Energy Efficient Underwater Acoustic Networks

Josep Miquel Jornet Montaña
Advisor: Milica Stojanovic

May 23, 2008

Resumen

Título: Diseño de redes subacuáticas energéticamente eficientes

La transferencia de información mediante múltiples enlaces consecutivos o transmisión multisalto es una técnica de comunicación habitual en redes inalámbricas. Este mecanismo es especialmente útil en redes de sensores o redes ad hoc en general, en las que los distintos elementos de red (habitualmente alimentados por baterías) deben minimizar su consumo energético sin comprometer el correcto funcionamiento de la misma. En el caso de redes subacuáticas, la transmisión mediante múltiples saltos cortos no sólo permite disminuir la potencia de transmisión, y su correspondiente consumo, sino que además posibilita la utilización de un ancho de banda mayor. Esta propiedad se debe al hecho que el ancho de banda acústico depende de la distancia de transmisión, aumentando cuando ésta disminuye.

Efectivamente, las comunicaciones subacuáticas se sustentan en la transmisión de ondas acústicas (las ondas electromagnéticas sufren de tal atenuación que no es posible trabajar en frecuencias superiores a unos 300Hz). El canal acústico se caracteriza por: unas pérdidas de propagación muy elevadas, que no sólo dependen de la distancia de transmisión sino también de la frecuencia de trabajo (aumentan con ambos parámetros); una velocidad de propagación relativamente baja (el sonido se propaga en agua salada a 1500m/s frente a los 300.000m/s de las ondas electromagnéticas); y distorsión tanto en el dominio temporal como frecuencial debido a la propagación multicamino y un pronunciando efecto Doppler.

La primera parte del proyecto está dedicada al diseño y estudio de los beneficios introducidos por el uso de un control de potencia discreto. A diferencia de lo habitual en los módems acústicos vigentes (transmisión fija a potencia máxima), se propone definir un número variable de niveles de potencia de transmisión. El objetivo es seleccionar aquel nivel de potencia que permita la comunicación entre un emisor y un receptor determinados, manteniendo una calidad de señal mínima (SNR) y minimizando la posible interferencia a otros elementos de la red.

El número de niveles necesario así como la manera de definir la correspondencia entre cada nivel y su potencia son analizados para distintas densidades de red y optimizados para minimizar el consumo energético por bit. Asimismo, el rendimiento del sistema es analizado en función de la frecuencia de trabajo y el ancho de banda disponible, que a su vez dependen de la densidad de red.

En una red cuyos integrantes hacen uso del control de potencia, las funcionalidades a nivel de enrutado, de control de acceso al medio y de capa física están estrechamente relacionadas. La técnica de enrutado es la que decide cuál es el nivel de potencia a usar, ya sea en función de la ruta que se quiera seguir o del mecanismo de descubrimiento de rutas que se esté ejecutando. El protocolo de acceso al medio adaptará algunos de sus parámetros de acuerdo con éste (por ejemplo, el tiempo de espera antes de dar por perdido un paquete). Finalmente, es en la capa física en la que se selecciona un nivel de transmisión u otro.

Para el análisis, dos protocolos MAC han sido considerados: DACAP, un protocolo de enrutado explícitamente desarrollado para redes subacuáticas en el que se intenta minimizar el número de colisiones (y su consiguiente pérdida de energía) mediante una sofisticación del proceso de reserva del canal, y una versión avanzada del conocido ALOHA, en el que el medio es escuchado antes de transmitir. Ambos protocolos son adecuados para redes integradas por nodos fijos y móviles (pequeños submarinos autónomos o AUVs), que no requieren estar sin-

cronizados a un reloj global. En los escenarios que resultan de nuestro interés, el rendimiento de estos protocolos depende directamente de la potencia de transmisión. De manera simplificada, el uso de un nivel de potencia mayor al necesario, en el caso de ALOHA, incrementa las posibilidades de colisión y el número de paquetes descartados debido a una excesiva interferencia. En el caso de DACAP, un aumento en la potencia de transmisión se traduce en un aumento en el tiempo de espera mínimo para garantizar la ausencia de colisiones.

Para evaluar estos efectos se ha desarrollado y utilizado un simulador de redes subacuáticas programado en Python (este se puede descargar libremente bajo licencia GNU des de http://sourceforge.net/projects/auvnetsim/). La fase inicial del proyecto consistió en verificar el correcto funcionamiento de la estructura ya definida e incluir los nuevos protocolos o variaciones de estos introducidas en este proyecto. El apéndice de este documento contiene una introducción en forma de manual de usuario del simulador.

Los resultados muestran que el consumo energético por bit se puede reducir aumentando la frecuencia de trabajo del sistema, disminuyendo así la interferencia de fondo creada en cada transmisión y permitiendo el uso de un ancho de banda mayor. Un ancho de banda mayor permite habitualmente transmitir a una tasa más elevada. Esto tiene un doble efecto. En primer lugar, el consumo energético es menor, pues la duración temporal de un bit es menor. En segundo lugar, los paquetes transmitidos, para una misma duración en número de bits, son temporalmente más cortos y, por tanto, la probabilidad de que colisionen es menor. Esto anima a transmitir siempre a la máxima tasa de transmisión posible, aunque la aplicación en concreto no lo requiera. Al mismo tiempo, es este conjunto de efectos el que permite que un protocolo tan sencillo como ALOHA muestre un rendimiento muy cercano al del más sofisticado DACAP.

Como conclusión de esta primera parte, se demuestra que la transmisión multisalto es claramente útil cuando de minimizar la energía por bit se trate, siempre que la potencia de transmisión, la frecuencia de trabajo y el ancho de banda sean seleccionados de acuerdo con la densidad de red. En la segunda parte del proyecto, se introduce una nueva técnica de enrutado para redes subacuáticas en la que la creación de rutas, el mecanismo de reserva de canal y el control de potencia discreto están claramente vinculados. El protocolo presentado se basa en el conocimiento parcial de la posición de los elementos de red. Efectivamente, parece lógico pensar que los elementos fijos de la red puedan saber su propia posición y, en el caso de los vehículos subacuáticos, éste es necesario para su correcto funcionamiento. El segundo requisito es que un nodo, además de su propia posición, debe saber la posición del destinatario final de la información. Nuevamente, este suposición tiene sentido en redes en las que hay uno o varios centros de recolección de información, cuya posición puede saberse de antemano. Sin embargo, un nodo no tiene porqué saber qué otros nodos le separan de su destino.

A diferencia de la mayoría de protocolos de enrutado existentes, no hay necesidad de definir una ruta antes de poder transmitir o de mantener una tabla de rutas actualizada constantemente, sino que a lo largo del camino hacia su destino, los distintos nodos irán proponiéndose como repetidores si realmente pueden serlo. De manera simplificada, el transmisor de un paquete iniciará el proceso de reserva del canal usando el nivel de potencia mínimo, especificando el destinario final y, si procede, solicitando un repetidor válido. Si hay uno o más nodos en una posición útil (el paquete de reserva del canal incluye las posiciones de origen y destino del paquete), estos contestarán enviando su posición. Un nodo está en una

posición válida si se encuentra dentro del cono de transmisión, definido por la distancia de transmisión y su apertura. Si toda va bien, de manera simultánea se habrá reservado el canal y definido el siguiente punto en la ruta hacia el destino final. Si no se ha encontrado ningún nodo válido, se incrementará el nivel de potencia de transmisión.

Usando el simulador introducido anteriormente, se ha verificado el correcto funcionamiento del protocolo, optimizando los distintos parámetros del sistema para el mínimo consumo energético por bit posible. Su rendimiento es comparado al uso de rutas preestablecidas, determinadas usando el algoritmo de Dijkstra para obtener las rutas de mínimo coste. Los resultados muestran que no sólo en la mayoría de casos las rutas coinciden, obteniendo de manera dinámica rutas energéticamente óptimas, sino que el retardo adicional introducido en todo el proceso es mínimo. Este protocolo da pié a muchas variaciones y sofisticaciones, pero el objetivo inicial ha sido crear una mecanismo válido y suficientemente sencillo como para poder ser implementado en la práctica.

Contents

List of Figures

List of Tables

Abstract

In this thesis, discrete power control is introduced as a practical means of enabling multi-hop communications for large coverage area in bandwidth-limited underwater acoustic networks. The document is divided into two parts. In the first part, the overall system performance is evaluated for different power selection schemes as well as for different frequency allocation patterns (center frequency f_c and bandwidth B). In the second part of the thesis, the Focused Beam Routing protocol (FBR), a scalable routing technique based on location information, is introduced and optimized for minimum energy per bit consumption. The appendix contains an introduction to the underwater acoustic network simulator that has been developed and used to obtain all the results shown in this thesis.

Acknowledgements

I would like to deeply thank the people who, during the several months in which this endeavor lasted, provided me with useful and helpful assistance. First of all, I would like to thank Milica for the opportunity she has given to me when, already one year ago, she accepted my application to develop my master thesis under her advice, and also for her energetic attitude and tending-to-infinite patience. I would also like to thank Justin for sharing with me the simulator; I would have not been able to start obtaining interesting results that fast without it. In any case, working at the MIT Sea Grant College has been a pleasure thanks to the people that research here and *buoy* the department. Thanks Janice for your incalculable help during these nine months.

I would also like to thank the ETSETB faculty that during the years of the degree has made my curiosity for knowledge and love for technology grow, as well as for offering these opportunities to their students. Thanks Montse for your fast answered emails.

Living in Cambridge would not have been that easy and exciting without my friends: Alberto, Alex, Carles, Carlos, Cris, Edu, Gerard, Happy Lau, John, Jordi, Juan, Laura, Linda, Marc, Marc Duo, Marta, Popi and Tere, among others. I have not forgotten you Patri, don't worry, but you deserve to be mentioned out of the list.

Last but not least, I would like to give a very special thanks to my family for the endless Skype calls, your visits during this period, and the freedom you gave me the day we all decided that coming to MIT would be a great asset for my personal career.

Part I

Distributed Power Control for Underwater Acoustic Networks

Introduction

Multi-hopping is a well-established transmission technique in wireless communication systems. This concept can usually be related to high density sensor or ad hoc networks, in which low-cost battery-powered nodes should minimize their energy consumption without compromising the network connectivity and the ability to deliver data to a final destination. In underwater acoustic networks, multi-hopping offers not only the benefits of power savings, but also the possibility to utilize a greater per hop data rate. This property is a consequence of the fact that the useful acoustic bandwidth depends on the transmission distance, increasing as the distance shortens [1]. The capacity of an acoustic relay link thus increases with the number of hops used to span a given distance [2].

The analysis presented in [2] is obtained for a noise-limited scenario, i.e. it does not take into account the presence of interference. As such, it serves as an upper bound on all practical systems in which the channel access must be regulated, in either a deterministic or a random fashion. The capacity of a cellular underwater network where multiple access is regulated either by TDMA or FDMA was assessed in [3]. The effects of interference on the system capacity in a contention-based acoustic network have been assessed in [4], showing similar results. In [5], the design of minimum energy routes is assessed, showing that in dense networks, there is an optimal number of hops over which the system performance does not improve.

In this first part of the document, we focus on random channel access for underwater acoustic networks, and address the design of discrete power control in light of minimum energy per bit consumption. Two MAC protocols are considered to verify the design correctness: the simple Carrier Sensing ALOHA (CS-ALOHA), and a recently proposed virtual carrier sensing method, the Distance Aware Collision Avoidance Protocol (DACAP) [9]. In a multi-hop scenario, the performance of both protocols depends on the transmission power. Simply stated, too little power may lead to a loss of connectivity, while too much power causes unnecessary interference which prolongs the contention phase. The increasing levels of interference cause repeated transmissions with CS-ALOHA, thus increasing the overall energy consumption. With DACAP, harmful collisions (those between data packets from nearby sources) are entirely avoided, so there is no energy wasted on repeated transmissions, but longer waiting times become necessary.

Chapter 1

The Underwater Channel

Acoustic communications are the typical physical layer technology in underwater networks. Radio waves propagate at long distances through conductive sea water only at extremely low frequencies (30-300 Hz), which require large antennas and high transmission power. Optical waves do not suffer from such high attenuation, but are affected by scattering. Moreover, transmission of optical signals requires high precision in pointing the narrow laser beams. Thus, links in underwater networks are based on acoustic wireless communications.

The main inherent characteristics of the underwater acoustic channel are summarized in the following sections.

1.1 Acoustic Propagation

1.1.1 Attenuation

Attenuation, or path loss that occurs in an underwater acoustic channel over a distance l for a signal of frequency f is given in dB by

$$10 \log A\left(l, f\right) = k \cdot 10 \log l + l \cdot 10 \log a\left(f\right) \tag{1.1}$$

where k denotes the spreading factor and $a\left(f\right)$ is the absorption coefficient, which is expressed empirically using Thorp's formula as [7]:

$$10 \log a(f) = 0.11 \frac{f^2}{1+f^2} + 44 \frac{f^2}{4100+f^2} + 2.75 \cdot 10^{-4} f^2 + 0.003 \tag{1.2}$$

where f is in kHz. This formula is generally valid for frequencies above a few hundred Hz, our range of interest. For lower frequencies, the following formula may be used:

$$10 \log a(f) = 0.002 + 0.11 \frac{f^2}{1+f^2} + 0.011 f^2 \tag{1.3}$$

The absorption coefficient increases rapidly with frequency, thus imposing a limit on the maximal usable frequency for an acoustic link of a given distance.

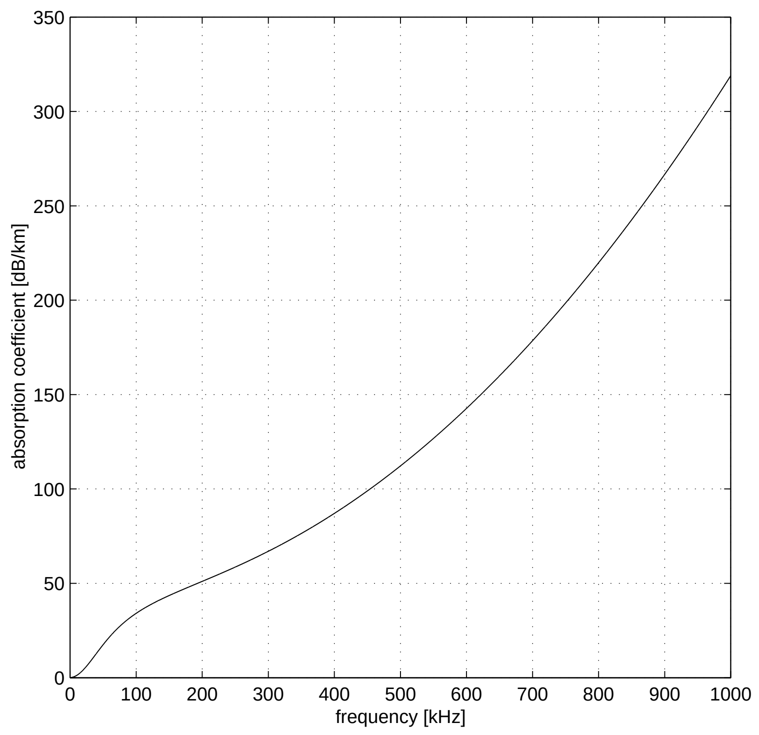

Figure 1.1: Absorption coefficient, $a(f)$ [dB/km].

1.1.2 Noise

The ambient noise in the ocean can be modeled using four different sources: turbulence, shipping, waves and thermal noise. The following empirical formulas give the p.s.d. of the four noise components in dBreμPa per Hz as a function of frequency in kHz [8]:

$$\begin{aligned} 10\log N_t(f) &= 17 - 30\log f \\ 10\log N_s(f) &= 40 + 20(s-0.5) + 26\log f - 60\log(f+0.03) \\ 10\log N_w(f) &= 50 + 7.5\omega^{1/2} + 20\log f - 40\log(f+0.4) \\ 10\log N_{th}(f) &= -15 + 20\log f \end{aligned} \tag{1.4}$$

Turbulence noise influences only the very low frequency region, f <10 Hz. Noise caused by distant shipping is dominant in the frequency region 10 Hz - 100 Hz, and it is modeled through the shipping activity factor s, whose value ranges between 0 and 1 for low and high activity, respectively. Surface motion, caused by wind-driven waves (ω is the wind speed in m/s) is the major factor contributing to the noise in the frequency region 100 Hz - 100 kHz, our frequency range of interest. Finally, thermal noise becomes dominant for f >100 kHz.

The overall p.s.d. of the ambient noise, $N(f) = N_t(f) + N_s(f) + N_\omega(f) + N_{th}(f)$ is illustrated in Fig.1.2, for two different cases of wind, with varying degrees of shipping activity in each case. Noise decays with frequency, thus limiting the useful acoustic bandwidth from below. In our range of interest, it can be approximated as:

$$10\log N(f) = 50 - 18\log f \tag{1.5}$$

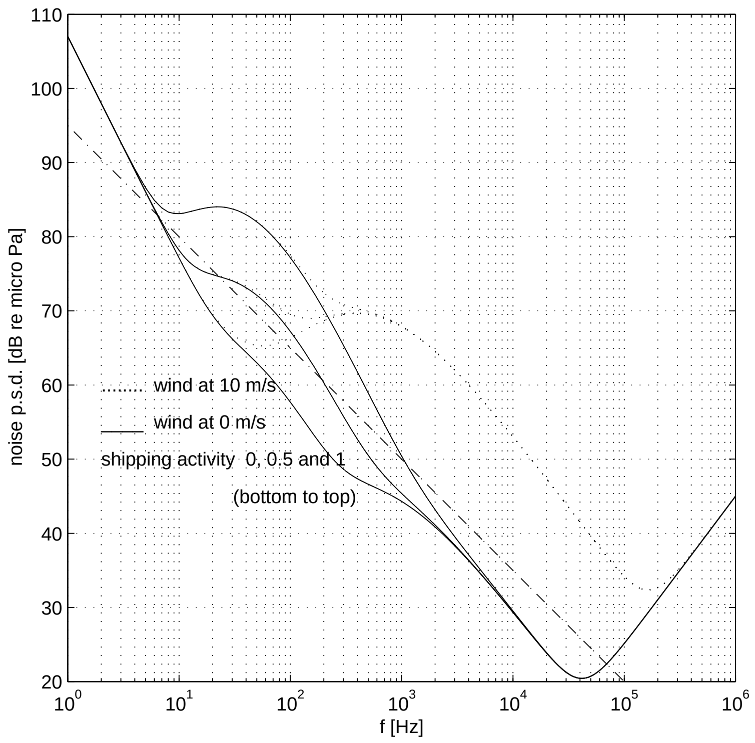

Figure 1.2: Power spectral density of the ambient noise, $N(f)$ [dBreμPa]. The dash-dot line shows the approximation (1.5).

1.1.3 Propagation Delay

The nominal speed of sound in water is 1500m/s, which is 200,000 lower than the speed of electromagnetic waves in open-air (300,000 km/s). This causes long propagation delays in underwater acoustic systems. While in wireless radio networks delays between two nodes are on the order of several microseconds, in underwater acoustic networks they can reach several seconds for each hop through the network.

1.1.4 Doppler Effect and Multi-path Propagation

The Doppler effect is caused by the relative motion of the transmitter-receiver pair. This effect causes a shifting in the transmitted signal frequency and time scaling. These effects are governed by the factor $\frac{v_r}{c}$, where v_r is the relative velocity between transmitter and receiver, and c is the signal propagation speed. Since propagation speed is much lower in the underwater acoustic channel, the Doppler effect will also be more pronounced. The frequency shift for an underwater acoustic channel whose nodes are moving at 2 m/s is on the order of several Hertz, which is an important shift in the frequency range of interest.

Multi-path propagation is another common problem of the underwater acoustic channel. Transmitted signal follows different paths to arrive to the receiver, thus causing the reception of different echoes of the same signal, with different phase and amplitude. This effect is caused by reflection and refraction of the acoustic waves. Reflective multi-path is a phenomenon usual in shallow water communications and is caused by the reflections of the signal at the water surface and bottom. Refractive multi-path is dominant in deep water communications and it is caused by the changes in the speed of sound at different depths.

These effects are a subject of study in signal processing and won't be taken into consideration in this study.

1.2 Resource Allocation

In contrast to the radio-frequency spectrum, the underwater channel is not regulated (yet). However, taking into account both the acoustic path loss and the ambient noise, the frequency allocation possibilities are not that numerous. In addition, currently available transducers impose a further limitation on the signal bandwidth.

In this section, we introduce the frequency allocation methodology that we will use throughout the document and address the impact of changing the center frequency f_c and its corresponding bandwidth B.

1.2.1 The AN product and the SNR

The narrow-band signal to noise ratio (SNR) is given by

$$SNR\left(l,f\right)=\frac{S\left(f\right)\Delta f/A\left(l,f\right)}{N\left(f\right)\Delta f}=\frac{S\left(f\right)/A\left(l,f\right)}{N\left(f\right)} \tag{1.6}$$

where $S\left(f\right)$ is the power spectral density of the transmitted signal and Δf is a narrow frequency band around f. The factor $1/A\left(l,f\right)N\left(f\right)$ is illustrated in Fig.1.3. For each transmission distance l, there clearly exists an optimal frequency $f_o\left(l\right)$ for which the maximal narrow-band SNR is obtained.

1.2.2 Bandwidth definition

We define the 3dB bandwidth $B_{3dB}\left(l\right)$ as the range of frequencies around $f_o\left(l\right)$ for which $A\left(l,f\right)N\left(f\right) < 2A\left(l,f_o\left(l\right)\right)N\left(f_o\left(l\right)\right)$. The optimal frequency $f_o\left(l\right)$ and its corresponding $B_{3dB}\left(l\right)$ as a function of the transmission distance l are plotted in Fig.1.4. The center frequency $f_c\left(l\right)$ and the bandwidth/center-frequency ratio are also included. As the transmission distance is reduced, the optimal frequency is higher, as well as its corresponding 3dB bandwidth.

1.2.3 Transmission Power

Assuming that the transmitted signal p.s.d. is flat across the 3dB bandwidth, the transmission power necessary to provide a target SNR$_0$ at a distance l from the source is determined as:

$$P\left(l\right)=SNR_0B_{3dB}\left(l\right)\frac{\int_{B_{3dB}(l)}N\left(f\right)df}{\int_{B_{3dB}(l)}A^{-1}\left(l,f\right)df} \tag{1.7}$$

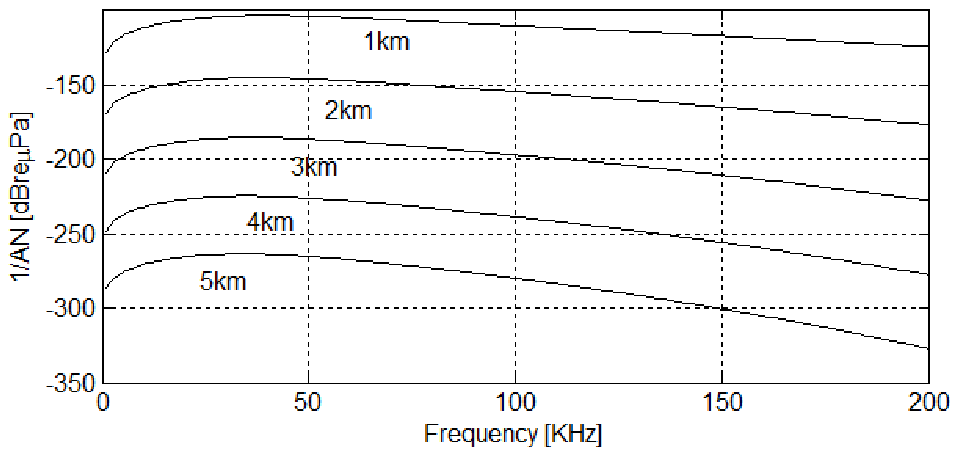

Figure 1.3: Frequency-dependent part of the narrow-band SNR, $1/A\,(l, f)\,N\,(f)$, for different transmission distances (spreading factor k=1.5).

1.2.4 Interference: Minimum Power versus Minimum Energy

When several nodes are sharing the same channel, interference must be taken into account in the system design. As pointed out in [4], the high absorption of the underwater acoustic channel (1.1) plays a key role in multi-hop communications.

The acoustic absorption coefficient increases with the system center frequency (Fig.1.1). This implies a higher transmission power needed to cover the same link distances (1.7). At the same time, interference suffers a greater interference too. Therefore, the number of received packets that are discarded due to interference is expected to decrease and, consequently, the energy lost in retransmissions is reduced.

When moving to higher center frequencies, the available bandwidth is greater. A greater bandwidth supports transmitting at higher bit-rates, which has a twofold effect: first, the total energy consumption is reduced because the transmission time is shorter, and second, shorter packets are less likely to collide (Fig.4.4b).

Due to the complexity of addressing these relations analytically, we will refer back to them in the simulation results included in Sec.4.3.

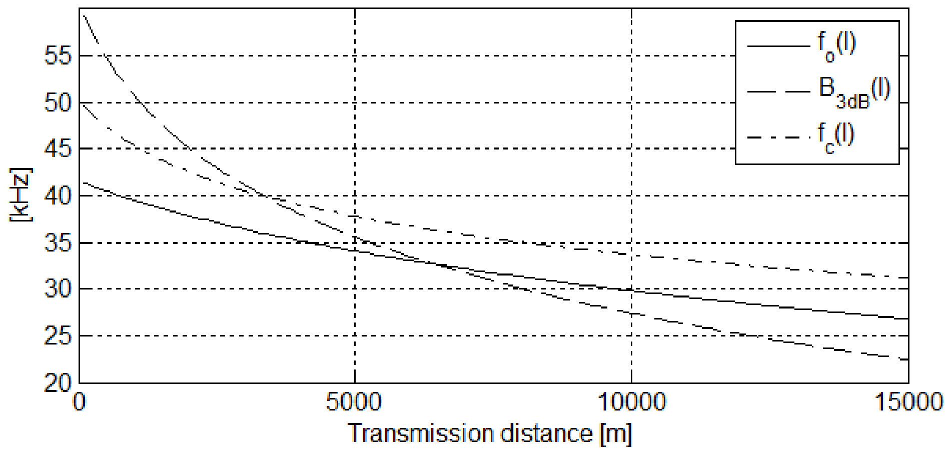

Figure 1.4: Optimal frequency $f_0\,(l)$, 3dB bandwidth $B_{3dB}\,(l)$ and center frequency $f_c\,(l)$ (spreading factor k=1.5).

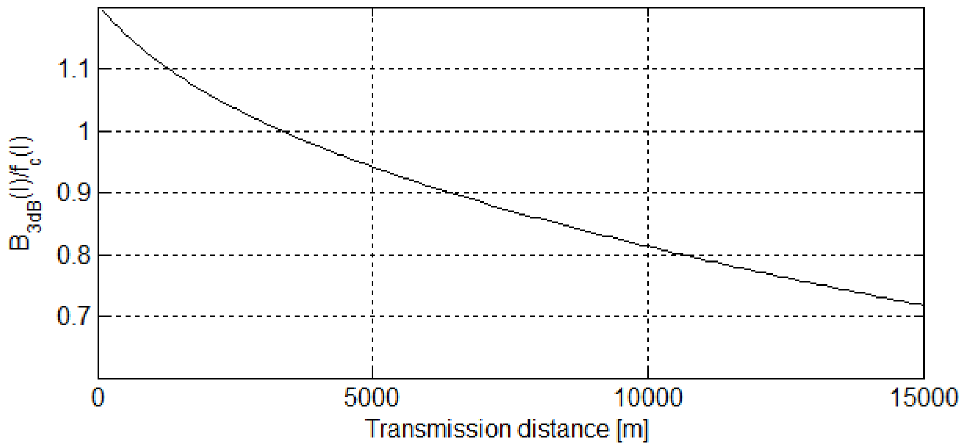

Figure 1.5: Bandwidth-center frequency ratio as functions of transmission distance l (spreading factor k=1.5).

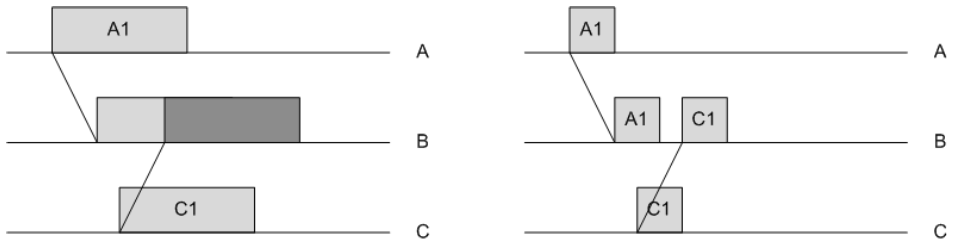

Figure 1.6: Illustration of the benefits of increasing the transmission bandwidth: shorter packets are less likely to collide.

Chapter 2

Power Control

We look at an underwater acoustic network containing both static and mobile nodes. The nodes are able to switch their transmission power P over a finite set of power levels, ranging from some minimum to a maximum, $P_0, P_1, ..., P_{N-1}$.

2.1 Network topology and maximal power

We define the maximal transmission power as the minimum that still guarantees connectivity between any two nodes in the network [10]. In light of multi-hop communications, two nodes are virtually connected if there exists at least one path of physically connected nodes between them. Two nodes are physically connected if they can reach each other with a target SNR_0.

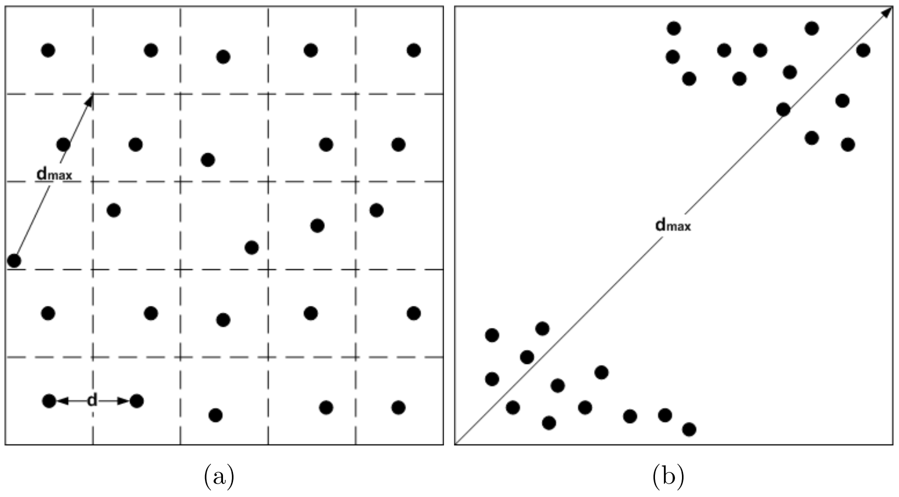

Figure 2.1: Two possible scenarios: a) random location of the nodes within a grid and b) completely random location of the nodes.

We focus on the scenario presented in Fig.2.1a. In this case, the space over which the network is deployed can be divided into a virtual grid, where each square contains one node randomly placed inside it. The network node density is:

$$\rho = \frac{n}{S} = \frac{1}{d^2} \tag{2.1}$$

where S is the network area and n is the total number of nodes in it. The maximum power P_{N-1} is associated with a transmission distance equal to

$$d_{max} = \sqrt{5}d = \sqrt{\frac{5}{\rho}} \tag{2.2}$$

For a fixed area S equal to 100 km^2, the maximum transmission range as a function of the number of nodes is shown in Fig.2.2.

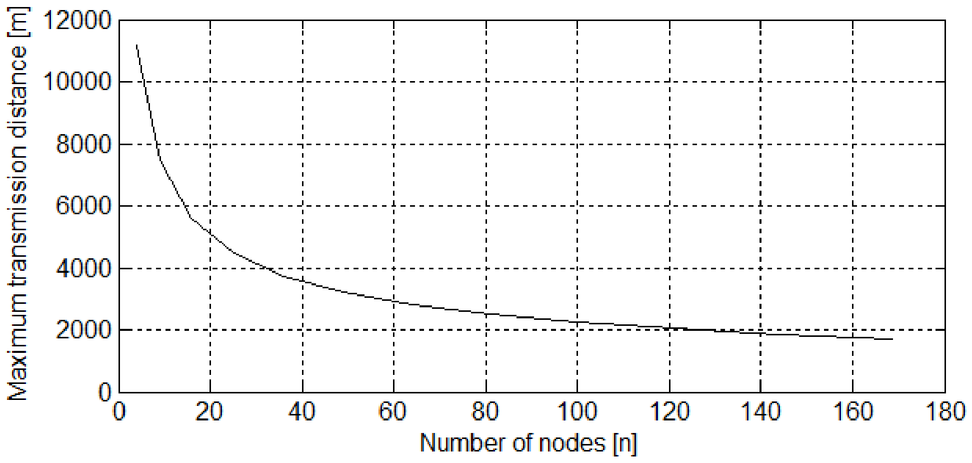

Figure 2.2: Maximum transmission distance as a function of the number of nodes deployed over a fixed area S=100 km^2 (2.2).

There are other situations which may be of interest. For example, the n nodes can be randomly placed over the entire area S as shown in Fig.2.1b. In this case, the maximum transmission power does not scale with the network node density. The worst case, illustrated in the figure, implies that the maximum transmission distance can be much greater than the one given in (2.2).

In what follows we will assume the first network topology, as one that is more likely to occur in practice. If this assumption is violated, i.e. two nodes on a path become separated by more than d_{max} given in (2.2), some parts of the network will loose connectivity. However, in a network that contains mobile nodes, this situation may be only temporary.

2.2 Step size between levels

Assuming a uniform separation of power levels in dB, the step size Δ between two consecutive levels is defined by

$$P_0 = P(d_0), P_n = \Delta^n P_0 = P(d_n), P_{N-1} = P(d_{max}) \tag{2.3}$$

Alternatively, the separation between two consecutive levels can be defined in terms of a uniform increase in the coverage distance, δ:

$$P_0 = P(d_0), P_n = P(d_0 + n\delta), P_{N-1} = P(d_{max}) \tag{2.4}$$

Fig.2.3 shows the two power distribution patterns when using $N = 4$ levels, in a network with 64 nodes over 100 km^2. The step size is defined in dB as:

$$10 \log \Delta = \left(10 \log P\left(d_{max}\right) - 10 \log P\left(d_0\right)\right)/N = 14dB \tag{2.5}$$

and in meters as:

$$\delta = \left(d_{max} - d_0\right)/N = 765m \tag{2.6}$$

In this example, the differences between the two power allocation patterns are small, i.e. $P\left(d_n\right) \approx P\left(d_0 + n\delta\right)$.

In Fig.2.4, the transmission power is plotted versus distance for two different center frequencies. Both curves show a nearly constant slope, for the range of distances considered, which makes the previous affirmation valid in our frequency range of interest. Due to the convenience of defining the power levels in terms of a uniform increase in the distance, we use this definition in what follows.

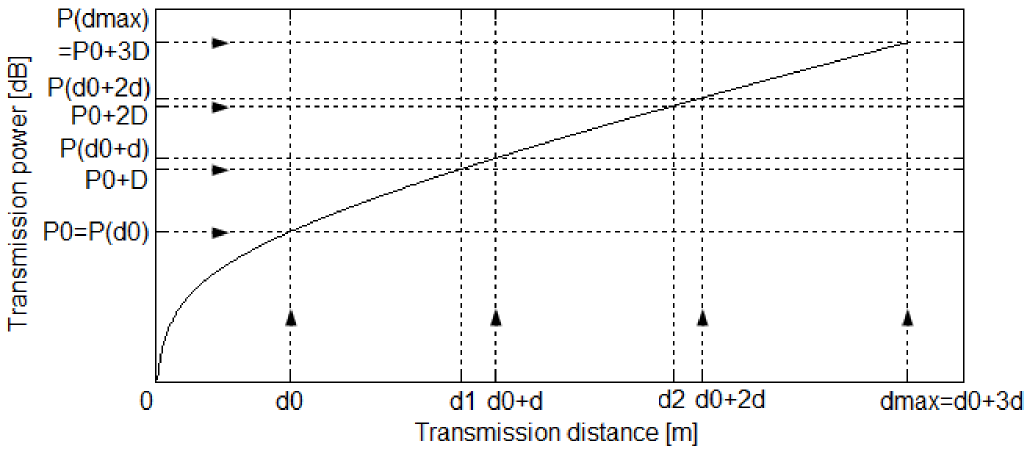

Figure 2.3: Power levels and corresponding distances for the two strategies in (2.3) and (2.4) using a step size of 14dB and 765m, respectively, in a network with 64 nodes deployed over 100 km^2 in a grid-uniform manner (d_0=500 m, d_{max}=2795 m, f_c=40 kHz,B=30 kHz, SNR$_0$=20 dB).

We can also think of a non uniform distribution of the power levels. For example, in a completely random scenario (Fig.2.1b), it makes sense to keep the maximum power level to cover d_{max}. However, in order to avoid using more power than necessary for the more likely short distance communications, the $N - 1$ remaining levels can be separated by a smaller step size, for example, $\delta = \left(d_{max} - d_0\right)/2N$ (see Fig.2.5).

2.3 Number of power levels

The last parameter required to completely characterize the power control is the number of power levels N. Increasing the number of levels allows finer tuning of the power; however, a small number of levels is of interest to practical implementation. We conjecture that there is

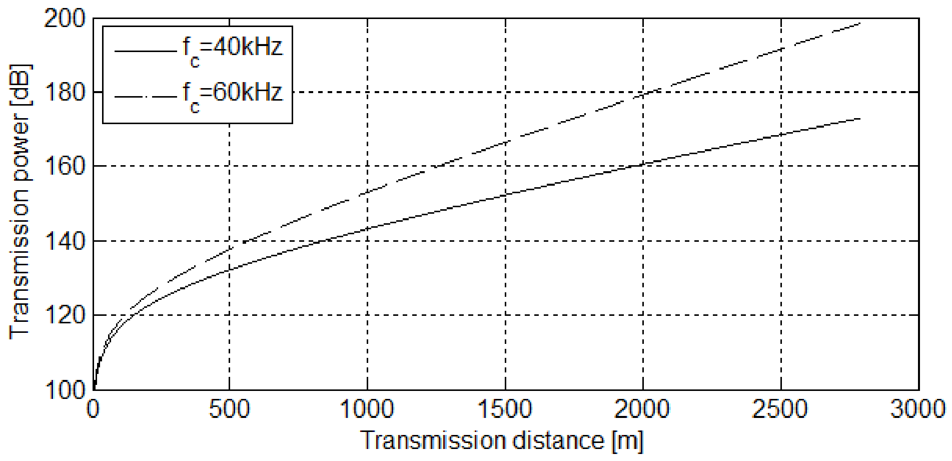

Figure 2.4: Transmission power as a function of distance for two center frequencies (B=30 kHz, SNR$_0$=20 dB).

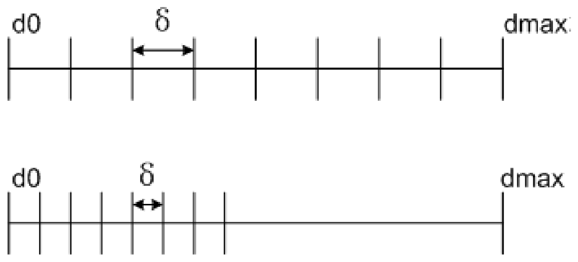

Figure 2.5: (Top) Uniform separation of 8 power levels and (Bottom) Non uniform separation of the same power levels.

an effect in energy savings of diminishing returns when it comes to increase the number of levels beyond some point. We will assess this issue in the Sec.4.2.

Chapter 3

Medium Access Control

Two MAC protocols are considered: on the one hand, the simple carrier sensing variation of 1-persistent ALOHA and, on the other hand, the Distance Aware Collision Avoidance Protocol, a recently proposed virtual carrier sense like protocol [9].

3.1 Carrier Sensing ALOHA

A node using CS-ALOHA will listen to the channel before transmitting, and if finds the channel idle, the data transmission will start. Taking into account the half-duplex operation of current acoustic modems, the transmitter cannot detect collisions, and it will always transmit the entire data packet. The transmitter will deduce that its transmission has collided if after a certain waiting time, it has not received a positive acknowledgement. In that situation, it will retransmit following the same procedure, unless the maximum number of retransmission attempts has been reached.

This MAC protocol stands out for its simplicity and average end-to-end delay, but it is not the best option in terms of energy consumption, due to the energy lost in retransmissions.

3.2 Distance-Aware Collision Avoidance Protocol

DACAP is a virtual carrier sense like MAC protocol that has recently been proposed for underwater networks [9]. It implements an exchange of short control packets for avoiding data packet collisions, thus maximizing the network throughput.

The protocol is based on the following steps:

- Upon receiving a request to send (RTS), a node sends a clear to send (CTS), and waits for a data packet. If during the waiting time, another RTS is overheard, the node sends a short warning to its partner.

- Upon receiving a CTS, the transmitter waits for those nodes whose attempts to transmit may result in collisions. If during this time, another CTS is overheard or a warning packet arrives, the transmission is deferred by a random back-off time. Otherwise, the transmission of the data packet proceeds.

This protocol was shown to improve the system performance in terms of energy per bit consumption, at the expense of increasing the average end-to-end delay due to postponements.

3.3 Integration of the Power Control

When using a discrete power control, the routing protocol, the medium access control and the physical layer functionalities are tightly coupled. By choosing a path, the routing protocol decides which power level should be used. The medium access control should then adapt specific parameters, such as waiting, or back-off times, according to the new transmission distance. Finally, switching of the transmission power to the new level occurs at the physical layer.

In our simulations, we assume that the network topology is known, i.e. all nodes know all others' positions. Then, routes are geographically pre-established using Dijkstra's algorithm, in which the cost between two nodes is defined as the power level required to guarantee physical connectivity between them. The target SNR_0 for physical connectivity is set to be 20dB in our simulations. A node will directly select the minimum power level required to reach the next node along the packet route.

A more general approach, in which each transmitting node only needs its current position and the position of the final destination, is introduced in the second part of the thesis.

3.3.1 Implicit Acknowledgement

Apart from an end-to-end acknowledgement which may be generated at the transport or application layer, each intermediate node expects a positive acknowledgement from the current receiver. If nodes use omnidirectional transducers, which is the case of mobile nodes, the transmitter can deduce that its last data transaction has been properly completed if it overhears its own packet being transmitted to the next relay. However, this may not be always possible. If the power level used to reach the next node is lower than the one used for the previous transmission, the acknowledgement should be sent explicitly using a higher power level. The same should be done when the packet reaches its final destination.

At the same time, if for any reason a node receives a RTS from the same transmitter for a packet that has been successfully transmitted (each packet has an unique ID), an acknowledgement is explicitly sent, avoiding the long data packet retransmission.

Fig.3.1 illustrates the concept of implicit acknowledgement for the two considered protocols. The source node, S, is transmitting to the destination node D through the intermediate node $IN1$. This can be reached with the power level P1, which covers a distance equal to $d1$. $IN1$ does not need to explicitly send an acknowledgement to the source S if $d1 \leq d2$. $T_{control}$ and T_{data} stand for the duration of the control and data packets for a given bit-rate and c is the underwater speed of sound.

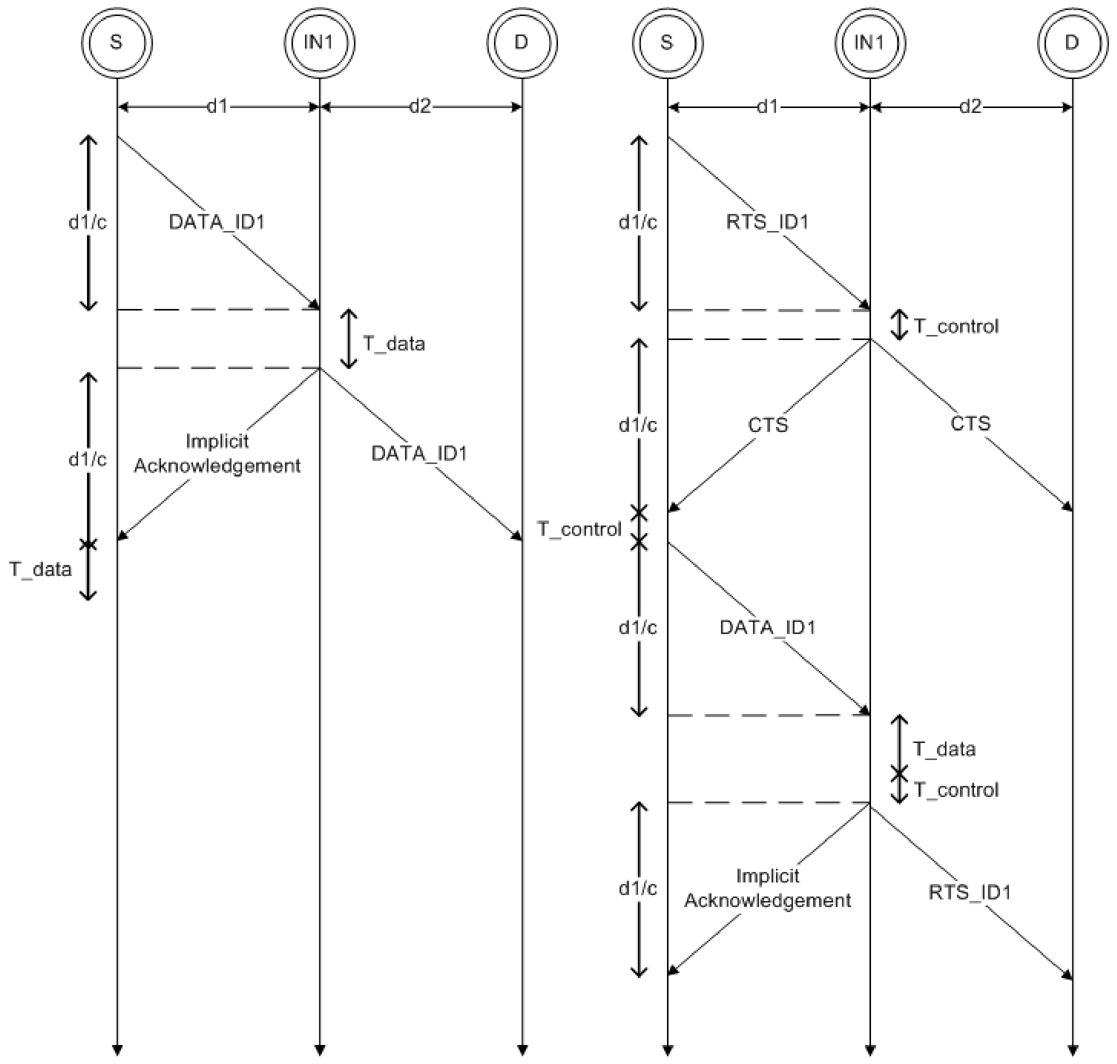

Figure 3.1: Implicit acknowledgement example when using CS-ALOHA (left) and DACAP (right).

Chapter 4

Simulation Results

To verify the concepts introduced, we have used a discrete-event underwater acoustic network simulator implemented in standard Python. It is described in the Appendix.

The simulation scenario corresponds to the one shown in Fig.2.1a, in which the maximum transmission range scales with the node density. The network is composed of four active nodes, a common sink in the center, and a varying number of relay nodes. A Poisson distribution with an average packet generation rate λ=1 packet/min is assumed. The relay nodes are not generating packets.

The system performance is measured in terms of average energy per bit consumption, which is given by

$$E_b = \frac{P_n}{\alpha B_{3dB}} \tag{4.1}$$

where α stands for the bandwidth efficiency and it is assumed to be constant and equal to 1bit/Hertz. The total number of collisions as well as the average packet end-to-end delay are also measured to illustrate the performance.

4.1 Effects of relay density

Before focusing on the effects of different power distribution schemes and frequency allocation patterns, the benefits of increasing the relay density are analyzed. Fig.4.1 shows the average energy per bit consumption when varying the number of relays. For each node density, d_{max} is obtained from (2.2). The system center frequency and the available bandwidth are determined following the principles outlined in Sec.1.2, as $f_c(d)$ and $B_{3dB}(d)$, where d is the average internode distance. The number of available power levels is $N = 8$.

As the number of nodes increases, the average energy per bit consumption is clearly reduced. This is due not only to the fact that the transmission power is lower, but also to the fact that the bandwidth available to shorter links is greater. We also observe that there is a density of relay nodes above which there is no improvement in the system performance in terms of energy per bit consumption. The differences between CS-ALOHA and DACAP are negligible when compared to the overall system improvement.

In addition to the grid-like scenario, Fig.4.1 illustrates the results for a completely random scenario. The distance d_{max} is determined as in Fig.2.1b to guarantee connectivity. Specifically, the maximum power level is defined as $P_{N-1} = P(d_{max})$, but, in order to avoid using more power than necessary for the more likely short distance communications, the $N-1$ remaining levels are separated by a smaller step size, $\delta = (d_{max} - d_0)/2N$. In this case, the energy per bit consumption decays with the number of relay nodes, but at a smaller rate. The differences between both scenarios are smaller in highly dense networks, as situations in which the maximum power level is required are less likely to occur.

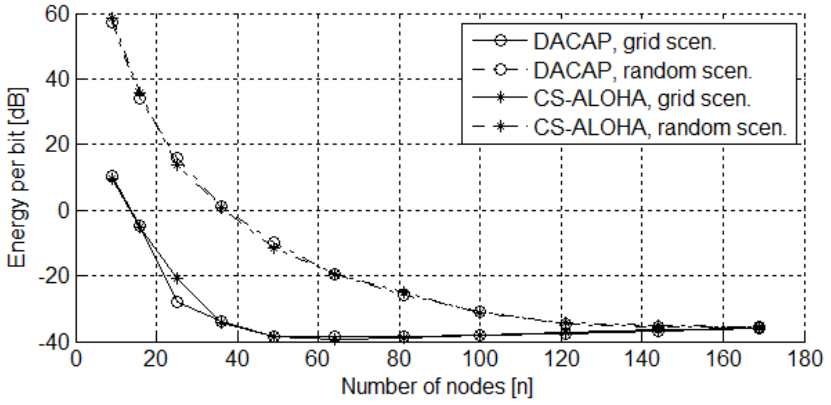

Figure 4.1: Energy per bit consumption for CS-ALOHA and DACAP using power control with 8 uniformly distributed levels, when increasing the number of relay nodes.

4.2 Effect of the number of levels N

When increasing the number of relay nodes, the energy consumption is reduced mainly due to the fact that the maximum transmission distance scales with the network density as (2.2). Therefore, even for a constant transmission power, i.e $N = 1$, the energy consumption is clearly reduced.

Using more than one power level allows the system to allocate the power in a better way. Only those nodes that require the highest transmission power will use it. By increasing the number of power levels, the power can be chosen more accurately, reducing the total power consumption, as well as interference.

Fig.4.2 shows the average energy per bit consumption when the number of relay nodes is increased for a varying number of power levels. It reveals that using more than 4 levels does not significantly improve the system performance, which support the conjecture we made in Sec.2.3. Therefore, we can use $N = 4$ power levels for the network under consideration as a good compromise between energy per bit consumption and implementation complexity.

4.3 Resource Allocation

In the previous sections, both f_c and B_{3dB} have been optimized for power consumption according to the channel model introduced in Sec.1.2. Here we illustrate the effect of independently changing these two parameters when using power control with 4 uniformly spaced levels.

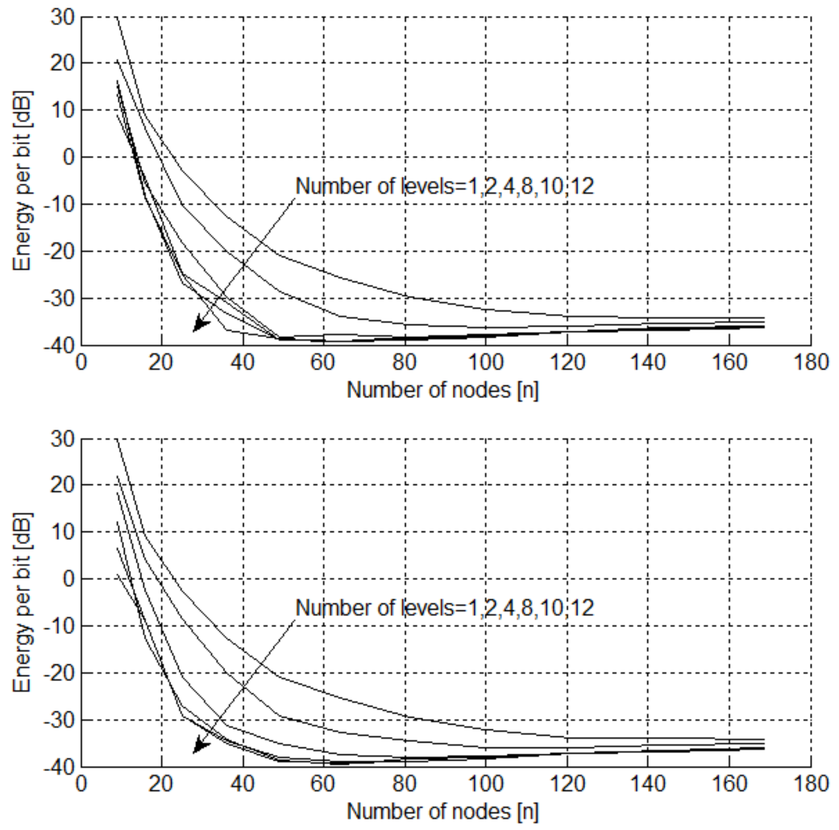

Figure 4.2: Energy per bit consumption for CS-ALOHA (top) and DACAP (bottom) when increasing the number of power levels.

4.3.1 Center Frequency, f_c

Fig.4.3 shows the energy per bit consumption, number of collisions and average packet end-to-end delay for the two MAC protocols under study for two choices of center frequency: f_c=40 kHz and f_c=60 kHz.

At a higher center frequency, the power consumption for the same inter-node distance is increased because the acoustic path loss is higher. For this reason, the performance is better at 40 kHz than at 60 kHz, but this is only so for lower densities. In dense networks, where the transmission distance is small, the energy per bit consumption is nearly the same (Fig.4.3a).

As the center frequency increases, the interference coming from other nodes also suffers a greater attenuation. Therefore, when using CS-ALOHA, the total number of collisions can be reduced by increasing the center frequency. This effect can be more clearly seen at high node densities, for which the number of collisions is usually higher (Fig.4.3b). Because the transmission power scales with the network density, the total energy lost in collisions is small, and the energy performance of CS-ALOHA is close to that of DACAP. At the same time, the end-to-end delay clearly benefits from this reduction in the number of collisions due to a smaller number of retransmissions (Fig.4.3c). When using DACAP, a reduction in interference translates into a reduction in the waiting time necessary to avoid collisions. This effect is easier to identify at low densities, where the waiting time, which is proportional to the propagation delay, is higher.

4.3.2 Bandwidth, B

Fig.4.4 shows the energy per bit consumption, number of collisions and average packet end-to-end delay for the two MAC protocols under study for two choices of available bandwidth: B=30 kHz and B=1 kHz.

The energy per bit consumption benefits from a greater bandwidth for two reasons. First, the bit duration $1/B$ is reduced; thus, the energy per bit reduces too, independently of the MAC protocol used (Fig.4.4a). Secondary, packets are shorter and, therefore, less likely to collide (Fig.4.4b).

This combined effect is what allows a very simple MAC protocol, such as CS-ALOHA, in which collisions are not prevented, to achieve an overall energy per bit performance very close to the more sophisticated collision avoidance DACAP. This fact encourages transmission at high bit-rates: even if the application does not require it, the system performance in terms of energy consumption and end-to-end delay will improve.

4.3.3 Combined Effects

When choosing the center frequency and the bandwidth using the minimum power approach introduced in Sec.1.2, both the optimal center frequency and its corresponding 3dB bandwidth increase with the node density. In other words, the average inter-node distance is shorter for a higher node density (Fig.2.2), and, hence, the optimal frequency and the available bandwidth are higher (Fig.1.4). As the bandwidth increases, both energy per bit consumption and end-to-end delay are reduced because the packets are shorter and they are less likely to collide.

Fig.4.5 shows the energy per bit consumption, number of collisions and average packet end-to-end delay for the two MAC protocols under study, for the optimal choice of f_c and B_{3dB}.

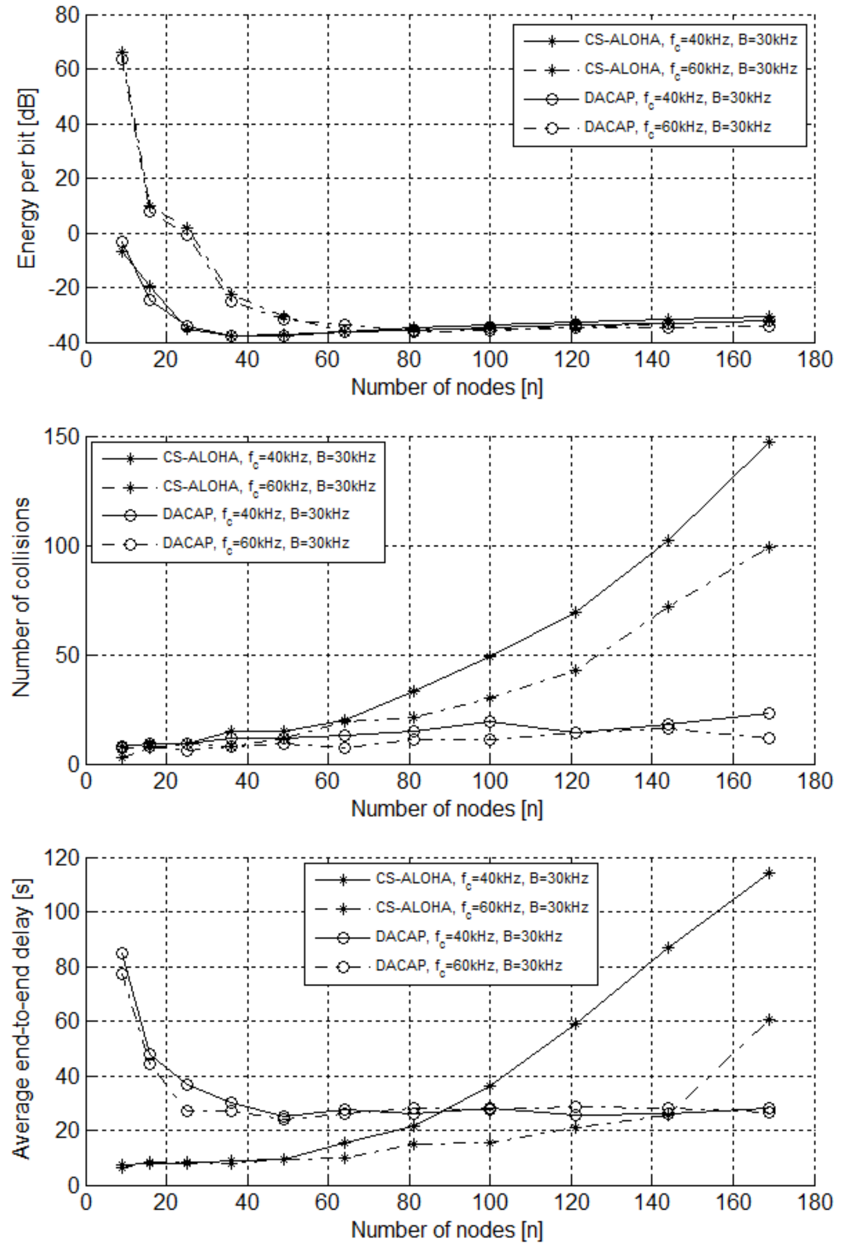

Figure 4.3: Energy per bit consumption, number of collisions and average packet end-to-end delay for CS-ALOHA and DACAP for two choices of center frequency: f_c=40 kHz and f_c=60 kHz; N=4 levels.

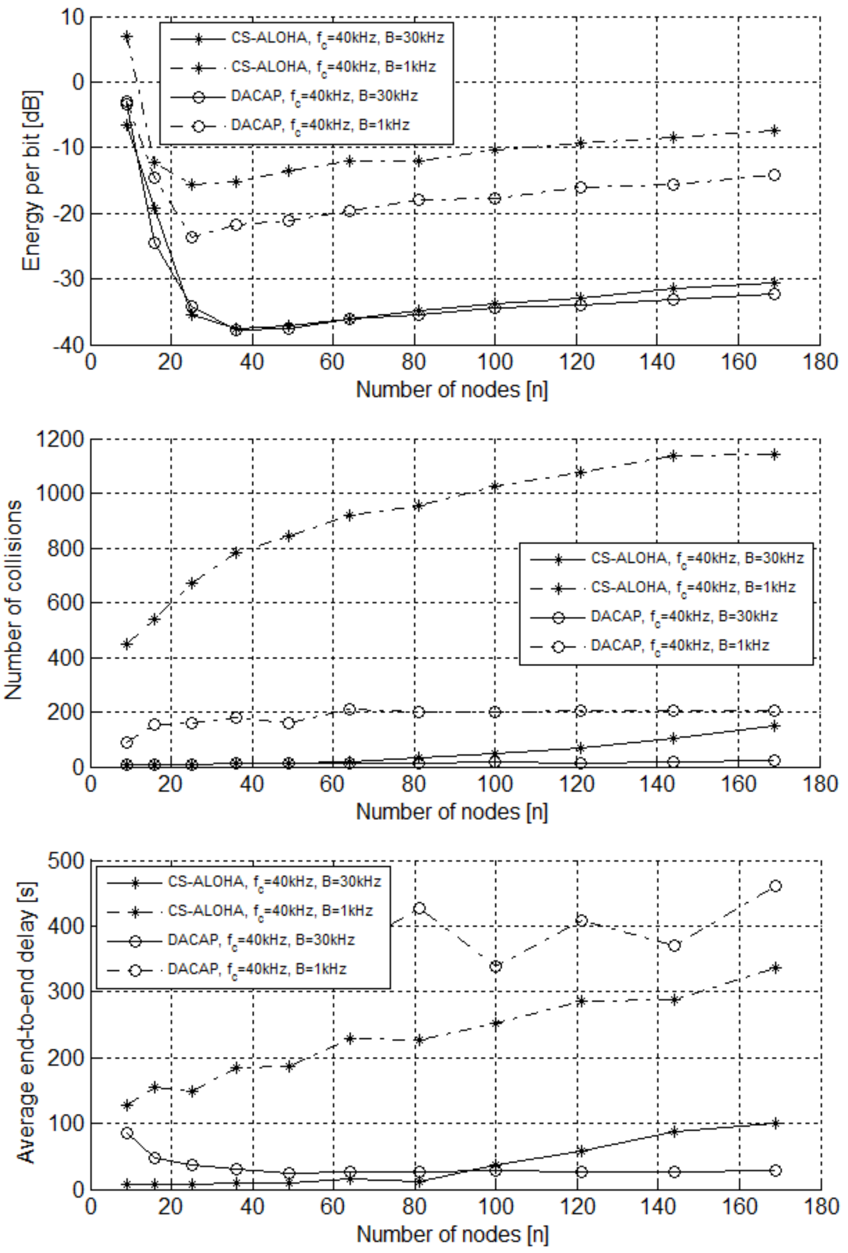

Figure 4.4: Energy per bit consumption, number of collisions and average packet end-to-end delay for CS-ALOHA and DACAP for two choices of bandwidth: B=30 kHz and B=1 kHz; N=4 levels.

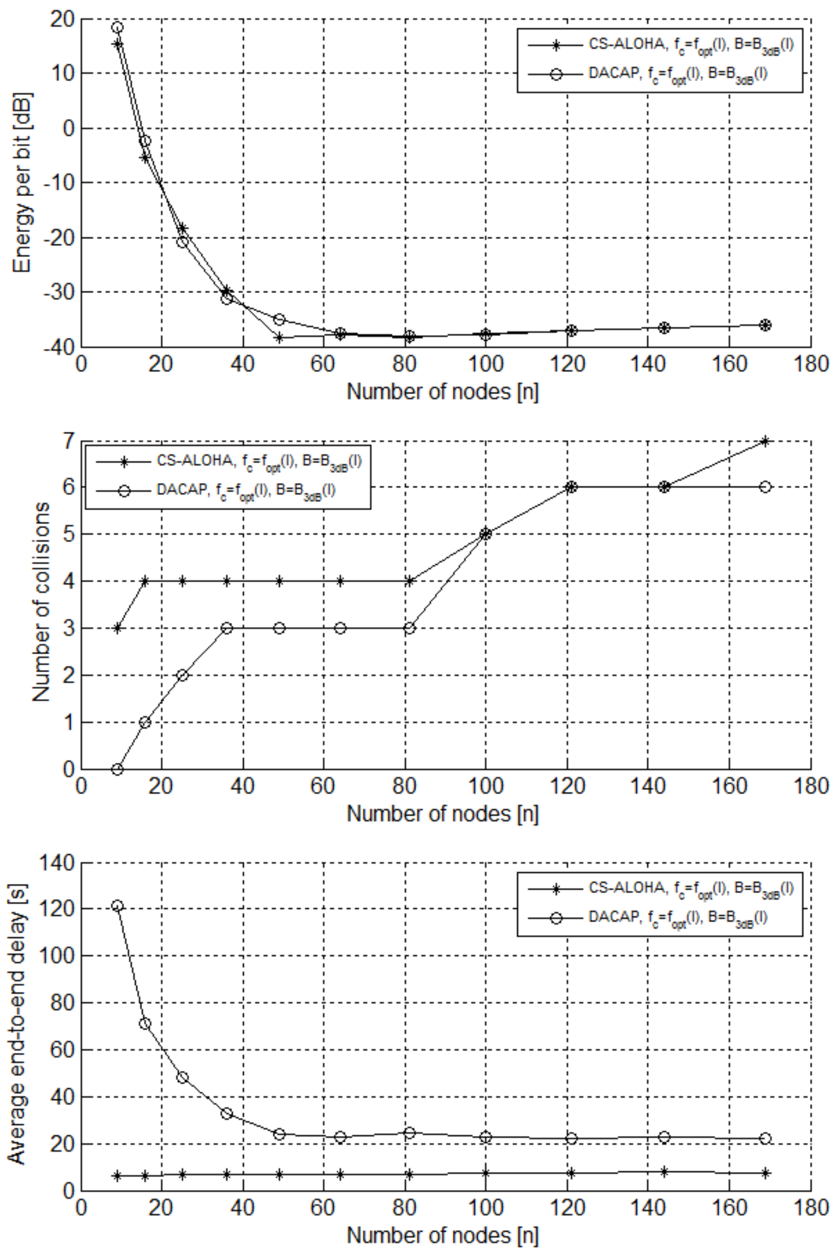

Figure 4.5: Energy per bit consumption, number of collisions and average packet end-to-end delay for CS-ALOHA and DACAP for the optimal choice of f_c and B_{3dB}; N=4 levels.

Chapter 5

Conclusions

Discrete power control was considered as a practical means for enabling multi-hop communications for scalable, large coverage in bandwidth-limited underwater acoustic networks. Different power allocation schemes as well as number of available levels were considered for varying network densities. For an example scenario, it was shown that four uniformly-distributed levels suffice to achieve energy consumption close to minimum and implementation complexity. This number is low enough to motivate a practical implementation of power control.

Due to the dependence of the acoustic path loss on both the distance and the frequency, shorter links are able to utilize higher center frequencies, allowing the system to exploit greater bandwidths. The center frequency and bandwidth were shown to have an effect on both MAC protocols considered. For a higher center frequency, the power required to make up for the greater acoustic path loss is higher, but the interference also attenuates more. With CS-ALOHA, this turns into a reduction in the number of collisions, whereas with DACAP, the necessary waiting time to avoid transmissions interfering nodes becomes shorter, thus effectively reducing the average end-to-end delay.

The main benefit however comes from the increase in the available bandwidth. The total energy consumed decreases not only because the transmission time per bit is shorter, but also because shorter packets are less likely to collide. Therefore, the total energy consumption due to retransmissions becomes smaller. At the same time, a reduction in the number of collisions, implies shorter end-to-end delay. This fact encourages us to always transmit using high bit-rates: even if the application does not require it, the system performance in terms of energy consumption and end-to-end delay will clearly be higher.

Most notably, these effects make CS-ALOHA performance comparable to that of DACAP in terms of energy per bit consumption, and better in terms of average end-to-end delay in high density networks. Hence, by optimizing the frequency allocation, it becomes possible to take full advantage of the simplicity of CS-ALOHA, which is otherwise compromised by channel latency.

Part II

Focused Beam Routing protocol for Underwater Acoustic Networks

Motivation

In the first part of this document, nodes were able to route and forward packets when following pre-established routes, which were defined in light of minimum power consumption. This can only be done when each node has a complete knowledge of the network topology.

In this second part, we propose a routing methodology, and evaluate its performance when coupled with power control. This routing technique assumes that nodes know their own locations. Such assumption is justified in underwater systems where fixed bottom-mounted nodes have location information upon deployment, while the mobile nodes, i.e. autonomous underwater vehicles (AUVs), are equipped with internal navigation systems. In addition, a source node also knows the location of its desired final destination, but not the locations of other nodes. This case is representative of an underwater network in which the distributed nodes are required to transmit to a common sink, or a set of sinks.

Without location information, a large number of broadcast or multicast queries may cause unnecessary network flooding, thus reducing the user perceived throughput. This is one of the main limitations in non geographical ad-hoc routing protocols. In proactive protocols (e.g., DSDV [11], OLSR [12]), or reactive protocols (e.g., AODV [13], DSR [14]), large signaling overhead and high latency, compromise the network performance.

The knowledge of location can eliminate this effect. In wireless sensor networks, location awareness has been previously considered leading to geographical routing protocols such as GeRaF [15, 16], a forwarding technique based on geographical location of the nodes involved and random selection of the relaying nodes via contention among the receivers. An integrated MAC/Routing protocol based on geographical information and which makes use of power control is introduced in [17]. In this case, a competition is triggered at each hop in order to select the next relay node, so that the most energy efficient one is chosen.

Routing protocols based on location information have been also designed explicitly for the underwater channel. In [18], a location aware variation of DSR in which link quality measurements are considered in the relay selection process is shown to reduce the system latency. In [19], the authors propose a vector-based forwarding protocol for sensor networks, in which a virtual transmission pipe is defined at each hop of the transmission path. In [5], the design of minimum energy routes is assessed, showing that in dense networks, there is an optimal number of hops over which the system performance does not improve.

The routing methodology that we are proposing is described in the following section.

Chapter 6

Routing and Power Control

To illustrate the routing protocol, let us refer to the example of Fig.6.1. Shown in this figure is a network of nodes, distributed in an arbitrary manner across some area. A simple two-dimensional scenario can be envisioned without the loss of generality.

Referring to Fig.6.1, let us assume that node A wants to transmit to node B. To do so, node A will issue a request to send (RTS) to its neighbors. This request is a short control packet that contains the location of the source node (A) and of the final destination (B). Note that this is in fact a multicast request.

The initial transaction is performed at the lowest power level and the power is increased only if necessary. Power control is performed as an integral part of routing and medium access control. We assume open loop power control, in which the transmitting node decides which power level to use, rather than being instructed explicitly by a receiving node. In a practical system, power control will be achieved by choosing from several discrete levels. At the moment, we are not concerned with the exact way in which the power levels are determined, but simply assume that there is a finite number of increasing power levels, P_1 through P_N.

Corresponding to each power level P_n there is a transmission radius d_n. Only those that are within this radius are assumed to receive the signal at a level sufficient for detection. The signal of course propagates beyond this distance and can be overheard, but because of attenuation it cannot be detected. As such, it causes interference to other nodes, which will be taken into account when evaluating the system performance.

Returning to our example, let us draw an imaginary line between nodes A and B. All the nodes that receive A's multicast RTS, first calculate their location relative to the AB line. The objective in doing so is to determine whether they are candidates for relaying. Candidate nodes are those that lie within a cone of angle $\pm\theta/2$ emanating from the transmitter towards the final destination. If a node determines that it is within the transmitter's cone, it will respond to the RTS. Those nodes that are outside of the cone will not respond.

In our example, there are no nodes within the transmission cone that can be reached at the power level P_1. Hence, after an expected round-trip time ($2d_1/c$ for the power P_1), node A receives no responses. It now increases the transmission power to P_2, and sends a new RTS. In general, a transmitting node will keep increasing the power until it reaches someone, or until all power levels have been exhausted. If it cannot reach anyone at the maximal level P_N, the transmitter will shift its cone and start looking for candidate relays left and right of the

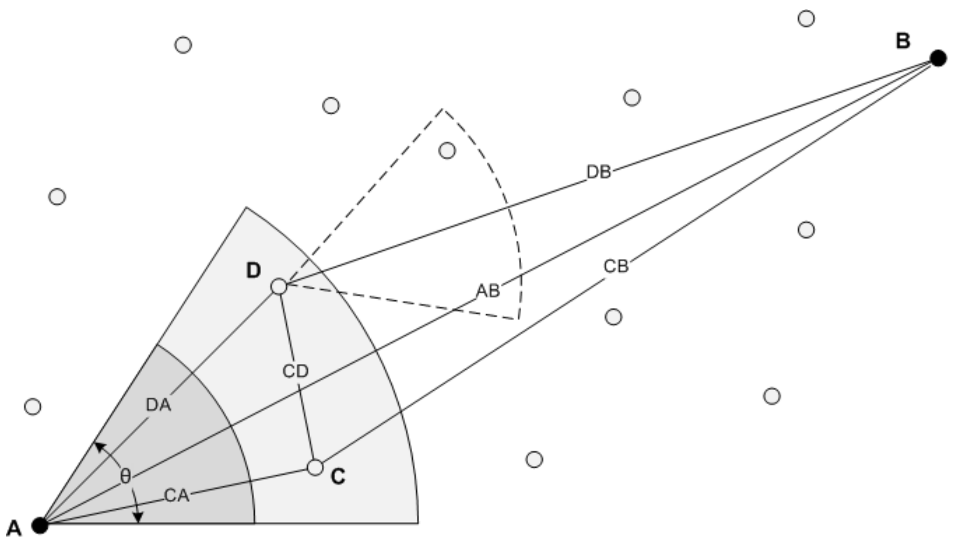

Figure 6.1: Illustration of the routing protocol: nodes within the transmitter's cone $\boldsymbol{\theta}$ are candidate relays.

main cone. This strategy favors paths with minimal amount of zigzagging, while guaranteeing that all possible paths will eventually be searched. Alternatively, a node will first search in the d_1 vicinity by shifting its cone, then decide to increase the power to advance in distance.

Other strategies are also possible; for example, the strategy used in GeRaF[15] involves defining relay zones as the intersections of concentric circles around the receiver and the coverage area centered in the transmitter. A relay is then chosen from the region that provides the largest advancement. Specifically, the ring farthest from the transmitter is queried first, then the closer rings. The objective in doing so is not to conserve the power (transmission power is set to reach the farthest circle), but to find the shortest path (given the finite transmission power, equal for all nodes).

If the transmitter, after increasing the power to some level, reaches a single neighbor, it passes the data packet on to that neighbor who becomes a relay. A positive acknowledgement at each hop is expected. The relay now initiates an identical procedure, looking for candidate nodes within *its* cone. It has become an effective transmitter, searching for the next relay towards the final destination. If there is more than one candidate relay, the current sender will have to decide which one will become the next relay. In our example, A reaches two candidates, C and D, at power P_2. (The protocol does not change if there are more than two candidates.) When they receive the RTS from A, each one knows that it can help in relaying, and each replies to A's request using a very short control packet, akin to the clear to send (CTS) signal. Note that there is a subtle difference between the traditional CTS, issued by the destination node, and this one, which is issued by a candidate relay. A candidate's CTS contains the address of the node issuing it (C or D) as well the addresses of the source and destination (A and B). The two candidate relays are not (yet) aware of each other's existence, so it is possible that their replies will collide. However, because the CTS is very short, and the distances CA and DA are unlikely to be exactly the same, the chances of the two CTSs colliding at A are minimal. For example, with 500 bits in a CTS packet, and a bit rate of 5 kbps, there will be no collision if the distances CA and DA differ by more than 300 m.

Transmission times may also be randomized in order to avoid node synchronization effects.

If there is no collision, A receives both replies. A reply includes the sender's address, and, hence, A knows which candidate is closer to the final destination -node D in this case. It may then choose D as the relay, and pass the data packet on to it. Node B will overhear the data packet transaction and deduce from its header that it has not been chosen as relay. Alternatively, more intelligence can be incorporated into making this decision. For example, A could know from overhearing previous transactions that D is already engaged elsewhere and is thus becoming a bottleneck; it could therefore choose B as its relay. Alternatively, the CTS packet can include information about the network activity that each one of the candidates is measuring. In that case, routing is performed by exploiting first and second order neighborhood information for more efficient, integrated MAC/routing schemes [20]. As the authors show in [20], this information can be used as part of the relays' decision of whether and when to respond a multicast RTS. However, such details are of no concern for the basic routing principle. An important observation to be made is that the (long) data packet is transmitted only after the relay has been chosen, i.e., the link is secured and there are no risks of *data* packet collisions. In other words, the only packets that *can* collide are the (short) control packets.

Although the chances of collision are small, it can still happen. If A detects a collision, it will send the RTS again, using the same power level. In this round, however, C and D may know of each other's existence. This can only be guaranteed if they are inside a cone with an aperture smaller or equal to 60°. In this case, they have also learned each other's location, and only that node that knows to be closest to the final destination will reply. Hence, the next CTS collision will be avoided. In a more general case, C and D may not be aware of each other either because of the half-duplex operation of acoustic modems, or because the distance CD is greater than the transmission range associated with the power level in use. In this situation, they are able to detect that the previous query has not been completed successfully because they will have received exactly the same request as before. Then, they may delay their CTS retransmissions by $T_{wait} = N_{rtx} \cdot T_{CTS} \cdot x$ seconds, where N_{rtx} is the number of retransmissions, x is a uniformly distributed random variable, and T_{CTS} stands for the duration of the reply packet.

When the next relay has been chosen, the procedure continues. The cone emanating from node D is illustrated by dashed lines in Fig. 6.1. As the algorithm progresses, and a cone is formed at each relay, the route will zoom in on the final destination so long as there are candidate relays within reach of one another. Fig. 6.2 illustrates the region of candidate relay locations for the case when a relay can be found in each hop within a single cone, i.e., no node needs to shift its cone and look outside of the angle θ. Note that this region is bounded, as dictated by the definition of *transmitter's* cone.

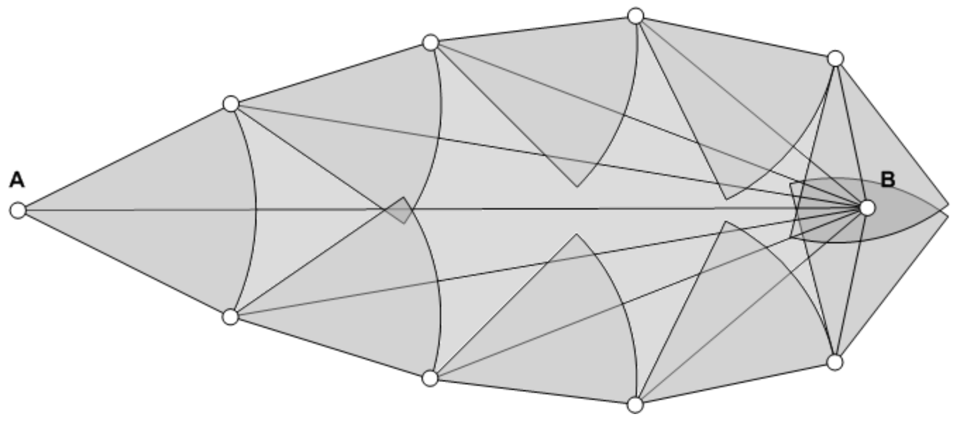

Figure 6.2: The region of candidate relay locations is contained in a cone emanating from each relay. The region of all reachable relays at the lowest power level is the shaded beam-like area.

Chapter 7

Medium Access Control

The algorithm we propose can be coupled with any MAC protocol. Since the exchange of short control packets is an inherent part of the proposed routing protocol, DACAP, a collision avoidance protocol based on virtual carrier sensing [9], seems a suitable choice. Below, we summarize the more important aspects of coupling the MAC and the routing layers.

7.1 Multicast Requests

When requesting a route, the transmitter sends a multicast RTS. Each control packet contains three $\{ID, Position\}$ pairs: one for the current transmitter, one for the final destination and one for the next intermediate node, i.e. the relay. In a multicast RTS, this field is left empty. A node proposing itself as a relay overwrites it with its own ID and position. After sending a multicast RTS, the transmitter will wait twice the maximum propagation delay corresponding to the current transmission power level even if it has already received one or more CTSs (plus the corresponding additional delay if it is a retransmitted packet).

7.2 Silence Packets

After a multicast RTS, the requesting node may receive no answers. This will occur if there are no neighbors, or there are, but they are already engaged in another communication. In the latter case, if the transmitter is not aware of the situation, it will decide to increase the transmission power, increasing the chances of disturbing ongoing transmissions. To prevent this situation, a node aware of a concurrent communication that overhears a multicast RTS will send a *very short silence packet* to the requesting node. A node receiving a *silence* packet will defer its transmission. The length of this kind of packet minimizes the chances of interfering with the ongoing communication.

	back-off	timeout
RTS	$2\left(2\frac{d_i}{c}-\frac{d_s}{c}\right)+T_{control}+T_{data}+T_{ack}$	-
CTS	$2\frac{d_i}{c}+T_{data}+T_{ack}$	$2\frac{d_i}{c}+2T_{control}$
DATA	$2\left(\frac{d_i}{c}-\frac{d_s}{c}\right)+T_{ack}$	$2\frac{d_i}{c}+T_{control}+T_{data}$
ACK	-	$2\frac{d_i}{c}+T_{data}+T_{ack}$

Table 7.1: Back-off: time that a node must stay in the back-off state after having overheard each packet. Timeout: time to wait for each packet before assuming that it was lost. T_i stands for the maximum propagation delay corresponding to the power level being used. DACAP specific parameters: $2T_{min} = 2T_i$,$Tw_{min} = 0$,$\Delta d = d_i$.

7.3 Dynamic Backoff and Waiting Times

Some valuable information can be obtained from the knowledge of the power level that it is being used. This is why it is specified in each control packet. By doing this, any node that overhears an ongoing communication can dynamically adjust its backoff and waiting times. An example is illustrated in Fig.7.1. The source node S is transmitting to the destination node D using a power level covering a distance $d1$, and node O is able to overhear the communication. In this case, it may overhear the initial RTS and go into the backoff state for the time specified in Table.7.1. $T_{control}$ and T_{data} stand for the duration of the control and data packets for a given bit-rate and T_{ack} is equal to $T_{control}$ when an explicit acknowledgement is sent, or to T_{data} if the packet is implicitly acknowledged. c is the underwater speed of sound.

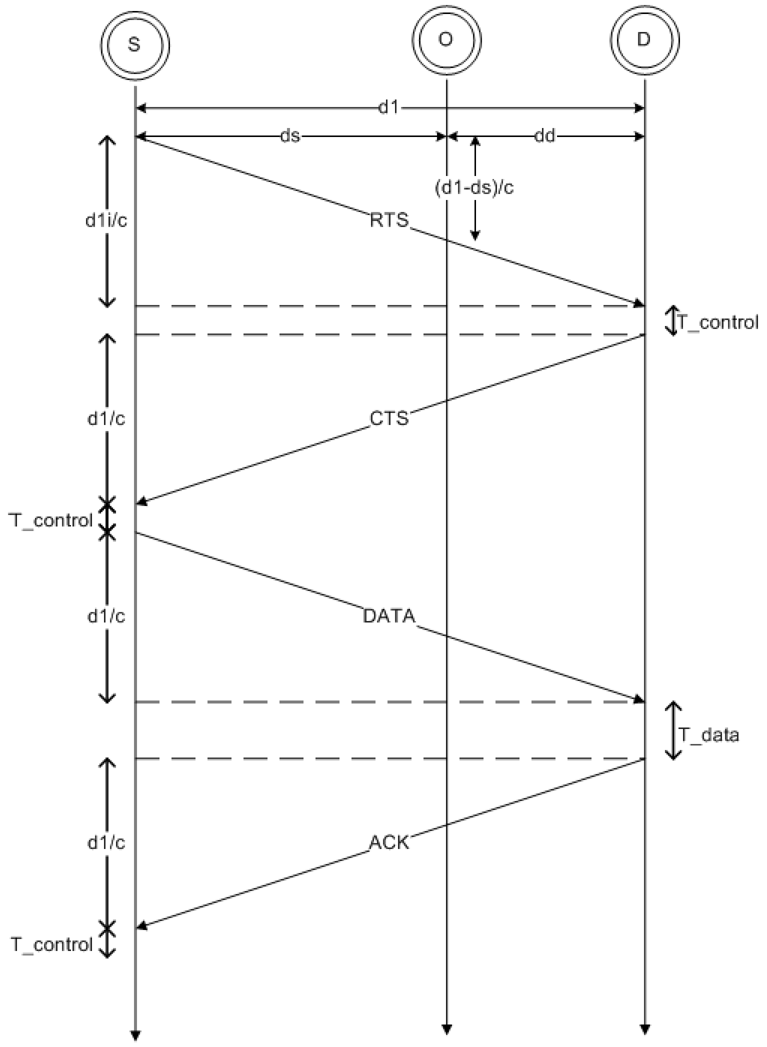

Figure 7.1: Dynamic backoff example, in which a passive node O can overhear an ongoing communication between S and D and goes into backoff state for the corresponding time specified in Table.7.1.

Chapter 8

Performance Analysis

Now the scenario for simulation is the following. The network is composed of a varying number of active nodes, randomly located over a square area of 100 km^2. There are 4 sinks, located at the edges. A Poisson distribution with an average packet generation rate λ in packets/second for each transmitter is assumed. An active node chooses the closest sink.

Each node makes use of discrete power control with four uniformly separated levels. The system frequency allocation (center frequency f_c and bandwidth B) is made so as to optimize the system performance in terms of energy per bit consumption, end-to-end delay and number of collisions as explained in Sec.1.2. The average energy per bit consumption takes into account the energy invested in transmission, listening and active reception of control and data packets, as well as their possible retransmissions.

8.1 Increasing Network Density

We first investigate the impact of node density on the protocol performance. A higher node density implies a shorter internode distance. We compute the optimal frequency allocation scheme (f_c and B) for every node density as explained in 1.2.

Fig.8.1 shows the energy per bit consumption, the average packet end-to-end delay and the total number of collisions in the network, as functions of the network node density. The performance is evaluated for two choices of the cone aperture, θ=60° and θ=120°, and it is compared to the case in which routes are established using Dijkstra's algorithm, with the cost between two nodes defined as the minimal power required to guarantee connectivity. Two nodes are connected if they can reach each other with a reference SNR$_0$, 20dB in our simulations.

In [5], it was shown that, specially in very dense networks, paths following minimum power routes (maximum number of hops) are not the optimum in terms of energy savings, but there is a minimum distance that should be advanced in each hop. However, the authors also show that for the node densities that we are considering both options are the same. This is why we use this minimum power routes as the gold standard. Alternatively, in very dense networks, this problem could be managed by instead of starting sweeping the power levels from the lowest one, doing it from a higher level.

First of all, we note from Fig.8.1 that both the energy per bit and packet end-to-end delay are very close to those of the network with minimum-power pre-established routes. The method we are proposing is able to dynamically discover minimum energy routes with minimal network knowledge. In addition, by combining the channel reservation process with the route discovery phase, the extra delay introduced by routing is small. As described in Sec.6, after a multicast RTS, the current transmitter should wait twice the maximum propagation delay corresponding to its current transmission power. For a given transmission range, the node offering the maximum advance towards the destination is the best relay. Therefore, even when packets are already directed, the waiting time necessary to reserve the channel tends to this value. At the same time, the number of collisions can be less than for pre-established routes because bottleneck situations are dynamically resolved (this is not the case of Dijkstra's algorithm in the way it is defined).

8.2 Optimal Cone Aperture

The cone aperture θ plays an important role in the system performance. In sparse networks, limiting the area with potential relays to a cone with an aperture of 60° turns to be less energy efficient than opening the cone. Indeed, in low density networks, rather than reducing the amount of zigzagging, too small θs can force the protocol to switch to higher power levels than necessary. This reduces the end-to-end delay but increases the energy consumption. On the contrary, in dense networks, forwarding a packet over too many relays (large θ) can overload the network, thus reducing the system performance, as illustrated in Fig.8.1c. As we are using DACAP with power control, network congestion translates into higher delays but not into a noticeable energy consumption increase.

For each node density, an optimal cone aperture can be determined for a given scenario. In Fig.8.2, the cone aperture that minimizes the average energy per bit consumption for each node density is shown for our example network. The energy per bit consumption, the average packet end-to-end delay and the total number of collisions when using this optimal cone aperture are shown as functions of the network node density in Fig.8.3. The results validate the issue pointed out before. When the network is composed of a few nodes, opening the cone can reduce the energy per bit consumption on average (in this case, zigzagging is not detrimental). When moving to higher network densities by *focusing* on the receiver closing the cone avoids making more hops than necessary by preventing zigzagging.

8.3 Increasing User Packet Generation Rate

For a specific network density, the protocol performance as a function of the packet generation rate is measured and compared to the case in which pre-established routes are followed. By doing this, the actual load in the network due to the route discovery mechanism is obtained.

Fig.8.4 shows the energy per bit consumption, the average packet end-to-end delay and the total number of collisions in the network with 64 active nodes and an increasing packet generation rate. The performance is evaluated for two choices of the cone aperture θ=60° and θ=150°, which corresponds to the optimal. Similarly to the previous case, the protocol performance is close to that of pre-established routes. Only at high packet generation rates, can the lack of route information can increase the latency, but in practice the routes will not change that fast. Therefore, instead of having to discover them, nodes can just follow the last valid route and start the multicast query only if necessary.

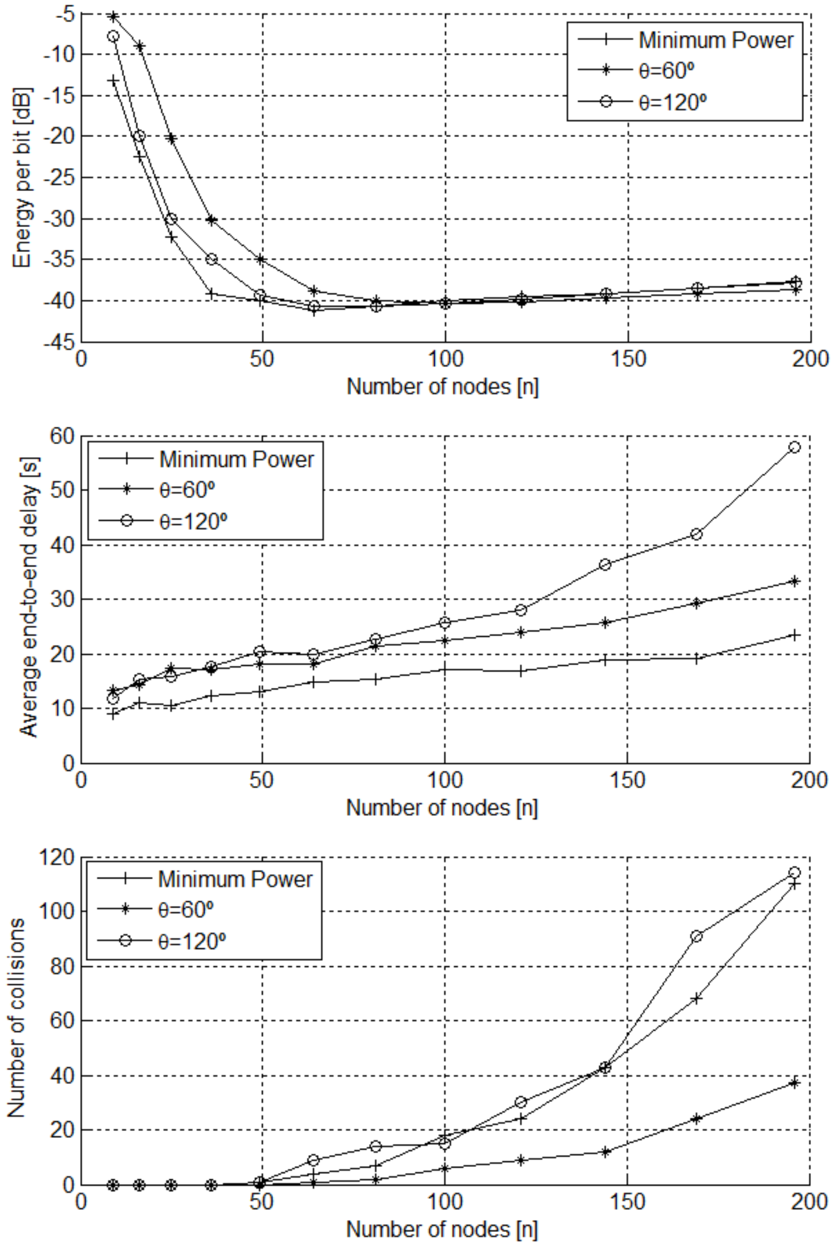

Figure 8.1: Energy per bit consumption, average packet end-to-end delay and number of collisions when using a transmission cone with $\boldsymbol{\theta}$=60° and $\boldsymbol{\theta}$=120° as functions of the number of active nodes over a fixed area of 100 km$^{\mathbf{2}}$ ($\boldsymbol{\lambda}$=0.2 packets/minute, packet length=9600 bits).

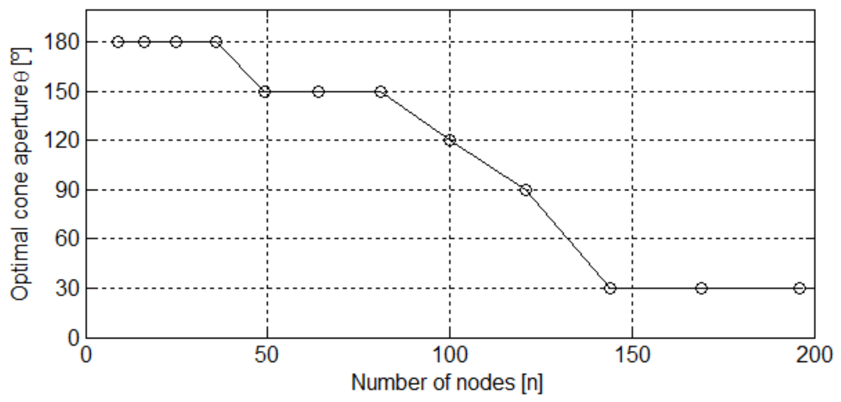

Figure 8.2: Optimal cone aperture $\boldsymbol{\theta}$ as a function of the number of active nodes over a fixed area of 100 km$^{\mathbf{2}}$ ($\boldsymbol{\lambda}$=0.2 packets/minute, packet length=9600 bits).

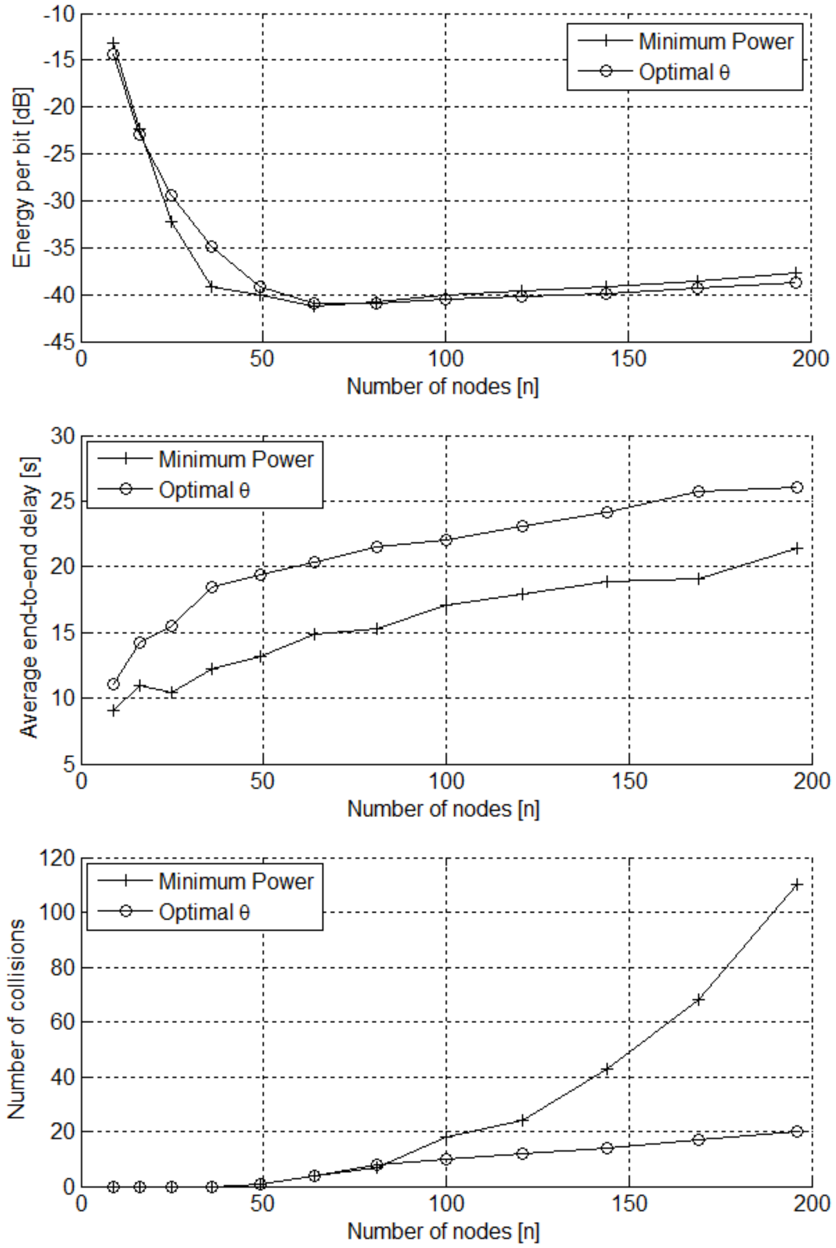

Figure 8.3: Energy per bit consumption, average packet end-to-end delay and number of collisions when using a transmission cone with the optimal aperture of Fig.8.2, as functions of the number of active nodes over a fixed area of 100 km$^{\mathbf{2}}$ ($\boldsymbol{\lambda}$=0.2 packets/minute, packet length=9600 bits).

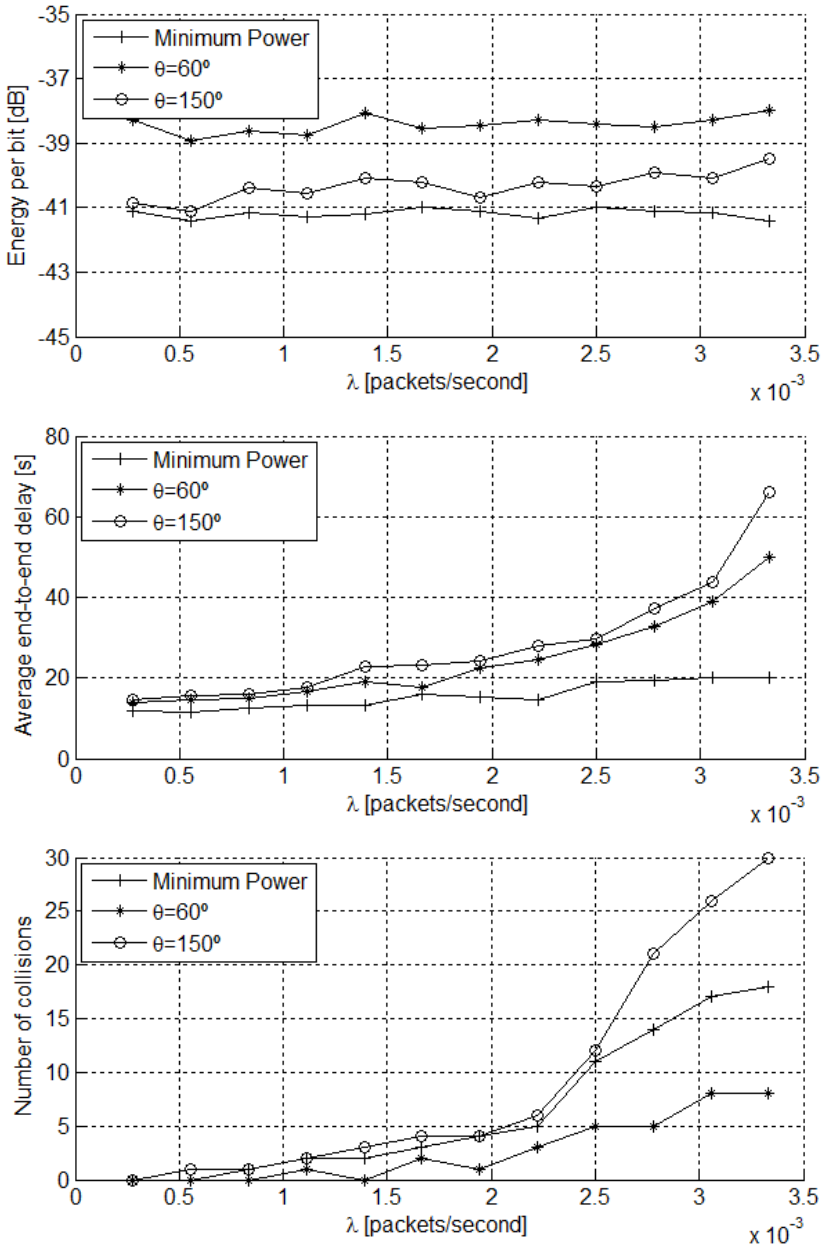

Figure 8.4: Energy per bit consumption, average packet end-to-end delay and number of collisions when using a transmission cone with $\boldsymbol{\theta}$=60° and $\boldsymbol{\theta}$=150° as functions of the packet generation rate of 64 active nodes, over a fixed area of 100 km$^{\mathbf{2}}$.

Chapter 9

Conclusions and Future Work

A routing technique, based on location information, was proposed for energy-efficient multi-hop communications in underwater acoustic networks. The system performance was evaluated for different node densities and network loads. It was shown that, by properly coupling routing and MAC functionalities with power control, routes can be established on demand with a minimum impact on the network performance. Energy per bit consumption and average packet end-to-end delay were compared to those of a network in which packets follow minimum-power pre-established routes.

9.1 Improving the routing protocol

Future work related to the protocol involves several refinements and extensions, as well as validation of the system performance in different scenarios. In particular, issues which should be considered are additional cost metrics in the candidate selection process (now only the location information is considered), alternative MAC protocols, and the effect of different power allocations strategies (e.g., defining the power levels according to average number of neighbors that can be reached).

9.2 New Applications

The routing methodology that we have introduced in this second part of the thesis is well suited for networks in which the position of the destination (one or more common sinks) is known. If the receiver is an AUV, this is not that easy. One way to solve this could be letting the AUV inform the nodes around him when he is close enough to receive a transmission (see Fig.9.1). We can also think of an AUV as a mobile relay node, connecting disconnected networks, as illustrated in Fig.9.2. These are just some examples of situations in which efficient MAC and Routing protocols are necessary.

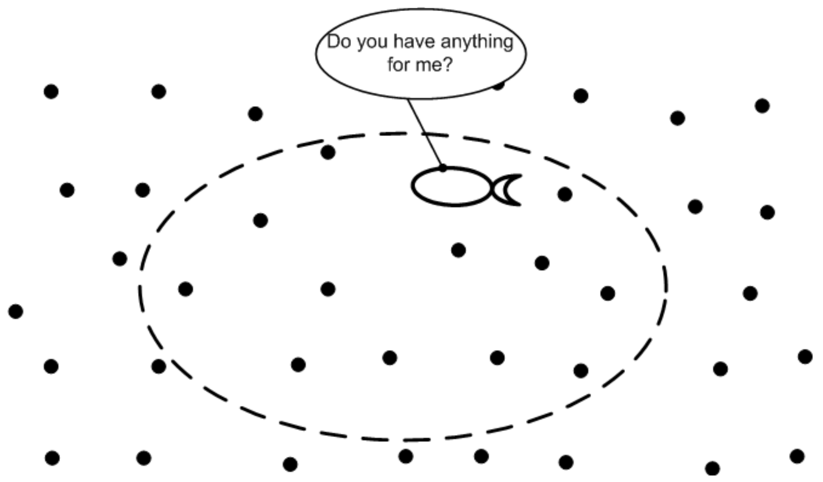

Figure 9.1: An AUV announcing its position to the static nodes in a node field.

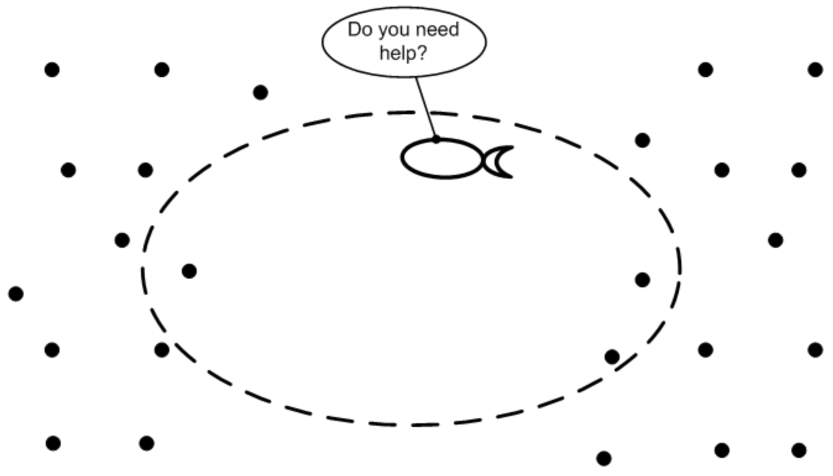

Figure 9.2: An AUV offering its services as ”mobile relay”.

9.3 Reducing the energy consumed in listening to the channel

When transmission power control is used, the total energy consumed in reception or in just listening to the channel can be even higher than the one invested in transmitting data (see Fig.9.3). This problem has been assessed in the literature and the solution that different authors propose is to make the nodes periodically go into sleeping mode. That is to say, a node will periodically disconnect the power-hungry communication part of itself in order to reduce the total energy consumption. This can be easily done, what is difficult is to do it without drastically damaging the system performance. It does not make sense to disconnect the radio part of an AUV, as it is only a few hours *on duty*. Static nodes can be programmed to periodically enter into sleeping mode, following some kind of pattern that maximizes the coverage of those nodes that are awake.

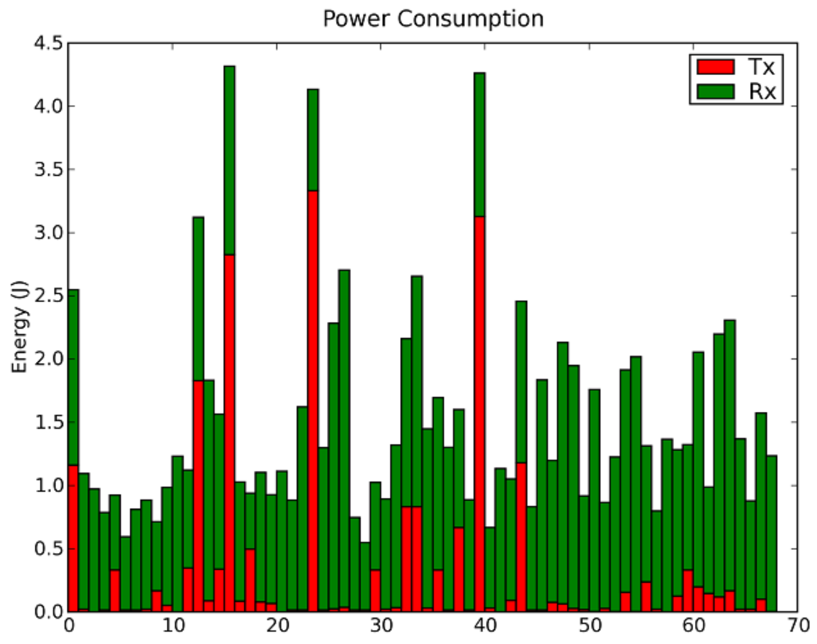

Figure 9.3: Energy consumption for the different nodes in a network. Energies consumed in transmitting data and listening to the channel are both plotted (J).

9.4 Improving the simulator

The current channel model is very simple: just the propagation delay and the acoustic path-loss are taken into account. One thing that can be easily included is a certain bit error probability which at the end is the one that summarizes the effect of multi-path, doppler effect, modulation, etc. However, a more sophisticated channel model could be a great asset for the simulator.

Appendix A

AUVNetSim: A Simulator for Underwater Networks

All the results included in this document have been obtained using AUVNetSim [21], a simulation library for testing acoustic networking algorithms. It is written in Python [22] and it makes extensive use of the SimPy discrete event simulation package [23]. AUVNetSim is redistributed under the terms of the GNU General Public License.

AUVNetSim is interesting for both *end users* and *developers*. A user willing to run several simulations using the resources that are already available, can easily modify several system parameters without having to explicitly deal with python code. A developer, who for example, wants to include a new MAC protocol, can simply do so by taking the advantage of the existing structure.

A.1 Prerequisites

To run this software, the following packages are necessary:

- Python Environment: the python core software.
- SimPy Package: a discrete-event simulation system.
- MatplotLib: a python plotting library.
- Numpy: a package that provides scientific computing functionalities.

All of these packages are freely available under the GNU license.

A.2 AUVNetSim for Users

The simulator already contains a great variety of parameters and protocols that can be selected. Rather than having to compile the code each time a new simulation is required, a user just needs to set up the simulation file (*.py) and the configuration file (*.conf).

A.2.1 Simulation File

This file contains the *main* function that will be invoked when the simulator is launched. The following python code is an example:

```
import Simulation as AUVSim     # Inclusion of the resources
import pylab                    # Inclusion of the visualization class

def aSimulation():              # Main Function

    if(len(sys.argv) < 2):
        print "usage: ", sys.argv[0], "ConfigFile" # A configuration file is expected
    exit(1)

    config = AUVSim.ReadConfigFromFile(sys.argv[1])

    print "Running simulation"
    nodes = AUVSim.RunSimulation(config)    # The simulation is launched
    print "Done"

    PlotScenario(nodes)         # Visualization of the scenario for simulation
    PlotConsumption(nodes)      # Visualization of the consumption per node
    PlotDelay(nodes)            # Visualization of the delay per node
    pylab.show()

if __name__ == "__main__":
    aSimulation()
```

After the inclusion of the AUVNetSim library, the simulation is launched. After that, some of the results or statistics that are monitored throughout the simulation are displayed. The user can either use the already defined visualization functions, create new ones or save the required information in plain text files which later could be read using, for example, Matlab. Several examples are included with the downloadable package.

A.2.2 Configuration File

Several parameters should be specified in the configuration file before each simulation. In the following lines, there is an example of the content of this type of files.

```
# Simulation Duration (seconds)
SimulationDuration = 1800.00

# Available Bandwidth (kHz)
BandWidth = 48.00

# Bandwidth efficiency (bps/Hz)
BandwidthBitrateRelation = 1.00
```

```
# Frequency (kHz)
Frequency = 44.00

# Maximum Transmit Power -> Acoustic Intensity (dB re uPa)
TransmitPower = 500.00

# Receive Power (dB) -> Battery Consumption (dB)
ReceivePower = -10.00

# Listen Power (dB) -> Battery Consumption (dB)
ListenPowerW = -10.00

# DataPacketLength (bits)
DataPacketLength = 9600.00 #bits

# PHY: set parameters for the physical layer
PHY = {"SIRThreshold": 15.00, "SNRThreshold": 20.00,
"LISThreshold":  3.00, "variablePower":True,
"multicast2Distance":{0:1600.00,1:2300.00,2:2800.00,3:3200.00,5:6000.0}}

# MAC: define which protocol we are using & set parameteres
MAC = {"protocol":"ALOHA", "max2resend":10.0, "attempts":4,
"ACK_packet_length":24, "RTS_packet_length":48, "CTS_packet_length":48,
"WAR_packet_length":24, "SIL_packet_length":24, "tmin/T":2.0,
"twmin/T":0.0, "deltatdata":0.0, "deltad/T":0.0, }

# Routing: set parameters for the routing layer
Routing = {"Algorithm": "FBR", "variation":0, "coneAngle":60.0}

# Nodes: here is where we define individual nodes
# format: AcousticNode(Address, position[, period, destination])
Nodes = [["A", (0,9000,1000), 4*60, "Sink"],["D",(9000,0,1000), 4*60, "Sink"],
["B", (9000,9000,1000), 4*60, "Sink"], ["C", (0,0,1000), 4*60, "Sink"],
["Sink",(4500,4500,1000), None, "A"]]
```

All the possible parameters that can be currently specified are summarized in Table A.1. The simulation can be started by just typing from the OS command line:

```
!> python simulation_file.py configuration_file.conf
```

A.3 AUVNetSim for Developers

Before reading this section, we encourage the user to familiarize with the Python programming language and the SimPy library [22, 23].

Physical Layer		
Center Frequency	[kHz]	
Bandwidth	[kHz]	
Bandwidth Efficiency	[bit/Hz]	
Transmitting mode max power	[dB]	
Receiving power consumption	[dB]	
Listening power consumption	[dB]	
Listening threshold	[dB]	
Receiving threshold	[dB]	
Power Control	True/False	
Power Levels	name:value[km]	Only if P.Control is used
Medium Access Control		
Protocol	CS-ALOHA, DACAP	
RTS length	[bit]	Only if DACAP is used
CTS length	[bit]	Only if DACAP is used
ACK length	[bit]	
WAR length	[bit]	Only if DACAP is used
SIL length	[bit]	Only if DACAP is used
DATA length	[bit]	
Retransmission attemps		
Maximum waiting time	[s]	Only if ALOHA is used
Tmin		Only if DACAP is used
Twmin		Only if DACAP is used
Interference region		Only if DACAP is used
Routing Layer		
Protocol	No routes, Static Routes, FBR	
Variation	0,1,2	Only if FBR is used
Cone aperture	[degree]	Only if FBR is used
Retransmission attemps		Only if FBR is used
Nodes		
Name		
Period	[s]	Can be None
Destination	[node]	Can be None
Position or Path	List of points	
Simulation Duration	[s]	

Table A.1: AUVNetSim parameters that should be specified in the configuration file

A.3.1 Simulator Structure

The way in which AUVNetSim is programmed eases the task of including new features. Like many other wireless network simulators, the description of the different layers functionalities is specified in different classes or files. A programmer willing to introduce, for example, a new routing technique does not need deal with the MAC or the physical layer.

The communication between layers is performed by the exchange of short messages. For example, a packet coming from the application layer is sent to the routing layer, which will update the packet header and, on its turn, will send it to the MAC layer. Finally, the message will be transmitted to the channel through the physical layer, following the protocol policy.

In the following lines, an overview of each of the files that compose the simulator is offered.

Simulation.py

This is the main file for a project. In here, the 3D scenario for simulation is created and the simulation is conducted.

```
import SimPy.Simulation as Sim          # Inclusion of the discrete-event mechanism
from AcousticNode import AcousticNode   # Contains the definition of a node

def RunSimulation(config_dict):
    Sim.initialize()

    # Signal all nodes that a message has been transmitted
    AcousticEvent = Sim.SimEvent("AcousticEvent")

    nodes = SetupNodesForSimulation(config_dict, AcousticEvent)

    Sim.simulate(until=config_dict["SimulationDuration"])

    return nodes
```

A scenario is defined according to the description in the configuration file. There are two ways of specifying the nodes that the system contains:

- Each node can be specified by its name and position (or a path if it is a mobile node). If it is an active node, the packet generation rate and the packets' destination (a new destination can be randomly chosen before each transmission) should also be included.
- A node-field can be defined by the 3D region that it is covering and the number of nodes that are positioned in it. Nodes can be completely randomly positioned or randomly positioned within a grid. The nodes in a node-field are usually just relays, but it is easy to make them generate information too.

```
def SetupNodesForSimulation(config_dict, acoustic_event):
    nodes = []

    # Single nodes
    if "Nodes" in config_dict.keys():
        for n in config_dict["Nodes"]:
            cn = [config_dict,] + n
            nodes.append(AcousticNode(acoustic_event, *cn))

    # Node field
    if "NodeField" in config_dict.keys():
        nodes += CreateRandomNodeField(acoustic_event, config_dict,
                 *config_dict["NodeField"])

    return nodes
```

AcousticNode.py

This file contains the description of an acoustic node. Within this class, the different functionalities of a single node are initialized according to the configuration file. A node is determined by:

- Name and position.
- Characteristics of its physical layer.
- Medium Access Control protocol in use.
- Routing technique.
- Packet Generation Rate and packets' destination.

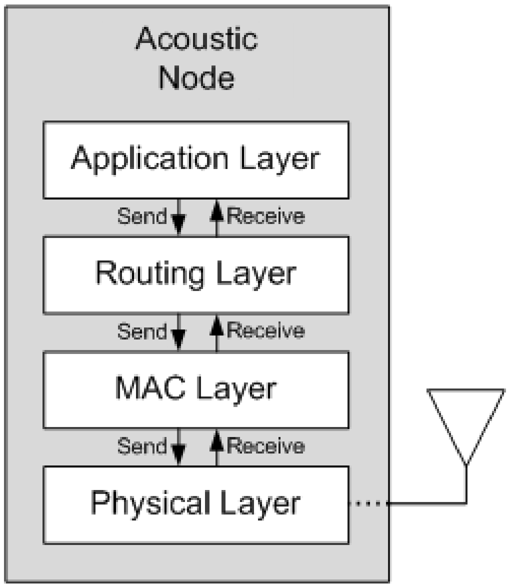

Figure A.1: AUVNetSim: node programming structure.

```
class AcousticNode():

    def __init__(self, event, config, label, position, period=None, destination=None):

        self.config = config
        self.name = label        # Node name
        self.random_dest = False # Indicates if the destination is fixed
                                 # or randomly selected

        self.SetupPath(position_or_path) # Position or path of the node
        self.total = nHigh*nWide

        # Physical layer
        self.physical_layer = PhysicalLayer(self, config["PHY"], event)

        # MAC Layer
        self.MACProtocol = SetupMAC(self, config["MAC"])

        # Routing Layer
        self.routing_layer = SetupRouting(self, config["Routing"])

        # Application Layer
        self.app_layer = ApplicationLayer(self)
        if period is not None:
            # Schedules the first transmission
            self.SetUpPeriodicTransmission(config["Period"], destination)
```

PhysicalLayer.py

The physical layer of an acoustic node is modeled by a modem and a transducer. The modem operates in half-duplex mode (it can only receive or transmit at a time). When a packet is received from the MAC layer, the modem will automatically change to transmission mode, even if there were packets being received. It is the MAC protocol's duty to check if the channel is idle before transmitting. This information is obtained from the modem. When a packet is being received through the acoustic transducer, the modem is in reception mode. At the end of the reception, a packet can be either properly received and passed on the MAC layer; just overheard (received but without a pre-specified SNR); or discarded because of interference. When a modem detects a collision, it is possible to inform the MAC layer.

Several system performance parameters are measured at this level, including the energy consumed in transmissions, the energy consumed in listening to the channel, and the number of collisions detected.

The channel model is also contained in this file. A packet being transmitted will be delayed according to acoustic propagation and its power will be attenuated according to the path-loss model defined in the same file and introduced in Chap.1.

A developer can easily introduce new channel models, variables to monitor, modem functionalities, but there are two functions that should be always preserved. These are the ones that are used to communicate from and to the layer immediately above, in this case, the

MAC layer.

```
def TransmitPacket(self, packet):
    ''' Function called from the upper layers to transmit a packet.
    '''
    # It is MAC protocol duty to check before transmitting if the channel is idle
    # using the IsIdle() function.
    if self.IsIdle()==False:
        self.PrintMessage("I should not do this ... the channel was not idle!")

    self.collision = False # Initializing the flag
    if self.variable_power:
        distance = self.multicast2distance[packet["level"]]
        power = distance2Intensity(self.bandwidth, self.freq,
                                   distance, self.SNR_threshold)
    else:
        power = self.transmit_power # Default maximum power

    new_transmission = OutgoingPacket(self)
    Sim.activate(new_transmission, new_transmission.transmit(packet, power))

def OnSuccessfulReceipt(self, packet):
    ''' Function called from the lower layers when a packet is received.
    '''
    self.node.MACProtocol.OnNewPacket(packet)
```

MAC.py

Different MAC protocols are defined in this file. CS-ALOHA, DACAP and DACAP for FBR are already included in the library. It is not the aim of this document to explain the way in which these are implemented. As in the previous case, there are some functionalities that should be always preserved:

```
def InitiateTransmission(self, OutgoingPacket):
    ''' Function called from the upper layers to transmit a packet.
    '''

    self.outgoing_packet_queue.append(OutgoingPacket)
    self.fsm.process("send_data")


def OnNewPacket(self, IncomingPacket):
    ''' Function called from the lower layers when a packet is received.
    '''
    self.incoming_packet = IncomingPacket
    if self.IsForMe():
        self.fsm.process(self.packet_signal[IncomingPacket["type"]])
    else:
        self.OverHearing()
```

The FSM.py class is used to implement Finite State Machines (common among MAC protocols). Any state diagram can be easily reproduced by defining the different states and all the possible transitions between them.

It is also common in MAC protocols to make use of timers to schedule waiting or back-off periods. A timer will trigger an event if it is not stopped once the time is consumed.

```
class InternalTimer(Sim.Process):
    def __init__(self, fsm):
        Sim.Process.__init__(self, name="MAC_Timer")
        random.seed()
        self.fsm = fsm

    def Lifecycle(self, Request):
        while True:
            yield Sim.waitevent, self, Request
            yield Sim.hold, self, Request.signalparam[0]
            if(self.interrupted()):
                # Just ignores the time finalization
                self.interruptReset()
            else:
                # Triggers a new transition in the state diagram
                self.fsm.process(Request.signalparam[1])
```

Routing Layer

A programmer willing to include a new routing technique should do it in this file. As in the previous layers, independently of the protocol, the functions that interact with the MAC protocol and the application layer should be preserved:

```
class SimpleRoutingTable(dict):

    def SendPacket(self, packet):
        packet["level"]=0.0
        packet["route"].append((self.node.name, self.node.GetCurrentPosition()))
        try:
            packet["through"] = self[packet["dest"]]
        except KeyError:
            packet["through"] = packet["dest"]

        self.node.MACProtocol.InitiateTransmission(packet)

    def OnPacketReception(self, packet):
        # If this is the final destination of the packet,
        # pass it to the application layer
        # otherwise, send it on...
        if packet["dest"] == self.node.name:
            self.node.app_layer.OnPacketReception(packet)
        else:
            SendPacket(packet)
```

At the same time, as the coupling between the MAC protocol and the routing technique increases, there are more functionalities that are cross-referenced between classes.

Application Layer

In the application layer, packets may be periodically generated and relayed to the lower layers. In addition, some system performance parameters are monitored such as the packet end-to-end delay or the number of hops that a packet has made before reaching its final destination.

```
def PeriodicTransmission(self, period, destination):
    while True:
        self.packets_sent+=1
        packet_ID =  self.node.name+str(self.packets_sent)

        if self.random_dest and destination==None:
            num = randint(0, self.node.total)
            destination = self.node.prefix+'%03d'%(num,)

            if destination == self.node.name or destination == "S000":
                destination = "Sink"

        packet = {"ID": packet_ID, "dest": destination, "source": self.node.name,
                  "route": [], "type": "DATA", "initial_time": Sim.now(),
                  "length": self.node.config["DataPacketLength"]}

        self.node.routing_layer.SendPacket(packet)
        next = poisson(period)
        yield Sim.hold, self, next

def OnPacketReception(self, packet):
    self.log.append(packet)
    origin = packet["route"][0][0]
    if origin in self.packets_received.keys():
        self.packets_received[origin]+=1
    else:
        self.packets_received[origin]=1

    delay = Sim.now()-packet["initial_time"]
    hops = len(packet["route"])

    self.PrintMessage("Packet "+packet["ID"]+" received over "+str(hops)+
                      " hops with a delay of "+str(delay)+
                      "s (delay/hop="+str(delay/hops)+").")

    self.packets_time.append(delay)
    self.packets_hops.append(hops)
    self.packets_dhops.append(delay/hops)
```

Visualization Functionalities

Last but not least, there are several functions that are included in the downloadable package that can be used to illustrate the results and check at a glance the system overall performance. All them make an extensive use of the MatPlotLib/Pylab package and can be found in the Sim.py file. The main idea is to process the different variables that are monitored during the simulation, from the structure containing all the nodes.

In Fig.A.2, the scenario for a simulation containing four active nodes (A, B, C, D), transmitting to a common sink, and 64 relays is shown. Within this graph, we are able to show plenty of information:

1. Only the nodes that participated in the packets transmissions are shown by name.
2. The color of a node is related to the energy that it has consumed by just listening to the channel. That is to say, a node surrounded by active nodes will have a red-like color.
3. The size of a node is proportional to the energy that it has consumed transmitting packets in the network. A bigger node has taken part in more packet exchanges than a smaller node.
4. Routes can then visually be identified.

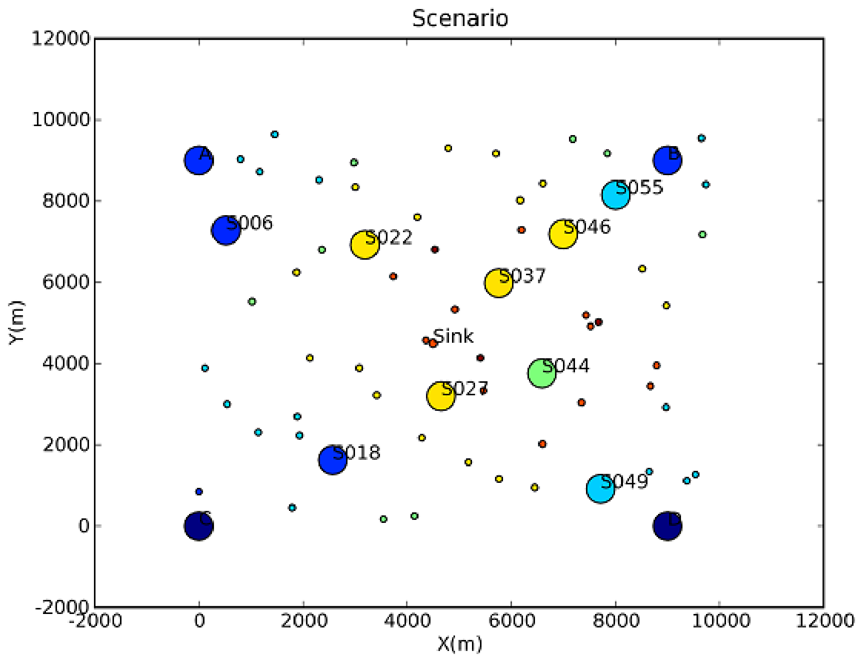

Figure A.2: AUVNetSim: scenario for simulation and final node status.

Alternatively, the energy consumed in both transmitting and receiving packets can be plotted in a bar diagram, such as the one shown in Fig.A.3. Fig.A.4 contains a histogram of the end-to-end delay of the packets received at the common sink. 3D graphs are also possible. For example, in Fig.A.5 a network containing an AUV and a node-field with 16 relays is shown. In the same plot, the route that each packet has followed is included.

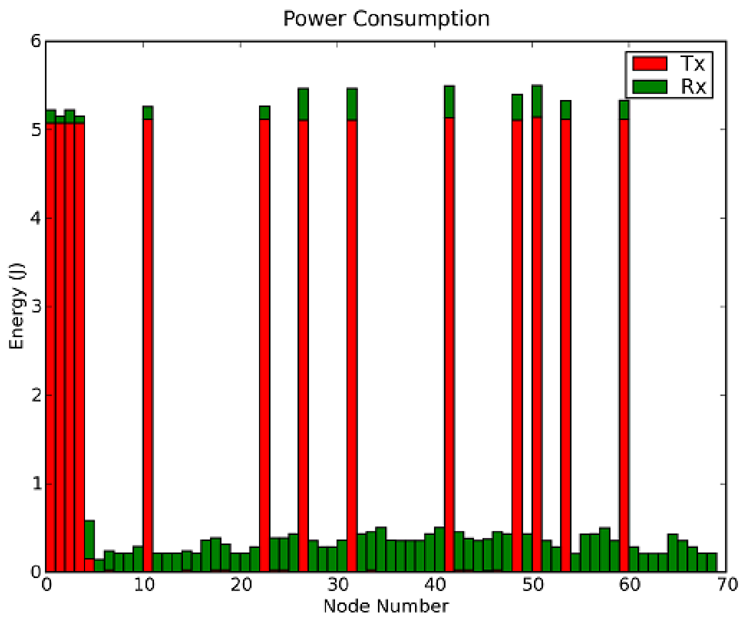

Figure A.3: AUVNetSim: energy consumption at each node of the scenario.

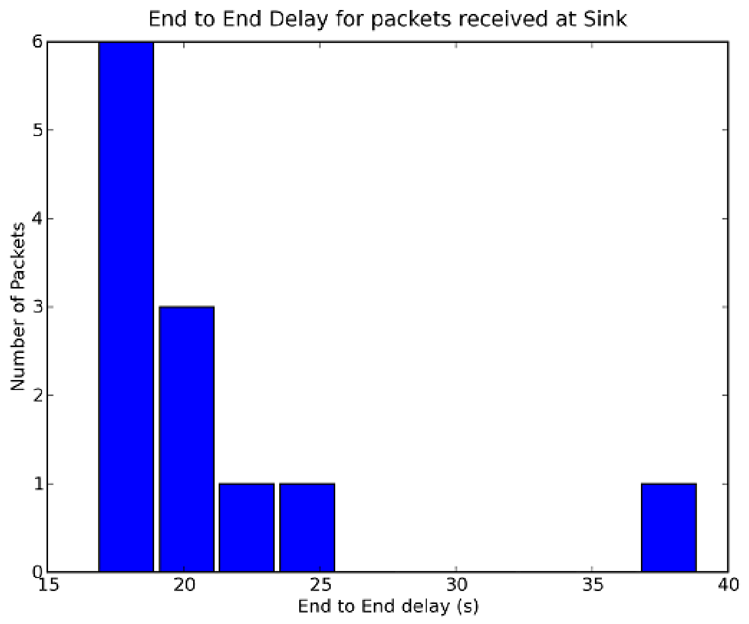

Figure A.4: AUVNetSim: histogram of the end-to-end delay of the packets received at the common sink.

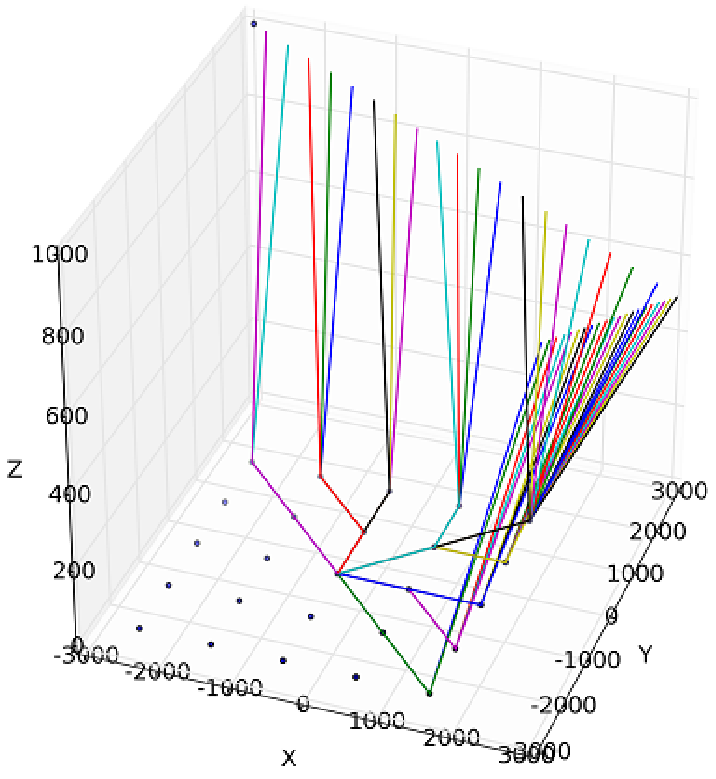

Figure A.5: AUVNetSim: 3D scenario containing an AUV and a node-field with 16 relays.

Bibliography

[1] M.Stojanovic, "On the Relationship Between Capacity and Distance in an Underwater Acoustic Channel," First ACM International Workshop on Underwater Networks (WUWNet), September 2006.

[2] M.Stojanovic, "Capacity of a Relay Acoustic Link," Oceans 2007, vol., no., pp.1-7, Sept. 29 2007-Oct. 4 2007.

[3] M.Stojanovic, "On the Design of Underwater Acoustic Cellular Systems," Oceans 2007 - Europe , vol., no., pp.1-6, 18-21 June 2007.

[4] C.R.Benson, M.J.Ryan, M.R.Frater, "On the Benefits of High Absorption in Practical Multi-hop Networks," Oceans 2007 - Europe , vol., no., pp.1-6, 18-21, June 2007.

[5] A.F.Harris, M.Zorzi, "On the Design of Energy-efficient Routing Protocols in Underwater Networks," Sensor, Mesh and Ad Hoc Communications and Networks, 2007. SECON '07. 4th Annual IEEE Communications Society Conference on , vol., no., pp.80-90, 18-21 June 2007.

[6] A.Quazi and W.Konrad, "Underwater acoustic communications," IEEE Comm. Magazine, pp. 24-29, March 1982.

[7] L.Nerkhovskikh and Y.Lysanov, "Fundamentals of Ocean Acoustics," New York: Springer, 1982.

[8] R.Coates, "Underwater Acoustic Systems," New York: Wiley, 1989.

[9] B.Peleato, M.Stojanovic, "Distance aware collision avoidance protocol for ad-hoc underwater acoustic sensor networks," Communications Letters, IEEE , vol.11, no.12, pp.1025-1027, December 2007.

[10] A.Porto, M.Stojanovic, "Optimizing the Transmission Range in an Underwater Acoustic Network," Oceans 2007, vol., no., pp.1-5, Sept. 29 2007-Oct. 4 2007.

[11] C.E.Perkins , P.Bhagwat, "Highly dynamic Destination-Sequenced Distance-Vector routing (DSDV) for mobile computers," Proceedings of the conference on Communications architectures, protocols and applications, p.234-244, August 31-September 02, 1994, London, United Kingdom.

[12] P.Jacquet, P.Muhlethaler, T.Clausen, A.Laouiti, A.Qayyum, and L.Viennot, “Optimized Link State Routing Protocol for Ad Hoc Networks,” In Proc. of IEEE INMIC, pages 62–68, Pakistan, Dec. 2001.

[13] C.Perkins, E.Belding-Royer, S.Das, “Ad hoc On-Demand Distance Vector (AODV) Routing,” RFC Editor, 2003.

[14] D.B.Johnson, D.A.Maltz, J.Broch, “DSR: the dynamic source routing protocol for multi-hop wireless ad hoc networks,” Ad hoc networking, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 2001.

[15] M.Zorzi, R.R. Rao, “Geographic random forwarding (GeRaF) for ad hoc and sensor networks: multihop performance,” Mobile Computing, IEEE Transactions on, vol.2, no.4, pp. 337-348, Oct.-Dec. 2003.

[16] M.Zorzi, R.R. Rao, “Geographic random forwarding (GeRaF) for ad hoc and sensor networks: energy and latency performance,” Mobile Computing, IEEE Transactions on, vol.2, no.4, pp. 349-365, Oct.-Dec. 2003.

[17] D.Ferrara, L.Galluccio, A.Leonardi, G.Morabito, S.Palazzo, ”MACRO: an integrated MAC/routing protocol for geographic forwarding in wireless sensor networks,” INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE , vol.3, no., pp. 1770-1781 vol. 3, 13-17 March 2005.

[18] E.A.Carlson, P.P.Beaujean, E.An, “Location-Aware Routing Protocol for Underwater Acoustic Networks,” Oceans 2006, vol., no., pp.1-6, Sept. 2006.

[19] N.Nicolaou, A.See, Peng Xie, Jun-Hong Cu, D.Maggiorini, “Improving the Robustness of Location-Based Routing for Underwater Sensor Networks,” Oceans 2007 - Europe , vol., no., pp.1-6, 18-21 June 2007.

[20] M.Rossi, M.Zorzi, “Integrated Cost-Based MAC and Routing Techniques for Hop Count Forwarding in Wireless Sensor Networks,” Mobile Computing, IEEE Transactions on , vol.6, no.4, pp.434-448, April 2007.

[21] J.M.Jornet, J.Eskesen, “AUVNetSim,” http://sourceforge.net/projects/auvnetsim/.

[22] Guido van Rossum, “Python Tutorial”, http://docs.python.org/tut/.

[23] T.Vignaux, K.Muller, “The SimPy Manual”, http://simpy.sourceforge.net/.