dell'acqua: temperaturà, salinità, densità, acidità, ed aspetti chimici come: conduttività, pH e torbidità. Tutti i dati raccolti sono poi elaborati dalla CPU ivi installata e successivamente inviati attraverso un modem acustico all'entità presente nel fondale, quale potrebbe essere il sink, che a sua volta le invierà alla base in superficie.
Il modem ha un'elevata importanza poiché: determina la velocità di trasmissione

dei pacchetti dati, limita la quantità di dati che i sensori possono immagazzinare (la memoria installata non è infinita) e incide sul consumo di energia del sistema.

I modem acustici possono essere posizionati a varie profondità, a seconda dei materiali con cui sono realizzati: valori tipici partono da 200 m, se installati in contenitori realizzati in poliossimetilene, fino ad arrivare a 2000 m per quelli contenuti in box realizzati in titanio. Considerando inoltre i modem prodotti da EvoLogics, ed utilizzati per le simulazioni, il loro range di operatività, cioé la distanza entro la quale due modem possono comunicare, è compreso tra i [1500÷3500] m e l'intervallo di frequenze utilizzabili per la trasmissione è [18÷34] kHz.

Come già accennato, inoltre, un parametro da non sottovalutare per i nodi sensori è il loro consumo energetico, essendo questi alimentati da batterie la cui durata è limitata. A tal proposito è molto importante ridurre lo spreco energetico durante le fasi di stand-by e di operatività, soprattuto in fase di trasmissione che è quella a maggiore consumo.
I parametri presi da datasheet e, recuperabili da [16], sono relativi ai modem utilizzati nelle simulazioni che saranno presentate di seguito e indicano alcuni valori tipici di consumo, che: nella fase di stand-by è pari all'incirca 2.5 mW, in modalità di *listening* circa 100 mW, mentre in fase di *receiving* si raggiungono valori di 1.3 W.

Le tabelle 4.1 e 4.2 riassumono alcune caratteristiche di due tipi di modem utilizzati comunemente per reti acustiche sottomarine, e utilizzati poi nelle simulazioni effettuate e presentate nel capitolo 5, dati presi da [16].

L'entità di livello intermedio tra nodi sensori e AUV, dove appunto l'AUV è il terzo elemento della scala gerarchica di cui è costituito il protocollo UFETCH, è quella denominata *sink*.
Il sink dal punto di vista delle operazioni svolte è l'elemento maggiormente complesso e due sono le principali funzioni da esso svolte: la prima richiedere e memorizzare i dati provenienti dai nodi sensori, la seconda, trasferire tutti i dati raccolti al passo precedente all'AUV.

Dal lato della componentistica invece, il nodo sink è assimilabile ad un semplice nodo sensore, presumibilmente con componenti di prestazioni maggiori, ma avente sempre il compito di monitorare l'area che lo circonda entro un certo raggio, come fanno i nodi sensori, ma oltre a ciò è colui che ha il ruolo esclusivo di comunicazione con l'AUV, fatto che ai nodi sensori è vietato. Quindi in definitiva, un sink è

Specifiche tecniche modem 1	
Profondità di installazione	200 m involucro in poliossimetilene 1000 m involucro in lega di alluminio 2000 m involucro di acciaio inox 2000 m involucro di titanio
Raggio di comunicazione	1000 m (2000 m in caso di ottime condizioni)
Banda di frequenza	48-78 kHz
Connessione acustica	fino a 31.2 kbit/s
Bit error rate	inferiore a 10^{-10} (in condizioni ideali)
Memoria interna per raccolta dati	1 MB configurabile
Potenza consumata	Modalità stand-by: 2.5 mW Modalità ascolto: [5-285] mW Modalità ricezione: inferiore a 1.1 W Modalità di trasmissione: 5.5 W per distanze di 250 m 8 W per distanze di 500 m 18 W per distanze di 1000 m 60 W per le distanze massime disponibili

Tabella 4.1: Modem 1: adatto per comunicazioni a corto raggio, con elevato bit rate e con predisposizione del livello data-link per AUV e ROV per reti acustiche sottomarine di tipo *shallow-water*.

sempre costituito da una serie di sensori interfacciati con una CPU che elabora i dati, li memorizza e poi attraverso un modem acustico li trasferisce al nodo AUV.

L'elemento di più alta priorità, considerando la scala gerarchica delle entità esposta inizialmente di cui è costituito il protocollo UFETCH, è denominato AUV. L'AUV è un robot che lavora in acqua ed è in grado di portare a termine delle missioni in maniera completamente autonoma. Si distinguono appunto dai ROV, veicoli operati da remoto, per il fatto che non hanno bisogno di essere collegati via cavo ad un pilota umano.
Tra i modelli più diffusi si può citare il *Glider*, strumento che consente di effettuare missioni di durata lunga grazie al proprio basso consumo energetico, esso si muove seguendo una preimpostata traiettoria e con una velocità che varia nel range di

Specifiche tecniche modem 2	
Profondità di installazione	200 m involucro in poliossimetilene 1000 m involucro in lega di alluminio 2000 m involucro di acciaio inox 2000 m involucro di titanio
Raggio di comunicazione	3500 m
Banda di frequenza	[18-34] kHz
Connessione acustica	fino a 13.9 kbit/s
Bit error rate	inferiore a 10^{-10} (in condizioni ideali)
Memoria interna per raccolta dati	1 MB configurabile
Potenza consumata	Modalità stand-by: 2.5 mW Modalità ascolto: [5-285] mW Modalità ricezione: fino a 1.3 W Modalità di trasmissione: 2.8 W per distanze di 1000 m 8 W per distanze di 2000 m 35 W per distanze di 3500 m

Tabella 4.2: Modem 2: adatto per comunicazioni a medio-lungo raggio con predisposizione del livello data-link per AUV e ROV per reti acustiche sottomarine di tipo *shallow-water*.

[4÷8] nodi.
Il compito principale quindi dell'AUV è quello di raccogliere i dati che i nodi sink, distribuiti nell'area monitorata in maniera strategica, hanno recuperato a loro volta dai nodi sensori. Naturalmente la comunicazione tra sink ed AUV potrà avvenire quando quest'ultimo è entro il raggio di azione del primo, raggio che dipende molto dalla potenza utilizzata per la trasmissione. Valori tipici sono 2000 m. I dati recuperati dall'AUV saranno poi, al termine della missione, riportati alla stazione base presente in superficie.

4.1.2 L'architettura del protocollo D-MAC.

Le funzionalità del protocollo di livello fisico basato sulla tecnologia S2C [17], hanno creato i presupposti per la realizzazione di un nuovo protocollo ibrido

applicabile a livello data link dello stack protocollare ISO/OSI, denominato D-MAC.
Il protocollo D-MAC offre l'opportunità di gestire l'utilizzo del canale sulla base della tipologia dei pacchetti da inviare per mezzo di due algoritmi definiti: *short term media access algorithms* e *burst media access algorithms*. La differenza tra le due tecniche citate è la seguente:

1. short-term media access algorithm è principalmente utilizzato per lo scambio di messaggi brevi sia in broadcast, sia in unicast, mediante un basso bit rate, valori nell'intorno di $[1 \div 2]$ kbps, e offrendo la possibilità di accedere al canale contemporaneamente a più nodi della rete;

2. burst media access algorithm è progettato invece per la trasmissione di grandi moli di dati utilizzando la bit rate ottimale e non consentendo ad altri nodi di accedere al canale se già occupato. L'algoritmo implementa uno schema di stima di canale, in modo da adattare la bit rate di trasmissione sulla base delle sue condizioni stimate.

Per quanto appena sopra detto, il protocollo D-MAC supporta quindi due distinte tipologie di pacchetti dati:

1. *burst data*, i quali richiedono forzatamente l'utilizzo di una connessione tra nodo locale e nodo remoto affinché possa avvenire la comunicazione.
   Stabilire una connessione è molto importante, infatti consente di effettuare una stima del canale, utilizzabile poi dagli algoritmi di routing per ottimizzare l'efficienza di utilizzo del canale, e di ricavare la bit rate più alta possibile da impiegare poi per le trasmissioni dei pacchetti.

   La creazione e il mantenimento della connessione consta di tre fasi:

   - 1 FASE: *Creazione della connessione*. In questo stadio nodo locale e nodo remoto effettueranno lo scambio di due messaggi: RTS e CTS. Inizialmente il nodo locale trasmette un pacchetto denominato RTS, con lo scopo di richiedere la possibilità di iniziare una trasmissione dati verso il nodo remoto. A seguito della ricezione del pacchetto RTS, il nodo remoto, se attualmente libero da qualsiasi altra comunicazione,

risponde con un messaggio CTS, che, una volta ricevuto dal mittente, indica l'avvenuta creazione con successo della connessione.
Il pacchetto CTS gioca un ruolo fondamentale all'interno di questa fase, dato che contiene la stima del canale ottenuta analizzando il segnale RTS ricevuto. Il suo contenuto è utile al nodo locale per settare la bit rate ottimale da utilizzare per la trasmissione dei pacchetti DATA.

- 2 FASE: *Consegna burst data.* A seguito dello stabilimento della connessione, cioé dopo aver ricevuto il pacchetto CTS entro il tempo prestabilito, il nodo locale, utilizzando la bit rate ottimale, inizia la trasmissione in maniera sequenziale dei pacchetti DATA verso il nodo remoto.
  Il protocollo D-MAC offre inoltre l'opportunità di implementare tecniche ARQ, quindi in tal caso, ritrasmissioni di pacchetti DATA saranno ammesse se questi sono o in errore a causa del canale, oppure arriveranno al ricevitore fuori sequenza.

- 3 FASE: *Chiusura della connessione.* Una volta trasmessi tutti i pacchetti DATA dal nodo locale, quest'ultimo invierà un messaggio al nodo remoto con il quale indica la volontà di chiudere la connessione.
  La spedizione di tale pacchetto non è opzionale, infatti mantenere aperta una connessione implica uno spreco di risorse, soprattutto di memoria. A tal fine, se il nodo remoto, dopo un certo intervallo temporale non riceve alcun pacchetto dal nodo locale, sarà esso stesso a richiedere la chiusura della connessione acustica. Solitamente la durata di tale timeout è pari al valore di RTT tra i nodi.

RTS, CTS sono messaggi utili per creare e gestire la connessione e a tal proposito sono anche denominati *Service messages.* Tutti questi saranno sempre trasmessi con la bit rate minima offerta dal sistema.

2. *instant message*, è la seconda tipologia di messaggi supportati dal protocollo D-MAC. Questi possono essere trasmessi senza la creazione di alcuna connessione.
   Solitamente gli instant messagge non hanno dimensioni superiori alle centinaia di bit e possono essere inviati in maniera indipendente dai burst data, oppure

semplicemente accodati ai service message.
Questa categoria a sua volta è costituita da tre diverse tipologie:

- *Robust message*, sono quei pacchetti che richiedono la conferma di corretta ricezione da parte del trasmettitore attraverso messaggi di acknowledgements (ACK).
  Nel caso in cui la trasmissione non vada a buon fine la si ripete fintantoché non si avrà esito positivo, oppure non si raggiungerà un valore limite di tentativi impostato in fase di progettazione;
- *Datagram message*, costituiscono la versione opposta dei robust message, questi messaggi non richiedono alcuna conferma di avvenuta e corretta ricezione e la loro trasmissione avvienne una sola volta, sia che vada o meno a buon fine;
- *Broadcast message*, sono pacchetti che non richiedono alcun ACK in risposta al pacchetto trasmesso, l'unica differenza rispetto ai datagram message è appunto che sono broadcast, cioé indirizzati a tutti i nodi che sono entro il raggio di copertura del nodo trasmettitore.

Si osservi che gli instant messagge possono essere trasmessi in un qualsiasi istante senza interrompere eventuali comunicazioni di burst data.

Come già accennato nel capitolo 1, nello specifico il sottoparagrafo 4.1.1, un nodo appartenente ad una rete acustica sottomarina può trovarsi in diverse fasi durante il suo periodo di vita. Secondo l'architettura del protocollo D-MAC queste sono:

- *Listen state*, in questo stato il nodo può ricevere messaggi di servizio che consento la creazione di una connessione, quindi RTS o CTS, oppure può trasmettere messaggi di tipo burst o instant data.
  Dal momento che l'accesso al canale è esclusivo, se già un nodo sta trasmettendo burst data, qualsiasi altro nodo, una volta percepita la presenza della comunicazione in atto, passerà allo stato di backoff;
- *On-line state*, è lo stato in cui il nodo locale ha il permesso di ricevere o inviare burst data da/a un nodo remoto. Anche in questo caso, se il nodo

percepisce una comunicazione attiva tra un nodo locale ed un nodo remoto, allora tratta questa come un conflitto, ed onde evitare collisioni, ritorna nello stato di backoff rimandando di fatto la sua trasmissione;

- *Backoff state*, è lo stato di riposo del nodo. Una volta entrato, il nodo rifiuta qualsiasi tipo di richiesta di invio e/o ricezione di burst data e instant message. Il nodo vi rimane per un tempo limitato, dopo il quale passerà allo stato listen.

In conclusione il protocollo D-MAC offre ai progettisti di protocolli per reti acustiche sottomarine un'architettura di livello link layer che consente di utilizzare due modalità di trasmissione per mezzo di due tipologie di messaggi: burst e instant messages.

4.1.3 I pacchetti del protocollo UFETCH

Il protocollo UFETCH è stato realizzato a partire dall'architettura base D-MAC descritta nel sottoparagrafo 4.1.2 di questo capitolo. Tale protocollo quindi utilizza i pacchetti RTS e CTS per la creazione della connessione tra sink ed AUV. Oltre a questi, ne saranno utilizzati altri che saranno descritti qui di seguito.

Poiché UFETCH è costituito principalmente da due flussi di comunicazione, il primo tra nodi sensori e sink, il secondo tra sink ed AUV, per ognuno di questi è utilizzata una specifica e distinta messaggistica.
Per quanto concerne la comunicazione nodo sensore-sink, l'instaurazione della connessione avviene atttraverso i seguenti messaggi:

- *BEACON*: pacchetto inviato dal sink in broadcast per risvegliare i nodi sensori entro il suo raggio di copertura. Mediante questo messaggio il sink avvisa il nodo sensore della volontà di ricevere i pacchetti DATA che quest'ultimo ha raccolto a partire dall'ultima comunicazione con esso avvenuta.
  Come tutti i pacchetti di segnalazione, il pacchetto di BEACON non contiene una parte dati, ma è costituito solamente dall'header, all'interno del quale è presente l'estremo inferiore e superiore, ($t\_min\_bck \div t\_max\_bck$), dell'intervallo entro il quale il nodo sensore dovrà scegliere il suo valore prima di trasmettere il pacchetto PROBE. Inoltre un terzo elemento, c_beacon_pck,

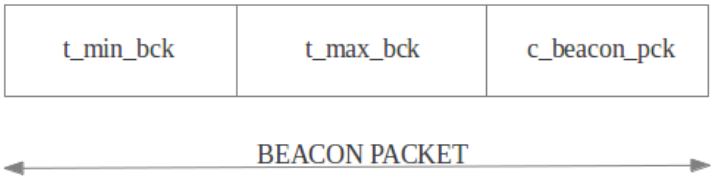


Figura 4.1: Struttura del pacchetto BEACON utilizzato per la creazione della connessione tra nodo sensore e nodo sink.

indica il numero di pacchetti CBEACON che il nodo sensore deve attendere dopo la ricezione del BEACON. Questo valore è importante dato che nega al nodo sensore la possibilità di elaborare altri pacchetti BEACON provenienti da altri nodi sink, prima che esso abbia ricevuto dal sink, sul quale si è sincronizzato, il numero massimo di CBEACON.
La struttura del pacchetto è rappresentata in figura (4.1).

- *PROBE*: pacchetto trasmesso dal nodo sensore in risposta al BEACON che è stato ricevuto da un qualsiasi nodo sink della rete. Questo messaggio è inviato dopo un tempo di backoff scelto nell'intervallo contenuto nel pacchetto BEACON ricevuto, e soltanto se il nodo sensore ha almeno un pacchetto DATA pronto per essere spedito.
  La struttura del messaggio, mostrata in figura (4.2), indica il contenuto del pacchetto PROBE. Due sono le informazioni che esso rende disponibili al sink: il numero di pacchetti DATA che vuole trasmettere durante la connessione, *num_data_pck*, e il tempo di backoff scelto da esso prima di inviare la risposta al BEACON, *bck_probe_before_tx*.
  Entrambe sono informazioni utili, la prima perché consentirà al nodo sink di porre un limite massimo ai pacchetti DATA che il nodo sensore può trasmettere, in modo tale che un solo nodo non si impossessi del canale oscurando tutti gli altri nodi della rete, la seconda sarà utile per il calcolo della distanza tra sensore e sink.

- *POLL*: pacchetto d'importanza fondamentale all'interno dell'economia per la creazione della connessione, poiché questo ha il compito di abilitare il

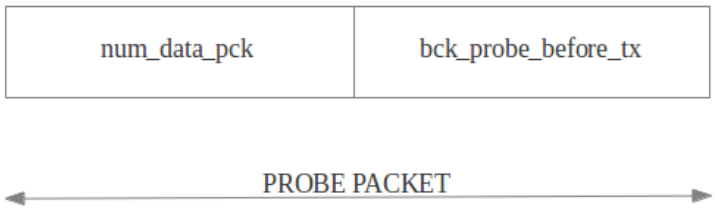


Figura 4.2: Struttura del pacchetto PROBE utilizzato per la creazione della connessione tra nodo sensore e nodo sink.

nodo sensore alla trasmissione dei propri pacchetti DATA promessigli con il PROBE inviatogli in precedenza.
Il POLL è quindi inviato in unicast al nodo sensore da cui il sink vorrà ricevere i pacchetti DATA.
La struttura rappresentata in figura (4.3) indica i due parametri inviati: il primo, *num_data_packet_want_rx*, denota il numero di pacchetti DATA massimo che il nodo sensore può trasmettere, mentre il secondo, *mac_addr_node_polled*, è l'indirizzo MAC del nodo sensore che dovrà analizzare tale pacchetto di POLL.
Per il motivo già spiegato durante l'illustrazione del pacchetto PROBE, il nodo sink pone un limite massimo di pacchetti DATA che vuole ricevere dal nodo sensore.
Il secondo parametro tra i due è quello di maggiore importanza dato che consente a tutti i nodi sensori in attesa del POLL, di intuire se è il proprio turno di trasmissione o meno, in caso positivo elaboreranno il messaggio, al contrario lo ignoreranno. Un valore errato di tale campo implica l'accesso alla trasmissione da parte di un nodo non autorizzato.

- *CBEACON*: pacchetto che è inviato dal nodo sink dopo aver ricevuto i pacchetti DATA da tutti quei nodi da cui aveva ricevuto il PROBE.
  Questo messaggio ha la stessa funzione del BEACON, cioé risvegliare i nodi sensori sotto il proprio raggio di copertura. A differenza del BEACON però, il CBEACON è elaborato solamente da quei nodi che, a seguito della ricezione del BEACON e della trasmissione del PROBE, non hanno avuto

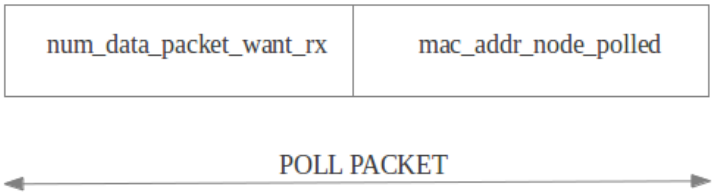


Figura 4.3: Struttura del pacchetto POLL utilizzato per la creazione della connessione tra nodo sensore e nodo sink.

la possibilità di trasmettere in quel ciclo i propri pacchetti DATA. Questo potrebbe accadere dal momento che il POLL trasmesso in risposta al PROBE non sia mai arrivato al destinatario, perché perso durante il tragitto a causa dell'elevato rumore del canale. Altro caso potrebbe essere quello in cui il timeout di attesa del POLL da parte del nodo sensore sia troppo stringente per le particolari condizioni di canae presenti all'atto della trasmissione, non consentendo così mai la ricezione del pacchetto.

La struttura di un CBEACON è uguale a quella di un BEACON, già rappresentata in figura (4.1).

La seconda comunicazione presente nel protocollo UFETCH è quella tra un nodo sink e l'AUV. In questo caso la messaggistica utilizzata per poter instaurare e trasmettere le informazioni raccolte dai nodi sink è quella di seguito presentata:

- *TRIGGER*: pacchetto che è inviato periodicamente dall'AUV in broadcast per poter risvegliare i nodi sink che sono entro il proprio raggio di copertura. Il TRIGGER è dunque il pacchetto equivalente al BEACON presentato per la comunicazione nodo sensore-sink, infatti il suo obiettivo è quello di far capire ad un qualsiasi nodo sink che, da quell'istante in poi e per un certo periodo temporale, l'AUV sarà disponibile per la ricezione delle informazioni recuperate.

  La sua struttura è raffigurata in figura (4.4). Come si può vedere, il TRIGGER è un pacchetto di segnalazione che comunica l'estremo inferiore e superiore, (t_min_bck÷t_max_bck), dell'intervallo temporale dal quale il

Figura 4.4: Struttura del pacchetto TRIGGER utilizzato per la creazione della connessione tra nodo sink e AUV.

nodo sink dovrà poi scegliere, in maniera aleatoria, il valore di attesa prima di trasmettere il pacchetto RTS di risposta.

- *RTS*: pacchetto trasmesso dal nodo sink che ha ricevuto correttamente il messaggio di TRIGGER. Anche in questo caso, come si era descritto per il pacchetto PROBE nella comunicazione nodo sensore-sink, l'RTS è inviato solo se il nodo in questione ha almeno un pacchetto DATA all'interno della propria memoria e pronto per la trasmissione.

  La figura (4.5) rappresenta la struttura del pacchetto RTS.
  Le informazioni da esso inviate sono solamente due: il numero di pacchetti DATA che il nodo sink sarebbe disposto a trasmettere all'AUV, *num_data_pck*, cioé tutti quei pacchetti che attualmente ha in memoria e sono già stati elaborati dal proprio processore e, il tempo di backoff scelto prima di iniziare la trasmissione dell'RTS appunto, *bck_rts_before_tx*. Quest'ultima informazione è poi utilizzata dall'AUV per il calcolo del RTT e successivamente per stabilire la lunghezza del timeout entro il quale ricevere tutti i pacchetti DATA concordati.

- *CTS*: messaggio che conclude la fase di instaurazione della connessione tra sink ed AUV. Questo pacchetto di segnalazione è trasmesso dall'AUV in modalità unicast al primo nodo sink da cui ha ricevuto il pacchetto RTS in maniera corretta.
  Lo scopo di tale pacchetto è quello di dare l'abilitazione al nodo sink per la trasmissione dei pacchetti DATA, per questo motivo, come rappresentato in figura (4.6), deve contenere i seguenti due campi: *num_data_packet_want_rx*,

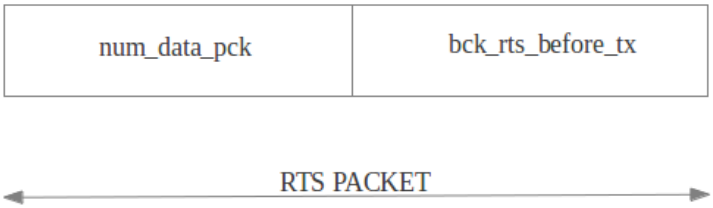


Figura 4.5: Struttura del pacchetto RTS utilizzato per la creazione della connessione tra nodo sink e AUV.

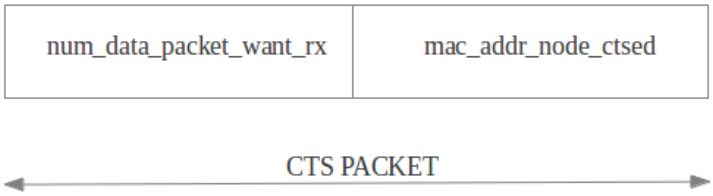


Figura 4.6: Struttura del pacchetto CTS utilizzato per la creazione della connessione tra nodo sink e AUV.

che indica la quantità di pacchetti massima che il nodo sink dovrà trasmettere, e *mac_addr_node_ctsed*, ovvero l'indirizzo MAC del nodo sink che avrà l'abilitazione alla trasmissione.
Come nel caso del pacchetto POLL, il secondo campo dell'header è di fondamentale importanza, dato che un eventuale errore causa l'abilitazione all'invio dei pacchetti DATA ad un nodo non appropriato, con conseguente perdita di tutte le informazioni.

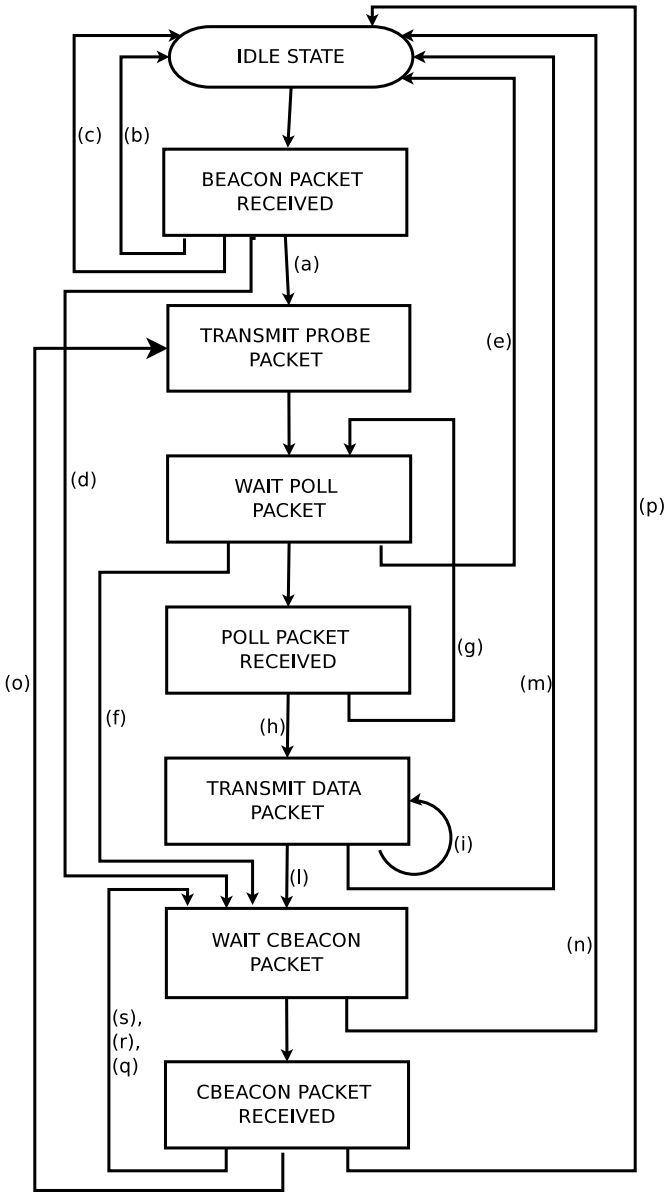
In conclusione, sopra sono stati descritti tutti i pacchetti che sono coinvolti dal protocollo UFETCH e, dato che sono pacchetti di segnalazione, sono tutti trasmessi alla bit rate minima utilizzata dal sistema.

4.1.4 Macchina a stati del nodo sensore

I nodi sensori, come già detto nella sezione 4.1.1, sono il cuore del protocollo UFETCH, dato che hanno il compito di recuperare la maggior quantità di in-

formazioni possibile dall'area circostante in cui essi sono installati. Solitamente la regione di osservazione assegnata a ciascun nodo ha un raggio di estensione all'incirca pari a 1500 m.
Periodicamente inoltre, i nodi sensori sono interpellati dal nodo sink a cui afferiscono e al quale dovranno trasmettere tutte le proprie informazioni raccolte dall'ultima volta che sono stati interrogati.

La figura (4.7) presenta i vari stati in cui un nodo sensore può trovarsi durante la sua fase di attività.
Si ipotizzi inizialmente che il nodo sia in *idle state*, ovvero nello stato in cui monitora l'ambiente e/o organizza le informazioni raccolte che dovranno poi essere trasmesse al nodo sink. In un qualsiasi momento della sua permanenza in *idle state*, il nodo sensore può essere interpellato dal nodo sink, affinché possa iniziare la creazione della connessione che porterà poi alla trasmissione dei dati a quest'ultimo. A tal proposito, quando il nodo sensore riceve un pacchetto di tipo BEACON, intuisce la richiesta del nodo sink di voler instaurare una comunicazione, quindi se in quell'istante è abilitato alla ricezione, esso inizia ad elaborare tale pacchetto entrando di fatto nello stato *beacon packet received*.
La comunicazione tra sink e nodo sensore può avvenire solamente se la distanza dal sink è inferiore a 1500 m, se ciò non è verificato, (b), allora il nodo ignora il BEACON e ritorna in *Idle state*. Allo stesso modo, il nodo rientra nel medesimo stato nel momento in cui: la condizione relativa alla distanza è verificata, ma non ha dati in memoria attualmente disponibili da potergli trasmettere e ha ricevuto il numero massimo di pacchetti di tipo CBEACON da parte del sink, (c). Un secondo caso in cui il pacchetto di BEACON è ignorato è quello in cui il nodo è ad una distanza dal nodo sink inferiore ai 1500 m, non ha informazioni da trasmettergli, ma non ha ancora ricevuto il numero massimo di pacchetti CBEACON dal sink. In questo caso il nodo sensore passa allo stato *Wait cbeacon packet* all'interno del quale si ripone in attesa, per un tempo pari a ($T\_wait\_CBEACON$), di un pacchetto di CBEACON, transizione (d).
Nel momento in cui nessuna delle condizioni *(b)-(c)-(d)* è verificata, allora il nodo ha la possibilità di instaurare una comunicazione con il nodo sink, (a), entra quindi nello stato *Transmit probe packet*, dove costruirà il pacchetto di *PROBE* da trasmettere come risposta al sink. Questo pacchetto sarà trasmesso solo dopo un



LEGENDA:
(a) distance SENSOR->HN <= 1500[m]
&& Node has at least one data pck to tx;
(b) distance SENSOR->HN >= 1500[m];
(c) distance SENSOR-HN <= 1500[m]
&& Node has 0 data pck to tx
&& Node has rx max allowed cbeacon pcks;
(d) distance SENSOR->HN <= 1600[m]
&& Node has 0 data pck to tx
&& Node hasn't rx max allowed cbeacon pcks;
(e) Poll timeout expired
&& Node has rx max allowed cbeacon pcks;
(f) Poll timeout expired
&& Node hasn't rx max allowed cbeacon pcks;
(g) Poll is not addressed to this node;
(h) Poll is addressed to this node;
(i) Node has rx max allowed cbeacon pcks;
(l) Node has tx last data pck
&& Node hasn't rx max allowed cbeacon pcks;
(m) Node has tx last data pck
&& node has rx max allowed cbeacon pcks;
(n) Cbeacon timeout expired;
(o) Cbeacon pck is addressed to this node
&& node hasn't rx max allowed cbeacon pcks
&& node hasn't yet tx his data pcks in previous cycle
&& node has at least one data to tx;
(p) Cbeacon is addressed to this node
&& node has rx max allowed cbeacon pcks;
(q) Cbeacon pck is addressed to this node
&& node hasn't rx max allowed cbeacon pcks
&& node has already tx his data pcks in previous cycle;
(r) Cbeacon is addressed to this node
&& node hasn't rx max allowed cbeacon pcks
&& node has 0 data pck to tx
&& node has not yet tx his data pcks in previous cycle;
(s) Cbeacon pck is not addressed to this node;

Figura 4.7: Macchina a stati che gestisce il comportamento di un nodo sensore durante la comunicazione con un nodo sink.

tempo di backoff scelto in modo aleatorio nell'intervallo [1÷6] s. L'attesa prima della trasmissione del PROBE riduce la probabilità che pacchetti di due o più nodi sensori tramessi collidano, dato che avevano ricevuto il BEACON nello stesso istante temporale.
Terminata la trasmissione del PROBE, il nodo sensore si ripone nello stato di *Wait poll packet*, all'interno del quale attende per un intervallo temporale pari a *T_wait_POLL* la risposta al pacchetto inviato, che è sotto forma di un nuovo pacchetto denominato *POLL*.

La durata del timer entro il quale il nodo sensore si aspetta di ricevere il pacchetto POLL, cioé *T_wait_POLL*, deve essere almeno pari al tempo necessario affinché il pacchetto PROBE si propaghi verso il nodo sink e quest'ultimo spedisca il messaggio di risposta, quindi almeno $(2RTT + T_{tx,PROBE} + T_{tx,POLL} + NT_{tx,DATA})$ s, con RTT il round trip time, $T_{tx,PROBE}$, $T_{tx,POLL}$ e $T_{tx,DATA}$ rispettivamente i tempi necessari per l'invio dei pacchetti PROBE, POLL e DATA. Questo timer però è molto stringente poiché il nodo sink, come sarà descritto più avanti, non risponde immediatamente con un POLL alla ricezione del PROBE, ma lo fa solamente dopo un periodo di durata T_wait_PROBE in cui ha atteso tutti gli eventuali PROBE trasmessi dai nodi sensori. Questo significa che deve valere la seguente diseguaglianza: T_wait_POLL $\geq 2RTT + T_{tx,PROBE} + T_{tx,POLL} \geq$T_wait_PROBE. Se entro tale tempo di attesa il nodo sensore non riceve alcun messaggio in risposta dal nodo sink, allora scatterà il *Poll timeout* e due saranno gli stati in cui al nodo sensore sarà acconsentito migrare.
La scelta dello stato in cui entrare alla scadenza del timeout di attesa del POLL, dipende dal numero di pacchetti CBEACON che il nodo ha ricevuto fino a quell'istante dal sink, nel caso in cui il numero recepito sia quello massimo, (e), esso ritorna in *Idle state*, contrariamente, (f), si pone in attesa di un CBEACON entrando nello stato *Wait cbeacon packet*.
Il nodo sensore durante la sua fase di attesa, se il timer T_wait_POLL è scelto abbastanza grande, riceverà un pacchetto di POLL dal nodo sink (cui aveva in precedenza inviato il PROBE) entrando così nello stato di *poll packet received* figura in (4.7). Qui il nodo analizza il pacchetto e verifica effettivamente se sia destinato o meno a sè; in caso negativo il nodo sensore ignora il *POLL* e ritorna nello stato *Wait poll packet*, transizione (g), mentre, in caso positivo, passa allo

stato *Data transmit packet*, transizione (h), dove inizierà la preparazione per la trasmissione dei pacchetti DATA.
Nel primo caso, prima del ritorno in *Wait poll packet*, il nodo sensore fa ripartire il timer di attesa del POLL, la lunghezza dell'intervallo sarà sempre pari al valore T_wait_POLL calcolato al suo primo ingresso in tale stato. Il motivo di questo azzeramento e ripartenza del timer, deriva dal fatto che il nodo sink può dare il consenso alla trasmissione dei pacchetti DATA ad un solo nodo sensore alla volta: quindi, se il PROBE relativo al nodo sensore è stato ricevuto dal sink dopo altri k PROBE, il nodo avrà la possibilità di trasmettere solamente dopo che $(k-1)$ nodi abbiano concluso la loro comunicazione. Il tempo necessario affinché questi $(k-1)$ utenti finiscano il loro invio dei dati è sicuramente maggiore del timeout impostato dal nodo sensore, questo significa una sicura scadenza del timer. Ciò comporta costi onerosi in termini di prestazioni, la scelta quindi di far ripartire il timer ad ogni POLL ricevuto consente di diminuire il problema espresso.

All'ingresso nello stato *Data transmit packet*, il nodo sensore inizia a trasmettere le informazioni che esso ha memorizzato a partire dall'ultima comunicazione avvenuta con il nodo sink.
In maniera sequenziale il nodo inizia a trasmettere i pacchetti *DATA* verso il sink; il numero di messaggi che saranno spediti è stato concordato tra i due estremi comunicanti durante la fase di scambio dei pacchetti PROBE e POLL. In figura (4.7) il procedimento è mostrato dalla transizione (i). Si osservi che il protocollo UFETCH in questo stadio iniziale di sviluppo non implementa forme di ARQ, in conseguenza di ciò, eventuali pacchetti persi e/o corrotti dal canale non saranno ritrasmessi.

Una volta che tutti i pacchetti DATA sono stati trasmessi dal nodo sensore, esso entra in uno dei seguenti due stati: *Wait cbeacon packet*, transizione (l), oppure *Idle state*, transizione (m). Il passaggio da (i) a (l) avviene quando il nodo sensore non ha ancora ricevuto il numero massimo di CBEACON che il nodo sink potrebbe inviargli, contrariamente, il passaggio da (i) a (m) si verifica quando sarà ricevuto dal sink il numero di pacchetti concordato durante lo scambio dei messaggi BEACON-PROBE-POLL.

Ipotizzando un ingresso del nodo sensore in *Wait cbeacon packet*, come già espresso in precedenza, esso qui attende un pacchetto di CBEACON fino ad un

massimo di (*T_wait_CBEACON*). Se, per ragioni di interferenza introdotta dal canale, il timer dovesse scadere, allora il nodo sensore ritorna allo stato di partenza *Idle state*, (*n*), da dove inizia nuovamente a monitorare la zona marina e attendere eventuali nuovi pacchetti di BEACON.
Diversamente, il nodo sensore entra in *Cbeacon packet received* nel momento in cui riceve un pacchetto CBEACON. Anche in tal caso si avranno varie opzioni tra cui il nodo potrà scegliere sulla base della sua attività passata. Infatti, se il nodo sensore a seguito di un BEACON oppure CBEACON antecedenti a quello ricevuto, ha già trasmesso una volta le proprie informazioni raccolte, allora nonostante abbia pacchetti DATA pronti da trasmettere è inibito alla trasmissione, ignorando di fatto il CBEACON ricevuto. Con questa tecnica è data priorità a tutti quei nodi che, per cause di interferenza o troppa congestione di traffico, non hanno ancora una volta inviato i propri pacchetti DATA al nodo sink a cui afferiscono. Quindi se il CBEACON viene ignorato, il nodo sensore ritorna nuovamente in *Wait cbeacon packet*, (*q*), se ha già ricevuto il numero massimo di CBEACON dal nodo sink, diversamente passa in *Idle state*, (*p*), rendendosi libero per una nuova comunicazione.
In altri casi, il CBEACON potrebbe essere non considerato dal nodo sensore. Il primo sicuramente è quello in cui il nodo abbia ricevuto un CBEACON che non sia ad esso indirizzato, in tale situzione ritorna nello stato *wait cbeacon packet*, (*s*). Un secondo caso potrebbe essere quello in cui il nodo abbia ricevuto il CBEACON correttamente, ma in quell'istante non abbia pacchetti DATA pronti da trasmettere, allora il nodo sensore farà ritorno allo stato *wait cbeacon packet*, (*r*).

Se nessuna delle condizioni che portano ad effettuare le transizioni *(p)-(q)-(r)-(s)* è verificata, allora questo implica che: il pacchetto CBEACON è stato correttamente ricevuto, il nodo deve ancora trasmettere i propri DATA al sink ed effettivamente in coda esso ha informazioni disponibili da trasmettere. In questo caso il nodo entra nello stato *Transmit probe packet*, (*o*), dando così via ad un nuovo ciclo di invio dati.

In conclusione, il nodo sensore continua per tutta la sua esistenza (durata della simulazione nel nostro caso) la sequenza di operazioni sopra indicate e rappresentate in figura (4.7), intervallando fasi di raccolta dati dall'area in cui esso è posizionato, a fasi di comunicazione con il sink, l'unica che gli è permessa essendo

inibito a comunicare direttamente con l'AUV, consegnandogli le informazioni precedentemente raccolte.

4.1.5 Macchina a stati del nodo sink

Il nodo sink è l'elemento più complesso del protocollo UFETCH dato che deve gestire due comunicazioni differenti, la prima verso i nodi sensori, la seconda verso il nodo AUV. Le due comunicazioni citate hanno priorità differenti: nel momento in cui il sink sia libero da qualsiasi compito, se gli viene concessa la possibilità di comunicare con l'AUV, e al tempo stesso è tentato a raccogliere i dati dai nodi sensori, esso privilegia la comunicazione con l'AUV, dando di fatto ad essa maggiore priorità.

Per descrivere il comportamento del nodo sink si farà riferimento alle due macchine a stati rappresentate in figura (4.8) e (4.9), la prima valida nel caso di comunicazione tra nodo sensore e sink, la seconda nel caso di comunicazione tra sink ed AUV.

Si consideri dapprima la comunicazione tra sink e nodi sensori, figura (4.8).
In uno scenario reale, ed uguale a quello utilizzato per le simulazioni presentate nel capitolo successivo, si ha che con l'utilizzo del nodo sink la rete è di fatto suddivisa in cluster, dove con il termine cluster si intende un sottoinsieme di nodi sensori posti entro un certo raggio di distanza dal sink, i quali possono inviare i propri dati raccolti al sink a cui sono collegati. Normalmente un nodo sink ha sotto la propria gestione tutti quei nodi che sono entro un raggio massimo da esso di 1500 m. Tale raggio potrebbe aumentare e il tutto dipende dalla potenza utilizzata per la trasmissione dei segnali acustici: più questa è alta, più il raggio aumenterà e maggiore sarà il numero di nodi in esso compreso.

Come già accennato nel sottoparagrafo 4.1.1, anche il nodo sink, come un semplice nodo sensore, effettua operazioni di monitoraggio raccogliendo quindi informazioni dall'area circostante in cui è installato.

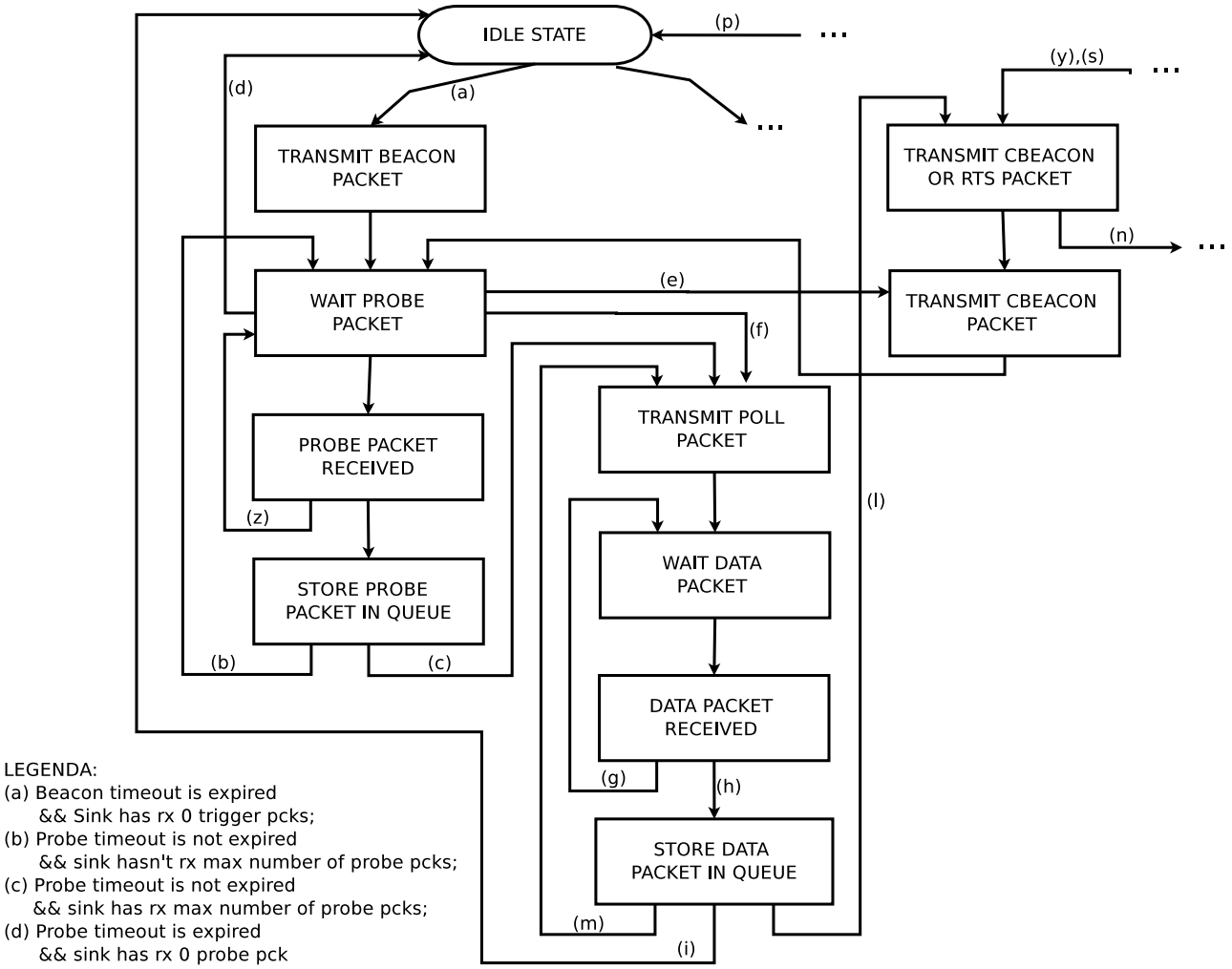
Ipotizzando che inizialmente il nodo sink si trovi nello stato *Idle state*, raffigurato in (4.8), qui il nodo, oltre a recuperare ed elaborare informazioni come accennato poco prima, deve essere pronto per iniziare una eventuale comunicazione, o con i nodi sensori che sono sotto il suo controllo, o con il nodo AUV, nel caso questo sia

nelle sue vicinanze.
Si osservi che l'istante più opportuno per iniziare la comunicazione con i nodi sensori è compito del nodo sink, mentre l'avvio della comunicazione con l'AUV non è a suo carico, infatti, quest'ultima comunicazione inizia solamente a seguito della ricezione di un particolare pacchetto inviato dall'AUV denominato TRIGGER.
L'attenzione, in questa prima parte del paragrafo, è rivolta alla descrizione della comunicazione tra sink e nodi sensori.

Dal momento che il nodo sink è in *Idle state*, dopo aver trascorso in tale stato un tempo pari a (*T_start_tx_SN*), e a seguito di nessuna ricezione di un pacchetto di tipo TRIGGER proveniente dall'AUV, il sink intuisce che deve richiedere le informazioni raccolte dai nodi sensori. A tal proposito, il sink entra nello stato *Transmit beacon packet*, (*a*), all'interno del quale costruisce il pacchetto denominato *BEACON*, che sarà poi trasmesso in broadcast con una potenza tale da raggiungere i nodi sensori entro un raggio R. Nella fase di simulazione, valori tipici utilizzati per *T_start_tx_SN* ed R saranno rispettivamente 40 s e 1600 m.
Una volta terminato l'invio del *BEACON*, il nodo sink entra nello stato *Wait probe packet*, dove si pone in attesa di tutti i pacchetti *PROBE* provenienti dai nodi sensori, che a loro volta, hanno ricevuto ed accettato con successo il BEACON. L'intervallo di attesa prestabilito a priori e d'ora in poi indicato con (*T_wait_PROBE*), deve essere abbastanza grande da far sì che anche il nodo a distanza maggiore dal sink riesca a ricevere il BEACON, e a sua volta inviare il pacchetto di PROBE in risposta a questo. Il valore minimo deve essere quindi maggiore di (τ_{MAX}+$T_{tx,BEACON}$+$T_{tx,PROBE}$), con: τ_{MAX} il ritardo di propagazione massimo dato dal seguente rapporto R/c, con c velocità di propagazione del suono in acqua, $T_{tx,BEACON}$ e $T_{tx,PROBE}$ rispettivamente il tempo necessario per l'invio dei pacchetti di BEACON e PROBE.

Durante la permanenza in *Wait probe packet*, il nodo sink per ogni pacchetto PROBE ricevuto effettua le seguenti operazioni:

- entra nello stato *Probe packet received*, dove verifica che il pacchetto sia privo di errori e sia ad esso destinato controllando gli header del PROBE. Se una delle due condizioni non è verificata, allora il PROBE viene ignorato e il sink ritorna in *Wait probe packet* in attesa di altri pacchetti, transizione (*z*). In caso contrario, il sink passa al punto successivo;



LEGENDA:
(a) Beacon timeout is expired
&& Sink has rx 0 trigger pcks;
(b) Probe timeout is not expired
&& sink hasn't rx max number of probe pcks;
(c) Probe timeout is not expired
&& sink has rx max number of probe pcks;
(d) Probe timeout is expired
&& sink has rx 0 probe pck
&& sink has tx max allowed cbeacon pcks;
(e) Probe timeout is expired
&& sink has rx 0 probe pck
&&sink hasn't tx max allowed cbeacon pcks;
(f) Probe timeout is expired
&& sink has rx at least one probe pcks;
(g) Sink has rx a data pck with errors
&& another data pck can be rx by sink;
(h) Sink has rx a data pck with no errors
&& data pck is addressed to the sink;
(i) Sink has rx last data pck from the node
&& it has 0 other node to probbed
&& sink has tx max number of cbeacon pcks;
(l) Sink has rx last data pck from the node
&& it has 0 other node to probbed
&& sink hasn't tx max number of cbeacon pcks;
(m) Sink has rx last data pck from the node
&& it has other node to probbed;
(n) Data queue of the node is not empty;
(p) distance Sink->AUV >= 1500[m];
(o) Data queue is empty;
(s) CTS timeout has expired
&& sink had interrupted cycle before to tx cbeacon pck;
(y) Sink has tx last data pck to the AUV
&& sink had interrupted cycle before to tx cbeacon pck;
(z) Probe timeout is not expired
&& probe is not addressed to the sink;

Figura 4.8: Macchina a stati che gestisce il comportamento della comunicazione tra nodo sink e nodo sensore.

- memorizza il pacchetto di PROBE con tutte le sue informazioni all'interno di una coda denominata Q_{PROBE}, operazioni svolte nello stato *Store probe packet*. Tale coda contiene informazioni del tipo: MAC address del nodo sensore mittente del PROBE, numero di pacchetti che il nodo sensore vorrebbe trasmettere e infine il tempo di backoff utilizzato dal nodo sensore prima di effettuare la trasmissione del PROBE. Al termine dell'immagazzinamento delle informazioni il nodo sink ritorna nello stato di attesa *Wait probe packet*, transizione (b).

Il ciclo sopra descritto può terminare, cioé il sink abbandona lo stato *Wait probe packet*, in due particolari situazioni: la prima quando il nodo sink riceve il numero massimo di PROBE stabilito in fase di progetto, la seconda, alla scadenza del timer *T_wait_PROBE*.
Nel primo caso, nonostante il timeout *T_wait_PROBE* non sia scaduto, il sink non può ricevere altri PROBE, probabilmente perché questo implicherebbe un'eccessiva quantità di DATA che dovranno essere poi trasmessi dai nodi sensori e che non potrebbero essere memorizzati dal sink non avendo quest'ultimo memoria illimitata. Al verificarsi di tale situazione, il timer *T_wait_PROBE* viene azzerato e il nodo sink entra nello stato di *Transmit poll packet*, transizione (f).
Nel secondo caso invece, cioé alla scadenza del tempo di attesa *T_wait_PROBE*, il nodo sink, sulla base del proprio stato interno, effettua una delle seguenti tre transizioni:

1. ritorna in *Idle state*, (d), se nessun pacchetto PROBE durante la sua attesa è stato ricevuto, cioé in tale fase di simulazione nessun nodo sensore ha dati disponibili da trasmettere, e al tempo stesso il sink ha già trasmesso il numero massimo di pacchetti CBEACON per tale ciclo di esecuzione;

2. migra in *Transmit cbeacon packet*, (e), se nessun pacchetto PROBE è stato ricevuto, ma il sink ha la possibilità di trasmettere ancora almeno un pacchetto di tipo CBEACON;

3. passa allo stato *transmit poll packet*, (c), se ha ricevuto almeno un pacchetto di PROBE dai nodi sensori, cioé almeno uno di questi ha raccolto informazioni che devono ancora essere ricevute dal sink;

Assumendo per il momento che una delle due condizioni (c) o (f) siano verificate, allora questo implica che il nodo sink entra nello stato *Transmit poll packet* come sopra accennato. In questo stato viene creato il pacchetto POLL, che verrà poi trasmesso in unicast al solo nodo sensore dal quale il sink vuole ricevere i pacchetti DATA ad esso notificati tramite il PROBE. Solitamente il POLL è inviato al primo nodo presente nella coda Q_{PROBE}: in questo modo, la priorità di ricezione dei pacchetti DATA è organizzata sulla base dell'arrivo dei PROBE al nodo sink. In altre parole sono ricevuti prima i pacchetti DATA dai nodi sensori più vicini al nodo sink. Altre tecniche potrebbero essere scelte, a discrezione del progettista del protocollo e dalle esigenze delle applicazioni supportate dalla rete.

Al termine dell'invio del POLL, il nodo sink entra quindi nello stato di *Wait data packet*, all'interno del quale si rende disponibile per la ricezione dei pacchetti DATA provenienti dal solo nodo che esso ha abilitato alla trasmissione.
Affinché il sistema non entri in stallo, il nodo sink calcola, a seguito del POLL inviato, il tempo massimo entro il quale vorrà ricevere tutti i pacchetti DATA promessi dal nodo sensore, timer indicato con (T_wait_DATA). Questo è calcolato tenendo conto: del numero di pacchetti che il nodo sensore deve trasmettere, n_{PCK},valore noto dal sink perché contenuto nell'header del PROBE, del tempo di guardia T_{GUARD}, del RTT ed infine del tempo necessario per trasmettere il pacchetto DATA, cioé ($T_{tx,DATA}$).
Riassumendo si ha che: T_wait_DATA=n_{PCK}(T_wait_DATA+T_{GUARD}+RTT).

Durante la fase di attesa, avente durata (T_wait_DATA), il nodo sink è disponibile alla ricezione dei pacchetti DATA provenienti dal nodo sensore che esso ha abilitato con l'invio del POLL.
Quindi, per ogni pacchetto DATA ricevuto, il sink effettua le successive operazioni passando attraverso i seguenti stati:

1. in principio entra nello stato *Data packet received*, all'interno del quale il sink verifica che: il pacchetto non sia corrotto e sia effettivemente proveniente dal nodo sensore a cui in precedenza aveva inviato il pacchetto POLL. Se solo una delle due condizioni non è verificata, allora il nodo sink ritorna nello stato *Wait data packet* in attesa di un altro pacchetto DATA, transizione (g). Si osservi che tecniche di ritrasmissione ARQ non sono qui implementate,

quindi eventuali pacchetti corrotti non saranno re-inviati;

2. se entrambe le condizioni in (1) sono state accertate, allora il sink passa allo stato *Store data packet*, (h), dove memorizza le informazioni del pacchetto all'interno di una particolare coda, denominata Q_{DATA}. I dati in ivi contenuti saranno poi trasferiti all'AUV, a seguito di una sua richiesta.

Il ciclo *Wait data packet - Data packet received - Store data packet* sarà effettuato per un numero di volte pari al numero di pacchetti DATA che il nodo sensore aveva promesso con l'invio del PROBE al nodo sink, oppure fintantoché il timer T_wait_DATA non scade.

Il secondo caso, ovvero la scadenza del timer, può essere dovuta a diverse cause, ad esempio il timer calcolato potrebbe essere troppo stringente per le attuali particolari condizioni di canale e di traffico, di conseguenza non tutti i pacchetti arrivano al ricevitore. Potrebbe essere che il POLL non sia mai stato ricevuto, o ricevuto con errori dal nodo sensore, quindi quest'ultimo non inizierà mai l'invio dei propri pacchetti DATA. Tutti questi elementi comportano la scadenza del timeout T_wait_DATA. Al verificarsi di questa situazione, il sink, sulla base del proprio stato interno, può scegliere tra tre possibili operazioni:

1. ritornare allo stato iniziale *Idle state*, se la coda Q_{PROBE} è vuota, ovvero altri nodi sensori da cui ricevere dati non sono presenti, e ha già trasmesso il numero massimo di pacchetti di CBEACON per l'attuale ciclo di esecuzione;

2. passare allo stato *Transmit cbeacon or rts packet*, se la coda Q_{PROBE} è vuota, ma ha la possibilità di trasmettere almeno un pacchetto CBEACON;

3. rientrare nello stato *Transmit poll packet* se almeno un elemento è ancora presente nella coda Q_{PROBE}, ovverosia esiste un nodo sensore da cui il sink ha ricevuto il pacchetto PROBE e dal quale vi deve ancora ricevere i pacchetti DATA;

Ritornando al caso più comune, cioè quello di ricezione completa di tutti i pacchetti DATA, il sink può nuovamente scegliere tra tre distinte opzioni:

1. ritornare nello stato iniziale *Idle state*, (i), se: la coda Q_{PROBE} non contiene nessun altro elemento, cioé nessun altro nodo ha avanzato la proposta di

trasmissione dei propri pacchetti DATA, e il nodo sink ha trasmesso il numero massimo di pacchetti CBEACON per l'attuale ciclo di esecuzione;

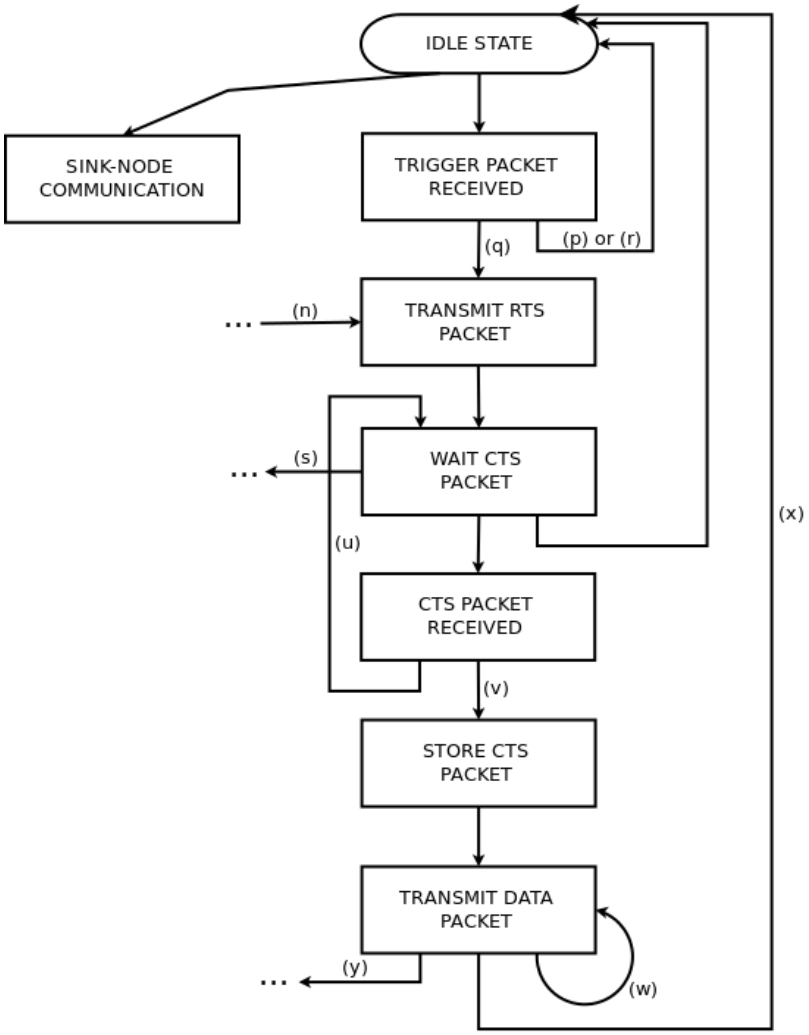
2. andare nello stato *Transmit cbeacon or rts packet*, (l), se: la coda Q_{PROBE} non contiene nessun altro elemento, ma il nodo sink ha ancora la possibilità di inviare almeno un pacchetto CBEACON;

3. passare allo stato *transmit poll packet*, (m), nel caso in cui almeno un nodo sensore da cui ricevere pacchetti DATA è ancora presente all'interno della coda Q_{PROBE}. In tal caso saranno effettuate tutte le operazioni descritte fino ad ora per le transizioni (c), (f).

I casi a seguito delle transizioni (i) ed (m) sono già stati analizzati in precedenza, si consideri invece ora la contingenza (l) oppure (e).
L'ingresso in *Transmit cbeacon or rts packet* è una sorta di spartiacque, nel senso che è l'unica situazione in cui il nodo sink può risvegliare l'AUV chiedendone la possibilità di trasmettere i propri pacchetti DATA ricevuti dai nodi sensori. Infatti, prima di trasmettere un pacchetto CBEACON ai nodi sensori, il sink verifica se l'AUV in quell'istante è libero da qualsiasi altra operazione. La richiesta è effettuata attraverso la trasmissione di un pacchetto RTS e solamente se il nodo sink ha pacchetti DATA all'interno della coda Q_{DATA}. In tal caso il sink entra nello stato *Transmit rts packet* raffigurato in (4.9), (n), e la comunicazione con l'AUV può avvenire solo se quest'ultimo riceverà correttamente il pacchetto RTS. Le fasi della comunicazione saranno successivamente illustrate nel dettaglio.

Tralasciando per un momento la comunicazione sink-AUV, il nodo sink a seguito delle condizioni (l) ed (e), prosegue la propria esecuzione entrando nello stato *Transmit cbeacon packet*, (o), all'interno del quale costruisce un pacchetto CBEACON che è poi trasmesso in broadcast a tutti i nodi sensori entro il proprio raggio di copertura.
A conclusione della trasmissione del CBEACON, il nodo sink ritorna nello stato di *Wait probe packet*, dove inizia nuovamente l'intero ciclo per l'attesa dei pacchetti di PROBE provenienti dai nodi sensori e successivamente la raccolta dei pacchetti DATA.

Come già detto all'inizio del paragrafo, il nodo sink oltre a gestire la comunicazione con i nodi sensori, deve anche interfacciarsi con il nodo AUV. A tal



LEGENDA:
(n) Data queue of the node is not empty;
(p) Diastance SINK->AUV >= 1500[m];
(r) Sink has data queue empty;
(s) CTS timeout is expired
&& sink has interrupted cycle before
to tx cbeacon pck;
(t) CTS timeout is expired
&& sink had not interrupted cycle before
to tx cbeacon;
(u) CTS packet is not addressed to the sink
&& cts packet is with errors;
(v) CTS is not addressed to the sink
&& CTS packet it's not with errors;
(w) Sink continue until it has not tx maximum
number of data pcks;
(y) Sink has tx last data pck to the AUV
&& sink had interrupted cycle before
to tx cbeacon pck;
(x) Sink has x last data pck
&& sink had not interrupted cycle before
to tx cbeacon;

Figura 4.9: Macchina a stati che indica tutte le possibili fasi in cui un nodo sink può trovarsi durante la comunicazione con il nodo AUV.

proposito, la figura (4.9) mostra in quali stati il sink può trovarsi durante la comunicazione con l'AUV.

L'AUV periodicamente richiede le informazioni recuperate dai nodi sink; una tale richiesta è effettuata mediante l'invio di un pacchetto speciale denominato TRIGGER. Alla ricezione di tale pacchetto, il sink intuisce la volontà da parte dell'AUV di ricevere i pacchetti DATA. L'accettazione del TRIGGER può avvenire, da parte del nodo sink, solamente durante una sua permanenza in *Idle state*, dove rimane per un tempo pari a (T_start_tx_SN).

Una seconda situazione che si può verificare, e che consente un successivo scambio di dati con l'AUV, è la trasmissione spontanea da parte del nodo sink di un pacchetto RTS, senza prima aver ricevuto un TRIGGER. Questo può accadere solamente quando il sink entra nello stato *Transmit cbeacon or rts packet*, transizione (l) di figura (4.8), dove prima di continuare la comunicazione con i nodi sensori, quindi trasmettere un nuovo CBEACON, esso tenta il risveglio dell'AUV.

Si consideri dapprima il caso in cui la comunicazione inizi solamente dopo la ricezione del pacchetto TRIGGER.
Sotto tale ipotesi, il nodo sink, nell'istante della ricezione del pacchetto, è in *Idle state* e quindi abilitato alla sua cattura. In tal caso il nodo sink entra in *Trigger packet received*, all'interno del quale verifica la correttezza del pacchetto. Se il TRIGGER è corrotto, oppure la coda Q_{DATA} è vuota, cioé nessun pacchetto DATA è pronto per essere trasmesso, allora il sink ritorna al suo stato iniziale, *Idle state* (transizioni rispettivamente (p) ed (r) di figura (4.9)); in tutti gli altri casi il pacchetto viene semplicemente ignorato e il nodo continua con l'operazione che attualmente sta eseguendo.

Se nessuna delle condizioni che portano alle transizioni (p) o (r) sono verificate, allora il nodo sink passa allo stato *Transmit rts packet*, (q), in cui costruisce il pacchetto RTS che sarà poi trasmesso dopo un tempo di backoff compreso nell'intervallo di estremi ($T_{min,tx,RTS} \div T_{max,tx,RTS}$).

All'interno dello stato *Transmit rts packet*, il nodo sink può arrivare a seguito anche della transizione (n), ovvero senza la ricezione del pacchetto TRIGGER, che coincide con la seconda modalità di instaurazione della comunicazione sink-AUV accennata poco sopra.
Questa tecnica è utilizzata dal nodo sink per aumentare le possibilità di comuni-

cazione tra sink ed AUV, dato che la probabilità che alla ricezione del TRIGGER il nodo sink sia abilitato alla sua elaborazione, cioè sia in *Idle state*, è molto bassa. Questa probabilità diminuisce all'aumentare del traffico di rete, ovvero all'aumentare del tasso di generazione dei pacchetti DATA da parte dei nodi sensori, infatti per più tempo il sink sarà occupato dalla comunicazione con i nodi sensori. In tal caso potrebbe accadere che i nodi sink accumulino una quantità elevata di informazioni, e non abbiano più memoria disponibile per immagazzinarne altre, dato che il suo svuotamento non avviene a causa delle poche volte con cui comunica con l'AUV. Inoltre più è alta la quantità di pacchetti DATA da trasmettere e maggiore è il tempo che l'AUV dovrà essere sotto il raggio di copertura del nodo sink. Questo secondo parametro non è impostabile a priori; l'AUV è un dispositivo che viaggia ad una velocità costante, rimanendo quindi entro il raggio di copertura di tutti i nodi sink per lo stesso intervallo temporale, in conseguenza di ciò il tempo disponibile per comunicare con esso è ripartizionato in maniera equa tra tutti i nodi sink presenti nella rete.

A conclusione della trasmissione del pacchetto RTS, il sink si ripone in uno stato di attesa, *Wait cts packet*, dove attende la ricezione di un pacchetto denominato CTS in risposta al proprio RTS.
L'attesa, onde evitare situazioni di stallo del sistema, non è illimitata, ma è pari a (*T_wait_CTS*). Il valore di tale timer, deve essere almeno pari alla somma del massimo tempo di propagazione del sistema τ_{MAX}, dato da R/c, con R la distanza dal nodo sink all'AUV, e dalla somma dei tempi di trasmissione dei pacchetti RTS e CTS, rispettivamente $T_{tx,RTS}$ e $T_{tx,CTS}$. Riassumendo si ha che T_wait_CTS $\geq$ (τ_{MAX}+$T_{tx,RTS}$+$T_{tx,CTS}$).

Due sono quindi le situazioni che si possono presentare:

1. il nodo sink riceve il pacchetto CTS entro l'intervallo di durata T_wait_CTS fatto partire dopo la trasmissione del pacchetto RTS;

2. il nodo sink non riceve alcun pacchetto CTS entro l'intervallo di durata T_wait_CTS, facendo così espirare il timer;

Nella prima ipotesi, il nodo sink entra nello stato *CTS packet received*, all'interno del quale verifica la correttezza del pacchetto. In caso negativo, ritorna nello stato

Wait cts packet, transizione (u), dove si ripone nuovamente in attesa del CTS corretto. In caso affermativo, il nodo sink entra nello stato *Store cts packet*, (v), dove memorizza tutte le informazioni in esso contenute, quali il MAC address del nodo AUV e il numero massimo di pacchetti DATA che potrà trasmettere.

Nella seconda ipotesi invece, alla scadenza del timer T_wait_CTS, il nodo sceglie lo stato in cui passare sulla base del proprio stato interno. Le possbilità sono:

1. una transizione in *Idle state*, (t), se lo stato da cui proveniva il sink era *Trigger packet received*. Da qui il sink può iniziare o una nuova comunicazione con i nodi sensori, sempre dopo un tempo T_wait_tx_SN, oppure instaurare un'ulteriore comunicazione con l'AUV;

2. una transizione verso *State cbeacon packet*, (s), se lo stato di provenienza era *Transmit cbeacon or rts packet*. Da qui il sink invia in broadcast un pacchetto CBEACON a tutti i nodi sensori entro il proprio raggio di copertura.

Ritornando al caso in cui il nodo sink fa ingresso nello stato *Store cts packet*, dopo la memorizzazione del pacchetto CTS, esso inizierà la vera e propria trasmissione dei pacchetti DATA con destinazione l'AUV. La procedura seguita dal sink è la seguente:

1. in principio entra in *Transmit data packet*, dove prepara il pacchetto DATA da trasmettere;

2. successivamente effettua la trasmissione del pacchetto DATA creato al punto (1);

3. al termine dell'invio, rientra al punto (1), transizione (w).

La trasmissione, transizione (w), ha luogo finché non saranno trasmessi tutti i pacchetti che esso ha in coda, oppure non ha raggiunto il numero massimo dettato dall'AUV e presente all'interno dell'header del pacchetto CTS da esso ricevuto.
Quando anche l'ultimo pacchetto DATA è stato spedito, equivalentemente al caso di scadenza del timer T_wait_CTS, il nodo sink può ritornare o in *Idle state*, (x), oppure inviare un nuovo CBEACON, ritornando di fatto nello stato da cui

era provenuto prima di iniziare la comunicazione con l'AUV, *State cbeacon packet*, transizione (y).

In conclusione il nodo sink, come si è potuto intuire dalla descrizione, ha un compito cruciale all'interno della rete, dato che deve gestire contemporaneamente due comunicazioni. Come si può inoltre notare dalle simulazioni presentate al capitolo successivo 5, è stato necessario effettuare una scelta oculata di tutti i timer, dato che potrebbero creare o dei colli di bottiglia in prossimità di tali entità, oppure aumentare di molto il ritardo di consegna end-to-end di un pacchetto DATA.

4.1.6 Macchina a stati del nodo AUV

Il terzo ed ultimo elemento che costituisce il protocollo UFETCH è l'AUV, cioé quell'entità che ha il compito di recuperare periodicamente le informazioni raccolte dai nodi sink, a loro volta acquisite dai nodi sensori distribuiti nella rete, e riportarle in una stazione di superficie dove saranno ulteriormente elaborati e condivisi con altre reti, probabilmente radio terrestri.

Il protocollo UFETCH, come nel caso di uw-POLLING, ipotizza che l'AUV sia un elemento mobile che viaggia ad una determinata velocità, variabile nell'intervallo [2÷4] nodi e che segue una prestabilita traiettoria, in modo da rientrare periodicamente all'interno del raggio di copertura di tutti i nodi sink della rete UWAN.
L'ottimizzazione della traiettoria e la scelta della velocità di spostamento, come si potrà osservare nel successivo capitolo 5, influiranno molto sulle prestazioni di sistema, sia in termini di tempo di consegna end-to end dei pacchetti DATA, sia in termini di packet error rate.

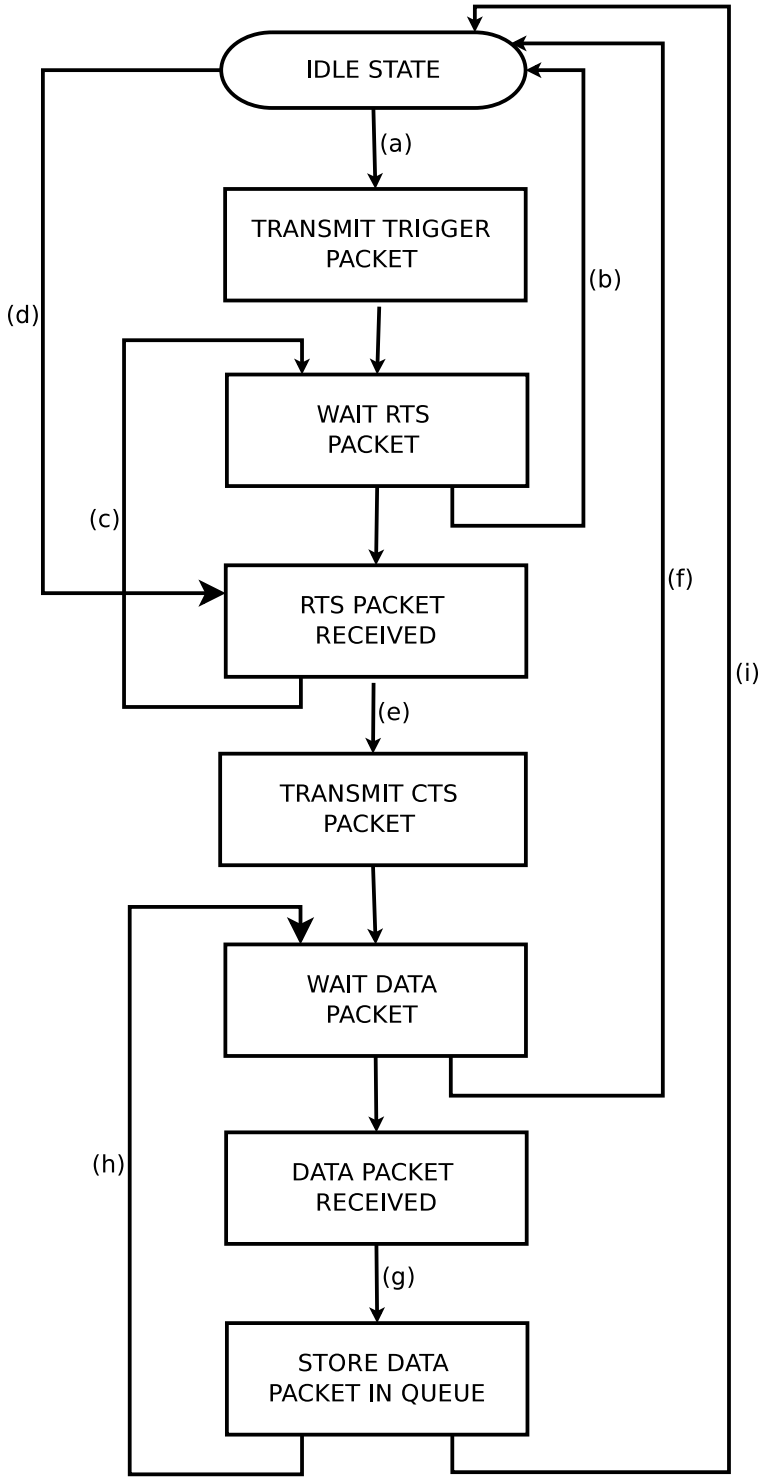
Di seguito, con l'aiuto anche della figura (4.10), saranno presentati tutti gli stati in cui l'AUV si può trovare durante la sua fase di operatività.

All'inizio dell'attività, l'AUV è nello stato base *Idle state* dal quale periodicamente tenta di stabilire una comunicazione con i nodi sink della rete. Per fare ciò, ciclicamente, ad intervalli regolari di (T_start_tx_AS), entra nello stato *Transmit trigger packet*, transizione (a). All'interno di questo, l'AUV crea un pacchetto denominato TRIGGER, che sarà poi trasmesso in broadcast a tutti i nodi della rete.

Una piccola osservazione deve essere fatta a riguardo del timer, infatti, nel sottoparagrafo 4.1.5 si è detto che il nodo sink attende i pacchetti TRIGGER, durante la sua permanenza in *Idle state*, per un tempo pari a (T_start_tx_SN), scattato tale timer esso dà inizio ad nuova una comunicazione con i nodi sensori. A tal proposito, affinché ci sia la possibilità che i nodi sink possano ricevere i TRIGGER, deve essere che T_start_tx_AS $\leq$ T_start_tx_SN, se tale diseguaglianza non è verificata, la comunicazione AUV-sink non può mai avvenire per mezzo di pacchetti TRIGGER.

Una volta conclusa la trasmissione del pacchetto TRIGGER, il nodo AUV si ripone in attesa di una risposta, ovvero di un pacchetto RTS, entrando nello stato *Wait rts packet*.
Onde evitare stalli di sistema, l'attesa, pari a (T_wait_RTS), non è illimitata, ma almeno pari alla somma tra il tempo di propagazione massimo della rete τ_{MAX}, dato dal rapporto R/c, con R la distanza tra l'AUV e il nodo sink a maggiore distanza raggiungibile da esso e c, la velocità del suono in acqua. Al ritardo di propagazione deve essere aggiunto il tempo di trasmissione del pacchetto TRIGGER e RTS, rispettivamente pari a $T_{tx,TRIGGER}$ e $T_{tx,RTS}$. In sostanza si ha che: T_wait_RTS $\geq (\tau_{MAX}+T_{tx,TRIGGER}+T_{tx,RTS})$.

Arrivato a tale fase, il nodo AUV può o ricevere nessun pacchetto RTS, perciò dopo (T_wait_RTS) dall'invio del TRIGGER scatterà il timeout relativo, oppure riceverne almeno uno e quindi proseguire con la comunicazione.
Nel primo caso, cioé alla scadenza del timeout, l'AUV ritorna nuovamente nello stato iniziale, *Idle state*, (b), dove attende ulteriori (T_start_tx_AS) prima di tentare di stabilire una nuova connessione con i nodi sink.
Nel secondo caso invece, una volta ricevuto il pacchetto RTS, l'AUV entra nello stato *Rts packet receive*, (c), dove controlla che il pacchetto sia privo di errori. In caso negativo, l'RTS viene ignorato e l'AUV ritorna in *Wait rts packet* dove rimane in attesa finché non scadrà il relativo timeout oppure riceve altri pacchetti RTS.
Se l'AUV riceve un pacchetto RTS senza aver trasmesso in precedenza un pacchetto TRIGGER, questo è accettato solamente se l'AUV è in *Idle state*. Se così non fosse, anche in questa situazione l'RTS viene ignorato, e il nodo continuerà con le operazioni che già stava eseguendo. In caso affermativo, il nodo AUV passa direttamente allo stato *Rts packet receive*, transizione (d).



LEGENDA:
(a) Trigger timeout has expired;
(b) RTS timeout has expired
&& AUV has rx 0 RTS pcks;
(c) RTS pck is in error;
(d) RTS received && TRIGGER packet
it was not transmitted;
(e) RTS pck is correct &&
&& AUV is enabled to received It;
(f) Data timeout has expired
(g) AUV has rx data pck with no errors
&& data pck is addressed to the AUV;
(h) AUV wait another data pck from SINK;
(i) AUV has rx the maximum data pck from SINK

Figura 4.10: Macchina a stati che indica tutte le possibili fasi in cui il nodo AUV può trovarsi durante la comunicazione con un nodo sink.

Se i controlli sopra citati si sono conclusi con successo, allora l'AUV memorizza tutte le informazioni in esso contenute, quali MAC address del nodo sink mittente e il numero di pacchetti DATA che il sink vorrebbe trasmettergli, azzera successivamente il timer T_wait_RTS e fa ingresso allo stato *Transmit cts packet*, transizione (e). All'interno di tale stato l'AUV crea il pacchetto denominato CTS, che sarà immediatamente trasmesso in unicast al nodo sink, da cui aveva in precedenza ricevuto il messaggio RTS.
A trasmissione avvenuta, l'AUV si pone in attesa dei pacchetti DATA che esso ha richiesto al sink, entrando nello stato *Wait data packet*. Il tempo di attesa deve essere calcolato tenendo conto: del numero di pacchetti che il nodo sink deve trasmettere, n_{PCK}, valore noto all'AUV perché contenuto nell'header del CTS; del tempo di guardia T_{GUARD}; del RTT; del tempo necessario per trasmettere il pacchetto DATA, cioé ($T_{tx,DATA}$). Indicando tale tempo con T_wait_DATA, si ha che: T_wait_DATA=n_{PCK}(T_wait_DATA+T_{GUARD}+RTT).

Alla ricezione di un pacchetto DATA, l'AUV passa dallo stato *Wait data packet* allo stato *Data packet received*, dove controlla il contenuto del pacchetto e se effettivamente esso proviene dal sink a cui in precedenza aveva inviato il CTS. Nel caso le due condizioni siano verificate, (g), il pacchetto DATA viene memorizzato in una particolare coda, denominata $Q_{DATA,AUV}$, gestita appunto dall'AUV.
Al termine della memorizzazione, il nodo AUV ritorna nello stato *Wait data packet* in attesa di altri pacchetti DATA, transizione (h). Solamente in uno specifico caso l'AUV ritorna nello stato iniziale *Idle state*, transizione (i), ovvero quando riceve l'ultimo pacchetto DATA dal nodo sink con cui stava comunicando.

Un caso particolare può verificarsi nel momento in cui l'AUV non riceve tutti i pacchetti DATA, o non ne riceve nemmeno uno dal nodo sink interpellato. In tale situazione, il timer T_wait_DATA scade e il nodo AUV semplicemente ritorna nello stato iniziale, *Idle state*, da dove successivamente potrà intraprendere una nuova comunicazione con un altro nodo sink. Le casistiche che portano alla scadenza del timer sono varie, infatti potrebbe essere che il nodo sink non riceva mai il pacchetto CTS trasmesso dall'AUV, conseguenza non inizia mai la trasmissione dei pacchetti DATA e quindi l'AUV non ne riceve nemmeno uno. Potrebbe essere inoltre che il timer calcolato T_wait_DATA sia troppo stringente per quelle particolari condizioni di canale, quindi non tutti i messaggi DATA riescono ad arrivare a

destinazione. A questi si potrebbero aggiungerne molti altri, che ora non staremo qui ad elencare.

In conclusione, il nodo AUV rispetto alle altre entità della rete che costituiscono il protocollo UFETCH, non ha molte operazioni da svolgere, l'unico suo scopo è appunto richiedere e memorizzare le informazioni dai nodi sink cercando di ottimizzarne il tempo di raccolta.

Capitolo 5

Simulazioni e risultati

L'obiettivo del seguente capitolo sarà quello di presentare una serie di grafici ottenuti a seguito di simulazioni in modo da poter dimostrare i pro e contro del protocollo UFETCH, descritto in maniera dettagliata nel capitolo 4. A tal proposito saranno effettuati alcuni confronti con altri protocolli quali uw-POLLING e MSUN, descritti nel capitolo 2.

L'analisi proposta di seguito si collega in parte con lo studio preliminare effettuato nel capitolo 3, estendendo i risultati ottenuti in essa.
Ciò che contraddistingue i due studi sono gli ambienti di simulazione: infatti nel primo caso, denominato analisi preliminare da qui in avanti, si era considerato un ambiente ideale privo di eventuali interferenze tra i nodi della rete, nel secondo caso invece, che sarà oggetto di discussione del capitolo seguente, si testerà il protocollo UFETCH in uno scenario reale.
Prima di essere immessa nel canale inoltre l'informazione è modulata BPSK.

Tutte le simulazioni sono state realizzate mediante il simulatore di reti ns-2, apposito programma utilizzato per simulare reti di telecomunicazioni e sviluppato presso l'University of Southern California's Information Sciences Institute (ISI). A questo sono state affiancate le librerie DESERT, sviluppate dall'Università degli studi di Padova all'interno del dipartimento di Ingegneria dell'informazione.
I risultati ottenuti sono stati poi elaborati dal programma MATLAB in modo da estrarre i grafici qui presenti nel capitolo.

Il capitolo è organizzato come segue: un primo breve sottoparagrafo è dedicato

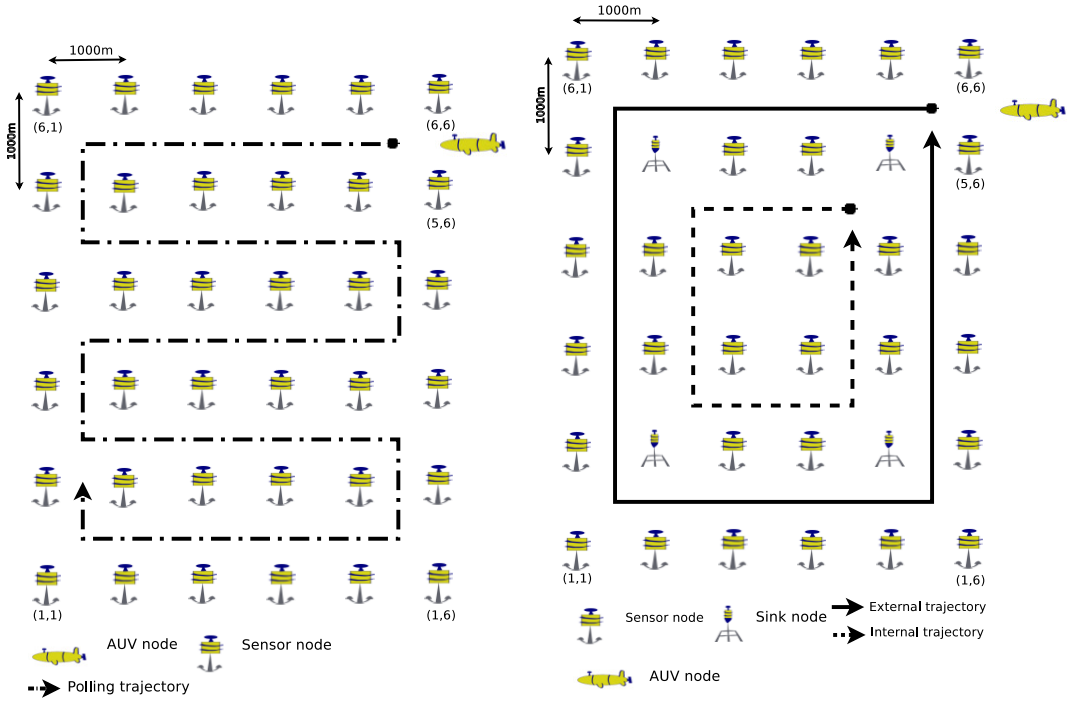
alla descrizione dello scenario di simulazione e dei parametri utilizzati per effettuare i test, i successivi quattro sottoparagrafi contengono una serie di grafici che consentono di evidenziare i pro e contro del protocollo UFETCH.

5.1 Ambiente e parametri di simulazione

La topologia di rete utilizzata per testare i protocolli MSUN, uw-POLLING e UFETCH è la stessa già descritta nel sottoparagrafo 3.2 del capitolo 3, ovvero 36 nodi sensori disposti su di una griglia di 6x6 elementi ognuno a distanza dai nodi ad esso perpendicolari di d=1000 m e ad una profondità di 50 m. In modo tale da rendere più realistiche le simulazioni, i nodi sono posizionati in maniera aleatoria, e mantenuti fissi in tale posizione per l'intera simulazione, all'interno di un cerchio di raggio 125 m con centro l'intersezione dei punti della griglia di dimensioni 6x6. Per comodità, si riporta lo scenario già discusso nel sottoparagrafo 3.2 in figura (5.1).

Come già indicato nei capitoli precedenti, nel caso di MSUN, il nodo AUV è fermo nella posizione indicata in figura (5.1), mentre nel caso di uw-POLLING e UFETCH, questo è considerato in movimento. Varie sono le traiettorie effettuate da tale nodo, le quali a loro volta si differenziano sulla base del protocollo utilizzato in modo da ottimizzare il tempo di raccolta dei pacchetti DATA. Si può già osservare dalla figura (5.1(b)) che, nel caso di UFETCH, il percorso eseguito è più breve rispetto a quello di uw-POLLING, questo perché l'AUV nel primo caso non ha la necessità di monitorare l'intera area, ma solamente i quattro nodi principali denominati sink. Nel caso di uw-POLLING, dovendo l'AUV invece interagire con ogni singolo nodo sensore per potergli richiedere le informazioni, necessita del monitoraggio dell'intera area ricoperta dalla rete.

Per quanto concerne la struttura dei nodi, siano essi semplici sensori o più complessi sink, ognuno implementa i sette livelli del modello ISO/OSI.
I pacchetti generati dai nodi sensori, quindi dal loro livello applicazione, per le simulazioni sono considerati di dimensione fissa e pari a 250 byte. Ogni nodo sensore nel caso di simulazione dello schema multihop routing implementa a livello di rete il protocollo MSUN, in caso contrario, quando non si necessita di un algoritmo di routing essendo presente un elemento supervisore che gestisce la comunicazione, il



(a) Traiettoria AUV percorsa utilizzando uw-POLLING.

(b) Traiettoria AUV percorsa utilizzando UFETCH.

Figura 5.1: Topologia di rete utilizzata per le simulazioni dei protocolli MSUN, uwPOLLING e UFETCH.

protocollo di livello rete implementato è ad un hop.

A livello MAC, essendo MSUN un protocollo di routing si appoggia al protocollo CSMA-ALOHA, mentre uw-POLLING e UFETCH sono protocolli specificatamente progettati per lavorare su tale livello e quindi ogni nodo sensore implementerà una loro versione.

Lo strato fisico, ovvero la parte atta all'interfacciamento con il mezzo trasmissivo svolge un ruolo molto importante dato che deve gestire due velocità di trasmissione durante i test per MSUN, uw-POLLING e UFETCH, una prima più bassa, che riguarda l'interazione tra semplici nodi sensori nel caso di MSUN e uw-POLLING, mentre tra nodi sensori e sink per UFETCH, ed è indicata con $B_{n,n}$ e pari a 1 kbps. La seconda invece, è quella utilizzata per tutte le comunicazioni verso l'AUV, denotata con $B_{n,AUV}$ e assume il valore di 10 kbps.

La frequenza portante utilizzata per la trasmissione dei segnali è f_0=26 kHz, mentre

la larghezza di banda è pari a 16 kHz. Tali parametri sono stati scelti in modo da coincidere con le caratteristiche dei modem S2C di EvoLogics [16].
Facendo riferimento sempre a tali modem acustici, sono stati settati i parametri riguardanti i consumi di potenza durante le varie fasi in cui un nodo si può trovare durante la propria attività. A tal proposito si è dunque scelto 100 W per la fase di trasmissione, 0.8 W per la ricezione ed infine 0.008 W per la fase di inattività.

Il canale su cui viaggeranno le informazioni sarà condiviso tra tutti i 36 nodi costituenti la rete e, come già specificato all'inizio del capitolo, questo non sarà considerato ideale, ma introdurrà tutti quei fenomeni solitamente presenti in reti acustiche sottomarine, quali: attenuazioni, fading e multipath. La struttura del mezzo utilizzato, le relazioni che legano l'attenuazione alla distanza percorsa dal segnale e che descrivono il problema del multipath e fading, faranno riferimento a quanto già descritto e discusso in maniera dettagliata nel sottoparagrafo 1.2 del capitolo 1.
La potenza impiegata per la trasmissione dei pacchetti, siano essi di segnalazione o semplici DATA, viene fatta variare nell'intervallo [130÷190] dB re μPa in modo da osservare il comportamento del protocollo in diversi scenari. Nella realtà è sufficiente una potenza di 160 dB re μPa dato che questa consente una buona comunicazione tra nodi a distanze di 1500 m che è l'obiettivo inizialmente preposto.

Un'ultima osservazione è la seguente: tutti i risultati presentati coincidono con valori medi ottenuti a seguito di svariate simulazioni, solitamente 50 test per ogni set di parametri. Ad ogni test l'unico parametro variabile sarà la posizione del nodo, che come detto è scelta in maniera aleatoria all'interno di una circonferenza di raggio 125 m dalla posizione iniziale ad esso assegnatagli, oltre ovviamente alle condizioni di canale.

5.2 Packet Delivery Ratio

Il primo parametro oggetto di discussione è la PDR, ovvero la percentuale di pacchetti correttamente consegnati al nodo AUV. Si ricorda che nel caso di UFETCH due sono gli hop che un pacchetto deve effettuare per arrivare a destinazione, ovvero da nodo sensore verso il nodo sink e successivamente da nodo sink all'AUV, il destinatario conclusivo della catena. Nel caso di uw-POLLING invece un solo hop

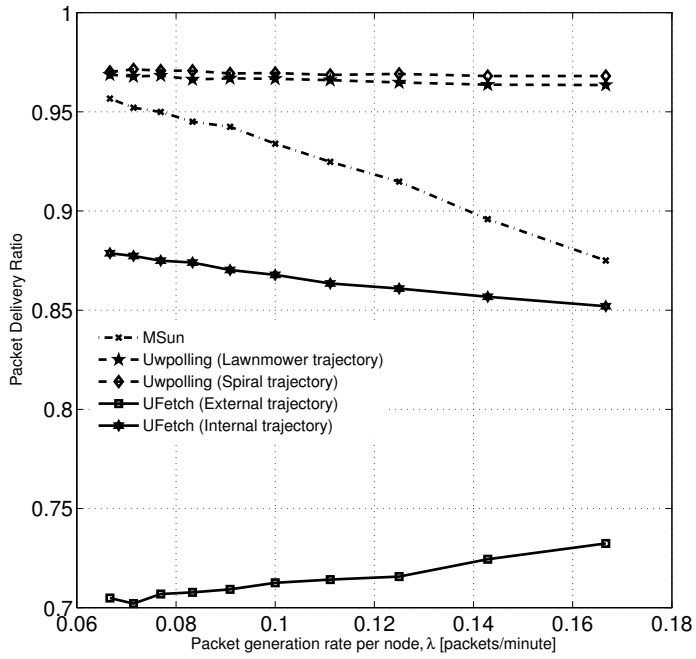


Figura 5.2: PDR ottenuta per i tre protocolli MSUN, uw-POLLING e UFETCH.

è necessario dato che il nodo sensore comunica direttamente con l'AUV, mentre per quanto concerne MSUN, essendo questo un protocollo multihop, il numero di hop effettuati dal pacchetto dipende dalla distanza del nodo sensore dall'AUV: più questa è elevata, maggiore sarà il numero di hop.

La figura (5.2) presenta la relazione che intercorre tra la PDR e il traffico generato dai nodi espresso in pck/min per i tre protocolli MSUN, uw-POLLING e UFETCH, oggetto di confronto.
Come si può notare uw-POLLING in termini di PDR è il migliore, infatti circa il [97÷98]% dei pacchetti arriva privo di errori a destinazione. Questo risultato ottimo è conseguenza del fatto che l'AUV crea un canale privo di interferenza con il nodo con cui dovrà comunicare, facendosi trasmettere solo la quantità di pacchetti DATA che esso ha generato. Ciò non accade con MSUN, dove ogni nodo appena ha un pacchetto DATA disponibile lo trasmette al suo vicino se esso è fuori dal raggio

di copertura dell'AUV, oppure direttamente a quest'ultimo se è nelle sue vicinanze. In questo caso la maggior quantità di pacchetti è ricevuta correttamente dall'AUV alla presenza di basso carico della rete, ma all'aumentare di quest'ultimo, la PDR diminuisce a causa dell'interferenza che si verrà creare in prossimità di quei soli nodi che hanno la possibilità di comunicare con l'AUV.
Prestazioni peggiori in termini di PDR sono date dal protocollo UFETCH, ma questo era anche un risultato preventivabile inizialmente, dato che è presente una elevata quantità di segnalazione per creare la comunicazione tra sensori-sink e sink-AUV. Tale messaggistica potrebbe causare interferenza nel momento in cui ci sono pacchetti DATA nel canale, aumentando quindi la probabilità d'errore.
Alcuni miglioramenti in termini di PDR nel caso del protocollo UFETCH, sono ottenibili modificando la traiettoria dell'AUV. Infatti come si può vedere nel grafico (5.3), il quale rappresenta sempre la PDR al variare del traffico di rete, ma in questo caso solamente per il protocollo UFETCH, si ha un miglioramento di circa il [10%÷15%] nel caso di utilizzo della traiettoria interna. Questo perché con la traiettoria esterna i nodi sink spesso dovranno comunicare con l'AUV che potrebbe

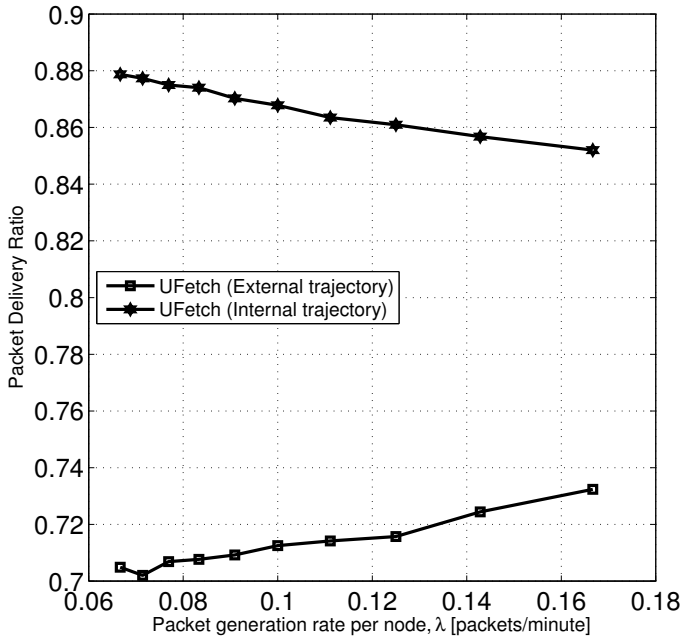
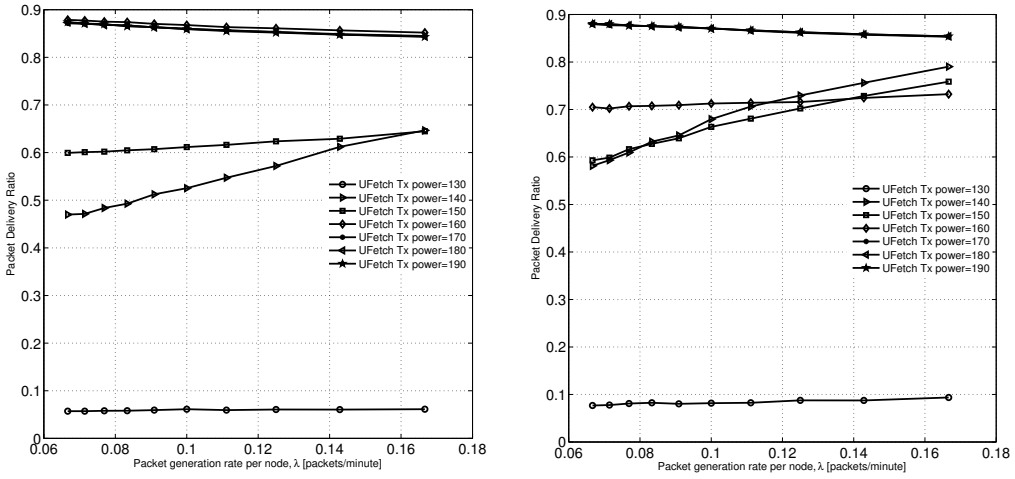


Figura 5.3: Guadagno in termini di PDR al variare della lunghezza del percorso eseguita dall'AUV.

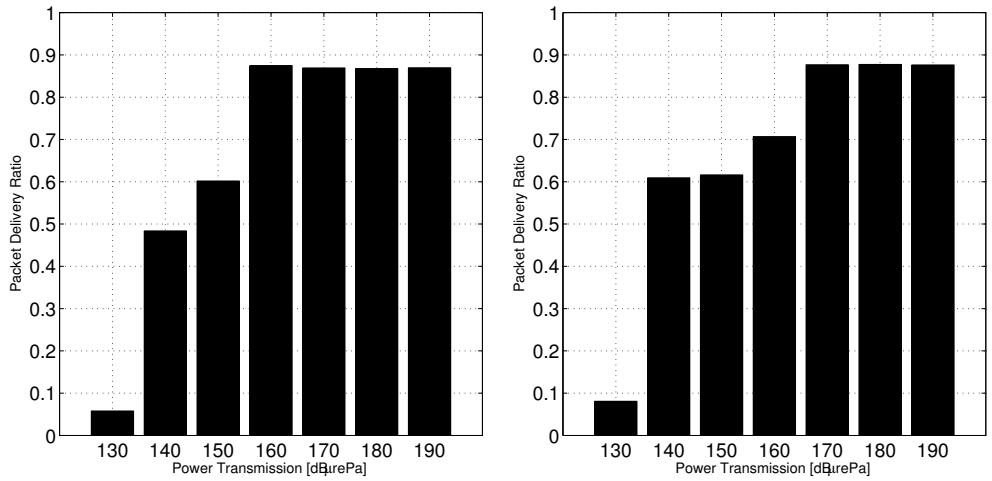
essere ad una distanza troppo elevata affinché, con la potenza utilizzata per la trasmissione (160 dB re μPa in questo caso) il pacchetto arrivi a destinazione con un rapporto SNR sufficiente. Con la traiettoria interna semplicemente l'AUV per più tempo rimane a contatto con il nodo sink non discostandosi mai da esso per più di 2500 m, di conseguenza la potenza utilizzata è sufficiente per poter raggiungere prestazioni accettabili e quasi comparabili con quelle ottenute mediante il protocollo MSUN.

Come già accennato, la potenza utilizzata per la trasmissione gioca un ruolo fondamentale per quanto riguarda la PDR, infatti più la potenza utilizzata per la trasmissione è alta e più robusto è il segnale trasmesso alle lunghe distanze. A tal proposito si possono osservare i grafici in figura (5.4) dove, considerando le due traiettorie a disposizione dell'AUV, è stata fatta variare la potenza nell'intervallo [130÷190] dB re μPa.
Si osserva immediatamente, sia nel caso di utilizzo della traiettoria interna, figura (5.4(a)), sia per quella esterna, figura (5.4(b)), che con potenze di [130-140-15] dB re μPa la maggior parte dei pacchetti DATA è corrotta a causa del ridotto SNR che caratterizza le trasmissioni, e la rete non raggiunge nemmeno il 50% di PDR.



(a) Traiettoria interna percorsa dall'AUV. (b) Traiettoria esterna percorsa dall'AUV.

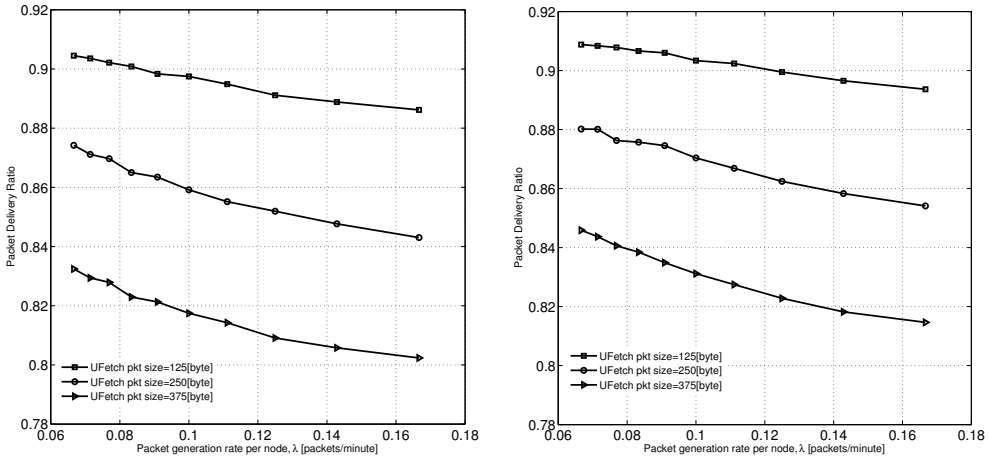
Figura 5.4: PDR ottenuta applicando il protocollo UFETCH al variare della potenza utilizzata per la trasmissione dei pacchetti DATA.



(a) Traiettoria percorsa dall'AUV interna. (b) Traiettoria percorsa dall'AUV esterna.

Figura 5.5: Guadagno in termini di PDR utilizzando il protocollo UFETCH al variare della potenza utilizzata per la trasmissione dei pacchetti DATA.

Nel caso di traiettoria interna invece, l'impiego di una potenza di 160 dB re μPa è sufficiente per raggiungere valori accettabili, ulteriori aumenti sarebbero pressoché inutili dato che grossi vantaggi non si ottengono, anzi potrebbero creare problemi in termini di consumo energetico da parte dei nodi come si vedrà in seguito.
Al contrario, se l'AUV segue il tragitto più esterno, una potenza di 160 dB re μPa consentirebbe la consegna corretta di circa il 70%, un valore limitato se confrontato con altri protocolli quali ad esempio MSUN e uw-POLLING. Il motivo, già descritto in precedenza, deriva dal fatto che l'AUV potrebbe raggiungere distanze anche di 4000 m da un nodo sink; questo fatto in concomitanza con l'interferenza di altri nodi presenti nella rete, costringe il sink all'utilizzo di potenze maggiori per effettuare la trasmissione dei pacchetti, ottenendo risultati confrontabili con quelli della traiettoria interna. Per una maggiore chiarezza, gli istogrammi presenti in figura (5.5), presentano, sempre sulla base delle due traiettorie interna, figura (5.5(a)) ed esterna, figura (5.5(b)), il guadagno ottenibile in termini di PDR all'aumentare della potenza di trasmissione. Questi valori sono ottenuti considerando il caso di una rete particolarmente carica, ovvero i nodi sensori generano un pacchetto ogni cinque minuti. I risultati confermano le conclusioni tratte a seguito della discussione dei grafici di figura (5.4).



(a) Traiettoria interna percorsa dall'AUV. (b) Traiettoria esterna percorsa dall'AUV.

Figura 5.6: PDR ottenuta applicando il protocollo UFETCH al variare delle dimensioni dei pacchetti DATA generati dai nodi sensori.

Un ultimo confronto in termini di PDR è stato effettuato ipotizzando di avere pacchetti DATA, generati dai nodi sensori, di dimensioni differenti, rispettivamente di [125-250-375] byte.
I risultati sono presentati in figura (5.6), sempre considerando le due traiettorie percorribili dall'AUV, interna figura (5.6(a)) ed esterna (5.6(b)).
Si nota che all'aumentare della dimensione dei pacchetti aumenta la possibilità di errore su di essi, questo perché maggiori sono i bit immessi nel canale, e maggiore è la probabilità che uno di questi venga corrotto.

5.3 Delay end-to-end

Un secondo parametro molto importante da considerare al fine del confronto tra protocolli è il ritardo di consegna end-to-end, ovvero il tempo che intercorre da quando il pacchetto è generato dal nodo sensore, a quando il medesimo arriva correttamente a destinazione, cioé all'AUV.

Il primo confronto, rappresentato in figura (5.7), è effettuato tra i protocolli MSUN, uw-POLLING e UFETCH. Si può subito osservare come UFETCH abbia

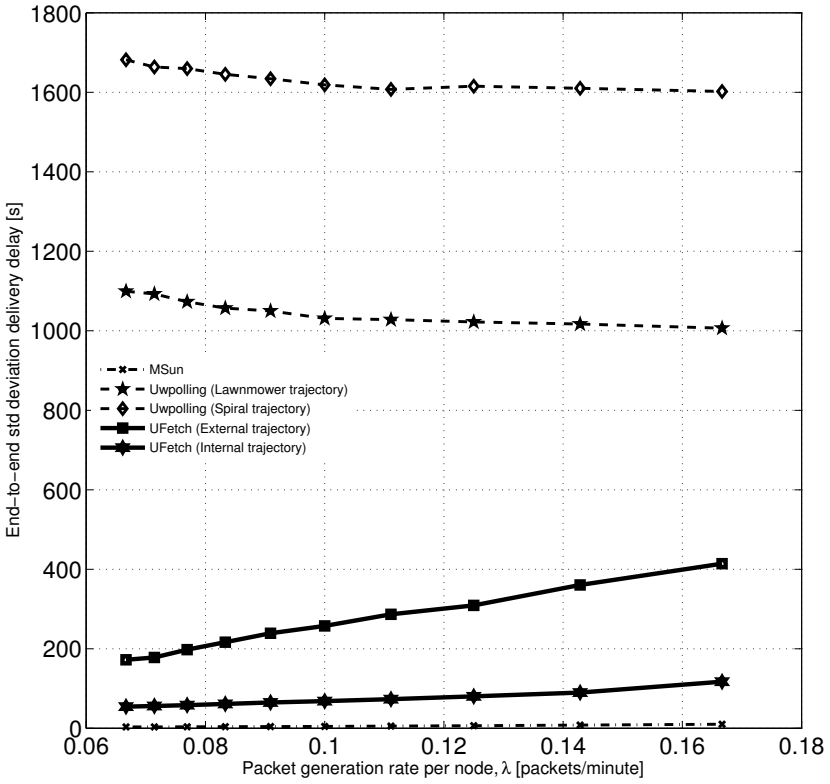


Figura 5.7: Confronto del ritardo di consegna end-to-end per i tre protocolli MSUN, uw-POLLING e UFETCH.

prestazioni intermedie tra i due protocolli MSUN e uw-POLLING, risultato in linea con il criterio di progetto che ha portato a realizzare UFETCH come soluzione intermedia tra il protocollo Multihop routing e Polling.
UFETCH raggiunge valori decisamente migliori rispetto uw-POLLING grazie alla presenza dei nodi sink. Infatti questi particolari nodi, se posizionati in maniera strategica all'interno dell'area marina monitorata, consentono il recupero dei pacchetti DATA dai nodi sensori e al tempo stesso permettono all'AUV di effettuare una traiettoria molto più breve rispetto a quella utilizzata nel caso uw-POLLING. Eseguire una traiettoria più breve implica che il tempo che intercorre tra due comunicazioni consecutive sink-AUV sia più piccolo, con conseguente diminuzione del ritardo di consegna end-to-end associato al pacchetto.

Utilizzando uw-POLLING, un nodo sensore potrà comunicare con l'AUV solamente quando questo è entro il proprio raggio di copertura, cioé circa ad una distanza di 1500 m. Si può ben capire che, se il nodo genera un pacchetto nel momento in cui l'AUV è dalla parte opposta della rete rispetto alla sua posizione, il tempo necessario affinché questo ritorni nelle sue vicinanze è molto elevato, fatto dato anche dalle basse velocità con cui l'AUV viaggia.
Si osservi inoltre che lavori di ottimizzazione della traiettoria percorribile dall'AUV ad oggi non sono stati eseguiti in maniera dettagliata, un confronto preliminare è stato effettuato solamente utilizzando due traiettorie, quindi ulteriori miglioramenti in termini di tempo di consegna dei pacchetti per il protocollo UFETCH potrebbero essere ottenuti svolgendo tale opera.
Al tempo stesso però, le prestazioni di UFETCH non sono migliori rispetto al protocollo MSUN dato che MSUN trae vantaggio dal fatto che un nodo, una volta stabilito un percorso che gli consenta di comunicare con l'AUV, lo utilizza immettendo un flusso di pacchetti DATA. UFETCH, invece, richiede tempo per l'accettazione della comunicazione(TRIGGER-RTS-CTS) che deve avvenire prima che i pacchetti DATA possano essere inviati dal nodo sink verso l'AUV.

Già dal grafico (5.7) si può notare come all'interno dello stesso protocollo UFETCH prestazioni migliori siano ottenibili utilizzando una traiettoria piuttosto di un'altra, si ricorda che ulteriori lavori di ottimizzazione non sono stati eseguiti. un semplice confronto è stato eseguito semplicemente considerando le due traiettorie raffigurate in figura (5.1), interna ed esterna. Da questo studio preliminare, si può notare dal grafico (5.8), come con la traiettoria più interna si abbia la consegna dei pacchetti in un tempo minore rispetto alla traiettoria più esterna, divario che tra le altre cose aumenta all'aumentare del traffico generato dai nodi.

Considerando sempre le due traiettorie sopra citate, si è cercato di vedere se utilizzando diverse potenze di trasmissione per i pacchetti sia possibile ottenere dei miglioramenti. La figura (5.9) mostra i risultati ottenuti.
Considerando la traiettoria interna, che è quella che ci consente di ottenere le maggiori prestazioni, si può notare come all'aumentare della potenza il ritardo di consegna si abbassi. Questo perché al nodo sink, utilizzando una potenza di trasmissione pari a 190 dB re μPa è consentito comunicare con l'AUV anche quando quest'ultimo è ad una distanza abbastanza elevata e nell'intorno dei 2000 m, cosa

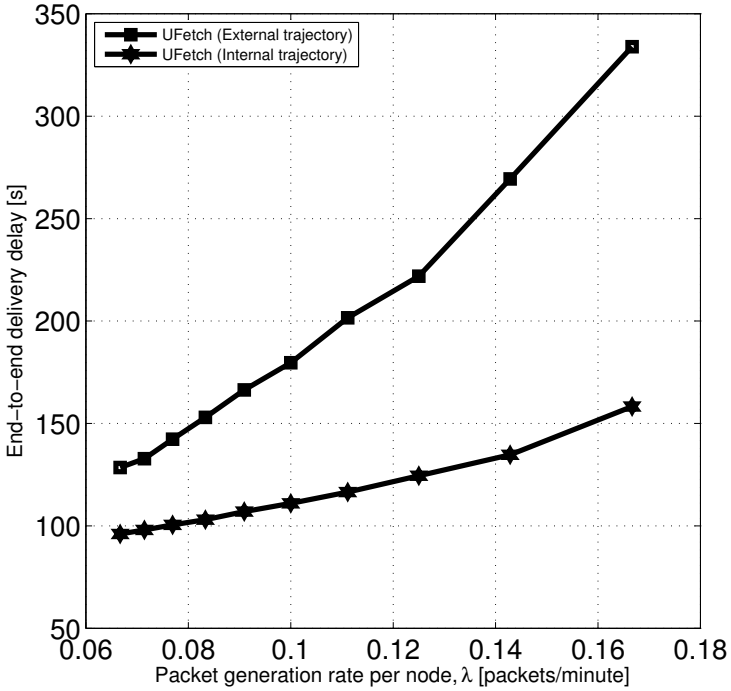


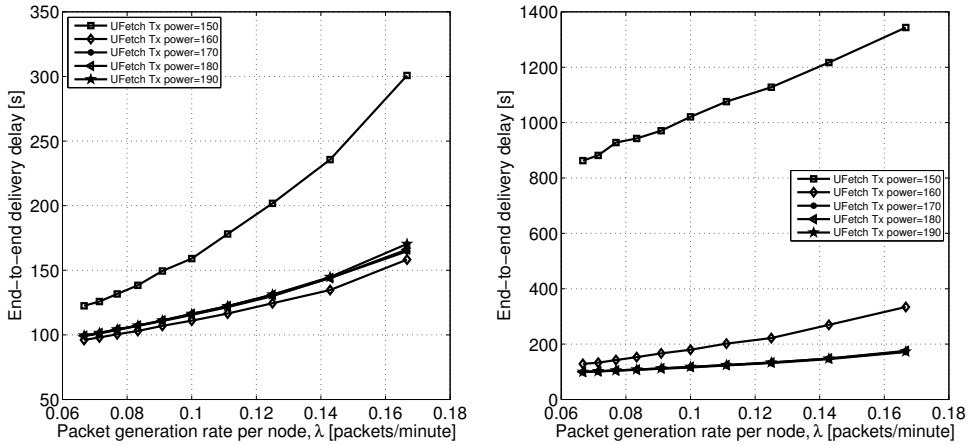
Figura 5.8: Confronto del ritardo di consegna dei pacchetti DATA end-to-end, utilizzando le due traiettorie (interna ed esterna) per il protocollo UFETCH.

che non accade nel caso di utilizzo di potenze pari o inferiori a 150 dB re μPa. Naturalmente un tempo minimo pari a circa 100 secondi è necessario affinché possa avvenire la consegna del pacchetto DATA, in quanto il nodo dovrà attendere un certo intervallo temporale prima di trovare l'AUV libero e pronto per la comunicazione.

5.4 Energia consumata dal sistema

All'inizio dell'elaborato si era detto che un parametro molto importante da tenere in considerazione per la progettazione di un protocollo di livello MAC fosse il consumo energetico dei nodi, dato che questi sono alimentati da batterie di durata limitata.
A tal proposito sono state ricavate alcune statistiche che di seguito si andranno a presentare.

Si parta dal fatto che un nodo sensore o sink durante la sua fase di attività si possa trovare in uno dei seguenti tre stati: inattività o *idling state*, ricezione o



(a) Traiettoria interna percorsa dall'AUV. (b) Traiettoria esterna percorsa dall'AUV.

Figura 5.9: Ritardo di consegna end-to-end dei pacchetti DATA al variare della potenza di trasmissione utilizzando il protocollo UFETCH.

reception state, trasmissione o *transmission state*. In ognuno di questi stati, anche nella fase di inattività, il nodo consuma energia che durante le simulazioni è stata impostata secondo i parametri dettati dai modem S2C di EvoLogics, come descritto nel sottoparagrafo 5.1.

Sommando l'energia consumata in ognuno di questi tre stati si è ottenuto il grafico di figura (5.10), il quale appunto mostra l'energia media consumata da ogni nodo al variare del carico di rete. Il confronto inizialmente è stato eseguito per i tre protocolli MSUN, uw-POLLING e UFETCH e si può vedere come, anche in questo caso, il protocollo UFETCH abbia prestazioni comprese tra quelle di MSUN e quelle di uw-POLLING.

Uw-POLLING presenta il minor consumo energetico poiché un singolo nodo è nella fase di trasmissione per una piccola percentuale di tempo rispetto alla sua intera fase di vita, dato che trasmette solamente quando l'AUV è entro il proprio raggio di copertura. Inoltre, la trasmissione avviene ad elevato bit rate, riducendo il tempo di trasmissione complessivo. Durante l'intervallo temporale necessario all'AUV per effettuare l'intero giro della traiettoria e ritornare nelle vicinanze di se stesso, il nodo sensore rimarrà inattivo consumando di conseguenza poca potenza.

Quanto detto per uw-POLLING non è valido per MSUN, infatti i nodi che implementano un tale protocollo hanno un consumo energetico decisamente elevato

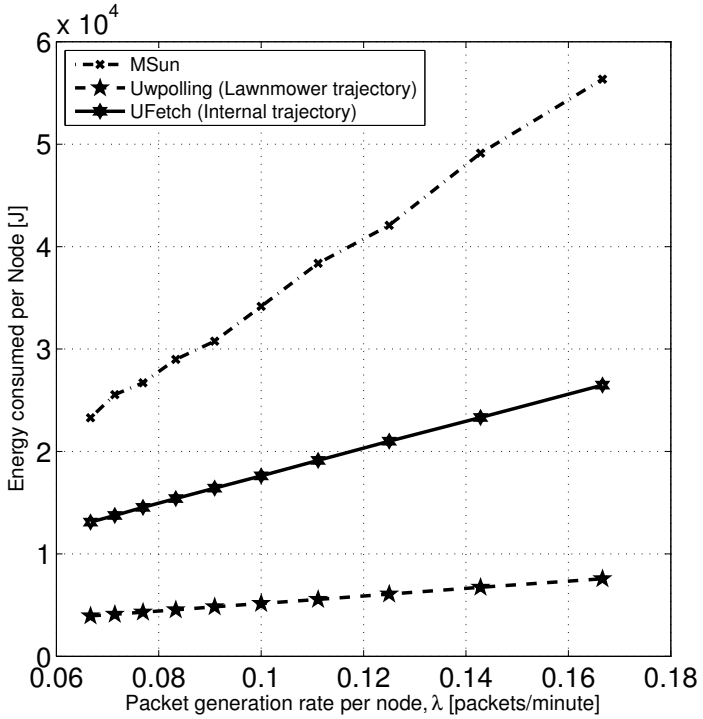
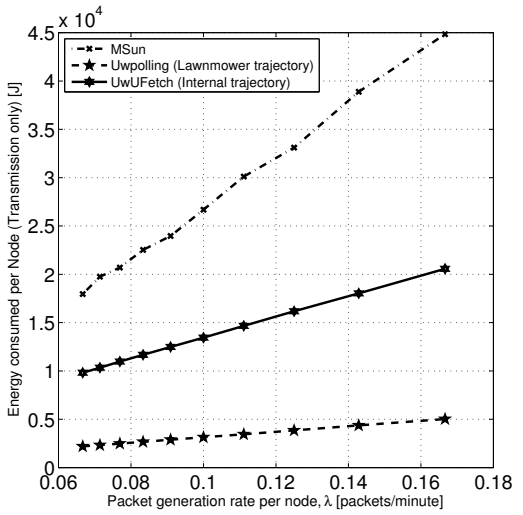


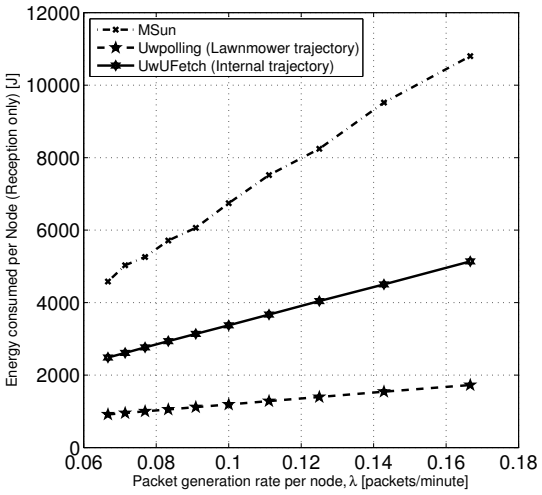
Figura 5.10: Energia totale consumata dai singoli nodi della rete utilizzando i tre protocolli MSUN, uw-POLLING e UFETCH.

rispetto anche ad UFETCH, dato che per la maggior parte del tempo i nodi sono in fase di trasmissione, poiché devono continuare a scambiare messaggi con l'obiettivo di creare una rotta su cui inviare poi i pacchetti DATA da far arrivare all'AUV.
Il protocollo UFETCH si pone nel mezzo in termini di energia consumata. Esso non raggiunge i valori di uw-POLLING dato che necessita di uno scambio di pacchetti di segnalazione, affinché si possano stabilire le comunicazioni tra nodi sensori-sink e tra sink-AUV, ma non raggiunge nemmeno gli elevati valori di consumo di MSUN poiché una volta stabilita la connessione non necessita di un continuo scambio di pacchetti per ricercare rotte da utilizzare per l'invio dei pacchetti DATA, questi sono direttamente consegnati dal nodo sensore al sink e successivamente dal sink all'AUV.

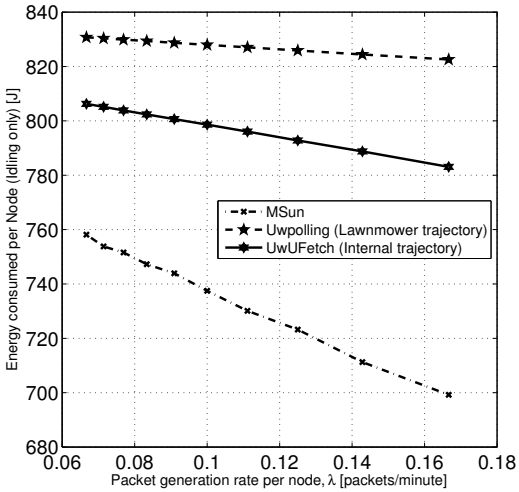
Il set di figure in (5.11) mostra in dettaglio il confronto tra i tre protocolli per quanto concerne l'energia media consumata dai singoli nodi durante le tre fasi appena sopra citate. La somma di questi tre grafici permette di ottenere la figura



(a) Energia media spesa da un nodo in stato di TRANSMIT.



(b) Energia media spesa da un nodo in stato di RECEIVE.



(c) Energia media spesa da un nodo in stato di IDLE .

Figura 5.11: Energia consumata dai singoli nodi della rete durante la loro permanenza nei tre stati principali di funzionamento: idling, receiving e transmitting state. Confronto effettuato tra i protocolli MSUN, uw-POLLING e UFETCH.

(5.10).

Da questo set è possibile notare come la maggior quantità di consumo la si abbia durante la fase di trasmissione.

Concentrando ora l'attenzione esclusivamente sul protocollo UFETCH, una

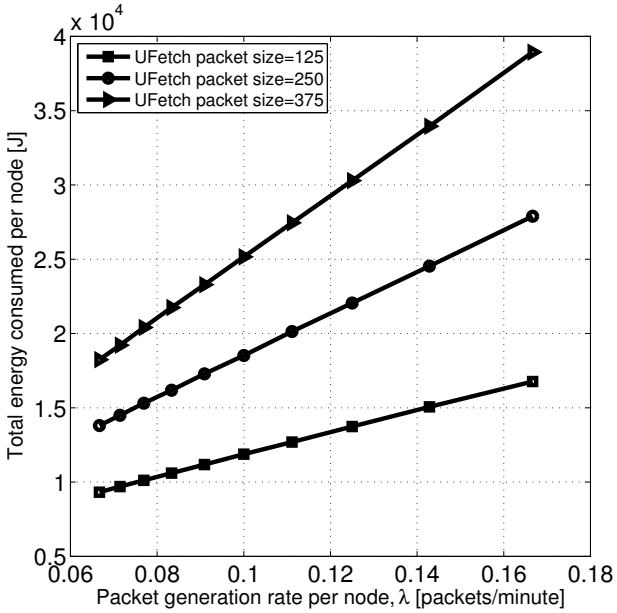
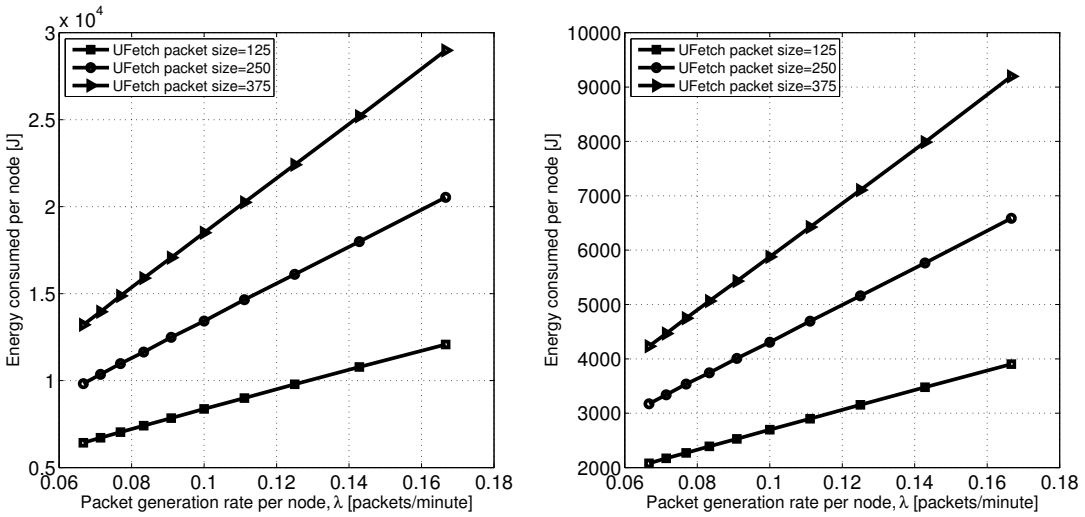


Figura 5.12: Energia totale consumata dai singoli nodi della rete utilizzando il solo protocollo UFETCH e facendo variare le dimensioni dei pacchetti DATA.

prima analisi è stata eseguita variando la dimensione dei pacchetti DATA generati dai nodi sensori. Le dimensioni considerate sono [125-250-375] byte, mentre come potenza di trasmissione si è utilizzato 160 dB re μPa, scelta dettata dal fatto che con tale valore si raggiungono risultati in termini di PDR e di ritardo di consegna end-to-end acccettabili.
La figura (5.12) riassume i risultati ottenuti, ovvero indica l'energia media totale consumata per nodo. Come da progetto, maggiore è la dimensione del pacchetto e maggiore è l'energia consumata, questo poiché mediamente per più tempo il nodo deve rimanere nello stato di trasmissione in cui ha luogo il maggior consumo energetico.

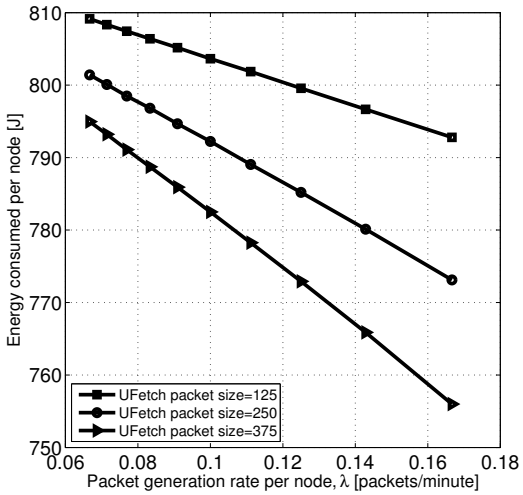
Osservando in dettaglio i consumi, come già fatto nel caso di confronto tra i tre protocolli, si riportano in figura (5.13) i consumi medi di un nodo durante i tre stati in cui si potrà trovare durante la sua fase di attività. In questo caso, come anche per i risultati ottenuti in figura (5.12) le simulazioni sono state eseguite considerando la traiettoria dell'AUV più breve, cioé quella interna. Gli stessi consumi energetici sono ottenibili nel caso di utilizzo della traiettoria più esterna da parte dell'AUV.

Altra statistica interessante da ricavare è il consumo energetico medio di ogni



(a) Energia media consumata dal nodo in stato di TRANSMIT.

(b) Energia media consumata dal nodo in stato di RECEIVE.



(c) Energia media consumata dal nodo in stato di IDLE .

Figura 5.13: Energia consumata dai singoli nodi della rete al variare delle dimensioni dei pacchetti DATA durante la loro permanenza nei tre stati principali di funzionamento: idling, receiving e transmitting state. Protocollo utilizzato: UFETCH.

singolo nodo al variare della potenza utilizzata per la trasmissione dei pacchetti circolanti nella rete.

A tal proposito la figura (5.14) mostra come l'energia totale consumata aumenti

all'aumentare della potenza di trasmissione utilizzata. I valori usati nei test sono compresi tra [130÷190] dB re μPa. La crescita del consumo energetico oltre ad aumentare con la potenza di trasmissione (soprattutto il passaggio da 130 a 140 dB re μPa), è inoltre lineare rispetto all'aumento del carico della rete. Si osservi però d'altro canto che l'utilizzo di 130 dB re μPa corrisponde ad una potenza con cui è impossibile trasmettere, poiché a causa del rumore introdotto dal canale, la maggior parte della segnalazione per instaurare la connessione viene corrotta, non consentendo così ai nodi sensori aventi informazioni pronte da trasmettere, di accedere al canale, ottenendo di conseguenza valori non accettabili di PDR e ritardo di consegna end-to-end.
Si dovrà quindi cercare un compromesso tra PDR, ritardo end-to-end e consumo energetico. Per avere PDR elevate e ritardi end-to-end più bassi possibili, si necessita di potenze di trasmissione elevate, che a loro volta porteano ad un maggiore consumo energetico; viceversa, avere un sistema che consumi poco in termini energetici significa perdere in termini di PDR e ritardo end-to-end. Per una maggiore completezza, si riportano anche in questo caso gli andamenti dei consumi medi energetici per le tre fasi di attività dei nodi al variare della potenza utilizzata per la trasmissione dei messaggi e del traffico generato dai nodi stessi. I risultati sono mostrati in figura (5.15), dove si sono visualizzati gli andamenti solo per tre potenze di trasmissione [140-160-190] dB re μPa, a differenza del grafico (5.14) dove si sono considerati sette valori distinti .

5.5 Throughput

Quarta ed ultima statistica discussa è relativa al confronto del throughput medio associato ad ogni singolo nodo della rete per i tre protocolli MSUN, uw-POLLING e UFETCH. Sono state inoltre effettuate successive indagini più specifiche a riguardo del solo protocollo UFETCH al variare della potenza utilizzata per la trasmissione.

Un primo confronto, come appena accennato, riguarda i tre protocolli. La figura (5.16) mostra i risultati ottenuti a seguito di una serie di simulazioni effettuate con potenza trasmissiva di 160 dB re μPa e al variare del traffico generato dai nodi sensori. In questo caso UFETCH presenta il throughput medio per nodo più basso rispetto agli altri due protocolli uw-POLLING e MSUN.

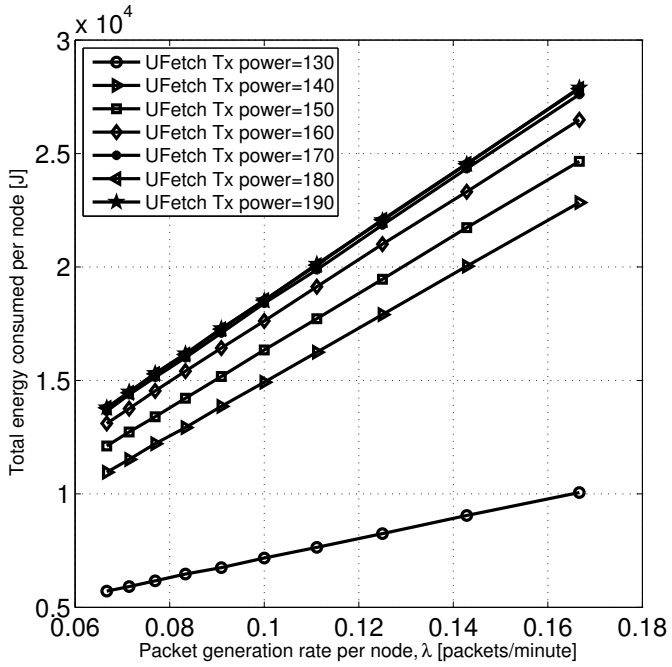
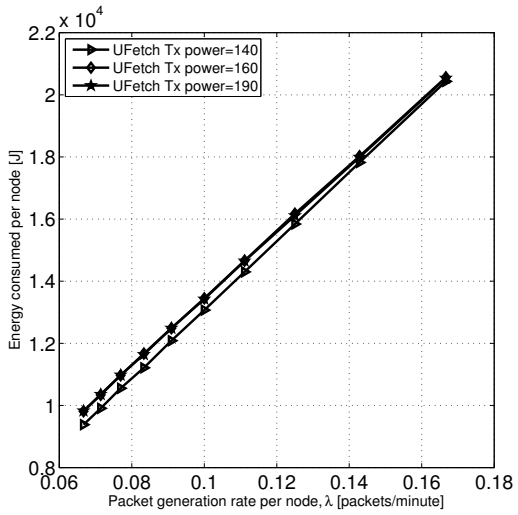


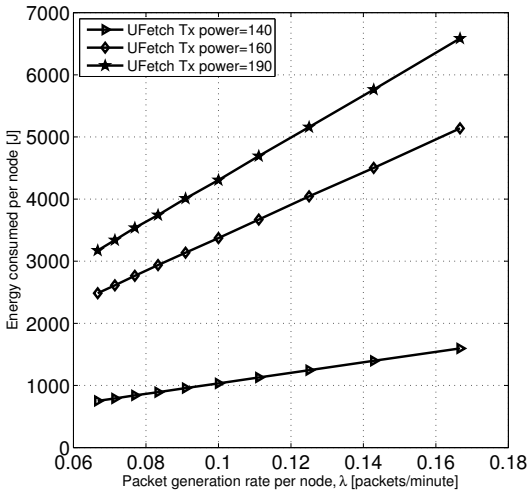
Figura 5.14: Energia totale consumata dai singoli nodi della rete utilizzando il solo protocollo UFETCH e facendo variare la potenza utilizzata per trasmettere i pacchetti DATA.

Il migliore in questi termini è sicuramente uw-POLLING. Pochi pacchetti con un tale protocollo saranno corrotti poiché il rumore è inibito grazie all'utilizzo di una potenza trasmissiva elevata considerando le distanze tra sensore e AUV all'atto della trasmissione dei pacchetti DATA, valori nell'intorno dei 1000 m.
MSUN invece, ha un throughput leggermente migliore rispetto ad UFETCH poiché implementa anche una tecnica di ritrasmissione dei pacchetti DATA nel momento in cui questi risultano essere corrotti.

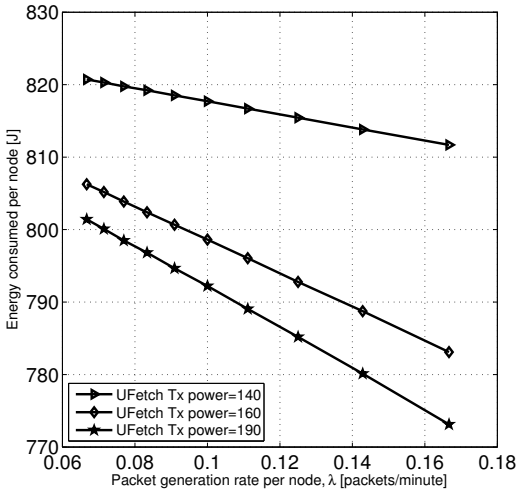
Un ingrandimento della figura (5.16) ci consente di osservare quali miglioramenti sono ottenibili utilizzando due distinte traiettorie percorribili dall'AUV. Osservando infatti la figura (5.17) si nota come all'interno del solo protocollo UFETCH si abbia la possibilità di aumentare il throughput medio per nodo considerando entrambe le rotte percorse dall'AUV.
Con il tragitto esterno evidenziato in grassetto in figura (5.1), il throughput è leggermente inferiore rispetto all'impiego di un percorso più breve qual é quello interno, linea tratteggiata della medesima figura. Il motivo di ciò è quello già



(a) Energia media consumata dal nodo in stato di TRANSMIT.



(b) Energia media consumata dal nodo in stato di RECEIVE.



(c) Energia media consumata dal nodo in stato di IDLE .

Figura 5.15: Energia consumata dai singoli nodi della rete al variare della potenza utilizzata per la trasmissione dei pacchetti DATA durante la loro permanenza nei tre stati principali di funzionamento: idling, receiving e transmitting state. Protocollo utilizzato: UFETCH.

descritto in precedenza, a parità di potenza di trasmissione, il segnale con la traiettoria più esterna deve eseguire un percorso più lungo per raggiungere l'AUV, di conseguenza subisce una maggiore attenuazione e quindi una più elevata probabilità

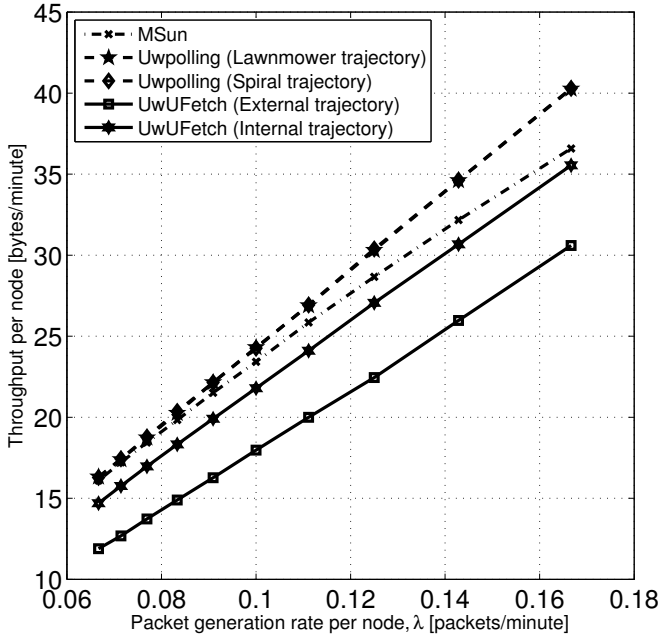


Figura 5.16: Throughput per nodo ottenuto utilizzando i tre protocolli MSUN, uw-POLLING e UFETCH.

che il pacchetto sia corrotto.

Per rafforzare ulteriormente quanto appena detto, il set di grafici (5.18) mostra il guadagno in termini di throughput al variare del tasso di generazione dei pacchetti e all'aumentare della potenza utilizzata per la trasmissione. Il guadagno lo si può notare maggiormente analizzando la figura (5.18(b)), dove è stata fatta eseguire all'AUV la traiettoria più lunga (esterna).
Per quanto riguarda invece il percorso interno si ha che l'utilizzo di potenze superiori a 160 dB re μPa, a parità di traffico generato, non produce incrementi di prestazioni. L'utilizzo quindi di una tale potenza è sufficiente per avere valori soddisfacenti di throughput consentendo anche un certo risparmio energetico, come già descritto nel sottoparagrafo 5.4.

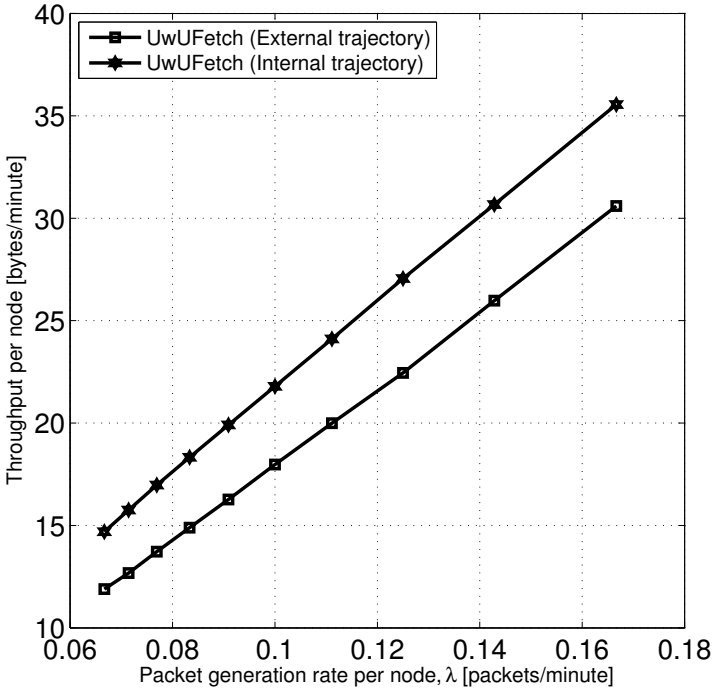
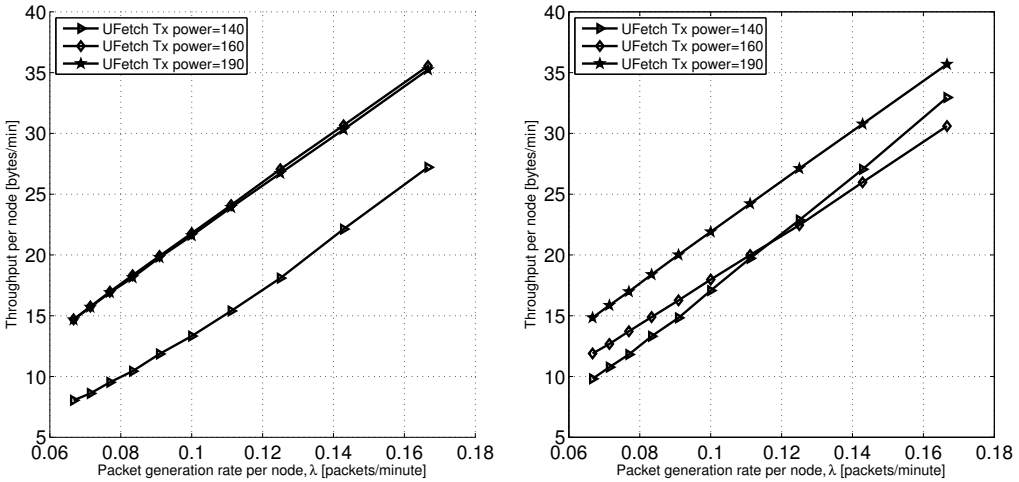


Figura 5.17: Throughput per nodo calcolato utilizzando il protocollo UFETCH con le due varianti dettate dal percorso effettuato dall'AUV.



(a) Traiettoria percorsa dall'AUV interna. (b) Traiettoria percorsa dall'AUV esterna.

Figura 5.18: Throughput per nodo ottenuto utilizzando il protocollo UFETCH nelle due varianti dettate dal percorso effettuato dal nodo AUV. Risultati ottenuti facendo variare la potenza utilizzata per la trasmissione dei pacchetti DATA.

Conclusioni

A seguito dello studio effettuato e descritto dettagliatamente per mezzo di grafici all'interno del capitolo 5, si può affermare come il protocollo UFETCH possa aggiungersi a tutti quei protocolli presentati e validi per reti acustiche sottomarine visti all'inizio dell'elaborato.
Pregi e difetti del protocollo UFETCH sono usciti dal confronto con i due protocolli MSUN e uw-POLLING. E' possibile affermare che UFETCH sia una valida alternativa, o un buon compromesso, tra i due protocolli da utilizzare per tutte quelle applicazioni da cui l'utente vuole ottenere risultati soddisfacenti sia in termini di PDR, sia di ritardo di consegna end-to-end, sia di risparmio energetico dei nodi, non privilegiando di fatto alcun parametro.
Si può notare come dai grafici presentati nel capitolo 5, sia emerso che il protocollo uw-POLLING presenti valori di PDR praticamente pari al 100% con un consumo energetico da parte dei nodi della rete estremamente basso, ma un ritardo di consegna end-to-end elevato se confrontato con gli altri due protocolli. Con MSUN invece quello che si presenta è: PDR leggermente inferiore rispetto ad uw-POLLING, valori di ritardi di consegna molto bassi, ma il tutto al costo di un elevato consumo energetico da parte dei nodi. Quest'ultimo parametro non va trascurato nel progetto di una rete acustica sottomarina. Si osservi che inoltre risultati migliori, che consentirebbero ad MSUN di raggiungere valori prossimi a quelli ottenuti da uw-POLLING, potrebbero essere ottenuti in termini di PDR nel momento in cui tale protocollo implementasse tecniche di ritrasmissione ARQ (cosa non applicata durante le simulazioni effettuate), possibilità che potrebbe essergli fornita dato che la trasmissione non è regolata da alcuna entità come in UFETCH ed uw-POLLING.
Infine UFETCH consente di ottenere valori leggermente inferiori di PDR rispetto

agli altri due protocolli, ma contemporaneamente permette l'implementazione di un sistema in cui si abbia un consumo energetico che raggiunge quasi quello ottimale di uw-POLLING, ed un ritardo di consegna dei pacchetti DATA molto vicino a quello di MSUN. La perdita in termini di PDR rispetto agli altri due protocolli (MSUN, uw-POLLING) è di circa il 5% in termini percentuali, gap che potrebbe essere ulteriormente ridotto se si operano delle ottimizzazioni della traiettoria percorsa dal nodo AUV. Infatti, dai primi studi effettuati sulla base solamente di due traiettorie aventi lunghezze di percorso diverse, ma percorse alla stessa velocità, si è visto come questa giochi un ruolo molto importante per quanto concerne le prestazioni finali del sistema. Un'analisi approfondita sotto questo punto di vista potrebbe essere quindi un argomento molto interessante da effettuare in futuro per poter ottimizzare ulteriormente i risultati già preliminarmente ottenuti.

In conclusione, come nella maggior parte dei casi, all'inizio del progetto l'utente dovrà scegliere il parametro al quale dare maggiore priorità sulla base dell'applicazione che vorrà implementare. Se la necessità è una comunicazione che non ammette consegne di dati errati, e si ha a disposizione un AUV mobile, allora il protocollo maggiormente indicato è uw-POLLING. Se la velocità di consegna dei pacchetti è invece la priorità dell'applicazione, e non è possibile avere un AUV mobile, allora MSUN, al costo di una PDR inferiore rispetto a quello offerta da uw-POLLING, è quello consigliato. Creando e organizzando però una particolare struttura di rete, e avendo a disposizione un AUV mobile, è possibile implementare un protocollo che sia applicabile a dei sistemi in modo da ottenere in media una leggera diminuzione delle prestazioni in termini di PDR e ritardo di consegna end-to-end e allo stesso tempo contenere il consumo energetico della rete rispetto ai due protocolli appena sopra citati. Questo compromesso è ottenibile implementando ai nodi della rete acustica sottomarina il nuovo protocollo fin qui presentato UFETCH.

Bibliografia

[1] A.B. Baggeroer, *"Acoustic telemetry-An overview"*, Aug.1984, IEEE J Ocean Eng, vol. 9, no. 4, pp.229−235.

[2] J.A. Catipovic, *"A performance limitations in underwater acoustic telemetry"*, 1990, IEEE J Ocean Eng, vol. 15, no. 3, pp.205−206.

[3] D.B. Kilfoyle. and A.B. Baggeroer, *"The state of the art in underwater acoustic telemetry."*, 2000, IEEE J Ocean Eng, vol. 25, no. 1, pp.4−27.

[4] A. Goldsmith, *"Wireless communications"*, Cambridge University Press, 2005.

[5] Y.S. Cho, J. Kim, W.Y. Yang, C.G Kang, *"MIMO-OFDM: Wireless Communications with Matlab"*, John Wiley and Sons Asia, Ltd, 2010.

[6] E.M. Sozer, M. Stojanovic, J.G. Proakis *"Underwater acoustic network"*, Luglio, Boston, USA, 1999.

[7] M. Stojanovic, *"On the relationship between capacity and distance in an underwater acoustic communication channel"*, Settembre, WUWNet, Los Angeles, California, USA, 2006.

[8] L. Hong, F. Hong, Z. Guo, X. Yang *"A TDMA-based MAC protocol in underwater sensor networks"*, Proc. WiCOM, Dalian, China, 2008.

[9] N. Chirdchoo, W.-S. Soh, K.C. Chua *"ALOHA-based MAC protocols with collision avoidance for underwater acoustic networks"*, in Proc. IEEE INFOCOM, Anchorage, AK, 2007.

[10] N. Chirdchoo, W-S. Soh, K.C. Chua *"MACA-MN: a MACA-based MAC protocol for underwater acoustic networks with packet train for multiple neighbors"*, in Vehicular technology conference, Maggio 2008, IEEE pp:46-50.

[11] X. Guo, M. Frater, M. Ryan *"Design of a propagation-delay-tolerant MAC protocol for underwater acoustic sensor networks"*, IEEE J.Ocean.Eng 34, Febbraio 2009.

[12] NG Hai-Heng, W-S. Soh, M. Motani *"MACA-U: a Media Access Protocol for underwater acoustic networks"*, in Global Telecommunications Conference, Dicembre 2008, IEEE GLOBECOM pp:1-5.

[13] B. Pelato, M. Stojanovic *"Distance Aware Collision Avoidance Protocol for Ad-Hoc underwater acoustic networks"*, Dicembre 2007, IEEE COMMUNICATION LETTERS, Vol. 11, No. 12.

[14] M. Molins, M. Stojanovic *"Slotted FAMA: a MAC protocol for underwater acoustic networks"*, Dicembre 2007, IEEE COMMUNICATION LETTERS, Vol. 11, No. 12.

[15] F. Favaro, P. Casari, F. Guerra, M. Zorzi *"Data Upload from a static underwater network to an AUV: Polling or Random Access?"*, Dicembre, Oceans, Yeosu, Corea, 2012.

[16] Evologics, *"http://www.evologics.de/en/products/acoustics/"*

[17] K.G. Kebkal, R. Bannasch *"Performance of S2C Acoustic modem in horizontal underwater acoustic channels"*, Giugno, Hereklion, Grecia, 2007, pp. 1351-1358.

[18] M. Chitre, S. Shahabudeen, M. Stojanovic *"Underwater Acoustic Communications and Networking: Recent Advances and Future Challenges"*, Spring. 2008, IEEE J Ocean Eng, vol. 42, no. 1, pp.103−115.

[19] O. Kebkal, M. Komar, K. Kebkal, *"D-MAC: Hybrid media access control for underwater acoustic sensor networks"*, IEEE, 2010.

[20] K.G. Kebkal, R. Bannasch *"Sweep spread carrier for underwater communication over acoustic channels with strong multpath propagation"*, Novembre 2002, J.Acoustical Society of America.

[21] A.S. Tanenbaum, D.J Wetherall *"Reti di calcolatori:5 Edizione"*, Pearson, Settembre 2011.