ProCoopa: um Protocolo Cooperativo para Redes de Sensores sem fio Aquáticas

Lucas S. Cerqueira[1], **Alex B. Vieira**[1],
Luiz F. M. Vieira[2], **Marcos A. M. Vieira**[2], **José A. M. Nacif**[3]

[1] Departamento de Ciência da Computação – Universidade Federal de Juiz de Fora
Juiz de Fora – MG – Brasil

[2]Departamento de Ciência da Computação – Universidade Federal de Minas Gerais
Belo Horizonte – MG – Brasil

[3]Instituto de Ciências Exatas e Tecnológicas – Universidade Federal de Viçosa
Florestal - MG, Brasil

lucas.saar@ice.ufjf.br, alex.borges@ufjf.edu.br,
{lfvieira,mmvieira}@dcc.ufmg.br, jnacif@ufv.br

Abstract. *Monitoring underwater environments is still a hard and costly task. In this paper, we present ProCoopa: a Cooperative Protocol for Underwater Wireless Sensor Networks. ProCoopa synchronously/asynchronously works on top of TDMA method combined with an ARQ scheme based on selective repeat technique. It uses idle sensor nodes as relay nodes, enhancing communication space diversity. Our simulations show that, when compared to a non-cooperative protocol, ProCoopa enhances overall network performance metrics. For instance, it reduces packet error rate by 28.32% and increases goodput by 16.87% while spending less than 1% more energy.*

Resumo. *O monitoramento de ambientes aquáticos ainda é uma tarefa difícil e dispendiosa. Neste artigo, apresentamos ProCoopa: um Protocolo Cooperativo para Redes de Sensores sem fio Aquáticas. O ProCoopa funciona de forma síncrona/assíncrona sobre o método TDMA combinado com um esquema ARQ baseado em* Selective Repeat. *Ele usa os nós sensores ociosos como nós retransmissores, aumentando a diversidade do espaço de comunicação. Nossas simulações mostram que, quando comparado a um protocolo não cooperativo, o ProCoopa reduz a taxa de erro de pacotes em 28,32% e aumenta o* goodput *em 16,87% enquanto gasta menos de 1% a mais de energia.*

1. Introdução

Redes de Sensores Aquáticas (RSAs) é uma importante área de pesquisa com crescente interesse da comunidade acadêmica e da indústria. Oceanos, rios e lagos [Vieira et al. 2018] são fundamentais para a vida em nosso planeta e uma série de segmentos da sociedade exige monitoramento subaquático. Por exemplo, a indústria petrolífera usa RSAs para monitorar e explorar recursos naturais, como petróleo e gás. Os governos monitoram os oceanos para alertar a população sobre desastres naturais como tsunamis.

Apesar de sua importância, o monitoramento de ambientes subaquáticos ainda é uma tarefa difícil e dispendiosa. As condições do meio aquático impõem grandes dificuldades à comunicação. Abaixo da superfície da água, as ondas eletromagnéticas e ópticas

sofrem alta atenuação, sendo absorvidas em alguns metros [Vieira et al. 2010]. Mesmo a comunicação acústica é desafiadora e apresenta três grandes problemas: (i) a largura de banda limitada e dependente da distância, (ii) o desvanecimento multi-caminho variado pelo tempo, e (iii) a baixa velocidade do som na água quando comparada à ondas de rádio. Além disso, como quase todas as redes de sensores, as RSAs apresentam severas restrições de energia.

Para melhorar a qualidade da comunicação e superar os desafios das RSAs, muitas técnicas foram estudadas desde a camada física até a camada de redes. Uma série de estudos envolve o desenvolvimento de modems acústicos e uso eficiente do canal de comunicação [Pinto et al. 2012, Demirors et al. 2014, Renner and Golkowski 2016], acesso múltiplo ao canal (detalhado na Seção 2) e roteamento de dados entre os nós sensores [Lee et al. 2010b, Vieira 2012, Coutinho et al. 2013, Coutinho et al. 2014, Coutinho et al. 2015, Basagni et al. 2015, Coutinho et al. 2016a, Coutinho et al. 2016b]. Nesse cenário, o uso de protocolos ARQ (*Automatic Repeat Request*) é obrigatório para obter uma comunicação livre de erros [Sozer et al. 2000]. Entretanto, para os protocolos baseados em ARQ como *stop-and-wait* e *selective repeat*, o longo atraso de propagação combinado com a elevada taxa de erro de bits (BER) do canal acústico aquático fazem com que seja difícil alcançar uma comunicação com alta vazão [Ghosh et al. 2013]. Além disso, os protocolos ARQ também aumentam o uso de energia e a latência de transmissão, o que não é desejável em RSAs.

Neste trabalho, apresentamos ProCoopa: um Protocolo Cooperativo para Redes de Sensores Aquáticas. O ProCoopa tira proveito da natureza de *broadcast* das transmissões sem fio, na qual cada nó da rede observa outras transmissões e pode atuar como cooperador, transmitindo simultaneamente os dados que recebe de um nó de origem para um nó destino na rede. Isso aumenta a diversidade de caminhos [Laneman et al. 2004], e, como consequência, também aumenta a chance de uma transmissão ser bem-sucedida. Além disso, por meio da cooperação, é possível alcançar a diversidade de tempo transmitindo os mesmos dados (ou sinal), que foi ouvido através de uma transmissão em *broadcast*, em um instante de tempo diferente. Nesse caso, o nó de origem pode transmitir um novo pacote de dados enquanto o cooperador tenta recuperar a transmissão de dados anteriormente perdida, aumentando a taxa de transferência do sistema.

Mais detalhadamente, o ProCoopa leva em consideração as duas subcamadas da camada de enlace de dados: o controle de acesso ao meio e o controle de enlace lógico. O ProCoopa funciona de forma síncrona/assíncrona em cima do método de acesso múltiplo por divisão de tempo (TDMA) combinado com um esquema ARQ baseado em *selective repeat*. Ele explora a ociosidade dos nós para aumentar a diversidade espacial, ou mais especificamente, a diversidade de cooperação. Diferente das abordagens existentes que também usam comunicação cooperativa [Lee et al. 2010a, Lee and Cho 2011, Azad et al. 2011, Ghosh et al. 2013, Azad et al. 2013], nosso protocolo é projetado para integrar a cooperação com um esquema ARQ em um protocolo de controle de acesso médio sem colisão.

Avaliamos o ProCoopa através de simulações no ns-3 em um cenário em que vários nós sensores subaquáticos tentam se comunicar com um nó sorvedouro. Quando comparado ao mesmo protocolo TDMA sem cooperação, os resultados mostram que o ProCoopa consegue melhorar as métricas de desempenho da rede. Por exemplo, no me-

lhor dos casos, a taxa de erro do pacote é reduzida em 28,32% e o *goodput* aumenta em 16,87% enquanto gasta menos de 1% a mais de energia.

Em suma, nossas maiores contribuições são as seguintes: (i) ProCoopa, um protocolo cooperativo para redes de sensores sem fio aquáticas, (ii) seu funcionamento que opera de forma síncrona/assíncrona sobre o método TDMA combinado com um esquema ARQ baseado em *Selective Repeat*, (iii) a avaliação sistemática do protocolo, mostrando os ganhos de desempenho quando comparado a uma arquitetura tradicional, (iv) apontamos ainda os cenários apropriados para cada versão (síncrona/assíncrona) do protocolo.

O restante deste artigo está organizado da seguinte forma: na seção 2, mostramos os trabalhos relacionados. Na seção 3, apresentamos o protocolo ProCoopa. Na seção 4, detalhamos o cenário de avaliação e também mostramos os resultados. Finalmente, na seção 5, extraímos as conclusões.

2. Trabalhos Relacionados

Trabalhos anteriores, na maioria dos casos, se concentram em mostrar que transmissões cooperativas podem ser eficientemente aplicadas a redes aquáticas. Quase todos esses trabalhos surgiram como extensões de trabalho de cooperação em redes sem-fio terrestres [Carbonelli and Mitra 2006, Vajapeyam et al. 2008, Han et al. 2008a, Han et al. 2008b, Carbonelli et al. 2009, Wang et al. 2011]. Por exemplo, Carbonelli et al. [Carbonelli and Mitra 2006] abordam a eficiência energética em um cenário cooperativo com múltiplos saltos. Vajapeyam et al. [Vajapeyam et al. 2008] também apresentam um protocolo auxiliado por nós cooperadores, no qual estes usam o esquema de *amplify-and-forward* (que é, em maior parte, baseado na camada física). Han et al. [Han et al. 2008a] também propõem um esquema de cooperação baseado em *amplify-and-forward* e mostraram que, mesmo com a presença de ruído na camada física (e sua amplificação), a qualidade das transmissões é melhorada com a cooperação.

Han et al. [Han et al. 2008b] avaliaram os efeitos dos esquemas de *amplify-and-forward*, *decode-and-forward* e *estimate-and-forward*. Os autores apresentaram o *Wave Cooperative*, um protocolo baseado em *amplify-and-forward*. O resultado mostra que as técnicas cooperativas apresentam melhor desempenho do que os protocolos sem qualquer cooperação. Além disso, sua nova abordagem apresenta desempenho superior em relação à capacidade do canal. Wang et al. [Wang et al. 2011], também propõem um esquema de cooperação assíncrona, aplicável em cenários com atraso de propagação grande e variável. Os autores compararam esquemas de *amplify-and-forward*, *decode-and-forward* e a transmissão direta. Os autores mostram que a melhoria de cada esquema depende das condições de relação sinal-ruído (SNR). Por exemplo, a transmissão direta tem um melhor resultado para condições de SNR ótimas, enquanto o esquema de *amplify-and-forward* apresenta um melhor desempenho para cenários com condições de SNR ruins.

Uma série de trabalhos utilizam conjuntamente esquemas ARQ e transmissões cooperativas [Lee et al. 2010a, Lee and Cho 2011, Ghosh et al. 2013]. Por exemplo, Lee et al. [Lee et al. 2010a] propõem um esquema de ARQ S&W cooperativo em um canal acústico de salto único. Quando o nó de destino recebe um pacote errado, ele solicita uma retransmissão para o nó cooperador. O protocolo recruta os nós mais próximos primeiro e, neste caso, os autores assumem que cada nó conhece a distância inter-nó para seus nós vizinhos. Lee et al. [Lee and Cho 2011] propuseram o uso do protocolo

cooperativo em um cenário de múltiplos saltos, que neste caso melhora a diversidade espacial da comunicação. Ghosh et al. [Ghosh et al. 2013] também utilizam um protocolo cooperativo, mas neste caso, os autores propõem o uso do ARQ cooperativo e do ARQ híbrido, onde os dados são codificados com um código FEC e os bits FEC redundantes são transmitidos juntamente com o dados ou solicitados pelo destino quando erros são detectados. Apesar de sua importância, esses três trabalhos consideram cenários simples, onde apenas um nó origina mensagens. Além disso, os problemas da camada MAC são assumidos como resolvidos ou ignorados.

Kim et al. [Kim and Cho 2016] consideram um protocolo MAC baseado em *handshake* com um esquema ARQ cooperativo. O processo de *handshake* é baseado no mecanismo de *request-to-send* (RTS) e *clear-to-send* (CTS) e as informações de cooperação são compartilhadas durante o processo de *handshaking*. Apesar da melhora observada no desempenho, sua abordagem apresenta uma grande quantidade de mensagens de controle, o que pode causar mais colisões e também prolongar a duração do processo de *handshaking*, diminuindo a vazão global da rede.

Em suma, diferente dos trabalhos anteriores, que baseiam suas soluções em esquemas de camada física e *amplify-and-forward*, levamos em consideração a camada MAC e seus problemas. Nossa abordagem coordena efetivamente a retransmissão de mensagens que falharam quando mais de um nó compartilha o meio, usando o acesso múltiplo por divisão de tempo (TDMA). Além disso, escolhemos dinamicamente os nós para serem cooperadores de forma síncrona ou assíncrona, de acordo com as especificidades do protocolo ou ambiente de comunicação.

3. Retransmissão Cooperativa

Neste trabalho, consideramos uma rede de sensores de um único salto onde N nós tentam enviar dados para um nó destino (*sink*). Também consideramos um esquema de acesso ao meio baseado em divisão de tempo (TDMA) onde cada nó é atribuído a um período de tempo –também chamado de *time slot*– com acesso exclusivo ao meio de comunicação.

Note que a comunicação aquática sem fio segue um padrão de transmissão em *broadcast*, assim como a transmissão por rádio. Quando um nó tenta enviar dados para o destino, todos os outros nós podem receber esses dados. Nesse sentido, os nós ociosos podem atuar como cooperadores, aumentando a diversidade espacial ou temporal. Por exemplo, suponha que n seja o nó de origem, s o nó de destino e r qualquer nó ocioso localizado entre n e s. Quando n tenta enviar dados para s, r é mais suscetível a recebê-lo corretamente do que s (devido ao desvanecimento do canal). Nesse caso, quando s não recebe os dados corretamente, mas r recebe, o nó r pode oferecer seu intervalo de tempo para reenviar os dados acima mencionados para s. Então, a retransmissão ocorrerá em um enlace mais curto, com maior probabilidade de sucesso.

Primeiramente, propomos o ProCoopa síncrono, um protocolo cooperativo sincronizado para RSA. Por este protocolo, cada quadro do TDMA apresenta um período de sinalização (PS) seguido de um período de dados (PD). Os nós do sistema usam o período de sinalização para trocar mensagens de controle de cooperação, mostrando seus interesses em cooperar.

Mais precisamente, um nó –em seu intervalo de tempo– só é capaz de transmitir um único pacote em um único quadro. O Algoritmo 1 apresenta o pseudocódigo da

transmissão de pacotes do ProCoopa síncrono. Neste algoritmo o nó decide qual pacote vai transmitir em seu intervalo de tempo. A prioridade é sempre dos pacotes gerados pelo próprio nó, primeiramente do que falhou no quadro anterior e em seguida de novos pacotes que foram gerados e entraram em sua fila de transmissão. Assim, o nó só irá cooperar caso não tenha pacotes próprios para transmitir.

Para retransmitir pacotes cooperativos, os nós devem guardar os pacotes que conseguem receber sem erro. Então, quando um nó recebe um pacote de dados com sucesso, o qual, por definição, é direcionado para o nó destino, ele armazena essa mensagem em um *buffer* de cooperação. Ao final do quadro, o nó destino transmite um único ACK/NACK para indicar quais os pacotes do quadro atual tiveram sucesso/falharam. Ao receber esta mensagem, cada nó pode liberar do seu *buffer* de cooperação os pacotes entregues com sucesso, e sincronizar a retransmissão dos que falharam no próximo período de sinalização.

Algoritmo 1 Pseudocódigo da transmissão de pacotes do ProCoopa síncrono no n-ésimo nó

```
1:  função TRANSMITEPACOTE
2:      se pacote do quadro anterior falhou e não há cooperador então
3:          retransmite pacote
4:      senão se possui pacote na fila para transmitir então
5:          desenfileira pacote e transmite
6:      senão se possui pacote cooperativo para transmitir então
7:          transmite pacote cooperativo
8:      fim se
9:      calcula próximo intervalo de tempo
10: fim função
```

Durante o período de sinalização, os nós ociosos podem tentar cooperar enviando uma mensagem especial, indicando que querem cooperar (*Want to Cooperate* - WTC). A mensagem WTC indica qual pacote o nó vai retransmitir. Além disso, permite que o nó que originou o pacote continue a sua transmissão. Ao final do período de sinalização, cada nó já sabe qual pacote irá transmitir no quadro atual e então os nós podem esvaziar seu *buffer* de cooperação. Como consideramos um período de cooperação de janela única, cada nó aloca apenas o correspondente ao tamanho de $n - 1$ pacotes para seu *buffer* cooperativo, onde n é o número de nós da rede.

A Figura 1 exemplifica o protocolo proposto, retratando um cenário contendo 3 nós. Durante o quadro i, o nó de origem O transmite uma mensagem m_1, que é recebida com sucesso pelo nó R, mas chega com erro no nó de destino S. Ao final do quadro, S sinaliza que a mensagem falhou transmitindo um pacote NACK, que é recebido por todos os nós. Como R está ocioso, ele indica que ele irá cooperar durante o seu próximo intervalo de tempo. Assim, o nó O pode transmitir sua próxima mensagem m_2 no mesmo quadro que R retransmite m_1.

A Figura 2 ilustra um cenário mais realista com quatro nós sensores e uma nó destino. Durante o primeiro período de dados, os quatro nós não estão ociosos e tentam transmitir dados. De acordo com esta figura, as mensagens dos nós 3 e 4 falham, enquanto as mensagens restantes são entregues com sucesso. O nó destino envia um NACK, sinali-

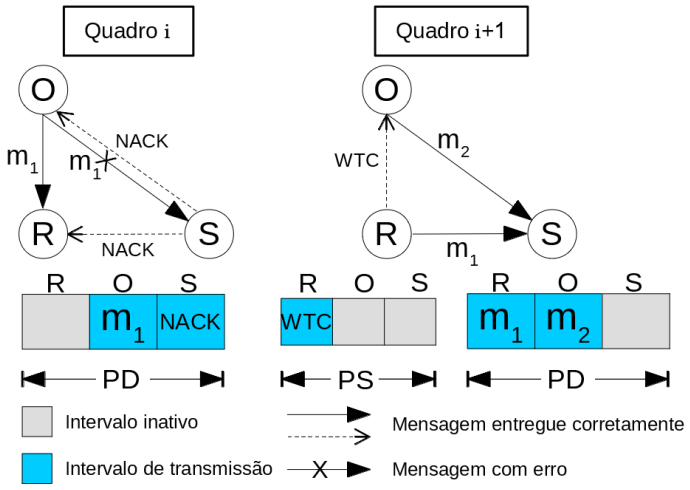


Figura 1. ProCoopa síncrono em um cenário com 3 nós.

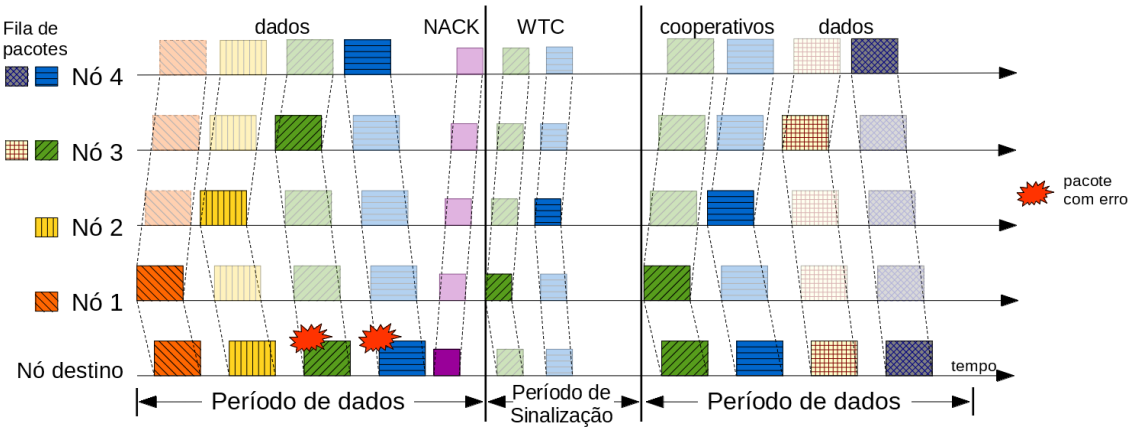


Figura 2. ProCoopa síncrono em um cenário com 5 nós.

zando que não recebeu corretamente as mensagens dos nós 3 e 4. Todos os nós, ao receber o NACK, verificam em seu *buffer* se eles têm alguma das mensagens perdidas. Caso positivo, eles sinalizam sua intenção de cooperar durante o próximo período de sinalização.

Nós assumimos uma ordem de prioridade para evitar a duplicação de retransmissões. O nó 1 é o primeiro a enviar sua mensagem WTC. Quando o nó 2 recebe a mensagem WTC do nó 1, percebe que o nó 1 cooperará retransmitindo a mensagem do nó 3. O nó 2 então sinaliza sua intenção de cooperar com o nó 4. Nós deixamos como trabalho futuro a investigação de outras políticas de de-duplicação, como por exemplo, seleção de nós com maior nível de energia. Ao receber a mensagem de cooperação, os nós 3 e 4 sabem que podem transmitir suas próximas mensagens. No próximo período de dados, as mensagens cooperativas dos nós 1 e 2 são enviadas e recebidas corretamente. Da mesma forma, as mensagens dos nós 3 e 4 também são entregues com sucesso.

A seleção do nó cooperador é crucial para o sucesso da cooperação. Intuitivamente, quanto mais perto o cooperador estiver do nó destino, melhor será a chance de uma retransmissão bem-sucedida. De fato, os nós mais próximos do nó destino apresentarão uma taxa de erro de pacote menor que os nós mais distantes, uma vez que a

taxa de erro do pacote é proporcional à distância. Em suma, ordenamos os intervalos de tempo dos nós em relação à sua distância do nó destino, semelhante ao que foi feito em [Domingo 2011]. Em outras palavras, o primeiro intervalo de tempo de cada quadro pertencerá ao nó mais próximo do destino. Desta forma, no ProCoopa síncrono, este nó apresentará uma maior oportunidade para atuar como um cooperador, pois será o primeiro nó a se anunciar como cooperador enviando uma mensagem WTC. A principal deficiência desta abordagem está relacionada ao balanceamento do consumo de energia. Os nós mais próximos do destino podem transmitir mais mensagens cooperativas do que outros nós, causando um consumo de energia desigual. Destacamos que diferentes políticas de seleção de retransmissão podem ser usadas. Por exemplo, as políticas podem considerar o nível de energia restante do nó.

Desenvolvemos também um protocolo assíncrono, onde os nós não precisam sinalizar para cooperar durante o período de sinalização. Um protocolo assíncrono pode simplificar o protocolo original e iniciar a transmissão de dados mais rapidamente (pois não é necessário um período de sinalização). Por outro lado, os nós não recebem a informação de quais mensagens cada nó cooperará, o que pode resultar na transmissão de mensagens duplicadas.

O ProCoopa assíncrono assume que os nós recebem mensagens NACK e simplesmente escolhem aleatoriamente uma das que possuem em seu *buffer* cooperativo. Nesta abordagem, a mesma mensagem que falhou pode ser retransmitida por vários nós, o que pode levar a um desperdício de energia. Por outro lado, pode aumentar o número de caminhos de transmissão e consequentemente, reduzir a taxa de erro do pacote.

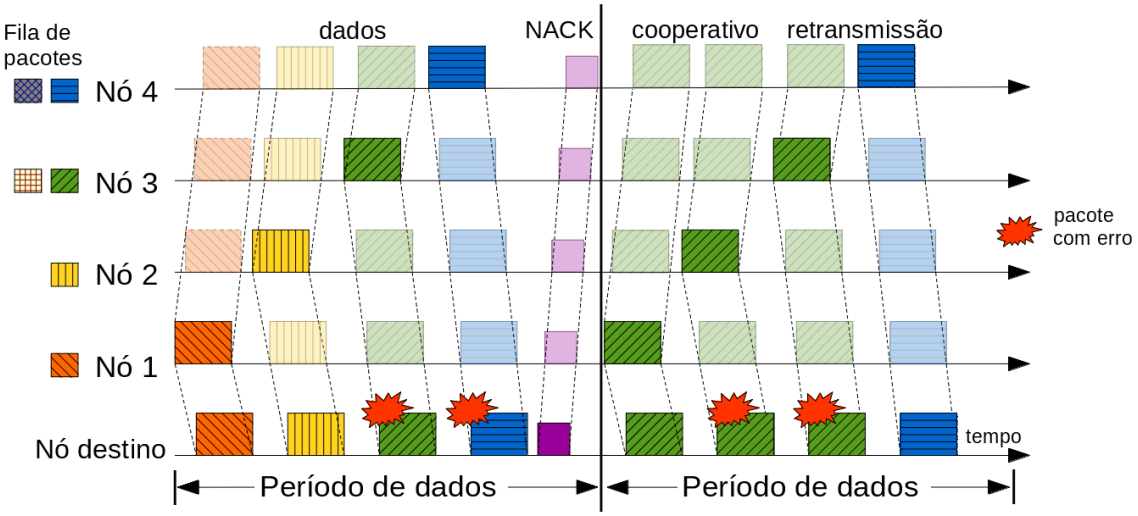


Figura 3. Dois quadros do ProCoopa assíncrono em um cenário com 5 nós.

A Figura 3 apresenta o protocolo assíncrono no mesmo cenário da Figura 2: 4 nós transmitindo mensagens para o nó *destino* em dois quadros. Observe que cada quadro contém apenas o período de dados, e não mais o período de sinalização e o período de dados. No primeiro quadro, cada nó possui uma mensagem para transmitir. No entanto, as mensagens dos nós 3 e 4 não são entregues corretamente. No final do quadro, o nó *destino* sinaliza com um NACK que as mensagens dos nós 3 e 4 falharam. No próximo quadro, os nós 1 e 2 não têm mensagens para transmitir, estando disponíveis para cooperar. Como

ambos receberam o NACK e as mensagens que falharam corretamente, escolhem aleatoriamente uma delas para cooperar. Neste exemplo, ambos acabam selecionando a mesma mensagem. O nó 3 também retransmite sua própria mensagem que falhou, totalizando três possíveis caminhos.

O protocolo proposto, tanto em sua versão síncrona quanto assíncrona, considera que todos os nós transmitem diretamente para o nó destino, ou seja, uma rede de salto único. Nos casos em que a área de cobertura é maior, uma rede de múltiplos saltos pode ser mais adequada. Para modificar o ProCoopa síncrono para permitir transmissões de vários saltos, os nós também devem transmitir mensagens de confirmação ACK/NACK. Uma possível solução seria diminuir o tamanho das mensagens de dados para que os nós comuns possam transmitir uma mensagem de confirmação após a mensagem de dados. Devido às restrições de espaço, deixamos como trabalho futuro a avaliação de um cenário de múltiplos saltos.

4. Avaliação

4.1. Metodologia de Avaliação

Nós avaliamos o ProCoopa através de simulações, usando o ns-3, um simulador de rede baseado em eventos discretos. Implementamos o ProCoopa síncrono e assíncrono, e um protocolo base TDMA.

Os nós da rede aquática são responsáveis por gerar pacotes de dados e transmitilos ao nó destino. Cada nó pode atuar como um cooperador quando inativo. O nó destino não gera dados e só recebe os pacotes e transmite mensagens confirmando ou não o recebimento dos dados (i.e. ACK/NACK). Neste trabalho, os nós são estáticos e são distribuídos aleatoriamente em uma topologia em estrela dentro de uma área quadrada de 200 m de lado e 70 m de profundidade. Os nós são reposicionados em cada execução da simulação, exceto o nó destino que está sempre posicionado no centro do quadrado.

Para gerar valores aleatórios, o ns-3 implementa um algoritmo baseado em fluxos e sub-fluxos. Cada fluxo gera um conjunto de sub-fluxos que não se sobrepõem. Assim, para produzir múltiplas execuções independentes, nós fixamos o fluxo escolhendo um valor para a semente e alteramos apenas o sub-fluxo para cada execução. Neste trabalho, usamos 138 como semente e i como sub-fluxo para a i-ésima execução ($i \in \mathbb{N}^*$). Cada simulação considera 50 execuções e, a menos quando citado o contrário, os resultados que apresentamos são valores médios com o intervalo de confiança, para um nível de confiança de 95%.

Nós geramos o tráfego de dados aleatoriamente de acordo com uma distribuição uniforme. Um nó pode gerar um pacote de acordo com uma probabilidade L em cada início de quadro. Chamamos essa probabilidade L de carga da rede. Em outras palavras, quando a carga de rede é $L = 0$, nenhum pacote é gerado em toda a simulação, e quando $L = 1$, todos os nós sempre terão um pacote para transmitir em cada quadro. Nós variamos L entre 10% e 100% de utilização ($L = 0,1$ a $L = 1$) para avaliar o desempenho do protocolo com diferentes cargas de rede.

Cada execução simula o funcionamento da rede durante uma hora. O tamanho dos pacotes de dados é configurado para 540 bytes, dos pacotes WTC para 3 bytes e dos pacotes NACK para 5 bytes, que estão na mesma ordem de grandeza

que [Ghosh et al. 2013, Azad et al. 2013]. As configurações do transdutor são baseadas no modem acústico UNET-2 [Chitre et al. 2012]: taxa de dados de 2.400 bps, frequência central de 4.000 Hz e modulação BPSK, largura de banda de 2.000 Hz, potência de transmissão de 138 dB, consumo de energia para transmissão de pacotes de 50 W, consumo de energia para recepção de pacotes de 158 mW e consumo de energia no modo inativo de 158 mW. Os intervalos de dados de cada nó apresentam duração de 2 s, com 1,8 s para transmissão de dados e 0,2 s como tempo de guarda para evitar colisões de pacotes. No período de sinalização, os intervalos de controle duram 0,2 s, o que representa apenas 10% do intervalo de dados.

Para calcular a probabilidade de sucesso de entrega de um pacote utilizamos modelos amplamente adotados na literatura. Para cada par de nós distantes d metros calculamos a atenuação do sinal acústico usando [Urick 1975]: $10\,log\,A(d,f) = k \cdot 10\,log\,d + d \cdot 10\,log\ a(f)$, onde o primeiro termo é a perda por espalhamento do sinal e f a frequência de transmissão. O segundo termo é a perda por absorção dada pela aproximação de Thorp [Brekhovskikh et al. 1991]: $0{,}11\,\frac{f^2}{1+f^2} + 44\frac{f^2}{4100+f} + 2{,}75 \cdot 10^{-4} f^2 + 0{,}003$, para frequências $f \geq 0{,}4$ Hz. Com isso calculamos a razão entre sinal-ruído $\gamma(d) = SL - A(d,f) - NL + DI$, onde SL é a potência do sinal de transmissão, NL é o barulho ambiente aproximado pela equação de Wenz em [Wenz 1962], e DI é o fator diretivo que para hidrofones (e.g. modens acústicos) omnidirecionais $DI = 0$ [Wang et al. 2016]. Finalmente, podemos descrever a probabilidade de erro no bit, para a modulação BPSK [Rappaport et al. 1996], como: $p_e(d) = \frac{1}{2}\left(1 - \sqrt{\frac{\Gamma(d)}{1+\Gamma(d)}}\right)$, onde $\Gamma(d)$ é dado por $\Gamma(d) = 10^{\gamma(d)/10}$.

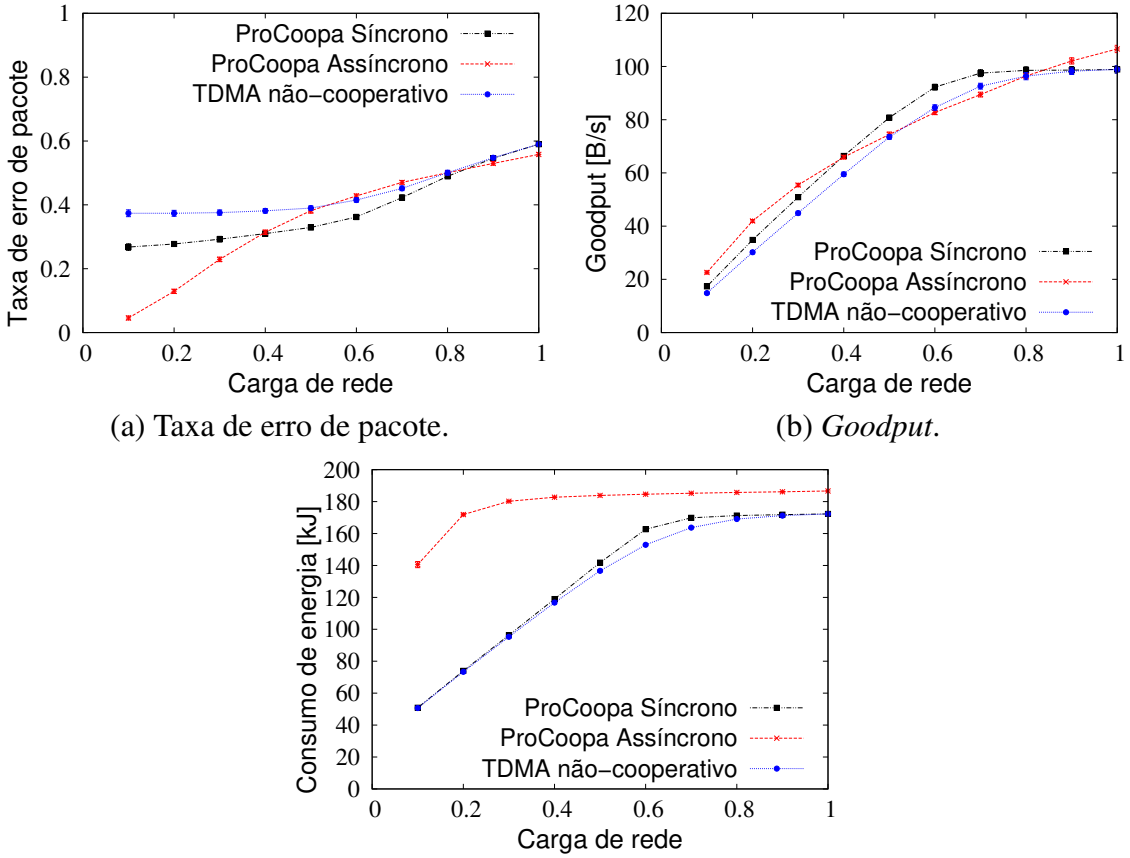
Neste trabalho, assumimos que a rede é capaz de manter o tempo sincronizado entre os nós. De fato, a sincronização dos períodos de tempo (*slots* de tempo) de todos os nós da rede é de extrema importância para a operação do protocolo [Viana et al. 2015]. Se um nó não estiver sincronizado, ele pode levar a colisões no nó destino, impactando negativamente o desempenho da rede. Para evitar esse problema, a sincronização do tempo pode ser realizada usando uma funcionalidade do próprio modem acústico UNET-2 [Anjangi and Chitre 2015]. Assim, o modem é usado para obter a diferença do relógio entre os nós e compensá-lo de acordo, de modo que os intervalos de tempo também podem ser sincronizados.

Avaliamos o ProCoopa síncrono/assíncrono e um sistema sem cooperação, de acordo com um conjunto de três métricas: (i) o *goodput*, que corresponde à quantidade de bytes de pacotes de dados que o nó destino recebe, em relação ao período total de simulação; (ii) taxa de erro de pacote (PER), que representa a porcentagem de pacotes de dados que não são devidamente recebidos pelo nó destino; (iii) energia total gasta.

4.2. Resultados

A Figura 4 apresenta os valores médios das métricas de desempenho (e intervalo de confiança) quando variamos a carga da rede de trabalho da rede. De acordo com a Figura 4a, o ProCoopa assíncrono apresenta menor PER quando comparado ao protocolo não cooperativo. Por exemplo, com uma carga de rede de 10%, o ProCoopa assíncrono é 87% melhor do que o TDMA não cooperativo. De fato, enquanto o primeiro apresenta média de cerca de 4% de PER, o último apresenta mais de 37%.

Analisando o cenário com baixa carga de rede, observamos que há um grande número de nós ociosos. Sem sincronização de nós e com um grande número de nós ociosos, muitos nós podem retransmitir um mesmo pacote e, como consequência, aumentar sua chance de uma transmissão bem-sucedida. Nesse caso, como há um grande número de nós ociosos, todos os pacotes que falharam têm maior chance de serem retransmitidos por pelo menos um nó cooperador. Por outro lado, quando a carga da rede aumenta (e.g., $> 0,4$), o ProCoopa síncrono supera a versão assíncrona. Nesse caso, a sincronização evita que dois ou mais nós retransmitam o mesmo pacote, o que permite a cobertura de um número maior de pacotes distintos que precisam ser retransmitidos.



(a) Taxa de erro de pacote.

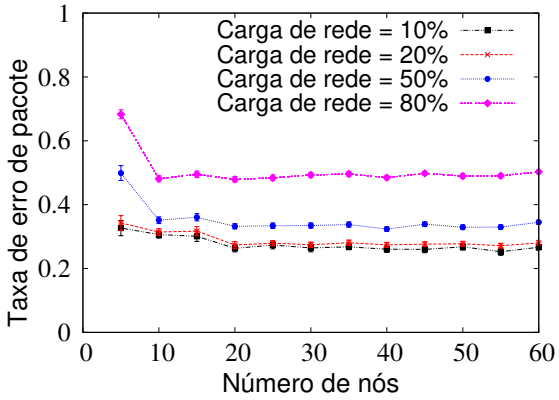
(b) *Goodput*.

(c) Consumo de energia.

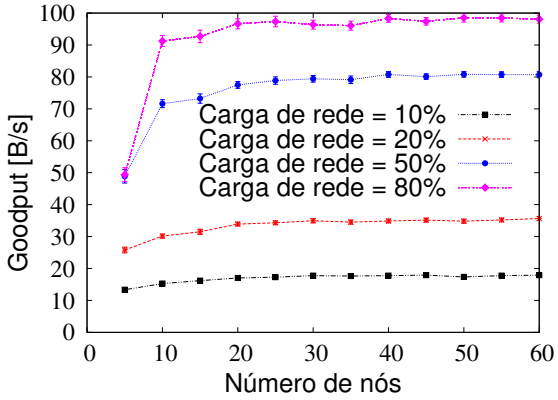
Figura 4. Desempenho dos protocolos avaliados para uma rede de sensor sem fio aquática com 50 nós.

Apesar das melhorias notáveis na taxa de erros de pacotes (PER) em cenários de carga de rede mais baixas, os três protocolos tendem a apresentar PER semelhantes em cenários de alta carga (por exemplo, $> 0,8$). Na verdade, em cenários de alta carga, todos os nós estão transmitindo dados praticamente o tempo todo. Nesse caso, os nós não estão ociosos e não têm a oportunidade de cooperar. Por outro lado, para uma menor carga de rede, o protocolo cooperativo obtém melhores resultados. Quanto menor a carga da rede, mais nós estão ociosos e, consequentemente, maior a chance de algum nó cooperar.

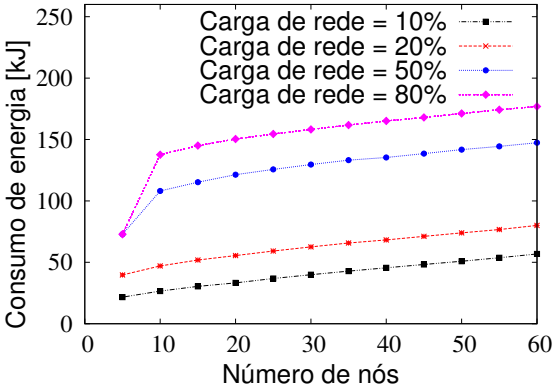
Em cargas de rede mais baixas, o ProCoopa assíncrono também apresenta melhor vazão (*goodput*) quando comparado à versão síncrona e ao protocolo não cooperativo. Como mostrado na Figura 4-b, o ProCoopa assíncrono é, em média, 52% melhor que o



(a) PER do ProCoopa síncrono.



(b) *Goodput* do ProCoopa síncrono.



(c) Consumo de energia do ProCoopa síncrono.

Figura 5. Escalabilidade do ProCoopa síncrono.

TDMA não cooperativo para 10% de carga de rede. No entanto, como acontece com o PER (e pelas mesmas razões), o ProCoopa síncrono apresenta melhor *goodput* quando a carga da rede aumenta.

Apesar dos ganhos em *goodput* e taxa de erro de pacote, o protocolo ProCoopa assíncrono exige mais energia para seu correto funcionamento. De fato, quando vários nós tentam cooperar sem qualquer sincronismo, eles podem desperdiçar energia, retransmitido o mesmo pacote de dados. De acordo com nossos resultados, com uma carga de trabalho de rede de 10%, o ProCoopa assíncrono consome quase três vezes mais energia, quando comparado ao TDMA não cooperativo. Nesse cenário –baixa carga de rede–, o ProCoopa síncrono apresenta melhor *goodput* e PER enquanto consume praticamente a mesma quantidade de energia do TDMA não cooperativo.

Por fim, a Figura 5 avalia a escalabilidade do ProCoopa. Nos concentramos no ProCoopa síncrono uma vez que a sincronização leva a uma sobrecarga (*overhead*) no protocolo que pode ser afetada pelo número de nós. De acordo com esta figura, o PER permanece aproximadamente estável –Figura 5-a– enquanto que o *goodput* e o consumo de energia –Figuras 5-b e 5-c respectivamente– aumentam ligeiramente com o número de nós da rede. Aumentar a rede também aumentará a diversidade de caminho entre os nós de origem, cooperadores e o nó destino. Por sua vez, a diversidade do caminho

pode reduzir a probabilidade de erro, o que levará a um melhor desempenho da rede. Como esperado, o consumo de energia aumenta com o número de nós de rede, mas, neste caso, notamos claramente que o consumo de energia escala. Finalmente, como afirmamos anteriormente, quanto maior a carga da rede, menor o desempenho (por exemplo, maior PER, menor *goodput* e maior consumo de energia).

5. Conclusões

Neste artigo, discutimos a comunicação cooperativa na camada de controle de acesso ao meio para redes de sensores aquáticos. Nós propomos o ProCoopa, um novo protocolo MAC cooperativo para RSAs. Nossa técnica de cooperação é baseada em um esquema de *automatic repeat request* com *selective repeat* e incorpora a sinalização de erros e a retransmissão de mensagens no controle de acesso ao meio. Os nós que, de outra forma, ficariam ociosos, são responsáveis pela retransmissão de mensagens com falha. Nossos resultados de simulação mostram uma melhora nas métricas de taxa de perda de pacotes e *goodput*, com pequeno impacto no consumo de energia. Mais especificamente, o ProCoopa síncrono alcança uma redução de 28,32% na taxa de perda de pacotes no melhor dos casos. O protocolo proposto funciona melhor em cargas de rede médias a baixas, o que corresponde à maioria das aplicações de redes de sensores aquáticos [Tomasi et al. 2015]. Ainda assim, o protocolo é capaz de manter ganhos em cargas de rede mais altas.

Trabalhos futuros incluem o estudo de diferentes técnicas de seleção de nós cooperadores, bem como a cooperação usando outros protocolos MAC, além da implementação das propostas em um ambiente real.

Referências

Anjangi, P. and Chitre, M. (2015). Design and implementation of super-tdma: A mac protocol exploiting large propagation delays for underwater acoustic networks. In *Proc. of the 10th International Conference on Underwater Networks & Systems*. ACM.

Azad, S., Casari, P., Guerra, F., and Zorzi, M. (2011). On arq strategies over random access protocols in underwater acoustic networks. In *Proc. of IEEE OCEANS*.

Azad, S., Casari, P., and Zorzi, M. (2013). The underwater selective repeat error control protocol for multiuser acoustic networks: Design and parameter optimization. *IEEE Transactions on Wireless Communications*, 12(10):4866–4877.

Basagni, S., Petrioli, C., Petroccia, R., and Spaccini, D. (2015). Carp: A channel-aware routing protocol for underwater acoustic wireless networks. *Ad Hoc Networks*, 34:92–104.

Brekhovskikh, L. M., Lysanov, Y. P., and Beyer, R. T. (1991). Fundamentals of ocean acoustics. *The Journal of the Acoustical Society of America*, 90(6):3382–3383.

Carbonelli, C., Chen, S.-H., and Mitra, U. (2009). Error propagation analysis for underwater cooperative multi-hop communications. *Ad Hoc Networks*, 7(4):759–769.

Carbonelli, C. and Mitra, U. (2006). Cooperative multihop communication for underwater acoustic networks. In *Proc. of the 1st ACM international workshop on Underwater networks*.

Chitre, M., Topor, I., and Koay, T.-B. (2012). The unet-2 modem—an extensible tool for underwater networking research. In *Proc. of IEEE OCEANS*.

Coutinho, R. W., Boukerche, A., Vieira, L. F., and Loureiro, A. A. (2015). Modeling and analysis of opportunistic routing in low duty-cycle underwater sensor networks. In *Proc. of the 18th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*.

Coutinho, R. W., Boukerche, A., Vieira, L. F., and Loureiro, A. A. (2016a). Geographic and opportunistic routing for underwater sensor networks. *IEEE Transactions on Computers*, 65(2):548–561.

Coutinho, R. W., Vieira, L. F. M., and Loureiro, A. A. F. (2013). DCR: Depth-controlled routing protocol for underwater sensor networks. In *2013 IEEE Symposium on Computers and Communications (ISCC)*.

Coutinho, R. W. L., Boukerche, A., Vieira, L. F. M., and Loureiro, A. A. F. (2014). GEDAR: geographic and opportunistic routing protocol with depth adjustment for mobile underwater sensor networks. In *IEEE Int. Conference on Communications*.

Coutinho, R. W. L., Boukerche, A., Vieira, L. F. M., and Loureiro, A. A. F. (2016b). Design guidelines for opportunistic routing in underwater networks. *IEEE Communications Magazine*, 54(2):40–48.

Demirors, E., Sklivanitis, G., Santagati, G. E., Melodia, T., and Batalama, S. N. (2014). Design of a software-defined underwater acoustic modem with real-time physical layer adaptation capabilities. In *Proc. of the ACM Int. Conference on Underwater Networks & Systems*.

Domingo, M. C. (2011). A distributed energy-aware routing protocol for underwater wireless sensor networks. *Wireless Personal Communications*, 57(4):607–627.

Ghosh, A., Lee, J.-W., and Cho, H.-S. (2013). Throughput and energy efficiency of a cooperative hybrid arq protocol for underwater acoustic sensor networks. *Sensors*, 13(11):15385–15408.

Han, J.-W., Ju, H.-J., Kim, K.-M., Chun, S.-Y., and Dho, K.-C. (2008a). A study on the cooperative diversity technique with amplify and forward for underwater wireless communication. In *Proc. of IEEE OCEANS*.

Han, Z., Sun, Y. L., and Shi, H. (2008b). Cooperative transmission for underwater acoustic communications. In *IEEE Int. Conference on Communications*.

Kim, H.-w. and Cho, H.-S. (2016). A cooperative arq-based mac protocol for underwater wireless sensor networks. In *Proc. of the 11th ACM International Conference on Underwater Networks & Systems*.

Laneman, J. N., Tse, D. N., and Wornell, G. W. (2004). Cooperative diversity in wireless networks: Efficient protocols and outage behavior. *IEEE Trans. on Information theory*, 50(12):3062–3080.

Lee, J. W., Cheon, J. Y., and Cho, H.-S. (2010a). A cooperative arq scheme in underwater acoustic sensor networks. In *Proc. of IEEE OCEANS*.

Lee, J. W. and Cho, H.-S. (2011). A cooperative arq scheme for multi-hop underwater acoustic sensor networks. In *IEEE Symposium on Underwater Technology and Workshop on Scientific Use of Submarine Cables and Related Technologies*.

Lee, U., Wang, P., Noh, Y., Vieira, L. F. M., Gerla, M., and Cui, J.-H. (2010b). Pressure routing for underwater sensor networks. In *Proc. of the IEEE Conference on Computer Communications (INFOCOM)*.

Pinto, D., Viana, S. S., Nacif, J. A. M., Vieira, L. F., Vieira, M. A., Vieira, A. B., and Fernandes, A. O. (2012). Hydronode: a low cost, energy efficient, multi purpose node for underwater sensor networks. In *Proc. of the IEEE Conference on Local Computer Networks (LCN)*.

Rappaport, T. S. et al. (1996). *Wireless communications: principles and practice*, volume 2. prentice hall PTR New Jersey.

Renner, C. and Golkowski, A. J. (2016). Acoustic modem for micro auvs: design and practical evaluation. In *Proc. of the ACM International Conference on Underwater Networks & Systems*.

Sozer, E. M., Stojanovic, M., and Proakis, J. G. (2000). Underwater acoustic networks. *IEEE journal of oceanic engineering*, 25(1):72–83.

Tomasi, B., Casari, P., Badia, L., and Zorzi, M. (2015). Cross-layer analysis via markov models of incremental redundancy hybrid arq over underwater acoustic channels. *Ad Hoc Networks*, 34:62–74.

Urick, R. J. (1975). Principles of underwater sound-2.

Vajapeyam, M., Vedantam, S., Mitra, U., Preisig, J. C., and Stojanovic, M. (2008). Distributed space–time cooperative schemes for underwater acoustic communications. *IEEE Journal of Oceanic Engineering*, 33(4):489–501.

Viana, S. S., Vieira, L. F., Vieira, M. A., Nacif, J. A. M., and Vieira, A. B. (2015). Survey on the design of underwater sensor nodes. *Design Automation for Embedded Systems*, pages 1–20.

Vieira, L., Loureiro, A., Fernandes, A., and Campos, M. (2010). Redes de sensores aquáticas. *XXVIII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos, Gramado, RS, Brasil*, 1:199–240.

Vieira, L. F. M. (2012). Performance and trade-offs of opportunistic routing in underwater networks. In *IEEE Wireless Communications and Networking Conference*.

Vieira, L. F. M., Vieira, M. A. M., Nacif, J. A., and Vieira, A. B. (Jan. 2018). Autonomous wireless lake monitoring. In *Computing in Science & Engineering (Print)*.

Wang, H., Wang, S., Zhang, E., and Zou, J. (2016). A network coding based hybrid arq protocol for underwater acoustic sensor networks. *Sensors*, 16(9):1444.

Wang, P., Feng, W., Zhang, L., and Li, V. O. (2011). Asynchronous cooperative transmission in underwater acoustic networks. In *Underwater Technology (UT), 2011 IEEE Symposium on and 2011 Workshop on Scientific Use of Submarine Cables and Related Technologies (SSC)*.

Wenz, G. M. (1962). Acoustic ambient noise in the ocean: Spectra and sources. *The Journal of the Acoustical Society of America*, 34(12):1936–1956.