TESIS – TE 142599

DESAIN PROTOKOL MEDIA ACCESS CONTROL (MAC) UNTUK SISTEM KOMUNIKASI KOOPERATIF PADA JARINGAN AKUSTIK BAWAH AIR

MUHAMMAD SYIRAJUDDIN S.
2213 203 204

DOSEN PEMBIMBING
Dr. Ir. Wirawan, DEA
Dr. Ir. Endroyono, DEA

PROGRAM MAGISTER
BIDANG KEAHLIAN TELEKOMUNIKASI MULTIMEDIA
JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2016

THESIS – TE 142599

MEDIA ACCESS CONTROL (MAC) PROTOCOL DESIGN FOR COOPERATIVE UNDERWATER ACOUSTIC NETWORK

MUHAMMAD SYIRAJUDDIN S.
2213 203 204

SUPERVISORS
Dr. Ir. Wirawan, DEA
Dr. Ir. Endroyono, DEA

MASTER PROGRAM
MULTIMEDIA TELECOMMUNICATION
DEPARTMENT OF ELECTRICAL ENGINEERING
FACULTY OF INDUSTRIAL ENGINEERING
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2016

Tesis disusun untuk memenuhi salah satu syarat memperoleh gelar
Magister Teknik (MT)
di
Institut Teknologi Sepuluh Nopember

oleh:

Muhammad Syirajuddin S.
NRP. 2213 203 204

Tanggal ujian : 18 Januari 2016
Periode wisuda : Maret 2016

Disetujui oleh:

1. **Dr. Ir. Wirawan, DEA** (Pembimbing I)
   NIP. 196311091989031011

2. **Dr. Ir. Endroyono, DEA** (Pembimbing II)
   NIP. 196504041991021001

3. **Dr. Ir. Achmad Affandi, DEA** (Penguji)
   NIP. 196510141990021001

4. **Dr. Ir. Suwadi, MT.** (Penguji)
   NIP. 196803181993031002

5. **Dr. Ir. Titiek Suryani, MT.** (Penguji)
   NIP. 19641130198903200l

Direktur Program Pascasarjana,

Prof. Ir. Djauhar Manfaat, M.Sc. Ph.D
NIP. 196012021987011001

DESAIN PROTOKOL MEDIA ACCESS CONTROL (MAC) UNTUK SISTEM KOMUNIKASI KOOPERATIF PADA JARINGAN AKUSTIK BAWAH AIR

Nama Mahasiswa	: Muhammad Syirajuddin S.
NRP	: 2213203204
Pembimbing	: Dr. Ir. Wirawan, DEA
	: Dr. Ir. Endroyono, DEA

ABSTRAK

Implementasi jaringan sensor nirkabel pada bawah air bisa dilakukan dengan menggunakan sistem komunikasi akustik. Sistem komunikasi akustik adalah sistem komunikasi yang gelombang pembawanya berupa gelombang suara. Sistem komunikasi akustik mempunyai tantangan yang unik bila dibandingkan dengan sistem komunikasi radio atau optik di udara, diantaranya adalah kecepatan perambatan gelombang suara yang lambat yang mengakibatkan durasi propagasi menjadi tinggi dan sempitnya lebar pita yang tersedia.

Disamping itu, bila sistem komunikasi akustik bawah air dikehendaki untuk jaringan sensor nirkabel yang pada umumnya terdiri atas banyak *node*, maka diperlukan pengaturan penggunaan kanal yang biasanya disebut sebagai protokol *Media Access Control* (MAC). MAC yang sudah ada untuk jaringan komunikasi nirkabel di udara tidak bisa langsung digunakan di jaringan bawah air karena tingginya delay propagasi gelombang akustik. Perlu desain unik agar bisa sesuai dengan karakteristik propagasi gelombang akustik di air.

Oleh karena itu, untuk mengatasi tantangan jaringan akustik bawah air, diusulkan protokol *Cooperative* MAC *for Underwater* (CoopMAC-U). Dari hasil simulasi menunjukkan bahwa CoopMAC-U memberikan *throughput* yang lebih bagus dibandingkan dua protokol yang lain, yaitu ALOHA dan MACA-U (*Multiple Access Collision Avoidance for Underwater*).

Kata Kunci: Akustik, Bawah Air, Jaringan, *Media Access Control*, Protokol.

MEDIA ACCESS CONTROL (MAC) PROTOCOL DESIGN FOR COOPERATIVE UNDERWATER ACOUSTIC NETWORK

Name	: Muhammad Syirajuddin S.
NRP	: 2213203204
Advisor	: Dr. Ir. Wirawan, DEA
	: Dr. Ir. Endroyono, DEA

ABSTRACT

Wireless sensor networks implementation in the underwater environment is possible using acoustic communication system. The Acoustic communication system is communication that is enabled by using sound wave. It has unique challenges, such as the sound wave propagates in underwater very slowly that makes delay propagation becomes high and the limited available bandwidth.

In addition to that, wireless sensor network usually consists of many sensor nodes which use a common channel to communicate to each other or to the sink. So, it requires a mechanism to control the channel usage using a protocol that is known as Media Access Control (MAC) Protocol. The existing MAC protocol for the wireless system in the air cannot be used directly in an underwater network because its high propagation delay.

Therefore, to overcome the challenges in the underwater acoustic network, Cooperative MAC for Underwater (CoopMAC-U) protocol is proposed. Simulation results show that CoopMAC-U gives the best throughput compared to two other different protocols, Aloha and MACA-U (Multiple Access Collision Avoidance for Underwater)

Keywords: Acoustic, Media Access Control, Network, Protocol, Underwater

KATA PENGANTAR

Dengan menyebut nama Allah Yang Maha Pengasih lagi Maha Penyayang.

Segala puji dan syukur kepada Allah atas segala rahmat dan karunia yang telah dilimpahkan sehingga penulisan tesis ini bisa diselesaikan dengan baik. Buku tesis ini disusun untuk memenuhi salah satu syarat memperoleh gelar Magister pada Program Studi Teknik Elektro, Bidang Keahlian Telekomunikasi Multimedia, Institut Teknologi Sepuluh Nopember.

Penulis mengucapkan banyak-banyak terima kasih kepada semua pihak yang telah memberikan segenap dukungan dalam penyusunan tesis ini khususnya kepada :

1. Orang tua tercinta, Aby Sudja'i, S.Ag. dan Ummy Rosyidah yang tanpa lelah terus mendukung, merestui dan mendoakan setiap langkah penulis. Serta kepada Neng Nur Rohmatul Laily S. sekeluarga yang telah memberikan dukungan yang tak ternilai.
2. Bapak Dr. Ir. Wirawan, DEA. yang telah berkenan menjadi pembimbing tesis sekaligus pembimbing akedemik.
3. Bapak Dr. Ir. Endroyono, DEA atas bimbingan dan motivasi dalam menyelesaikan tesis ini.
4. Prof. Tsuyoshi Usagawa dan Dr. Yoshifumi Chisaki yang telah mengijinkan tinggal di lab HICC Kumamoto University selama *spring semester* 2015. Serta semua anggota lab HICC, terutama Mr. Taira Onoguchi dan Mas Irwansyah yang membantu penulis beradaptasi di lingkungan baru.
5. Bapak dan Ibu Dosen S2 atas ilmu pengetahuan yang telah diberikan selama kuliah.
6. *Underwater Research Group* (Bapak Wirawan, Bapak Dhany Arifianto, Ibu Endang Widjiati, Bapak Tri Budi Santoso, Ibu Yuning Widiarti, Niken, Azran, Arum dan Arum (Fisika).
7. Rekan-rekan S2 dan S1 "penghuni" Lab Komunikasi Multimedia B304 yang telah berbagi kecerian di kondisi apapun.

Penulis menyadari bahwa dalam penulisan tesis ini masih jauh dari sempurna, untuk itu demi perbaikan dan penyempurnaan tesis, maka saran dan kritik membangun dari pembaca sangat penulis harapkan. Besar harapan penulis bahwa tesis ini dapat memberikan kontribusi untuk pengembangan keilmuan dan bermanfaat untuk masyarakan Indonesia.

Surabaya, Januari 2016

DAFTAR ISI

[*Halaman ini sengaja dikosongkan.*]

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1 Latar Belakang

Dengan luasnya lautan yang mencapai lebih dari 70% permukaan bumi [1] [2], ada banyak hal yang menarik untuk dipelajari, diteliti dan dimanfaatkan. Hal inilah yang mendorong untuk memasang jaringan sensor nirkabel di laut untuk bermacam keperluan seperti *environmental monitoring*, eksplorasi bawah laut, peringatan dini bencana, navigasi, *tactical surveillance* dan pendeteksian ranjau untuk keperluan militer [3]. Namun, untuk mengimplementasikan jaringan sensor nirkabel di bawah laut tidak semudah di permukaan laut atau di darat.

Pada umumnya, jaringan sensor nirkabel akan memanfaatkan gelombang RF (*radio frequency*) untuk mengirim dan menerima data. Namun, penggunaan gelombang RF ini akan sulit diterapkan untuk komunikasi jaringan bawah laut karena gelombang RF akan teredam dengan besar redaman yang berbading lurus dengan frekuensi yang dipakai dan jarak [4]. Peredaman ini terjadi karena sifat air laut yang mempunyai konduktifitas dan permitifitas yang tinggi.

Kemungkinan pemakaian gelombang optik pun juga sudah pernah diteliti untuk digunakan sebagai media komunikasi jaringan bawah laut. Komunikasi optik butuh kepresisian yang tinggi untuk mengarahkan pancaran laser yang sempit [3]. Oleh karena itu, jaringan akustik yang memanfaatkan gelombang suara menjadi alternatif paling baik dibandingkan dua metode sebelumnya karena suara bisa merambat dengan baik dalam air dan butuh daya yang lebih rendah dibandingkan RF dan optik untuk mencapai jarak yang sama dalam air [2].

Meski lebih unggul dibanding gelombang optik dan RF, penggunaan gelombang suara (akustik) untuk jaringan bawah laut bukan tanpa kendala, salah satunya adalah sinyal akustik merambat sangat lambat (kurang lebih 1500 meter/detik) dibandingkan dengan gelombang RF atau optik. Cepat rambat suara bawah laut bervariasi tergantung dari temperatur, salinitas dan tekanan atau

kedalaman air [5]. Dengan rendahnya kecepatan perambatan suara bawah air, maka *delay* propagasi sinyal akustik pun sangat tinggi.

Di samping masalah *delay* propagasi yang tinggi, *node* di suatu jaringan bawah laut pada umumnya dikehendaki berjumlah lebih dari satu dan akan berbagi kanal yang sama untuk saling berkomunikasi antar *node* dengan menggunakan mekanisme multiakses. Karena memakai kanal yang sama, maka perlu protokol yang mengatur bagaimana pemakaian medium yang biasa disebut dengan protokol MAC (*Media Access Control*) yang bertujuan untuk mengoptimalkan penggunaan kanal. Protokol MAC untuk kanal nirkabel di udara tidak bisa begitu saja diimplementasikan di kanal akustik bawah laut yang mempunyai *delay* propagasi yang tinggi. Perlu penyesuaian atau diperlukan desain khusus untuk menyesuaikan dengan kanal akustik bawah laut yang unik.

Selain itu, *bandwidth* yang tersedia dan frekuensi optimal yang bisa digunakan pada sistem komunikasi akustik berbeda-beda tergantung pada jarak transmisi. Semakin jauh jarak transmisi, semakin sempit *bandwidth* yang tersedia dan semakin rendah pula frekuensi optimalnya. Oleh karena itu, untuk meningkatkan *bandwidth* yang tersedia bisa dilakukan dengan cara membagi jarak transmisi yang jauh menjadi lebih pendek dengan menambahkan *relay* di tengahnya [6]. Dengan adanya *relay*, maka dibutuhkan tambahan slot waktu untuk meneruskan data dari *relay* ke tujuan. Kebutuhan slot waktu tambahan ini perlu dipertimbangkan juga dalam desain protokol MAC.

Terdapat beberapa contoh protokol MAC yang didesain khusus untuk meningkatkan efisiensi kanal bawah laut diantaranya adalah RIPT MAC, BiC-MAC dan ROPA MAC. Protokol RIPT dan ROPA meningkatkan pemanfaatan kanal dengan cara menggabungkan proses *handshaking*, selanjutnya *node* hanya mengirimkan data sesuai dengan urutan [7] [8]. BiC MAC mengusulkan pengiriman data dengan teknik *burst* dengan durasi paket yang pendek sehingga memungkinkan terjadi komunikasi dua arah dalam satu waktu tanpa terjadi tabrakan data [9]. Ketiga protokol di atas sudah memasukkan skenario *multi-hop* dalam desainnya. Namun, diperlukan reservasi kanal lagi untuk mentransmisikan ulang dari dari *relay*.

Selain dengan mendesain khusus protokol MAC, untuk meningkatkan kapasitas kanal akustik bawah laut bisa juga dengan memanfaatkan sistem komunikasi kooperatif [10] [11]. Pada sistem komunikasi kooperatif, di samping pemancar mengirimkan data langsung ke penerima, juga memanfaatkan *node* lain sebagai *relay* untuk mentransmisikan data ke penerima (*multi-hop*) [12]. Dengan demikian, pada sistem komunikasi kooperatif terdapat dua kali pemakaian kanal untuk satu pengiriman data. Sebatas pengetahuan penulis, belum ada protokol MAC untuk kanal akustik bawah air yang sudah mempertimbangkan sistem kooperatif dalam desainnya. Hal ini mendorong penulis untuk mengusulkan protokol kooperatif yang diberi nama dengan CoopMAC-U (*Cooperative MAC for Underwater*).

1.2 Perumusan Masalah

Berdasarkan latar belakang di atas, maka pada penelitian ini akan dilakukan modifikasi protokol MAC yang berbasis *contention* yang mempertimbangkan karakter kanal akustik yang unik sekaligus mempertimbangkan kebutuhan *time slot* tambahan untuk mendapatkan manfaat dari sistem komunikasi kooperatif yang akan coba diterapkan untuk meningkatkan kinerja jaringan bawah laut. Protokol yang dihasilkan akan dievaluasi menggunakan simulator.

1.3 Tujuan Penelitan

Tujuan dilakukan penelitian ini adalah sebagai berikut :

a. Mengusulkan protokol MAC yang *delay tolerant* untuk menyesuaikan dengan kondisi kanal akustik dan yang telah mempertimbangkan kebutuhan *time slot* tambahan untuk mendapatkan manfaat *cooperative diversity*.
b. Mengevaluasi kinerja protokol MAC *cooperative diversity* di kanal akustik.

1.4 Batasan Masalah

Batasan masalah pada penilitian ini adalah sebagai berikut :

a. Kanal akustik bawah air yang disimulasikan adalah untuk perairan dangkal.
b. Skema *relay* yang digunakan untuk *cooperative diversity* adalah skema DF (*Decode and Forward*).

c. Protokol MAC yang akan didesain menggunakan *virtual carrier sensing* menggunakan paket kontrol RTS/CTS.

1.5 Relevansi

Penelitian ini diharapkan dapat memberikan kontribusi berupa protokol MAC untuk *underwater acoustic network* dengan *cooperative diversity* untuk mengatasi kelemahan-kelemahan kanal akustik.

1.6 Sistematika Penulisan

Sistematika penulisan yang digunakan dalam Tesis ini adalah sebagai berikut:

BAB 1 : PENDAHULUAN

Bab ini mengandung hal-hal berikut diantaranya latar belakang, perumusan masalah, tujuan penelitian, batasan masalah, relevansi dan sistematika penulisan.

BAB 2 : TINJAUAN PUSTAKA

Bab ini berisi tinjauan pustaka yang akan menunjang pembuatan simulasi pada bab selanjutnya. Tinjauan pustaka yang digunakan meliputi: propagasi akustik bawah laut, *cooperative diversity*, protokol MAC dan *asynchronous cooperative transmission*.

BAB 3 : METODE PENELITIAN

Bab ini menjelaskan mengenai metode penelitian. Bagian ini berisi alur penelitan, model topologi, simulasi dan evaluasi.

BAB 4 : HASIL DAN ANALISA

Bab ini menjelaskan tentang hasil simulasi yang diperoleh serta analisa kinerja dari hasil simulasi.

BAB 5 : PENUTUP

Memberi kesimpulan tentang hasil yang telah diperoleh dan saran yang layak dilakukan bila penelitian ini dilanjutkan.

BAB II

TINJAUAN PUSTAKA

2.1 Model Propagasi Akustik Bawah Air

Jaringan akustik bawah air, sesuai dengan namanya, menggunakan gelombang akustik untuk mentransmisikan data [2]. Berikut ini beberapa hal yang berkaitan dengan propagasi akustik bawah air :

2.1.1 Rugi-rugi Transmisi

Unit standar untuk pengukuran propagasi akustik bawah air adalah intesitas (*I*) yang di didefinisikan sebagai aliran tekanan suara (daya) per satuan area. Biasanya dituliskan dalam satuan watt per meter persegi. Intensitas akustik bisa juga dihitung dengan persamaan (2.1) berikut ini :

$$I = \frac{p^2}{\rho c} \tag{2.1}$$

dimana p adalah amplitudo tekanan suara, ρ adalah kerapatan air laut dan c adalah cepat rambat suara di air laut [5]. Pada pengukuran, rugi-rugi transmisi dituliskan sebagai sepuluh kali *log* basis sepuluh dari perbandingan intensitas pada jarak satu meter dari sumber dibandingkan dengan intensitas pada jarak tertentu yang diinginkan dari sumber suara, atau bila dituliskan dalam rumus :

$$Rugi\,Transmisi = 10\log_{10}\left(\frac{I_{ref}}{I}\right) \tag{2.2}$$

Rugi-rugi transmisi pada propagasi gelombang akustik terjadi karena dua hal yaitu *geometric loss* dan absorbsi energi gelombang suara oleh air. Terdapat dua jenis bentuk geometri penyebaran gelombang akustik yang dipengaruhi oleh kedalam laut. Pada laut dangkal, bentuk geometri penyebaran gelombang akustik berbentuk silinder. Sedangkan pada laut dalam, bentuk geometri penyebarannya berbentuk bola. Bila digabungkan antara rugi-rugi geometri dan absorbsi, maka rugi-rugi transmisi total bisa dituliskan dalam persamaan (2.3) berikut ini:

$$A(d,f) = k \cdot 10\log(d) + d \cdot \alpha(f) \quad (2.3)$$

dimana *A* adalah *transmission loss* dengan satuan dB, *k* adalah koefisien spreading yang bernilai *k*=1 untuk laut dangkal dan *k*=2 untuk laut dalam, *d* adalah jarak antara pengirim dan penerima dalam satuan *yards*, $\alpha(f)$ adalah koefisien absorbsi dalam satuan dB/*kiloyards* [13] [14] [15].

Terdapat beberapa model yang merumuskan tentang besaran koefisien absorbsi energi suara di air, diantaranya adalah model *Thorp, Fisher & Simmon* dan model *Ainslie -McColm* [15]. Koefisien absorbsi yang sederhana dicontohkan oleh persamaan Thorp seperti dituliskan pada persamaan (2.4) berikut ini.

$$\alpha(f) = 0{,}1\frac{f^2}{1+f^2} + 40\frac{f^2}{4100+f} + 2{,}75.10^{-4} f^2 + 0{,}003 \quad (2.4)$$

dimana $\alpha(f)$ adalah koefisien absorbsi dalam satuan dB/*kiloyards* dan f dalam satuan kHz [14] [15] [16]. Persamaan Thorp diatas berlaku untuk $f \geq 0.4$ kHz.

2.1.2 Noise

Selain redaman, gelombang akustik juga terganggu oleh adanya *noise*. Terdapat dua jenis utama *noise*, yaitu *ambient noise* dan *site-specific noise. Ambient noise* adalah *noise* yang selalu ada meski di lautan dalam yang tenang. Berbeda dengan *site-specific noise* yang yang hanya ada di tempat tertentu saja seperti *noise* yang dihasilkan oleh kegiatan mamalia laut, penambangan lepas pantai, gempa atau runtuhan es di kutub bumi. Untuk *ambient noise* pada bawah air terdiri atas empat *noise* yang berbeda, yaitu *turbulence, shipping, wind* dan *thermal noise* [6]. Kerapatan spektrum daya dari keempat komponen *noise* tersebut bisa dihitung berurutan dengan persamaan (2.5), (2.6), (2.7) dan (2.8) [13] [14] [15].

$$10\log N_t(f) = 17 - 30\log(f) \quad (2.5)$$

$$10\log N_s(f) = 40 + 20(s-0{,}5) + 26\log(f) - 60\log(f+0{,}03) \quad (2.6)$$

$$10\log N_w(f) = 50 + 7{,}5w^{1/2} + 20\log(f) - 40\log(f+0{,}4) \quad (2.7)$$

$$10\log N_{th}(f) = -15 + 20\log(f) \quad (2.8)$$

dimana N_t adalah *noise* yang diakibatkan oleh turbulensi, N_s adalah *noise* karena adanya aktifitas kapal, s adalah faktor kapal yang bernilai antara 0 sampai dengan 1, N_w adalah *noise* karena angin, w adalah kecepatan angin dalam satuan meter/detik, dan N_{th} adalah *thermal noise*. Dari persamaan (2.5), (2.6), (2.7), (2.8), diatas bisa dihitung *noise* keseluruhan N_f dengan menggunakan persamaan (2.9) [13] [14] [15].

$$N(f) = N_t(f) + N_s(f) + N_w(f) + N_{th}(f) \tag{2.9}$$

2.1.3 Kecepatan Suara

Dikutip dari Etter (2013), *sound speed* atau kecepatan suara di bawah laut merupakan elemen dasar yang mempengaruhi tingkah laku propagasi suara di laut. Ada banyak rumus empiris yang dibuat untuk menghitung kecepatan suara menggunakan nilai suhu, salinitas dan tekanan (kedalaman) sebagai parameternya. Beberapa diantaranya yang sering dipakai adalah Wilson (1960), Leroy (1969), Frye & Pugh (1971), Del Grosso (1974), Medwin (1975), Chen & Millero (1977), Lovett (1978), Coppens (1981), Mackenzie (1981) dan Leroy dkk (2008, 2009).

Tiap-tiap formula di atas punya *range* masing-masing untuk suhu, salinitas dan tekanan. Penghitungan kecepatan suara dengan nilai di luar *range* akan menghasilkan penghitungan kecepatan yang salah. Beberapa ringkasan *range* parameter untuk menghitung kecepatan suara bisa dibaca di tabel 2.1 [5].

Sebagai contoh, untuk formula yang dibuat oleh Mackenzie (1981) ditunjukkan pada persamaan (2.10) berikut ini :

$$\begin{aligned} c = {} & 1448.96 + 4.591T - 5.304\times10^{-2}T^2 + 2.374\times10^{-4}T^3 \\ & + 1{,}340(S-35) + 1{,}630\times10^{-2}D + 1{,}675\times10^{-7}D^2 \\ & - 1{,}205\times10^{-2}T(S-35) - 7{,}139\times10^{-13}TD^3 \end{aligned} \tag{2.10}$$

dimana c adalah kecepatan suara (meter/detik), T adalah suhu air (-2℃ sampai dengan 35℃), S adalah salinitas (25‰ sampai 40‰) dan D adalah kedalaman laut (0 sampai 8000m) [5].

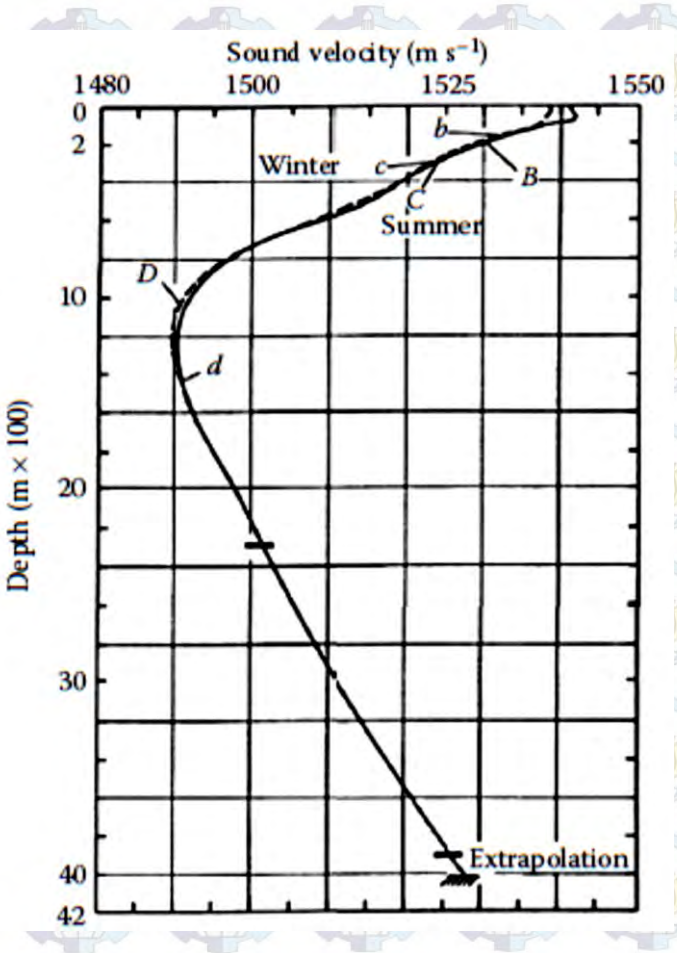
Tabel 2.1 Ringkasan *Range Parameter* untuk Penghitungan *Sound Speed.*

Referensi	Temperatur (℃)	Salinitas (ppt)	Tekanan/ Kedalaman	*Standard Error* (m/s)	Jumlah *Terms*
Wilson (1960)	-4 ~ 30	0 ~ 37	1 ~ 1000 kc/cm^2	0.3	23
Leroy (1969)	-2 ~ 34	20 ~ 42	0 ~ 8000 m	0.2	13
Frye & Pugh (1971)	-3 ~ 30	33.1 ~ 36.6	1.033 ~ 984.3 kg/cm^2	0.1	12
Del Groso (1974)	0 ~ 35	29 ~ 43	0 ~ 1000 kg/cm^2	0.05	19
Medwin (1975)	0 ~ 35	0 ~ 45	0 ~ 1000 kg/cm^2	~0.2	6
Chen & Millero (1977)	0 ~ 40	5 ~ 40	0 ~ 1000 bar	0.19	15
Lovett (1978)	0 ~ 30	30 ~ 37	0 ~ 10000 dbars	0.063	13
Coppens (1981)	-2 ~ 35	0 ~ 42	0 ~ 4000 m	0.1	8
Mackenzie (1981)	-2 ~ 30	25 ~ 40	0 ~ 8000 m	0.07	9
Leroy dkk (2008)	-1 ~ 30	0 ~ 42	0 ~ 12000 m	0.2	14

Sumber : Etter, 2013

2.1.4 Ray Bending

Dari sub bab di atas dijelaskan bahwa *sound speed / sound velocity* nilainya tergantung dari tiga parameter lingkungan, yaitu temperatur / suhu, salinitas dan tekanan atau kedalaman air. Perambatan gelombang akustik yang melewati medium air akan mengalami penurunan kecepatan atau sebaliknya tergantung dari tiga hal di atas [5] [17]. Untuk memudahkan mengamati perubahan kecepatan perambatan gelombang akustik bawah air pada tiap kedalam, biasanya disajikan dalam *sound speed profile* seperti yang ditunjukkan pada Gambar 2.1.

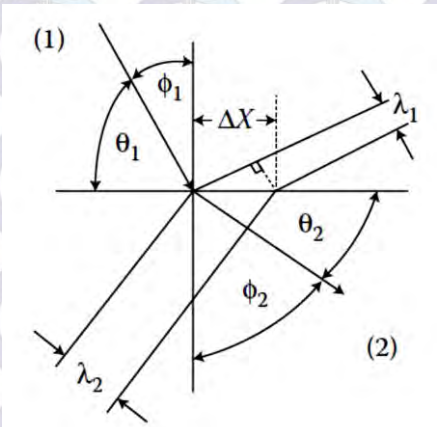


Gambar 2.1 *Sound Speed Profiles* pada saat Musim Dingin dan Musim Panas. [5]

Perbedaan cepat rambat gelombang akustik pada tiap kedalaman laut menimbulkan fenomena pembelokan arah perambatan gelombang akustik yang biasa disebut dengan fenomena *ray bending*. Fenomena *ray bending* ini mengikuti hukum pembiasan Snellius seperti yang dituliskan dalam persamaan (2.11) [5].

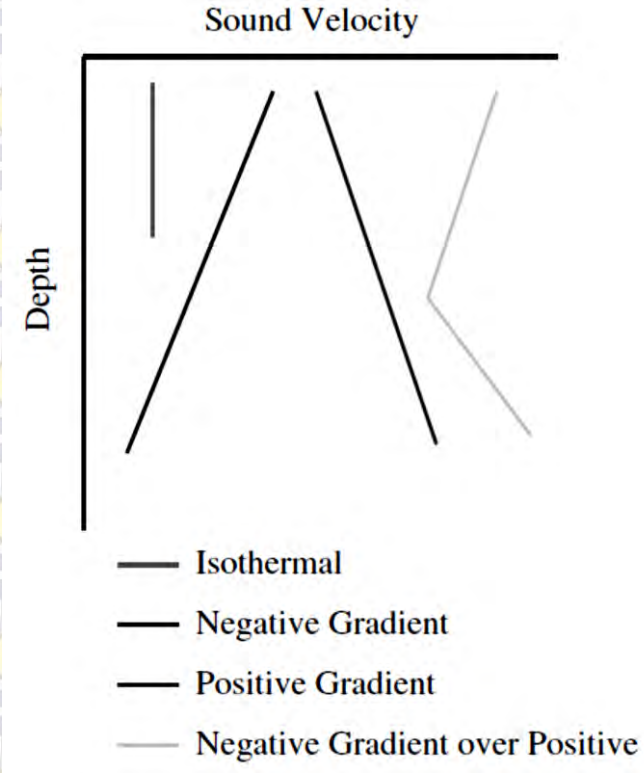
$$\frac{\sin\phi_1}{c_1} = \frac{\sin\phi_2}{c_2} \tag{2.11}$$

Dimana c_1 dan c_2 adalah kecepatan perambatan gelombang akustik masing-masing berurutan pada level kedalaman 1 dan level kedalaman 2, ϕ_1 adalah sudut yang terbentuk antara arah datangnya *ray* dengan garis normal dan ϕ_2 adalah sudut yang terbentuk antara *ray* yang dibiaskan dengan garis normal. Sudut-sudut yang terbentuk ini diilustrasikan lebih jelas dalam Gambar 2.2 berikut ini.



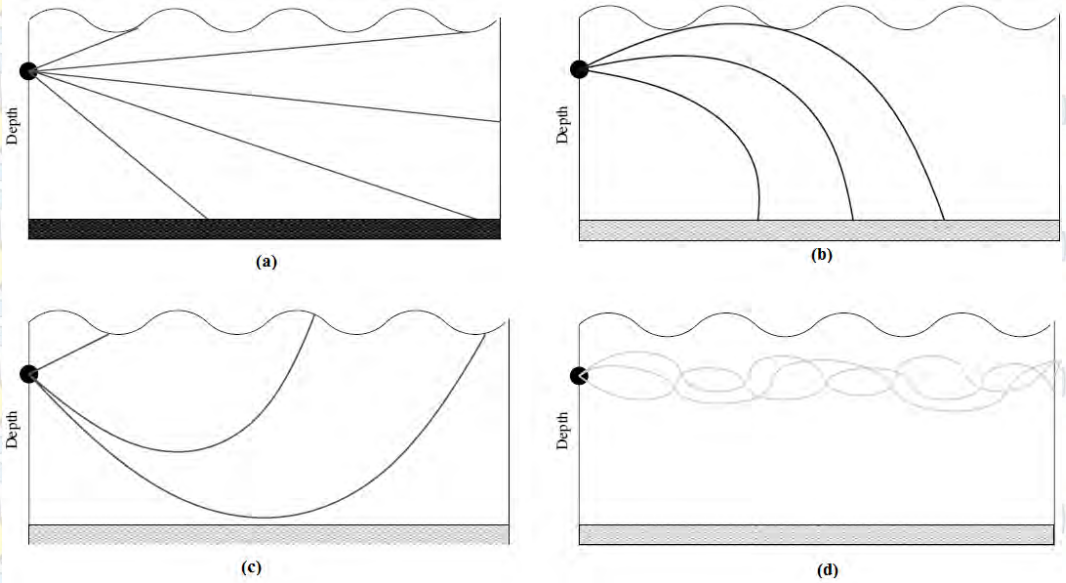
Gambar 2.2 Geometri untuk Hukum Snellius. [5]

Dengan adanya fenomena *ray bending* diatas, dikenal ada empat pola utama perambatan gelombang akustik di bawah air tergantung dari profil kecepatan perambatan suara di bawah laut, yaitu : *isothermal gradient*, *negative gradient*, *positive gradient* dan *negative over positive gradient* [17]. Masing-masing profil kecepatan perambatan gelombang suara di bawah air diilustrasikan pada Gambar 2.3 berikut ini.



Gambar 2.3 Profil Kecepatan Suara *vs* Kedalaman. [17]

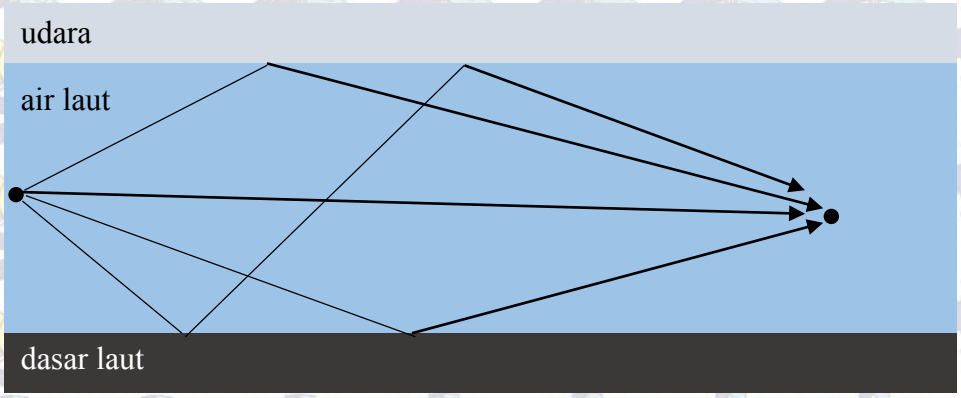
Dari masing-masing profil kecepatan perambatan suara pada Gambar 2.3, maka arah pembelokan perambatan gelombang akustik pada masing-masing profil diilustrasikan pada Gambar 2.4 berikut ini.



Gambar 2.4 (a) *isothermal gradient*, (b) *negative gradient*, (c) *positive gradient* dan (d) *negative gradient over positive.* [17]

2.1.5 Multipath

Dikutip dari (Lurton, 2002), propagasi gelombang akustik di bawah laut dibatasi oleh permukaan air laut dan dasar laut. Sehingga sinyal mengalami pemantulan oleh bidang batas (permukaan dan dasar laut) dalam proses perambatannya. Telah diketahui dari bab sebelumnya bahwa variasi kecepatan suara di medium bisa membentuk lintasan tertentu sesuai dengan *sound speed profile*. Hal ini bisa menyebabkan sinyal yang ditransmisikan akan merambat dari sumber ke tujuan melalui beberapa lintasan yang berbeda-beda. Sinyal utama yang merupakan komponen *direct path* akan diterima di tujuan dengan diikuti oleh gema (*reflection path*) yang amplitudonya akan semakin turun dengan banyaknya pantulan yang dilalui [18]. Fenomena kanal *multipath* yang terbentuk pada bawah air digambarkan seperti pada Gambar 2.5.

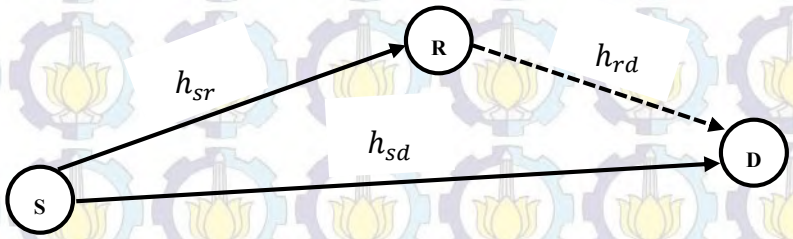


Gambar 2.5 *Multipath* di Perairan Dangkal.

Dengan menggunakan teknik *ray tracing*, gelombang akustik yang dipancarkan bisa dianggap sebagai berkas akustik (*ray acoustic*). Di sisi penerima, bisa dianggap sebagai superposisi dari beberapa *ray* yang diterima. Bila tiap *eigenray* melewati lintasan yang berbeda jaraknya dan atau *sound velocity* yang berbeda maka tiap *eigenray* tersebut akan diterima di penerima dengan amplitudo dan *delay* kedatangan yang berbeda-beda pula [13].

2.2 Cooperative Diversity

Cooperative diversity seperti diilustrasikan pada Gambar 2.6 adalah sistem komunikasi dimana penerima *D* menerima transmisi sinyal langsung dari pemancar *S* sekaligus menerima transmisi sinyal yang sama dari *node* lain yang bertugas sebagai *relay R*. Kedua sinyal tersebut digabung di sisi penerima dengan teknik MRC (*Maximum Ratio Combining*) untuk meningkatkan nilai SNR di sisi penerima [12].



Gambar 2.6 Ilustrasi Sistem Komunikasi Kooperatif Satu Relay.

Dalam sistem kooperatif satu *relay*, terdapat dua fase transmisi. Fase pertama, *S* mentransmisikan sinyal ke *D* yang sekaligus juga diterima oleh *R* karena sifat transmisi nirkabel yang *broadcast*. Fase kedua, sinyal yang telah diterima oleh *R* pada fase pertama akan diretransmisikan lagi ke *D*. Karena adanya dua fase inilah yang menyebabkan adanya kebutuhan *time slot* tambahan bila dibandingkan dengan sistem komunikasi non kooperatif.

Terdapat dua skema *relay* yang paling utama, yaitu AF (*Amplify and Forward*) dan DF (*Decode and Forward*). Secara garis besar, relay AF bekerja dengan cara menguatkan sinyal yang diterima kemudian langsung ditransmisikan kembali ke tujuan. Beberapa literatur menyebutnya sebagai *non regenerative relaying schemes*. Sedangkan untuk relay DF bekerja dengan cara melakukan demodulasi dan *decode* terlebih dahulu kemudian dilakukan *encode* dan dimodulasi lagi baru kemudian ditransmisikan lagi ke tujuan. Skema ini biasa juga disebut dengan *regenerative relaying schemes*.

2.2.1 Decode and Forward

Mengutip (Wang, 2011), pada fase pertama skema *relay* DF, S akan mentransmisikan sinyal ke *D* yang juga diterima oleh *R*. Bila kanal diasumsikan *flat fading* dan *time invariant*, maka persamaan sinyal yang diterima oleh R dan D bisa dituliskan masing-masing berurutan pada persamaan (2.12) dan persamaan (2.13) [19].

$$y_r = h_{sr}\sqrt{P_s}\,x_s + w_r \tag{2.12}$$

$$y_d^{(1)} = h_{sd}\sqrt{P_s}\,x_s + w_d^{(1)} \tag{2.13}$$

dimana y_r adalah sinyal yang diterima oleh *R*, $y_d^{(1)}$ adalah sinyal yang diterima oleh *D* pada fase pertama, h_{sr} adalah koefisien *fading S-R*, h_{sd} adalah koefisien *fading S-D*, P_s adalah daya pancar sinyal, x_s adalah sinyal yang ditransmisikan dengan $E[|x_c|^2] = 1$, w_r adalah *noise* di *R* dan yang terakhir $w_d^{(1)}$ adalah *noise* di *D* pada fase pertama.

Pada skema DF, sinyal yang diterima oleh *R* (y_r) pada persamaan (2.12) terlebih dahulu didemodulasi dan di-*decode* untuk mendapatkan kembali sinyal

informasi (x_s). Untuk mencegah R me-*relay* data yang salah, disyaratkan SNR pada kanal *S-R* (γ_{sr}) harus lebih besar dari SNR ambang (γ_{th}) yang ditentukan berdasarkan kebutuhan sistem (berdasarkan kapasitas / *probability of error* yang diinginkan) [19].

Pada fase kedua, sinyal informasi akan dilakukan *encoding* dan modulasi untuk ditransmisikan lagi ke *D*. Sinyal yang diterima oleh D pada fase kedua bisa dituliskan dalam persamaan (2.14) [19].

$$y_d^{(2)} = \begin{cases} h_{rd}\sqrt{P_r}x_s + w_d^{(2)} & , jika \quad \gamma_{sr} \geq \gamma_{th} \\ 0 & , untuk \quad \gamma_{sr} \quad yang\ lain \end{cases} \tag{2.14}$$

Bila kedua sinyal dari fase pertama dan kedua digabungkan dengan MRC (*Maximum Ratio* Combining) maka gabungan sinyal dituliskan dalam persamaan (2.15) berikut ini [19].

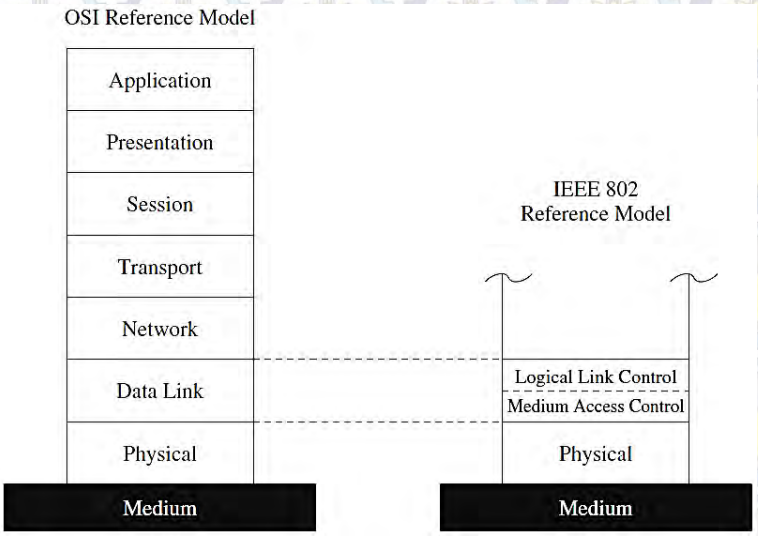
$$\tilde{y}_d = \begin{cases} \dfrac{\sqrt{P_s}h_{sr}^*}{\sigma_r^2}y_d^{(1)} + \dfrac{\sqrt{P_r}h_{rd}^*}{\sigma_d^2}y_d^{(2)} & , jika \quad \gamma_{sr} \geq \gamma_{th} \\ 0 & , untuk \quad \gamma_{sr} \quad yang\ lain \end{cases} \tag{2.15}$$

2.3 Protokol MAC

Sistem komunikasi nirkabel mempunyai sifat alami berupa *broadcast* yang berarti transmisinya menyebar ke segala arah sesuai dengan pola radiasi pemancarnya. Padahal, pada suatu jaringan nirkabel biasanya terdiri atas beberapa piranti / *node* yang menggunakan kanal yang sama dan tidak boleh ada transmisi data pada waktu yang sama di kanal yang sama, sehingga perlu ada mekanisme yang mengatur penggunaan medium (kanal komunikasi). Tugas pengaturan pemakaian kanal ini dilakukan pada *Data Link layer* yang merupakan layer kedua pada referensi OSI 7 *Layers*.

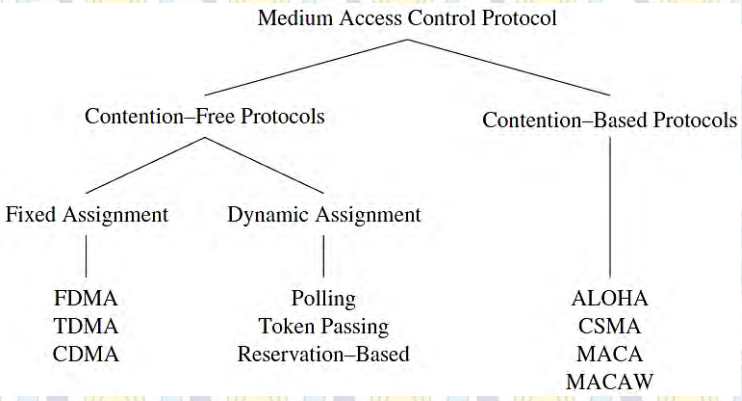
Dikutip dari (Dargie, 2010), menurut referensi standar IEEE 802, *data link layer* dibagi lagi menjadi dua *sub layer*, yaitu *logical link control* (LLC) dan *medium access control* (MAC) seperti diilustrasikan pada Gambar 2.7. Protokol

MAC berada pada *sub layer* yang paling bawah berdekatan dengan *physical layer* sehingga bisa mengendalikan *physical layer* secara langsung.



Gambar 2.7 MAC pada OSI 7 *Layers* dan Referensi IEEE 802. [20]

Protokol MAC bisa dikategorikan menjadi dua jenis seperti diilustrasikan pada Gambar 2.8, yaitu *contention free* dan *contention based*. *Contention free* adalah mekanisme pembagian pemakaian kanal tanpa perlu *node* yang ingin menggunakan kanal berebut dengan *node* yang lain. Protokol yang *contention free*, dalam perkembangannya, dibagi menjadi dua sub kategori yaitu *fixed assignment* dan *dynamic assignment*. Sedangkan untuk protokol yang *contention based*, ketika suatu *node* akan menggunakan kanal, maka *node* tersebut harus bersaing dengan *node* yang lain karena tidak ada penjadwalan secara eksklusif [20].



Gambar 2.8 Klasifikasi Protokol MAC. [20]

2.3.1 Contention Free MAC Protocol

Tabrakan (*colission*) terjadi ketika dua *node* atau lebih melakukan transmisi secara bersamaan menggunakan kanal yang sama. Hal ini bisa dihindari dengan cara mengalokasikan sumber daya kanal secara ekslusif. Sumber daya yang dimaksud bisa berupa frekuensi atau yang biasa disebut dengan FDMA (*Frequency Division Multipe Access*). Pada FDMA, pita frekuensi dibagi-bagi menjadi kanal-kanal dengan lebar pita yang lebih kecil. Setiap *node* akan dialokasikan pada kanal yang berbeda sehingga dalam transmisi simultan pun tidak akan saling berinterferensi atau bertabrakan.

Selain membagi frekuensi, bisa juga dengan membagi waktu pemakaian kanal yang biasa disebut dengan istilah TDMA (*Time Division Multiple Access*). TDMA memungkinkan beberapa *node* untuk saling berkomunikasi menggunakan kanal frekuensi yang sama dengan menggunakan penjadwalan pemakaian kanal. Terdapat periode waktu yang disebut dengan *frames* dan tiap *frame* dibagi-bagi menjadi *time slot*. Pengalokasian *time slot* yang berbeda ke setiap *node* bisa mencegah terjadinya transmisi simultan pada waktu yang bersamaan, sehingga interferensi atau tabrakan transmisi bisa dihindarkan.

Teknik multi akses yang lain adalah CDMA (*Code Division Multiple Access*). CDMA memungkinkan beberapa *node* untuk melakukan transmisi bersamaan dengan *node* yang lain dan tidak saling berinterferensi meskipun menggunakan kanal frekuensi dan waktu yang sama. Hal ini dicapai dengan cara memberikan kode yang berbeda dan saling ortogonal untuk setiap *node*.

Ketiga teknik multi akses di atas termasuk dalam kategori yang *fixed assignment*. Sehingga frekuensi, waktu atau kode yang sudah dialokasikan ke *node* satu tidak lagi bisa digunakan lagi untuk *node* yang lain. Hal ini bisa menurunkan efisiensi kanal ketika tidak semua *node* mempunyai data untuk dikirim. Belum lagi ketika topologi jaringan atau kebutuhan trafik berubah, pengalokasian frekuensi atau penjadwalan *time slot* membutuhkan usaha yang tidak mudah.

Untuk mengatasi masalah pada *fixed assignment*, ditawarkan teknik yang lebih flexibel dalam mengalokasikan sumber daya kanal, yaitu *dynamic assignment*. Intinya adalah dengan melakukan pengalokasian slot berdasarkan permintaan kanal oleh *node* di jaringan. Salah satu contoh dari *dynamic assignment* adalah teknik

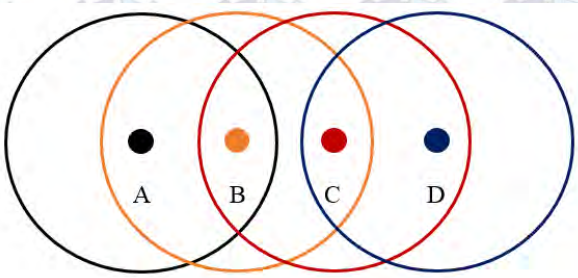
polling. Teknik *polling* memanfaatkan *base station* (pada jaringan yang mempunyai infrastruktur) yang mengirimkan *polling* dalam bentuk paket *frame* kecil ke *node* di jaringan, menanyakan setiap *node* apakah punya data untuk dikirim. Jika suatu *node* tidak mempunyai data untuk dikirim, maka *base station* akan mengirim paket *polling* lagi ke *node* yang lain dan seterusnya. Variasi yang lain adalah teknik *token passing*, dan *reservation-based*. *Dynamic assignment* bisa mengurangi kekurangan *fixed assignme*nt, namun protokol menjadi sangat komplek untuk memastikan tidak ada *node* yang konflik untuk mencegah terjadinya tabrakan [17] [20].

2.3.2 Contention Based MAC Protocol

Pada protokol MAC yang *contention free*, *node* tidak akan melakukan transmisi di luar alokasi yang diberikan ke *node* tersebut. Berbeda dengan protokol yang *contention based*, tidak ada pengalokasian kanal secara khusus. *Node* harus bersaing dan berebut kanal dengan *node* lain yang akan menggunakan kanal yang sama. Contoh protokol *contention based* yang paling primitif adalah protokol ALOHA yang mengijinkan *node* untuk menggunakan kanal kapan pun *node* punya data untuk dikirim. Karena tidak ada mekanisme untuk menghindari tabrakan, maka kinerja protokol ALOHA jelek apalagi untuk transmisi data kecepatan tinggi. ALOHA hanya mengatur mekanisme ketika transmisi gagal.

Protokol lain yang terkenal adalah CSMA (*Carrier Sense Multiple Access*) yang awalnya dipakai pada jaringan kabel bertopologi *bus*. Cara kerja protokol CSMA adalah dengan cara *node* yang akan menggunakan kanal harus terlebih dahulu melakukan *carrier sensing* untuk memastikan kanal tidak sedang dipakai. Bila tidak terdeteksi ada transmisi, maka *node* bisa mentransmisikan datanya. Namun bila pada saat proses *carrier sensing* terdeteksi ada transmisi, maka *node* akan melakukan proses backoff dengan durasi acak kemudian kembali ke fase carrier sensing lagi. Protokol CSMA kemudian disempurnakan dengan teknik *Colission Avoidance* yang kemudian disebut dengan CSMA/CA. Prinsip kerjanya adalah ketika proses *carrier sensing* dan *node* mendapati bahwa kanal tidak sedang dipakai, *node* tidak langsung mentransmisikan datanya namun menunggu selama periode waktu tertentu untuk mencegah terjadinya tabrakan dengan *node*.

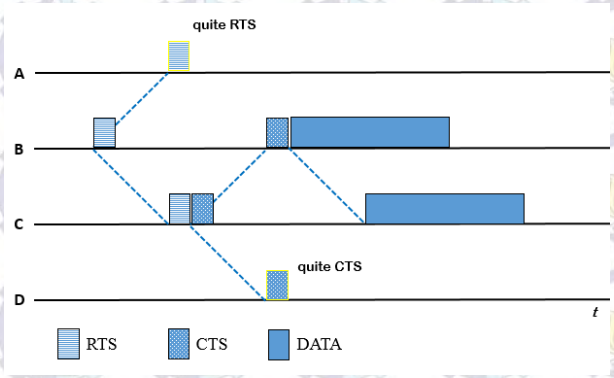
CSMA/CA telah memperbaiki kinerja ALOHA yang tidak punya mekanisme pencegahan tabrakan. Namun dengan menerapkan CSMA/CA, terdapat dua masalah yang muncul, yaitu *hidden-terminal problem* dan *exposed-terminal problem* yang diilustrasikan pada Gambar 2.9. Permasalahan *hidden terminal* muncul ketika A sedang mentransmisikan data ke B dan pada saat yang bersamaan C juga ingin mengirim data ke B. Area jangkauan transmisi A tidak sampai ke C sehingga saat C melakukan *carrier sensing*, C mendeteksi bahwa kanal kosong dan bisa melakukan transmisi. Dengan demikian, saat C melakukan transmisi data akan terjadi tabrakan di B. Sedangkan untuk permasalahan *exposed terminal* terjadi ketika C melakukan transmisi ke D dan pada saat yang bersamaan B ingin mengirim transmisi ke A. B melakukan *carrier sensing* dan mendeteksi kanal sedang dipakai oleh C sehingga B menunda melakukan transmisi. Penundaan transmisi oleh B menurunkan efektifitas penggunakan kanal karena dengan B melakukan transmisi pun tidak akan terjadi tabrakan di A ataupun di D [17] [20].



Gambar 2.9 *Hidden Terminal Problem* dan *Exposed Terminal Problem.*

Selain menggunakan teknik *carrier sensing*, untuk mencegah terjadi tabrakan transmisi bisa juga dengan memanfaatkan sinyal kontrol RTS dan CTS untuk mereservasi kanal sekaligus untuk proses *handshaking*. Protokol ini dikenal dengan sebutan MACA (*Multiple Access with Colission Avoidance*). Seperti diilustrasikan pada Gambar 2.10 dibawah ini, ketika sebuah *node* ingin menggunakan kanal, *node* harus mengirimkan sinyal kontrol RTS (*Request To Send*) ke *node* tujuan. Bila sinyal RTS bisa diterima oleh *node* tujuan dan siap untuk menerima paket, *node* tujuan akan membalas dengan paket CTS (*Clear To Send*). *Node* baru bisa mengirimkan datanya bila berhasil menerima paket CTS dari *node*

tujuan. Bila *node* gagal menerima CTS, maka *node* akan mengulangi proses dari awal kembali setelah jeda waktu tertentu.

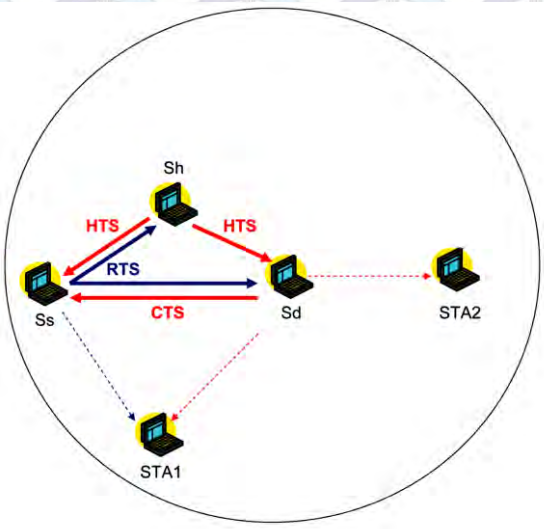


Gambar 2.10 Proses Reservasi Kanal dan *Handshaking*.

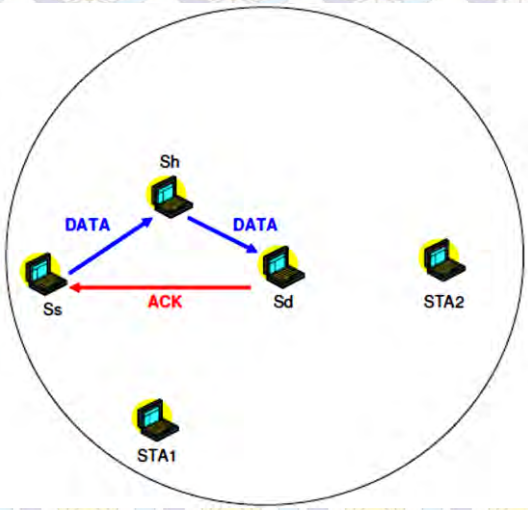
2.4 CoopMAC

CoopMAC adalah salah satu protokol MAC untuk jaringan kooperatif yang termasuk dalam kategori protokol yang *contention based*. CoopMAC cukup populer untuk untuk kanal nirkabel RF karena bisa *backward compatible* dengan standar yang dipakai pada 802.11. Sehingga bila syarat terjadinya jaringan kooperatif tidak terpenuhi, node masih bisa berkomunikasi langsung tanpa ada bantuan dari relay. Pada CoopMAC terdapat paket kontrol RTS (*Request to Send*) dan CTS (*Clear to Send*) layaknya standar yang dipakai 802.11, namun pada CoopMAC menggunakan paket kontrol tambahan yaitu HTS (*Helper ready to Send*) yang tidak ada pada standar 802.11 [21].

Algoritma dari protokol CoopMAC bisa dijelaskan pada Gambar 2.11(a). Saat *node* sumber (S_s) mempunyai antrian data, S_s akan menunggu selama jeda waktu acak tertentu untuk memastikan kondisi kanal sedang dipakai atau tidak. Setelah jeda waktu yang disebut dengan DIFS tersebut habis dan kanal masih kosong, (S_s) akan mengirim sinyal RTS ke *node* tujuan (S_d) sekaligus ke *relay* / *helper node* (S_h). S_h yang berpotensi menjadi relay akan menjawab dengan HTS ke S_s sekaligus juga didengar oleh S_d. Setelah S_d mendengar HTS, S_d akan membalas dengan mengirimkan CTS [21]. Baru kemudian proses pengiriman data dilakukan seperti dijelaskan pada Gambar 2.11(b).



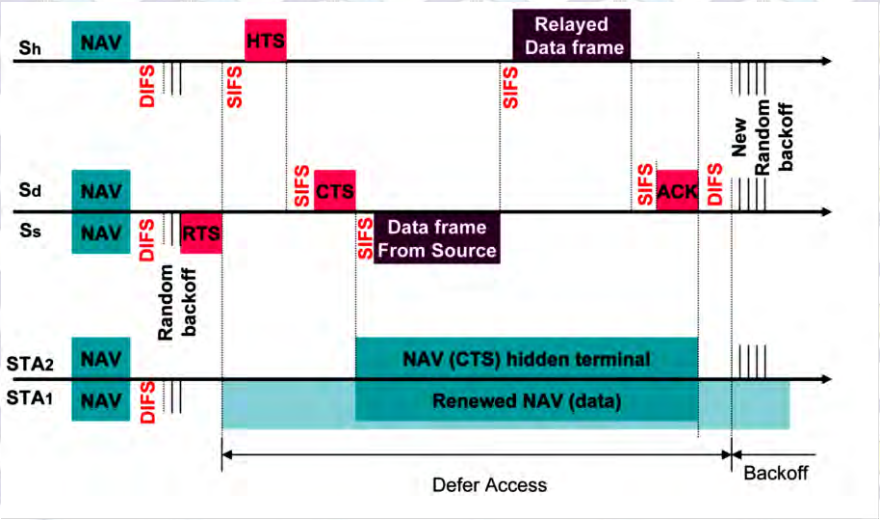
(a)



(b)

Gambar 2.11 (a) Pertukaran Paket Kontrol antar Node pada CoopMAC. dan (b) Proses Pengiriman Paket Data Pada CoopMAC. [21]

Setelah S_s menerima paket CTS, S_s mentransmisikan datanya ke S_d sekaligus juga diterima oleh S_h. Lalu, S_h akan mentransmisikan kembali sinyal yang diterimanya dari S_s setelah jeda waktu yang disebut dengan SIFS (*Short Interframe Space*) seperti diilustrasikan pada Gambar 2.12. Setelah paket data dari S_h diterima sepenuhnya, S_d akan mengirimkan paket ACK (*Acknowledge*) untuk memberitahu keberhasilan transmisi data dari S_s.

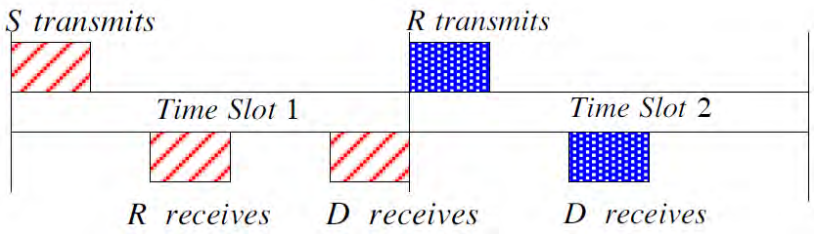


Gambar 2.12 Diagram CoopMAC. [21]

Pada kanal RF, durasi paket data umumnya lebih panjang dibandingkan dengan *delay* propagasi. Oleh karena itu S_h baru akan me-*relay*-kan datanya setelah memastikan transmisi pertama yang langsung dari S_s sudah selesai diterima oleh S_d untuk mencegah terjadinya tabrakan data. Dengan kata lain, transmisi fase pertama dan fase kedua mempunyai slot waktu yang berbeda. Pada kanal akustik bawah air yan mempunyai *delay* propagasi yang tinggi, bila S_h harus mengunggu S_d untuk menerima semua paket data dengan lengkap, akan menurunkan efisiensi kanal. Sehingga perlu desin protokol MAC yang lebih cocok untuk kanal bawah air.

2.5 Asynchronous Cooperative Transmission

Seperti dijelaskan di atas, bahwa untuk sistem komunikasi kooperatif terdapat dua fase yaitu *S-D* dan *R-D* sehingga dibutuhkan dua *time slot*. *Time slot* pertama digunakan untuk transmisi S-D sekaligus S-R dan *time slot* kedua digunakan untuk transmisi R-D. Proses ini diilustrasikan pada Gambar 2.13.

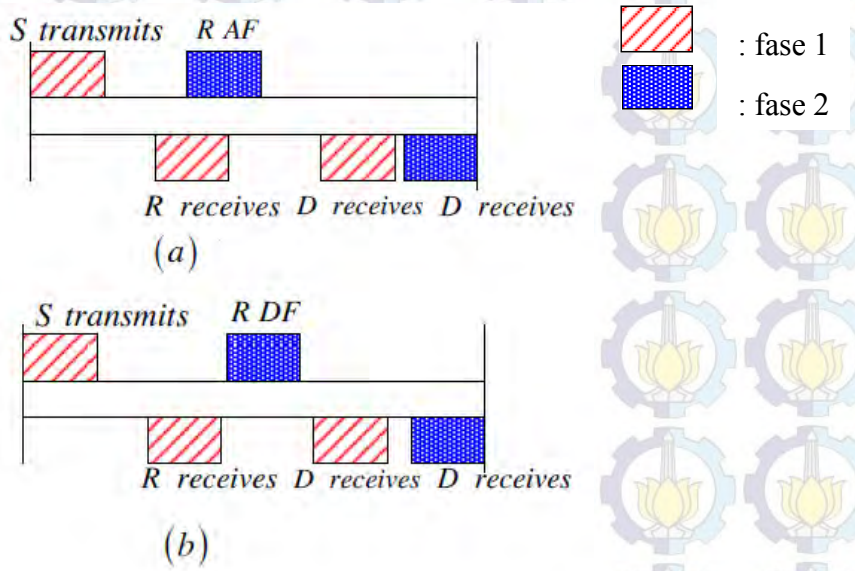


Gambar 2.13 Pembagian *Time Slot* untuk Kedua Fase. [19]

Time slot untuk *S-D* dan *R-D* perlu dialokasikan tersendiri agar tidak terjadi tabrakan paket di D. Hal ini menjadi kurang efektif ketika diimplementasikan dalam jaringan kooperatif bawah laut yang memanfaatkan gelombang akustik. Penurunan efektifitas kanal ini disebabkan oleh cepat rambat gelombang akustik pada media air laut yang berkisar pada 1500m/s menjadikan delay propagasi gelombang akustik sangat tinggi untuk satu kali transmisi.

Untuk mengatasi hal di atas, *Wave Cooperation* (WC) diusulkan oleh (Han, 2008) pada [11]. Prinsip kerja WC adalah dengan cara sinyal yang diterima oleh *relay* langsung dikuatkan dan diretransmisikan ke *D* tanpa perlu menunggu *time slot* selanjutnya. Hal ini mungkin dilakukan tanpa menyebabkan tabrakan transmisi di *D* karena *delay* propagasi jauh lebih tinggi dibandingkan dengan durasi data untuk komunikasi bawah laut.

Ide dari WC di atas lantas dikembangkan oleh (Wang, 2011) pada [19] dengan tidak terbatas pada *amplify and forward*. (Wang, 2011) mengistilahkan teknik yang diusulkan dengan istilah *Asynchronous Cooperation* yang prinsip pemakaian kanalnya sama dengan WC yaitu transmisi R-D bisa langsung dilakukan sesaat setelah pengolahan sinyal di *R* selesai dan siap diretransmisikan tanpa menunggu *time slot* berikutnya seperti diilustrasikan pada Gambar 2.14.



Gambar 2.14 *Asynchonous Transmission* (a) Underwater AF (b) Underware DF (Wang, 2011). [19]

BAB III

METODOLOGI PENELITIAN

Pada penelitian ini akan dirancang sebuah protokol MAC untuk jaringan kooperatif bawah air untuk keperluan jaringan sensor nirkabel bawah air. Protokol MAC yang didesain akan disimulasikan menggunakan *discrete event simulator* NS3. NS3 adalah salah satu simulator yang gratis digunakan dengan lisensi GNU GPLv2. Simulator ini dipilih karena dokumentasinya yang mudah dibaca dan diakses melalui internet dengan menggunakan bahasa pemrograman C++ secara keseluruhan. Di samping itu, sudah tersedia UAN (*Underwater Acoustic Network*) *framework* pada NS3 yang bisa mengurangi beban kerja pemrograman karena sebagian modul sudah dibuat dan siap digunakan [22]. Untuk hal yang berkaitan dengan NS3 dan cara instalasi akan dijelaskan di Lampiran A.

3.1 Algoritma CoopMAC-U Pada Source Node

CoopMAC-U (*Cooperative MAC for Underwater*) ini diadaptasi dari protokol CoopMAC yang ada di kanal nirkabel udara untuk digunakan dalam jaringan kooperatif bawah air. CoopMAC-U menggunakan *virtual carrier sensing* dengan memanfaatkan paket kontrol RTS (*Request to Send*), CTS (*Clear to Send*) dan HTS (*Helper ready to Send*) untuk reservasi kanal. Pada saat *source node* (S) punya data pada antrian di *buffer*, *node* akan mengubah *state* nya dari IDLE menjadi CONTEND dan mengambil nilai acak dan melakukan hitung mundur. Nilai acak yang disebut dengan *binary exponential backoff* (BEB) ini selain digunakan untuk pada periode *contention*, juga digunakan ketika terjadi kegagalan transmisi. Algoritma BEB dijelaskan pada persamaan (3.1) berikut ini :

$$B_{cnt} = \begin{cases} \min(2 \times B_{cnt}, B_{\max}), \textit{saat tabrakan} \\ B_{\min}, \textit{saat transmisi berhasil} \end{cases} \quad (3.1)$$

dimana B_{min} adalah nilai *backoff counter* terkecil pada protokol ini yang diset pada nilai 1, B_{max} adalah *backoff counter* terbesar yang diset bernilai 64 dan B_{cnt} adalah

backoff counter yang digunakan saat ini. B_{cnt} akan direset kembali ke nilai minimum bila transmisi data berhasil. Saat transmisi gagal, nilai B_{cnt} akan dikalikan dua sampai nilai maksimum tercapai.

Durasi waktu *backoff* atau *contend* bisa dihitung menggunakan rumus pada persamaan (3.2).

$$T_{bk} = Uniform\{0, B_{cnt}\} \times (T_{rts} + \tau_{\max}) \quad (3.2)$$

dimana T_{bk} adalah durasi *backoff*, B_{cnt} adalah *backoff counter*, T_{rts} adalah panjang durasi paket RTS, τ_{max} adalah durasi propagasi terlama dalam jaringan.

Setelah waktu hitung mundur habis dan *node* (S) tidak menerima paket kontrol dari manapun selama periode *contend*, maka bisa diasumsikan bahwa kanal sedang kosong. *Node* (S) akan mengecek pada tabel *cooptable* mengenai ketersediaan *node* tetangga yang berpotensi bisa menjadi *relay node* (R). Bila tidak ditemukan *node* (R) maka *node* (S) akan mengirimkan RTS biasa, bukan CoopRTS dan berganti *state* dari CONTEND menjadi WFCTS (*wait for CTS*). *Node* (S) akan menunggu datangnya paket CTS yang durasinya mengikuti rumus pada persamaan (3.3).

$$W_{cts} = T_{cts} + (2 \times \tau_{\max}) \quad (3.3)$$

dimana W_{cts} adalah durasi tunggu paket CTS, T_{cts} adalah durasi paket CTS, dan τ_{max} adalah durasi propagasi terlama dalam jaringan.

Bila setelah durasi tunggu W_{cts} habis dan paket CTS tidak diterima, maka *node* (S) akan menaikkan *backoff counter* dan kembali ke *state* CONTEND. Bila *node* (S) menerima paket CTS, *node* (S) langsung mengirimkan paket DATA dan mengubah *state*-nya menjadi WFACK (*wait for ACK*). Durasi tunggu untuk paket ACK mengikuti rumus pada persamaan (3.4).

$$W_{ack} = T_{ack} + (2 \times \tau_{\max}) \quad (3.4)$$

dimana W_{ack} adalah durasi tunggu paket ACK dan T_{ack} adalah durasi paket ACK. Bila paket ACK diterima sebelum durasi WFACK habis, data sesuai dengan nomer

frame akan dihapus dari daftar antrian data. Bila paket ACK tidak diterima sebelum waktu tunggu habis, maka *node* (S) kembali ke *state* CONTEND untuk melakukan reservasi kanal ulang.

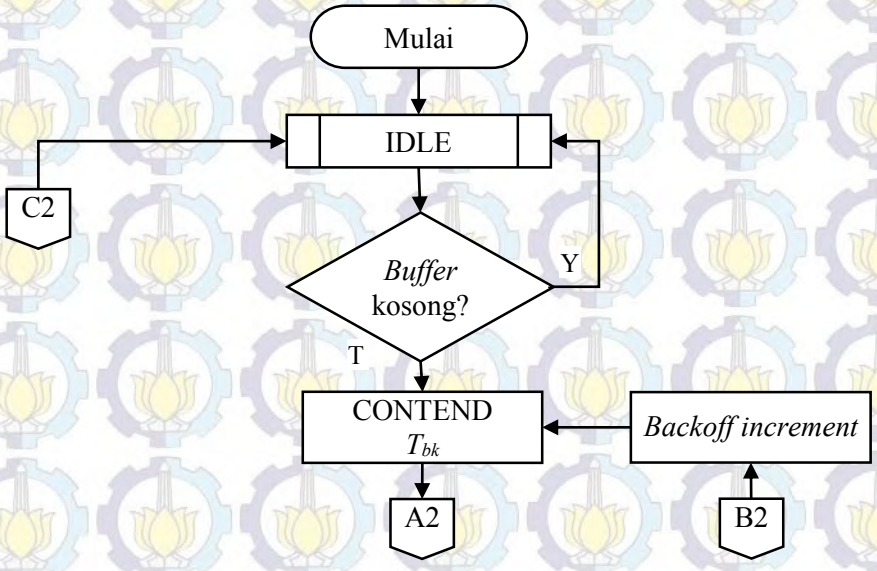
Saat *node* (S) punya data untuk ditransmisikan dan masa CONTEND habis serta punya *node* yang berpotensi menjadi *relay node* (R), maka *node* (S) akan mengirimkan paket CoopRTS dan mengubah *state*-nya menjadi WFHTS dengan durasi tunggu sesuai dengan rumus persamaan (3.5).

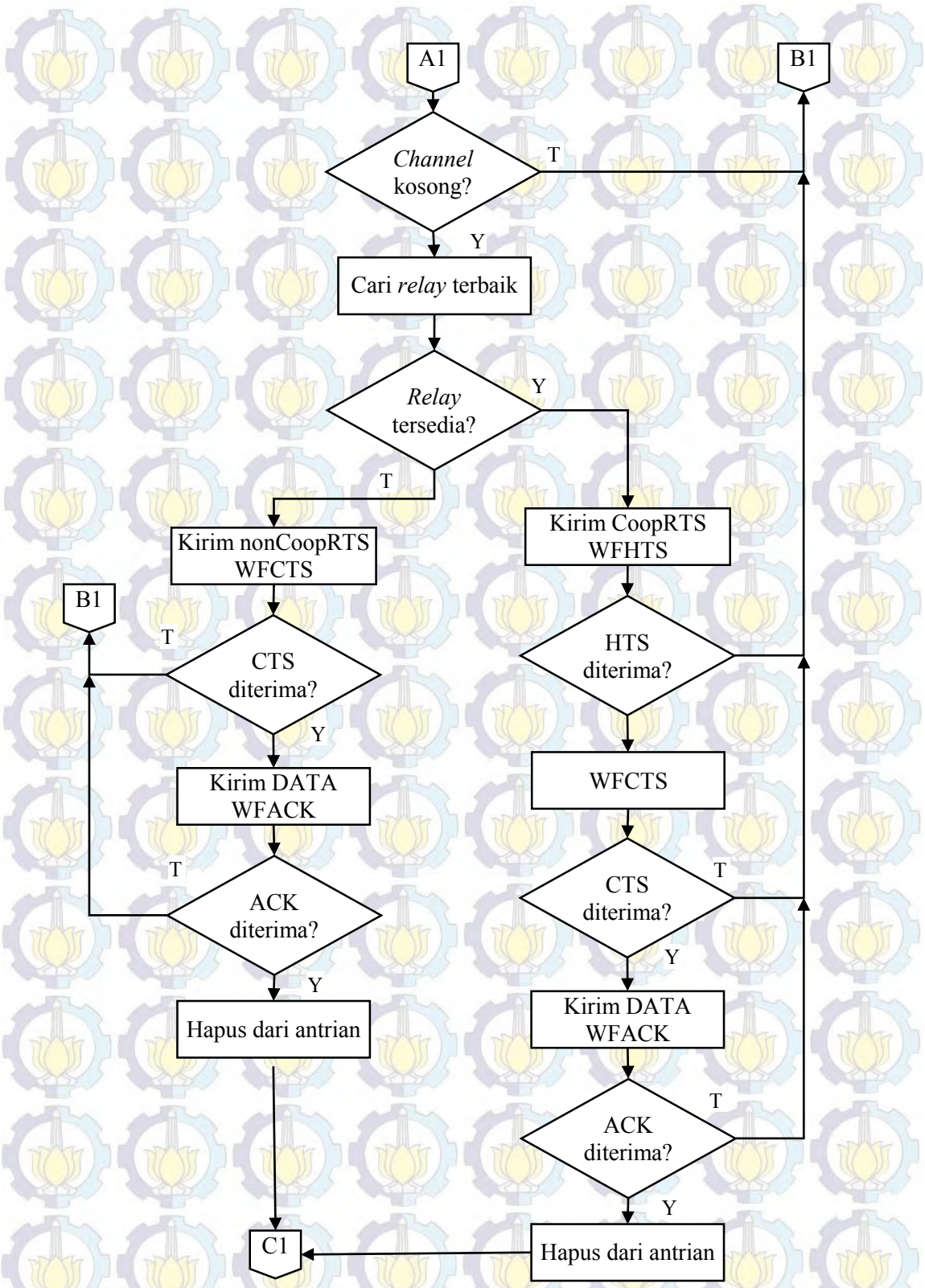
$$W_{hts} = T_{hts} + (2 \times \tau_{\max}) \tag{3.5}$$

dimana W_{hts} adalah durasi tunggu paket HTS dan T_{hts} adalah durasi paket HTS. Bila HTS gagal diterima sebelum waktu habis, maka kembali ke *state* CONTEND untuk reservasi kanal kembali. Sedangkan bila paket HTS diterima, *node* (S) akan mengubah *state*-nya menjadi WFCTS yang durasinya sama dengan pada persamaan (3.3).

Bila paket CTS diterima sebelum durasi WFCTS habis, *node* (S) akan mengirimkan DATA, dan mengganti *state*-nya menjadi WFACK dan menunggu selama durasi sesuai dengan persamaan (3.4). Saat paket ACK diterima, data yang sesuai dengan nomer *frame* yang berhasil dikirim akan dihapus dari antrian.

Algoritma CoopMAC-U pada *source node* (S) di atas sesuai dengan diagram alir seperti pada Gambar 3.1 berikut ini.





Gambar 3.1 Diagram Alir CoopMAC-U di *Source Node*.

3.2 Algoritma CoopMAC-U Pada Helper Node

Helper node atau *relay node* (R) adalah *node*) tidak selalu harus berupa *dedicated relay* yang hanya bertugas untuk me-*relay*-kan data dari *source node* (S) ke *destination node* (D). *node* (R) bisa *node* mana saja yang terletak antara *node* (S) dan *node* (D). *Node* (R) dipilih oleh *node* (S) menggunakan algoritma pemilihan relay yang akan dijelaskan di subbab selanjutnya.

Node (R) bekerja ketika mendapatkan paket RTS dari *node* (S) yang berfungsi untuk reservasi kanal, memberitahu *node* (D) akan ada transmisi data, sekaligus untuk meminta bantuan *node* (R) untuk me-*relay*-kan data. *Node* (R) bila dalam *state* IDLE dan bisa membantu, maka *node* (R) akan mengirimkan paket HTS dan mengganti *state*-nya menjadi WFCTS dengan panjang durasi tunggu mengikuti rumus pada persamaan (3.3).

Bila paket CTS tidak diterima sampai durasi tunggu habis, *state* kembali ke IDLE. Namun bila paket CTS diterima, maka *state* berubah menjadi WFDATA dengan durasi tunggu mengikuti rumus pada persamaan (3.6).

$$W_{data} = T_{data} + (2 \times \tau_{\max}) \tag{3.6}$$

dimana W_{data} adalah durasi tunggu paket data dan T_{data} adalah durasi paket DATA. Bila DATA diterima, *node* (R) akan langsung me-*relay*-kan data tersebut ke *node* (D) dan bila tidak ada DATA yang sampai sampai durasi W_{data} habis, maka *node* (R) akan kembali ke *state* IDLE. Algoritma ini diilustrasikan pada diagram alir pada gambar 3.2.

3.3 Algoritma CoopMAC-U Pada Destination Node

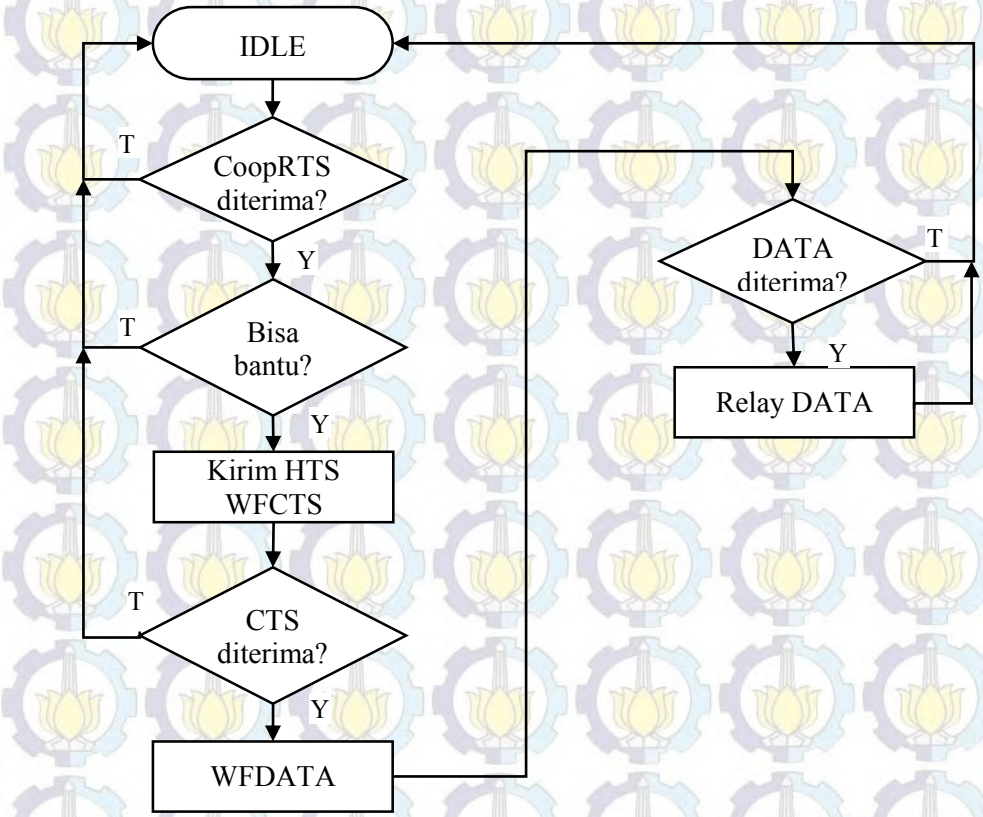
Destination node (D) punya kemampuan fleksibel untuk menangani sistem komunikasi kooperatif sekaligus komunikasi non kooperatif. Saat *node* (D) menerima paket RTS, akan dicek apakah CoopRTS atau nonCoopRTS. Bila yang diterima adalah RTS non kooperatif, *node* (D) akan mengirimkan paket CTS dan mengubah *state*-nya menjadi WFDATA dengan lama durasi mengikuti rumus pada persamaan (3.6). Bila RTS yang diterima adalah RTS kooperatif, *node* (D) akan

mengubah *state*-nya menjadi WFHTS dan menunggu paket HTS dengan durasi tunggu mengikuti rumus pada persamaan (3.7).

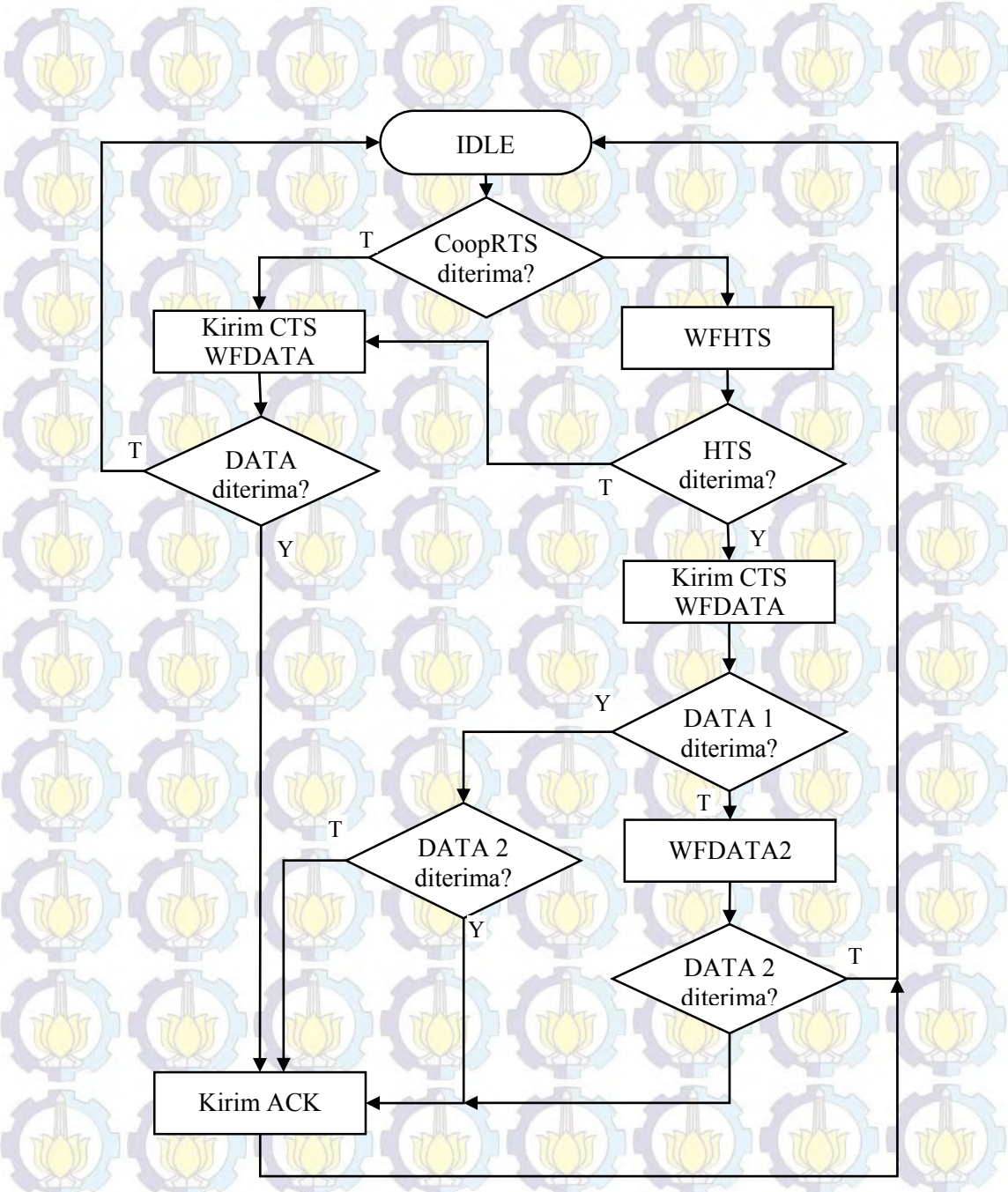
$$W_{hts} = T_{hts} + \tau_{\max} \tag{3.7}$$

dimana W_{hts} adalah durasi tunggu paket HTS, dan T_{hts} adalah durasi paket HTS.

Jika HTS tidak diterima sampai durasi W_{hts} habis, maka *node* (D) akan mengirimkan nonCoop CTS karena tidak ada konfirmasi dari *node* (R) tentang kesanggupan dalam membantu transmisi. Sebaliknya, bila HTS diterima, maka *node* (D) akan mengirimkan CoopCTS dan mengganti *state*-nya ke WFDATA yang durasinya mengikuti rumus pada persamaan (3.6). Selanjunya, *node* (D) akan mengirimkan ACK untuk memberitahu *node* (S) mengenai keberhasilan atau kegagalan DATA yang ditransmisikan dan kembali ke *state* IDLE. Algoritma CoopMAC-U pada *destination node* ini sesuai dengan diagram alir pada Gambar 3.3.



Gambar 3.2. Diagram Alir CoopMAC-U pada *Relay Node*.



Gambar 3.3. Diagram Alir CoopMAC-U di *Destination Node*.

3.4 Algoritma Pemilihan Relay

Seperti disinggung sebelumnya, pada jaringan kooperatif, *relay node* tidak selalu harus berupa *dedicated relay*. Namun, bisa saja *node* tetangga yang berdekatan yang kebetulan posisinya berada diantara *source node* dan *destination*

node. Meskipun demikian, tidak semua *node* yang berada dalam area tersebut bisa dijadikan *relay*, perlu ada proses seleksi untuk menentukan mana *relay* yang paling bagus. Pada protokol CoopMAC-U, proses pemilihan *relay* dilakukan dengan metode *preselected relay* berdasarakan *historical information*.

Yang dimaksud dengan *preselected relay* adalah *relay* sudah ditentukan terlebih dahulu sebelum proses pengiriman data dimulai. Pemilihan dan penunjukan *relay* dilakukan oleh *source node* dengan menggunakan paket RTS yang multifungsi, selain untuk reservasi kanal, memberi tahu *destination node* akan ada transmisi data, juga berfungsi untuk menunjuk *node* tetangga sebagai *relay node.* Sedangkan yang dimaksud dengan *historical information* adalah pemilihannya didasarkan pada informasi yang didapatkan sebelumnya dari proses pertukaran kontrol paket dari tiap-tiap *node* yang berdekatan.

Pada fase awal, setiap *node* mendengarkan pertukaran paket kontrol (RTS/CTS) untuk mengestimasi *link* antar tiap *node.* Diasumsikan kanalnya berupa kanal yang simetris, dengan kata lain, bila dimisalkan *node* R mengirim data dan diterima oleh *node* S dengan kuat sinyal sebesar X dB maka sebesar X dB itu pula kuat sinyal yang diterima oleh *node* R jika *node* S mentransmisikan sinyal ke *node* R. Dengan asumsi ini, *node* S bisa mengestimasi kuat sinyal yang diterima oleh *node* R dengan mendengarkan paket RTS yang dikirimkan oleh *node* R. Data estimasi ini disimpan dalam tabel yang disebut dengan *CoopTable* dengan format seperti pada Tabel 3.1 berikut ini. Estimasi kuat sinyal yang diterima melalui mendengarkan paket RTS akan disimpan pada kolom Hop1SINR dan alamat pengirim paket RTS akan disimpan pada kolom PotentialRelay.

Tabel 3.1 Format Penyimpanan Infomasi Node Tetangga Pada Tabel *CoopTable*.

Nama Kolom	Format
Destination	UanAddress (uint8_t)
PotentialRelay	UanAddress (uint8_t)
Hop1SINR	double
Hop2SINR	double
UpdateTime	time

Selain memanfaatkan RTS, CoopMAC-U juga memanfaatkan CTS untuk mendapatkan infomasi Hop2SINR. Hal ini dimungkinkan dengan cara menyisipkan informasi kuat sinyal yang diterima *destination node* pada saat menerima paker RTS ke dalam paket CTS. Dengan metode ini, *source node* bisa mengestimasi kanal S-R dan R-D dengan mengacu pada pertukaran paket kontrol tersebut.

Setelah dua *link* (S-R dan R-D) telah didapatkan estimasi SINR (*Signal to Interference plus Noise Ratio*) nya, maka langkah selanjutnya adalah memilih *relay* mana yang paling bagus dengan informasi yang telah dikumpulkan sebelumnya. Pada tiap-tiap *node* yang berpotensi menjadi *relay*, dicari nilai terkecil diantara Hop1SINR dan Hop2SINR . Nilai ini selanjutnya dibandingkan dengan nilai dari *node* yang lain dan dicari yang nilainya paling tinggi. Kemudian nilai tertinggi ini dibadingkan dengan SINR transmisi yang langsung, bila masih lebih besar dari SINR transmisi langsung maka *node* bisa dipilih menjadi *relay* namun bila sebaliknya, maka *node* akan mengirimkan data dengan transmisi *non-cooperative* karena tidak ditemukan *node* yang berpotensi menjadi relay.

Algoritma pemilihan *relay* ini sedikit berbeda dengan protokol CoopMAC yang didesain untuk Wi-Fi yang mencatat estimasi kanalnya dalam bentuk *data rate* (bit/detik). *Data rate* pada Wi-Fi bisa berubah secara adaptif tergantung pada SNR yang diterima. Dengan mengestimasi SNR saat menerima paket RTS dan membaca PLCP header dari paket DATA yang didengar dari *node* tetangga maka *node* bisa mengestimasi kedua link S-R dan R-D [21].

3.5 Format Frame Pada Paket CoopMAC-U

Sudah diketahui bahwasanya *bandwidth* yang tersedia pada sistem komunikasi bawah air sangat terbatas. Oleh karena itu, format *frame* pada paket-paket CoopMAC-U tidak bisa mengikuti standar TCP/IP yang biasa dipakai untuk jaringan komputer. Agar *bandwidth* yang tersedia tidak habis hanya untuk mengirim *header* data, maka *header* dan paket data dibuat lebih sederhana. Header dan dan format frame yang digunakan pada paket-paket CoopMAC-U bisa dibaca pada Tabel 3.2 sampai dengan Tabel 3.6.

Tabel 3.2 Format Paket RTS

Header	Ukuran
Nomer *Frame*	8 bit
Jumlah *Frame*	8 bit
Panjang *Frame*	16 bit
Jumlah *Retry*	8 bit
Source Address	8 bit
Destination Address	8 bit
Helper Address	8 bit
Tipe paket	8 bit

Tabel 3.3 Format Paket CTS

Header	Ukuran
Nomer *Frame*	8 bit
Jumlah *Retry*	8 bit
RxSINR	24 bit
Source Address	8 bit
Destination Address	8 bit
Helper Address	8 bit
Tipe paket	8 bit

Tabel 3.4 Format Paket HTS

Header	Ukuran
Nomer *Frame*	8 bit
Jumlah *Retry*	8 bit
Source Address	8 bit
Destination Address	8 bit
Helper Address	8 bit
Tipe paket	8 bit

Tabel 3.5 Format Paket DATA

Header	Ukuran
Nomer *Frame*	8 bit
Source Address	8 bit
Destination Address	8 bit
Helper Address	8 bit
DATA	x bit
Tipe Paket	8 bit

Tabel 3.6 Format Paket ACK

Header	Ukuran
Nomer *Frame*	8 bit
Frame yang di-*ack*	8 bit
Source Address	8 bit
Destination Address	8 bit
Helper Address	8 bit
Tipe Paket	8 bit

3.6 Model Kanal Propagasi

Untuk menyimulasikan protokol CoopMAC-U di atas, perlu juga menyimulasikan *layer* yang berada di bawahnya, yaitu *physical layer*. Pada penelitian ini akan dicoba dengan dua model *physical layer* yaitu model Thorp dengan modem BPSK *over* Rayleigh dan yang yang kedua yaitu model Bellhop dengan modem FH-FSK (*Frequency Hoping – Frequency Shift Keying*).

3.6.1 Model Kanal Thorp dengan Modem BPSK

Model kanal ini adalah model *physical layer* sederhana yang hanya menghitung redaman sinyal akustik yang besarnya tergantung oleh jarak dan frekuensi. Seperti telah dijelaskan pada Bab II, redaman gelombang akustik bawah air bisa dihitung dengan persamaan (2.3) dan untuk *noise* bisa dihitung dengan persamaan (2.9). Dari kedua persamaan tersebut, bisa dihitung nilai SNR (*Signal to Noise* Ratio) dengan persamaan (3.8).

$$SNR = \frac{P_t}{A(d,f) \cdot N(f) \cdot \Delta f} \tag{3.8}$$

dimana P_t adalah daya transmit, $A(d,f)$ adalah redaman yang tergantung pada jarak dan frekuensi, $N(f)$ adalah kerapatan daya *ambient noise* dan Δf adalah lebar pita.

Bila diasumsikan menggunakan modem BPSK dan kanal akustik diasumsikan berupa kanal *Rayleigh* maka probabilitas bit *error* nya bisa dihitung dengan persamaan (3.9) [23].

$$P_b = \frac{1}{2}\left(1 - \sqrt{\frac{\bar{\gamma}_b}{1+\bar{\gamma}_b}}\right) \quad (3.9)$$

Dimana (P_b) adalah probabilitas bit *error* dan $\bar{\gamma}_b$ adalah SNR rata-rata. Dari probabilitas bit *error* di atas, bisa dihitung probabilitas *packet error* dengan menggunakan persamaan (3.10) [24]:

$$P_p = 1 - (1 - P_b)^N \quad (3.10)$$

dimana P_p adalah probabilitas paket *error*, P_b adalah probabilitas bit *error* dan N adalah panjang paket dalam satuan bit.

Modem BPSK ini bekerja dengan spesifikasi yang dijelaskan dalam Tabel 3.7 berikut ini.

Tabel 3.7 Spesifikasi Modem BPSK

Data rate	4096 bit/detik
Panjang paket	4096 bit/paket
TxPower	137 dB re µPa (1 yards dari sumber)
Frekuensi *center*	25 KHz

3.6.2 Model Kanal Bellhop dengan Modem FH-FSK

Bila pada model kanal thorp sebelumnya hanya menghitung redaman yang sebagai fungsi frekuensi dan jarak saja tanpa mempertimbangkan parameter lingkungan bawah air, maka pada model kanal bellhop ini pemodelan kanal akustik dilakukan dengan lebih detail dengan memasukkan parameter lingkungan beserta efek lintasan jamak yang terjadi pada laut dangkal.

Model kanal bellhop ini memanfaatkan bellhop *simulator* yang merupakan salah satu model propagasi akustik yang menggunakan teknik *ray tracing*. Aplikasi bellhop ini tersedia dalam satu paket aplikasi yang disebut dengan *acoustic toolbox*. *Acoustic toolbox* ini terdiri atas kumpulan empat simulator kanal akustik bawah air dari tiga macam model simulasi, yaitu bellhop (*ray theory*), kraken (*normal mode*), krakel, (*normal* mode) dan scooter (*fast field*). Aplikasi simulator yang ditulis dalam bahasa pemrograman FORTRAN ini bisa diunduh dengan bebas dari situs

http://oalib.hlsresearch.com/ Rays/index.html namun untuk menggunakannya perlu dilakukan proses *compiling* terlebih dahulu yang urutan prosesnya akan dijelaskan pada Lampiran B [25].

Pada pemodelan kanal ini, simulator bellhop dimanfaatkan untuk membangkitkan *channel impulse response* dengan jarak dan kecepatan perambatan gelombang akustik di bawah air pada tiap kedalaman (*sound speed profile*) sebagai masukannya. Seperti dijelaskan pada Bab II sebelumnya, bahwa kecepatan suara di bawah air tergantung pada salinitas, temperatur dan kedalaman atau tekanan air laut. Oleh karena itu, untuk membuat *sound speed profile* dibutuhkan ketiga parameter lingkungan tersebut. Ketiga parameter lingkungan ini bisa didapat melalui *World Ocean Atlas* yang bisa diakses melalui situs web dengan alamat https://www.nodc.noaa.gov/ OC5/woa13/woa13data.html [26]

Pada penelitian ini, lokasi laut yang digunakan dalam simulasi berada pada koordinat *latitude*=-9.125 dan *longitude*=133.125 atau bila dalam peta lokasinya terletak di laut Arafura seperti yang ditunjukkan pada Gambar 3.4. Lokasi ini dipilih dengan pertimbangan bahwa kedalaman laut pada koordinat ini tidak lebih dari 200 meter. Dengan kata lain, laut pada koordinat tersebut masih dalam kategori laut dangkal.

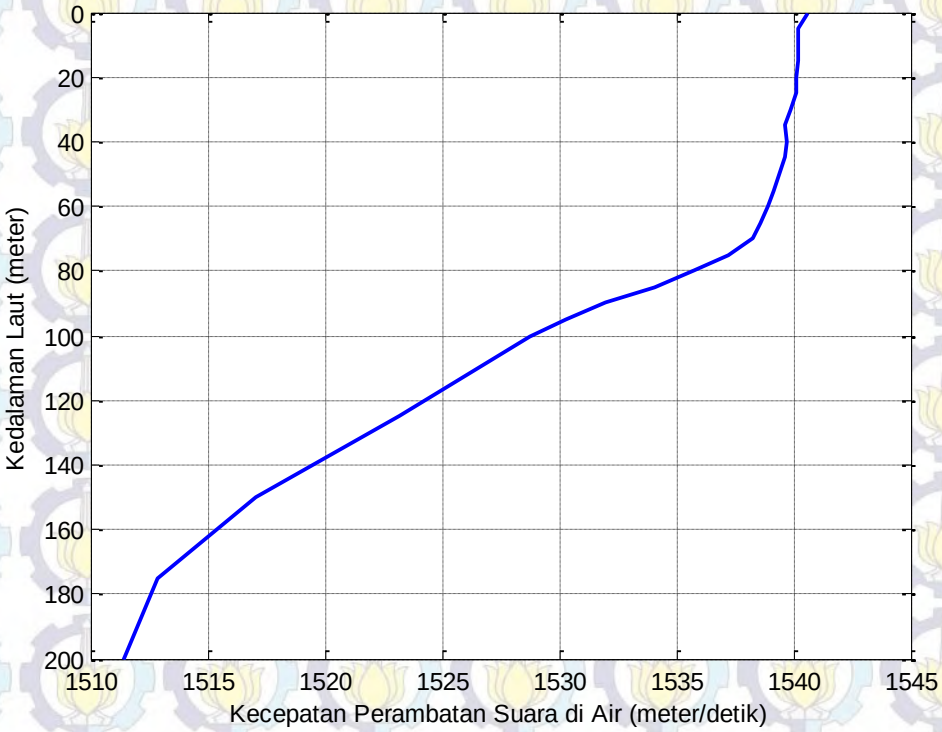
Gambar 3.4 Lokasi Sampel Untuk Masukan Simulator Bellhop. [27]

Di *World Ocean Atlas* terdapat beberapa versi data yang bisa dipilih, namun pada penelitian ini menggunakan data *World Ocean Atlas V2 2013 - annual statistical mean* dengan ukuran *grids* koordinat 1/4°. Ketiga parameter lingkungan di atas dituliskan dalam Tabel 3.8 berikut ini.

Tabel 3.8 Parameter Lingkungan Bawah Air Pada Koordinat (-9.125,133.125)

Kedalaman (meter)	Salinitas (*unitless*)	Temperatur (°C)
0	34.198	28.134
5	33.751	28.137
10	33.752	28.097
15	33.751	28.048
20	33.75	27.982
25	33.754	27.939
30	33.752	27.808
35	33.754	27.673
40	33.961	27.557
45	34.083	27.442
50	34.087	27.272
55	34.103	27.14
60	34.141	26.968
65	34.237	26.756
70	34.312	26.528
75	34.385	25.992
80	34.405	25.347
85	34.433	24.585
90	34.466	23.678
95	34.489	22.962
100	34.499	22.339
125	34.5	20.095
150	34.54	17.763
175	34.565	16.242
200	34.573	15.628

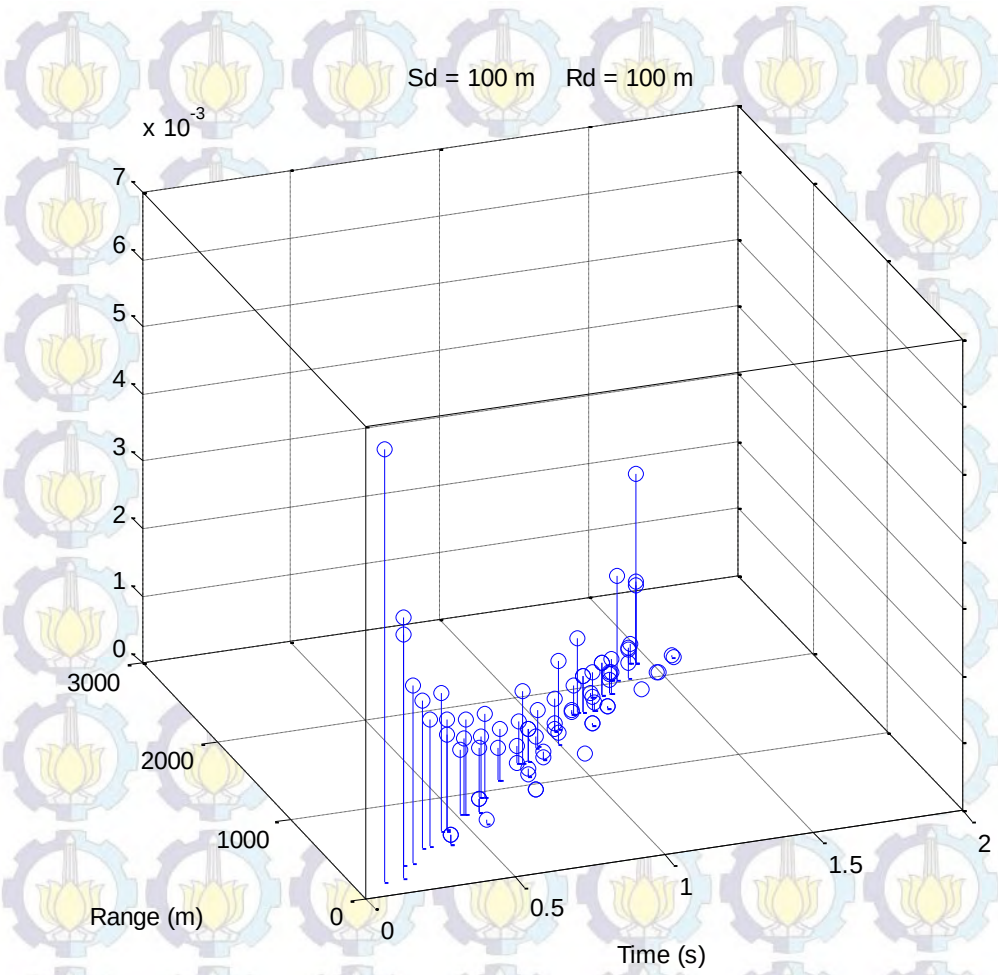
Dengan menggunakan rumus Mackenzie pada persamaan (2.10) dan ketiga data parameter lingkungan pada Tabel 3.8 di atas, bisa dihitung kecepatan suara pada tiap kedalaman laut yang hasilnya digambarkan dalam grafik *sound speed profile* pada gambar 3.5. Untuk *souce code* dalam bentuk m-file untuk mendapatkan *sound speed profile* dituliskan pada Lampiran C.



Gambar 3.5 Grafik *Sound Speed Profile* Pada Koodinat (-9.125,133.125)

Setelah *sound speed profile* (SSP) didapat, langkah selanjutnya adalah membangkitkan *channel impulse response* sesuai dengan jarak antara *transmitter* dan *receiver*. Simulator bellhop membutuhkan ENV *file* (*environment file*) untuk mencatat *Sound speed profile* dan jarak yang diinginkan sebelum dihitung oleh aplikasi bellhop. Detail mengenai ENV *file* dan penggunaan simulator belhop dijelaskan pada Lampiran D.

Jika semua *node* dalam jaringan sensor bawah air diasumsikan berada pada kedalaman yang sama, misal kedalaman = 100m, maka *channel impulse response* sebagai fungsi jarak dengan mengacu pada SSP pada gambar 3.5 bisa ditunjukkan pada gambar 3.6.

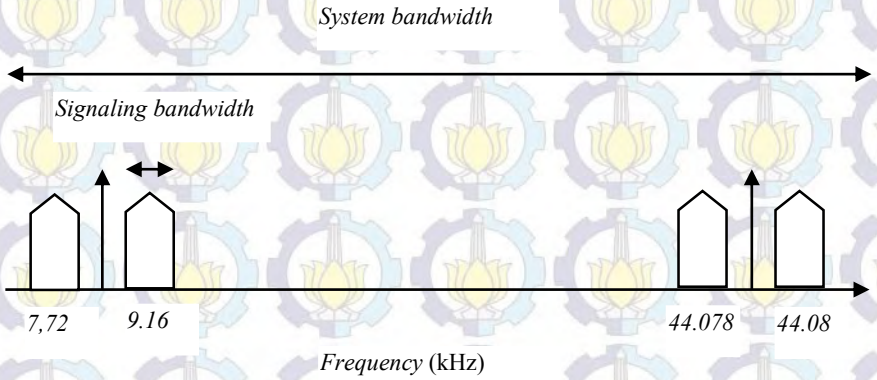


Gambar 3.6. *Channel Impulse Response* Sebagai Fungsi Jarak.

Model kanal akustik di atas akan digunakan untuk menyimulasikan *physical layer* untuk modem FH-FSK (*Frequency Hoping – Frequency Shift Keying*). Modem FH-FSK adalah modem dengan modulasi FSK namun frekuensi simbol tiap bit akan selalu berubah sesuai dengan jumlah *hop* atau *bin*. Pada simulasi ini, akan dibuat modem FH-FSK yang meniru desain pada [28] [29] dengan modifikasi peningkatan *bandwidth*. Bila pada [28], modem dengan *data rate* tertinggi adalah 320 bit/detik dengan jumlah *hops* sebanyak 3, maka pada simulasi ini akan dibuat modem dengan dengan *data rate* 720 bit/detik dengan jumlah *hops* sebanyak 13 dengan desain seperti ditunjukkan pada Tabel 3.7 dan Gambar 3.7 berikut ini.

Tabel 3.9 Tabel Konfigurasi FH-FSK

Data rate	720 bps
Panjang paket data	720 bit/paket
Jumlah *hops*	13 *hops*
Clearing Time	16.667 msec
Bandwidth sistem	37440 kHz
Signaling bandwidth	720 Hz
Frekuensi *low*	7720 Hz
Frekuensi *high*	44080 Hz



Gambar 3.7 *Bandwidth* untuk *Frequency Hopping*

Dengan menggunakan persamaan redaman dan *noise* pada persamaan (2.3) dan persamaan (2.9), maka SNR pada modem FH-FSK bisa dihitung dengan persamaan :

$$SNR = \frac{P_t}{A(d,f) \cdot N(f) \cdot \Delta f + ISI} \tag{3.11}$$

Dimana *ISI* adalah interferensi yang diakibatkan oleh efek *multipath* yang berada pada hop yang sama.

Dari SNR di atas bisa dihitung probabilitas *bit error* nya dengan persamaan (3.12) [23].

$$p_b = \frac{1}{2 + E_b / N_0} \tag{3.12}$$

Modem FH-FSK diasumsikan menggunakan *convolutional code* dengan rate ½ dan *constraint length* 9 ditambah dengan kemampuan CRC. Sehingga probabilitas *packet error* final nya dituliskan pada persamaan (3.13) [29].

$$P_b = 1 - \left((1 - P_{b-vit})^N + \binom{288}{1} \cdot P_{b-vit} \cdot (1 - P_{b-vit})^N \right) \tag{3.13}$$

Dimana *N* adalah panjang data dalam bit, P_{b-vit} adalah probabilitas *bit error* yang dihasilkan oleh viterbi *decoder* yang didapat dari persamaan (3.14) [23].

$$P_b < \frac{1}{k} \sum_{d=d_{free}}^{\infty} \beta_d P_2(d) \tag{3.14}$$

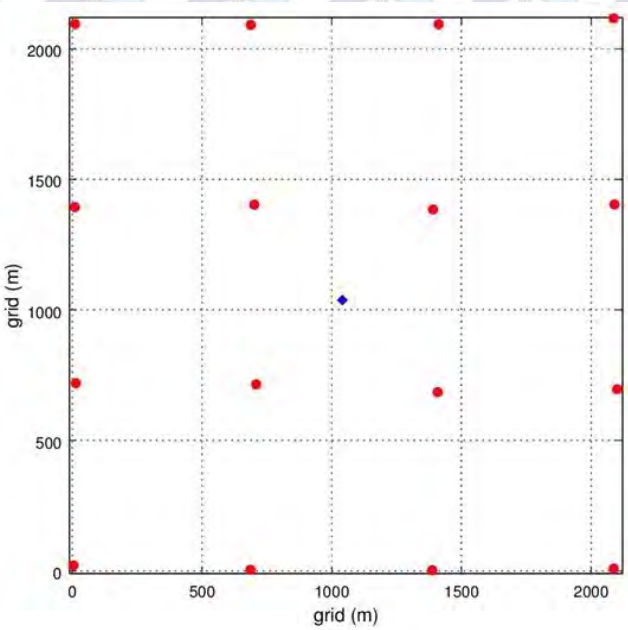
Dimana β_d adalah koefisien bobot yang nilainya diambil dari [30] dan $P_2(d)$ bisa dihitung dengan persamaan (3.15)

$$P_2(d) = {p_b}^d \sum_{k=0}^{d-1} \binom{d-1+k}{k} (1 - p_b)^k \tag{3.15}$$

Dimana *d* adalah nomer posisi bit dimana dua rangkaian berbeda dan p_b adalah nilai yang didapat dari persamaan (3.12).

3.7 Topologi Jaringan Sensor Nirkabel Bawah Air

Jaringan sensor nirkabel yang digunakan pada simulasi ini terdiri atas 16 sensor *node* dan 1 *sink*. Penempatan posisinya berbentuk *grid* dengan ukuran 2100m x 2100m yang terbentuk oleh susunan *grid* yang lebih kecil dengan ukuran 700m x 7000m. *Node* tidak diletakkan persis di koordinat kelipatan 700m namun diberikan nilai acak ± 25m. Diasumsikan *node* tidak berpindah posisi selama proses simulasi berlangsung. *Sink Node* diletakkan pada tengah-tengah *grid*. Semua *sensor node* dan *sink node* berada pada kedalaman air laut yang sama yaitu 100m dari kedalaman total air laut 200m. *Setup* posisi *node* tampak dari atas ditunjukkan pada gambar 3.8.



Gambar 3.8 Posisi Node Tampak Atas,
Merah=*Sensor Node*, Biru=*Sink Node.*

3.8 Pembangkitan Trafik

Trafik yang digunakan dalam simulasi diasumsikan panjangnya konstan di setiap paket dengan durasi paketnya adalah 1 detik. Jarak antar kedatangan paket rata-rata (*interarrival time*) diatur untuk mendapatkan *offered load* yang diinginkan. *Interarrival time* bernilai acak terdistribusi eksponensial dengan rata-rata tertentu. *Offered load* dihitung menggunakan persamaan (3.16).

$$\rho_{node} = (\lambda_{node} \cdot L_{DATA}) / R \tag{3.16}$$

Dimana ρ_{node} adalah *normalized offered load*, λ_{node} adalah rata-rata kedatangan paket per detik, L_{DATA} adalah panjang data (dalam bit) dan R adalah kecepatan transmisi (dalam bit/detik).

Pembangkitan trafik data pada simulasi ini menggunakan OnOffHelper yang sudah disediakan oleh NS3. Setidaknya dibutuhkan empat parameter untuk membangkitkan trafik menggunakan OnOffHelper yaitu OnTime, OffTime, DataRate, dan PacketSize. OnTime adalah durasi paket, OffTime adalah durasi tidak adanya paket, DataRate adalah laju kecepatan data dalam satuan bit/detik dan PacketSize dalam satuan byte.

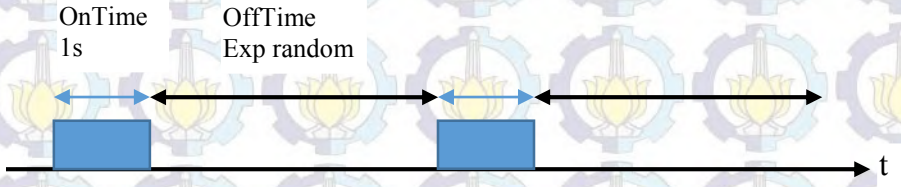
Semua paket yang dibangkitkan pada simulasi ini menggunakan OnTime yang nilainya konstan sebesar 1 detik dan OffTime yang nilainya acak mengikuti distribusi eksponensial dengan nilai rata-ratanya dittentukan sesuai dengan *offered load* yang diingikan.

BAB IV
HASIL DAN ANALISA

Pada bab ini akan disampaikan hasil simulasi dan analisa dari algoritma protokol dengan menggunakan model propagasi yang sudah dijelaskan pada bab sebelumnya.

4.1 Pengujian Pembangkitan Trafik

Seperti dijelaskan pada sub bab 3.9 tentang pembangkitan paket, ukuran paket akan selalu konstan dengan durasi paket 1 detik. Sedangkan kedatangan antar paket dibuat acak dengan mengatur durasi OffTime (tidak sedang mentransmisikan paket) seperti diilustrasikan pada gambar 4.1.



Gambar 4.1 Ilustrasi Pembangkitan Paket.

Pengujian pembangkitan trafik ini diperlukan untuk mempelajari perilaku *random generator* untuk keperluan analisa lebih lanjut. Terdapat tiga skenario pengujian, yaitu 1). *Single node* dengan *simulation time* konstan dan beban trafik diubah-ubah, 2). *Single node* dengan beban trafik konstan dan *simulation time* diubah-ubah dan 3). Dua *node* dengan *simulation time* konstan dan beban trafik diubah-ubah. Hasil pengujian ditunjukkan pada Tabel 4.1 , Tabel 4.2 dan Tabel 4.3 berikut ini.

Offered load seperti yang didefinisikan dalam persamaan (3.16) adalah besaran tanpa satuan (*unitless*). Namun, karena panjang paket dibuat tetap dengan durasi tiap paket adalah 1 detik, maka *offered load* di bab IV ini adalah ekivalen dengan jumlah paket per detik.

Tabel 4.1 Hasil Pengujian Pembangkitan Trafik pada Skenario 1, *Simulation Time* = 604800.

Offered Load (paket/detik)	Jumlah paket yang diharapkan	Jumlah paket yang dibangkitkan	%error
0.01	6048	6042	0.000992063
0.02	12096	12012	0.006944444
0.03	18144	18002	0.007826279
0.04	24192	24042	0.006200397
0.05	30240	29949	0.009623016
0.06	36288	36033	0.007027116
0.07	42336	42046	0.006849962
0.08	48384	48036	0.00719246
0.09	54432	54117	0.005787037
0.1	60480	60035	0.007357804
0.11	66528	66214	0.004719817
0.12	72576	72098	0.006586199
0.13	78624	78083	0.006880851
0.14	84672	84048	0.007369615
0.15	90720	90004	0.007892416

Tabel 4.2 Hasil Pengujian Pembangkitan Trafik pada Skenario 2, *Offered load* = 0.1

Simulation time	Jumlah paket yang diharapkan	Jumlah paket yang dibangkitkan	%error
3600	360	390	0.083333333
7200	720	743	0.031944444
14400	1440	1434	0.004166667
28800	2880	2878	0.000694444
57600	5760	5741	0.003298611
115200	11520	11442	0.006770833
230400	23040	22895	0.006293403
460800	46080	45754	0.007074653
921600	92160	91404	0.008203125

Tabel 4.3 Hasil Pengujian Pembangkitan Trafik pada Skenario 3, Dua *Node*, *Simulation time* = 18000

Offered load (paket/detik)	Jumlah paket yang diharapkan	Jumlah paket pada node 1	%error	Jumlah paket pada node 2	%error
0.01	180	165	0.083333333	183	0.016666667
0.02	360	357	0.008333333	332	0.077777778
0.03	540	536	0.007407407	518	0.040740741
0.04	720	732	0.016666667	696	0.033333333
0.05	900	946	0.051111111	869	0.034444444
0.06	1080	1140	0.055555556	1056	0.022222222
0.07	1260	1317	0.045238095	1242	0.014285714
0.08	1440	1503	0.04375	1414	0.018055556
0.09	1620	1684	0.039506173	1575	0.027777778
0.1	1800	1836	0.02	1769	0.017222222
0.11	1980	2031	0.025757576	1958	0.011111111
0.12	2160	2216	0.025925926	2139	0.009722222
0.13	2340	2407	0.028632479	2323	0.007264957
0.14	2520	2608	0.034920635	2510	0.003968254
0.15	2700	2785	0.031481481	2689	0.004074074

Dari ketiga tabel diatas bisa diketahui bahwa jumlah paket yang dibangkitkan tidak pernah sesuai dengan paket yang diharapkan. Namun selisih antara paket yang dibangkitkan dengan paket yang diharapkan masih dalam kategori normal terlihat dari %error yang kecil. Di samping itu, trafik yang dihasilkan tidak terpengaruhi oleh *offered load* dan *simulation time*. Dari percobaan skenario 3 juga bisa diketahui bahwa pembangkitan paket dengan menggunakan lebih dari satu *node* akan menghasilkan jumlah trafik yang berbeda antara satu *node* dengan yang lain namun jumlah trafik semuanya mendekati jumlah trafik yang diharapkan.

4.2 Pengujian Throughput Berdasarkan Probabilitas Paket Error

Keberhasilan paket yang diterima oleh *sink* dipengaruhi oleh kualitas kanal. Namun pada simulasi ini, kualitas kanal diwakili oleh perhitungan SNR untuk mendapatkan probabilitas paket *error*. Pada sub bab ini akan dilakukan pengujian penentuan keberhasilan paket berdasaran probabilitas paket *error*. Data hasil pengujian bisa dibaca pada Tabel 4.4.

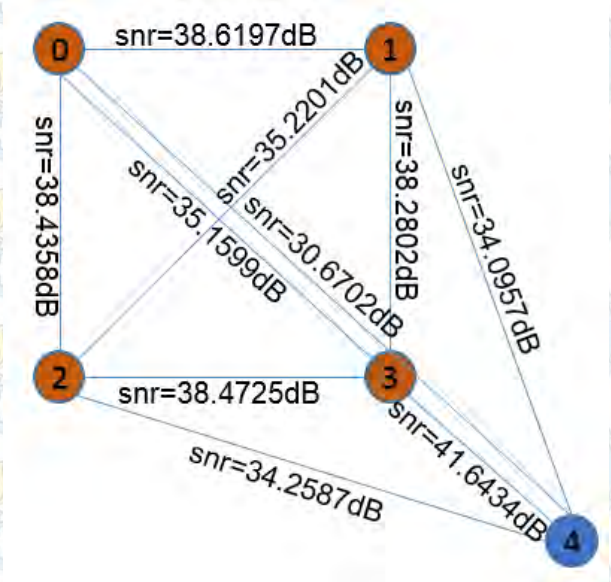
Tabel 4.4 Hasil Pengujian Keberhasilan Paket Berdasarkan Probabilitas Paket *Error* Dengan *Offered load*=0.1 dan *Simulation Time*=18000

Probilitas P_e	Jumlah paket yang diharapkan	Jumlah paket yang diterima	%error
0.1	1620	1622	0.001234568
0.2	1440	1422	0.0125
0.3	1260	1244	0.012698413
0.4	1080	1076	0.003703704
0.5	900	907	0.007777778
0.6	720	738	0.025
0.7	540	550	0.018518519
0.8	360	379	0.052777778
0.9	180	186	0.033333333

Dari Tabel 4.4 di atas bisa diketahui bahwa keberhasilan paket yang diterima dengan benar berdasarkan probabilitas paket *error* menghasilkan angka yang mendekati nilai yang diharapkan. Sehingga bisa disimpulkan bahwa model CalcPer sudah bekerja dengan semestinya.

4.3 Pengujian Algoritma Seleksi Relay

Pengujian algoritma seleksi relay dilakukan dengan penyederhanaan jaringan menjadi hanya menggunakan empat *node* dan satu *sink* untuk memudahkan pengamatan seperti diilustrasikan dalam Gambar 4.2.



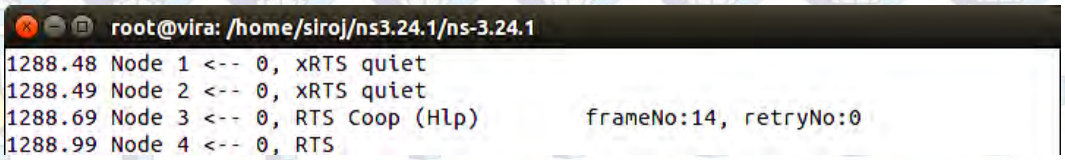
Gambar 4.2 Setup *Node* Untuk Pengujian Seleksi Relay.

Dengan *setup* pengujian diatas, maka jika *node* 0 ingin mengirim data ke *sink* (*node* 4) maka terdapat tiga *potential node* yang bisa menjadi *relay*, yaitu *node* , *node* 2 dan *node* 3. Bila *CoopTable* pada *node* 0 ditampilkan, hasilnya seperti ditunjukkan oleh Tabel 4.5.

Tabel 4.5 Tabel *CoopTable* pada *Node* 0

Destination node	Potential Relay	Hop1SINR (dB)	Hop2SINR (dB)
4	1	38.6197	34.0957
4	2	38.4358	34.2587
4	3	35.1599	41.6434

Dengan mengacu pada *CoopTable* pada Tabel 4.5 diatas, maka nilai SINR minimum dari kedua *hop* (S-R dan R-D) dari masing-masing *potential relay* berurutan dari *relay* 1, 2 dan 3 adalah 34.0956dB, 34.2587dB dan 35.1599dB. Dengan demikian nilai terbesar dari nilai minimum kedua *hop* diberikan oleh *node* 3 sehingga *node* 3 terpilih sebagai *relay* bagi *node* 0 untuk mentransmisikan data ke *node* 4. Hal ini terkonfirmasi dari *screenshot* terminal progam yang ditunjukkan oleh Gambar 4.3.



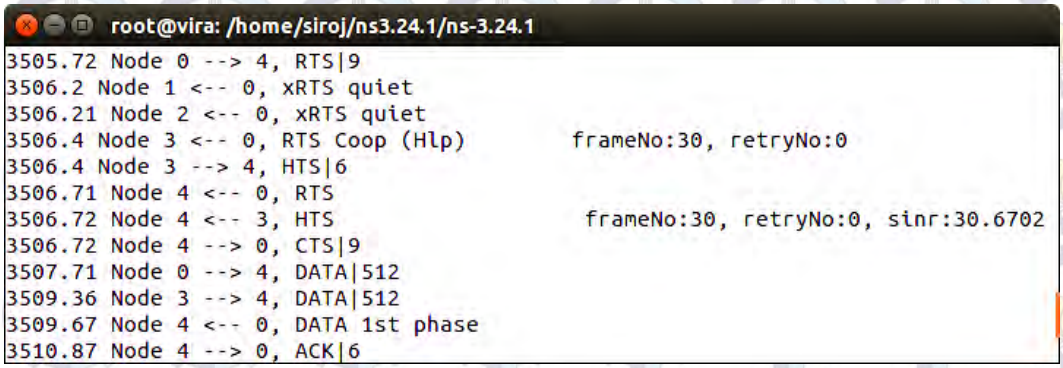
Gambar 4.3 *Screenshot* Terminal Program NS3 Untuk Pengujian Algoritma Pemilihan *Relay*.

Ketika *node* 0 mengirimkan paket RTS, *node* 1 dan *node* 2 yang tidak diikutsertakan dalam transmisi kooperatif akan mengubah *state* nya menjadi QUIET. Sedangkan *node* 3 statusnya kan menjadi *helper node* bagi *node* 0 untuk mentransmisikan data ke *node* 4. Dengan demikian, algoritma pemilihan *relay* telah teruji dengan benar.

4.4 Pengujian Algoritma Protokol CoopMAC-U

Untuk menguji algoritma Protokol CoopMAC-U, dilakukan dengan *setup* posisi dan jumlah *node* yang sama dengan pengujian algoritma seleksi *relay*, yaitu menggunakan empat *node* dan satu *sink* seperti pada Gambar 4.2. Gambar 4.4

berikut ini adalah *screenshot* terminal program NS3 ketika Protokol CoopMAC-U sudah bekerja dan informasi *node* tetangga sudah diketahui dengan lengkap.

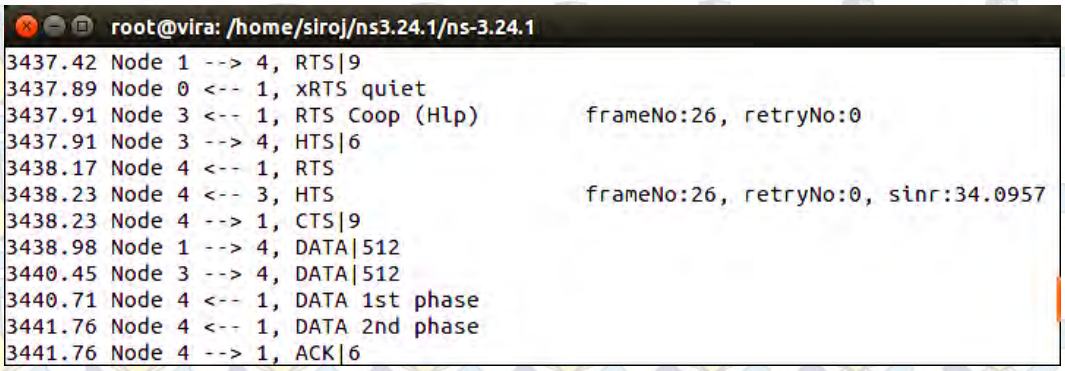


Gambar 4.4 *Screenshot* Terminal Program NS3 Untuk Pengujian Algoritma Protokol CoopMAC-U.

Dari Gambar 4.4 diatas bisa diketahui bahwasanya pada waktu 3505.72 detik, *node* 0 akan mengirimkan data ke *sink* (*node* 4). Sesuai dengan algoritma pemilihan relay pada sub bab 4.3, *potential relay* yang terpilih adalah *node* 3. Oleh karena itu, pada paket RTS sudah disisipkan permintaan ke *node* 3 untuk bersedia menjadi *relay / helper node*. Meskipun paket RTS ditujukan kepada *node* 3 dan *node* 4, namun karena sifat propagasinya yang menyebar ke semua arah maka *node* 1 dan *node* 2 pun juga mendapatkan paket RTS tersebut. Karena *node* 1 dan *node* 2 bukan *intended receiver* dan bukan pula *helper* maka kedua *node* tersebut mengubah *state* nya menjadi QUIET dan menahan segala jenis paket yang akan dikirimkan. *Node* 3 yang menerima paket RTS dan dalam kondisi bisa membantu, akan mengirimkan paket HTS.

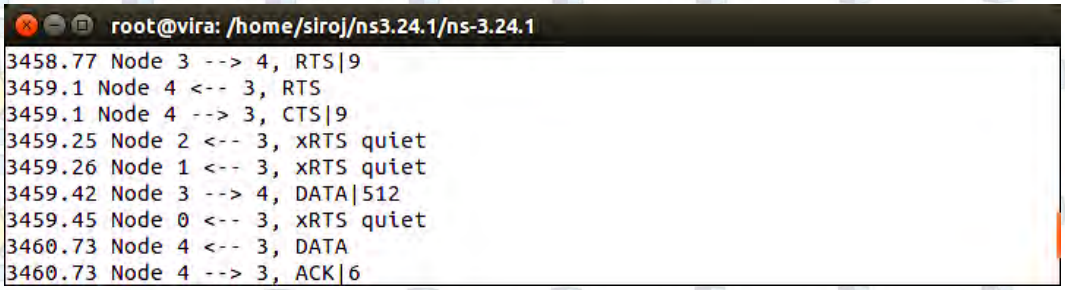
Setelah paket HTS diterima oleh *node* 4, *node* 4 akan mengirimkan paket CTS ke *node* 0 untuk memberi tahu bahwa *node* 4 sudah siap menerima transmisi data sekaligus memberi tahu bahwa *node* 3 juga siap menjadi *helper*. Setelah mendapat balasan paket CTS, *node* 0 langsung mengirimkan paket DATA disusul dengan *node* 3 mentransmisikan ulang paket DATA yang sudah diterima ke *node* 4. Namun pada akhirnya, paket DATA yang berhasil diterima oleh *node* 4 hanya satu saja dan *node* 4 menjawabnya dengan paket ACK untuk memberi tahu bahwa paket diterima dengan sukses.

Bila pada data Gambar 4.4 hanya menunjukkan satu paket DATA saja yang berhasil diterima dengan benar oleh *sink* (*node* 4), Gambar 4.5 berikut ini adalah *screenshot* transmisi dari *node* 1 ke *node* 4 yang dibantu oleh *node* 3 yang kedua paket dari fase *direct* dan *relay* keduanya diterima dengan benar.



Gambar 4.5 *Screenshot* Terminal Program NS3 Ketika Kedua Fase Paket DATA Berhasil Diterima dengan Benar oleh *Sink*.

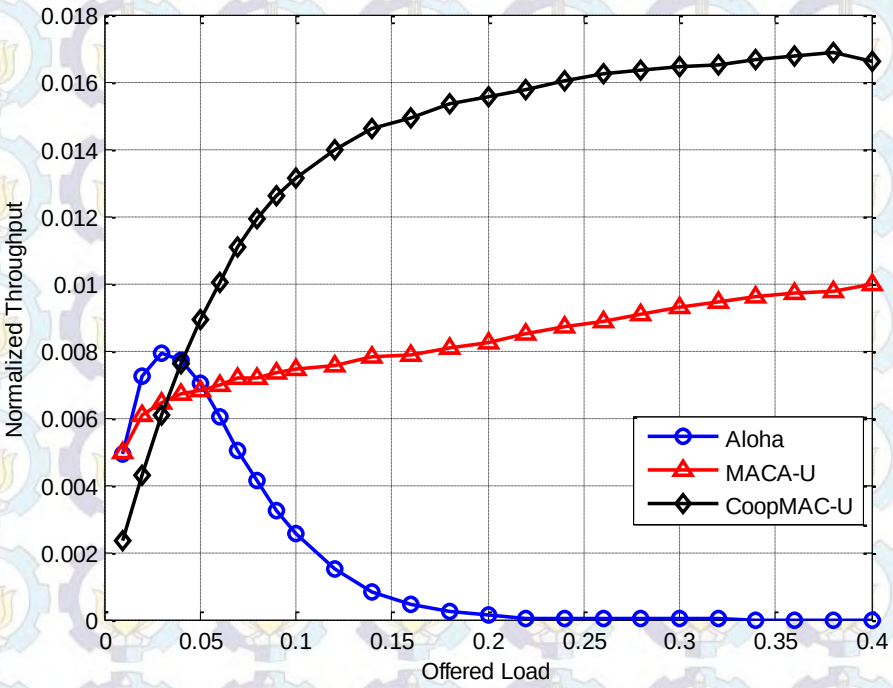
Untuk mendukung fleksibilitas protokol agar bisa digunakan pada setiap kondisi, maka node harus tetap bisa mentransmisikan data meski tersedia *potential relay* ataupun tidak ada. Hal ini terjadi pada fase awal ketika masih mengumpulkan infomasi *node* tetangga atau bisa juga disebabkan karena posisi antara *source node* dan *destination node* tidak ada *node* lain yang berpotensi menjadi *relay*. Seperti pada Gambar 4.6 yang menunjukkan bahwa *node* 3 tidak bisa menemukan *potential relay* sehingga memutuskan untuk mentransmisikan datanya secara non-kooperatif.



Gambar 4.6 *Screenshot* Terminal Program Ketika *Node* Tidak Menemukan *Potential Relay*.

4.5 Throughput Ternormalisasi Model Kanal Thorp

Model kanal Thorp adalah model kanal yang hanya menghitung *pathloss* dengan mengabaikan efek *multipath* yang terjadi pada kanal. Pengujian *throughput* dilakukan dengan membandingkan protokol CoopMAC-U dengan dua protokol yang lain, yaitu Aloha dan MACA-U. Simulasi dilakukan selama 608400 detik pada tiap *offered load*.



Gambar 4.7 Grafik *Throughput* Ternormalisasi Tiga Protokol dengan Model Kanal Thorp.

Aloha adalah mekanisme pengiriman paket yang tanpa ada penjadwalan ataupun reservasi kanal sebelumnya. Setiap kali *node* mempunyai data pada antrian, pada saat itu pula *node* mentransmisikan datanya. Oleh karena itu, probabilitas terjadinya tabrakan sangat tinggi. Hal ini bisa diamati dari grafik pada Gambar 4.7 dan Tabel 4.6, bahwasanya protokol Aloha hanya efektif digunakan untuk transmisi data dengan trafik yang rendah dan tidak cocok untuk transmisi trafik yang tinggi. Karena banyaknya kejadian tabrakan paket pada trafik data yang tinggi, utilitas kanal menjadi turun sampai pada kondisi paling parah, yaitu tidak ada satu pun

paket yang berhasil diterima di *sink*. Kondisi di mana tidak ada satu pun paket yang berhasil diterima dengan benar oleh *sink* terjadi saat *offered load* = 0,34 paket/detik.

Tabel 4.6 Tabel Throughput Ternormalisasi Protokol Aloha Pada Model Kanal Thorp dengan *Simulation Time* = 604800 detik.

Offered Load (paket/detik)	*Normalized Throughput* (paket/detik)
0.0100	0.004923011740000
0.0200	0.007246817130000
0.0300	0.007935371200000
0.0400	0.007705543150000
0.0500	0.007027426420000
0.0600	0.006039393190000
0.0700	0.005044539520000
0.0800	0.004114066630000
0.0900	0.003262855490000
0.1000	0.002566344250000
0.1200	0.001505559690000
0.1400	0.000837363591000
0.1600	0.000450252149000
0.1800	0.000234685020000
0.2000	0.000117600860000
0.2200	0.000046916336000
0.2400	0.000023871527800
0.2600	0.000009507275130
0.2800	0.000004960317460
0.3000	0.000001446759260
0.3200	0.000001033399470
0.3400	0
0.3600	0
0.3800	0
0.4000	0

Protokol yang kedua yang dijadikan pembanding adalah MACA-U yang mengacu pada [31]. MACA-U termasuk protokol MAC yang *contention based*. Yang membedakan MACA-U dengan MACA / MACAW adalah pada aturan perubahan *state*, algoritma *backoff*, strategi penerusan paket dan pada MACA-U tidak menggunakan paket ACK.

Throughput ternormalisasi untuk protokol MACA-U, seperti yang tertulis dalam Tabel 4.7, bila dibandingkan dengan yang dihasilkan oleh Aloha adalah pada

throughput Aloha lebih bagus dibandingkan dengan MACA-U ketika *offered load* di bawah 0,05 paket/detik. MACA-U mengungguli Aloha ketika *Offered load* sudah di atas 0,05 paket/detik. Karena MACA-U mempunyai mekanisme pengaturan pemakaian kanal, sehingga ketika *offered load* dinaikkan maka beban trafik yang dibangkitkan masih bisa dilayani oleh kanal dan *collision* bisa dihindari meskipun *throuhgput* pada MACA-U juga tidak bisa naik signifikan. Tabrakan data bisa dikurangi dengan adanya mekanisme reservasi kanal, namun keberhasilan paket untuk diterima oleh *sink* tidak ada peningkatan.

Tabel 4.7 Tabel Throughput Ternormalisasi Protokol MACA-U Pada Model Kanal Thorp dengan *Simulation Time* = 604800 detik.

Offered Load (paket/detik)	*Normalized Throughput* (paket/detik)
0.0100	0.004972408230000
0.0200	0.006060267860000
0.0300	0.006454613100000
0.0400	0.006689814810000
0.0500	0.006833974040000
0.0600	0.006977306550000
0.0700	0.007185846560000
0.0800	0.007189980160000
0.0900	0.007327835650000
0.1000	0.007445229830000
0.1200	0.007559937170000
0.1400	0.007789971890000
0.1600	0.007891968420000
0.1800	0.008074260090000
0.2000	0.008260582010000
0.2200	0.008484313000000
0.2400	0.008684792490000
0.2600	0.008865327380000
0.2800	0.009099185680000
0.3000	0.009304418820000
0.3200	0.009466869210000
0.3400	0.009592943950000
0.3600	0.009716125170000
0.3800	0.009756531080000
0.4000	0.009973028270000

Bila dibandingkan dengan kedua protokol yang sudah dibahas sebelumnya, CoopMAC-U memberikan *throughput* yang lebih bagus ketika *offered load* sudah diatas 0,04 paket/detik. CoopMAC-U menggunakan mekanisme pengaturan kanal yang sama dengan MACA-U yaitu menggunakan paket kontrol RTS dan CTS untuk reservasi kanal. Namun, yang membedakan antara CoopMAC-U dan MACA-U adalah penggunaan kontrol paket HTS dan ACK serta penggunaan *relay* untuk meretransmisikan data ke *sink*. Dengan adanya dua transmisi paket DATA yang sama, diharapkan keberhasilan transmisi paket bisa lebih bagus. Dan hal ini bisa dibuktikan dengan *throughput* CoopMAC-U yang lebih tinggi daripada Aloha dan MACA-U seperti yang ditunjukkan pada Gambar 4.7 atau pada Tabel 4.8.

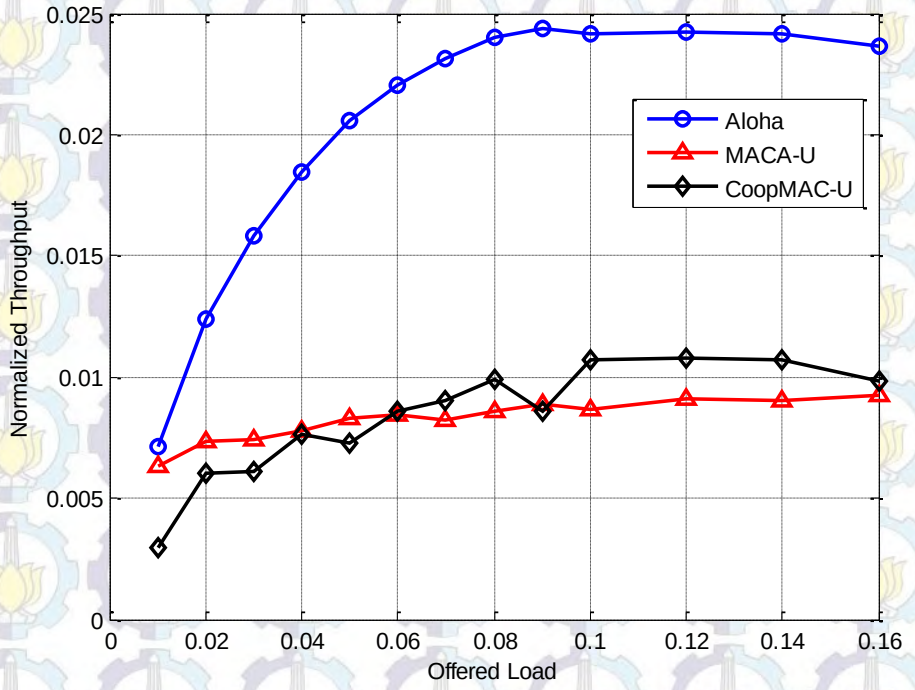
Tabel 4.8 Tabel *Throughput* Ternormalisasi Protokol CoopMAC-U Pada Model Kanal Thorp dengan *Simulation Time* = 604800 detik.

Offered Load (paket/detik)	*Normalized Throughput* (paket/detik)
0.0100	0.002335069440000
0.0200	0.004273106810000
0.0300	0.006070911870000
0.0400	0.007609333660000
0.0500	0.008933635090000
0.0600	0.010027901800000
0.0700	0.011084449400000
0.0800	0.011926360000000
0.0900	0.012615430700000
0.1000	0.013131820400000
0.1200	0.013976107800000
0.1400	0.014604518000000
0.1600	0.014942336300000
0.1800	0.015317150300000
0.2000	0.015546461600000
0.2200	0.015773396200000
0.2400	0.016047970400000
0.2600	0.016213727700000
0.2800	0.016362537200000
0.3000	0.016466393800000
0.3200	0.016511036700000
0.3400	0.016670696900000
0.3600	0.016778067100000
0.3800	0.016879236900000
0.4000	0.016620163700000

Bila *normalized throughput* yang dihasilkan dari semua *offered load* dihitung nilai rata-ratanya, maka dihasilkan nilai rata-rata *normalized throughput* untuk Aloha sebesar 0.002363942625805 paket/detik, nilai rata-rata *normalized throughput* untuk MACA-U adalah sebesar 0.007986301256800 paket/detik dan nilai rata-rata *normalized throughput* untuk CoopMAC-U adalah 0.013068712790800 paket/detik.

4.6 Throughput Ternormalisasi Model Kanal Bellhop dan Modem FH-FSK

Pengujian protokol CoopMAC-U selanjutnya adalah menggunakan model kanal Bellhop dengan menggunakan modem FH-FSK. Pada model ini, telah mempertimbangkan efek *multipath* pada kanal dengan *impulse response* yang dibangkitan oleh bellhop. *Impulse response* yang digunakan pada simulasi ini dibangkitkan untuk setiap *link* yang mungkin terjadi. Hasil simulasi untuk ketiga protokol dengan *simulation time* = 18000 detik ditunjukkan pada Gambar 4.8.



Gambar 4.8 *Throughput* Ternormalisai Tiga Protokol dengan Model Kanal Bellhop.

Nilai *normalized throughput* dari tiap *offered load* untuk protokol Aloha, MACA-U dan CoopMAC-U secara detail bisa dibaca berurutan pada Tabel 4.9, Tabel 4.10, dan Tabel 4.11.

Tabel 4.9 Tabel *Throughput* Ternormalisasi Protokol Aloha Pada Model Kanal Bellhop dengan *Simulation Time* = 18000 detik.

Offered Load (paket/detik)	*Normalized Throughput* (paket/detik)
0.0100	0.007128472222222
0.0200	0.012388888888889
0.0300	0.015805555555556
0.0400	0.018472222222222
0.0500	0.020541666666667
0.0600	0.022038194444444
0.0700	0.023128472222222
0.0800	0.024034722222222
0.0900	0.024413194444444
0.1000	0.024138888888889
0.1200	0.024239583333333
0.1400	0.024194444444444
0.1600	0.023621527777778

Tabel 4.10 Tabel *Throughput* Ternormalisasi Protokol MACA-U Pada Model Kanal Bellhop dengan *Simulation Time* = 18000 detik.

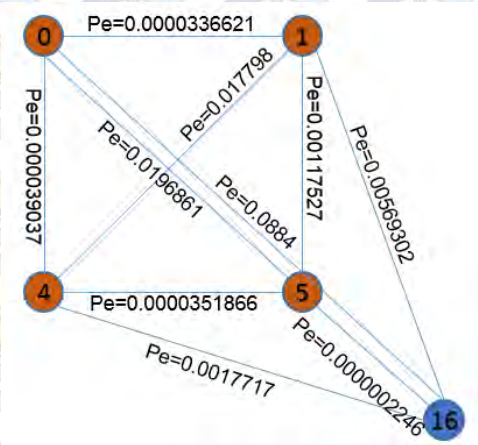
Offered Load (paket/detik)	*Normalized Throughput* (paket/detik)
0.0100	0.006298611111111
0.0200	0.007371527777778
0.0300	0.007430555555556
0.0400	0.007774305555556
0.0500	0.008333333333333
0.0600	0.008465277777778
0.0700	0.008208333333333
0.0800	0.008555555555556
0.0900	0.008906250000000
0.1000	0.008684027777778
0.1200	0.009079861111111
0.1400	0.009003472222222
0.1600	0.009274305555556

Tabel 4.11 Tabel *Throughput* Ternormalisasi Protokol CoopMAC-U Pada Model Kanal Bellhop dengan *Simulation Time* = 18000 detik.

Offered Load (paket/detik)	*Normalized Throughput* (paket/detik)
0.0100	0.002965277777778
0.0200	0.006048611111111
0.0300	0.006072916666667
0.0400	0.007670138888889
0.0500	0.007288194444444
0.0600	0.008625000000000
0.0700	0.009003472222222
0.0800	0.009937500000000
0.0900	0.008572916666667
0.1000	0.010680555555556
0.1200	0.010750000000000
0.1400	0.010722222222222
0.1600	0.009868055555556

Bila *normalized throughput* yang dihasilkan dari semua *offered load* dihitung nilai rata-ratanya, maka dihasilkan nilai rata-rata *normalized throughput* untuk Aloha sebesar 0.020318910256410 paket/detik, nilai rata-rata *normalized throughput* untuk MACA-U adalah sebesar 0.008260416666667 paket/detik dan nilai rata-rata *normalized throughput* untuk CoopMAC-U adalah 0.008323450854701 paket/detik.

Untuk memudahkan analisa, akan diamati bagian kecil dari topologi jaringan seperti diilustrasikan pada Gambar 4.9. *Pe* pada Gambar 4.9 adalah probabilitas paket *error* yang bernilai antara $0.0000002246 \leq P_e \leq 0.0884$. Dengan kata lain, tingkat keberhasilan pengiriman paket cukup tinggi meskipun tanpa skema kooperatif. Oleh karena itu *normalized throughput* rata-rata yang dihasilkan CoopMAC-U hanya lebih tinggi 0.000063034188034 bila dibanding dengan *normalized throughput* rata-rata yang dihasilkan oleh MACA-U.

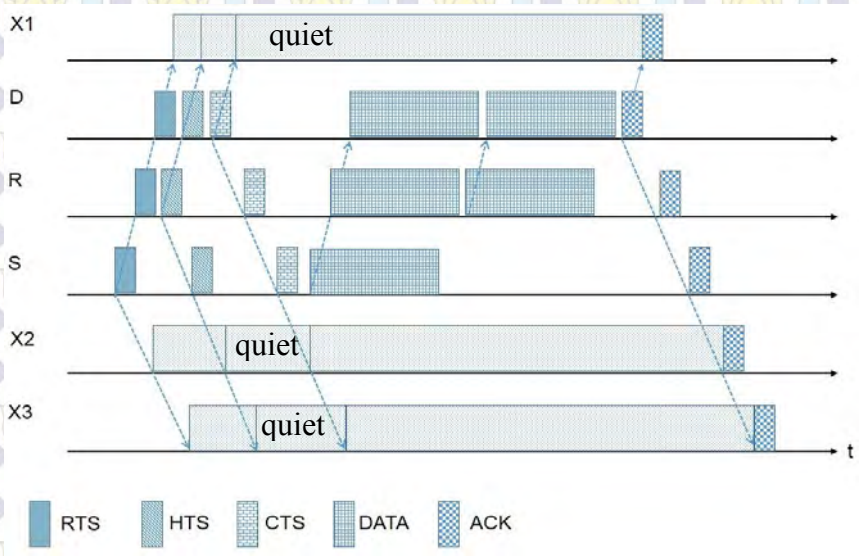


Gambar 4.9 Probabilitas Paket *Error* Pada Bagian Kecil dari Topologi

Di samping itu, modem yang digunakan dalam simulasi mempunyai kemampuan *frequency hoping* sehingga lebih tahan terhadap gangguan interferensi ataupun *jamming* yang menyebabkan protokol tanpa mekanisme reservasi kanal seperti Aloha masih tetap bisa memberikan *normalized throughput* yang tinggi.

4.7 Diagram Waktu dan Durasi Tiap Paket Pada Protokol CoopMAC-U

Untuk menggambarkan alur protokol CoopMAC-U yang telah dijelaskan pada Bab III secara lengkap, maka alur pengiriman paket kontrol sampai pengiriman paket DATA bisa dalam bentuk diagram waktu seperti pada Gambar 4.10 berikut ini.



Gambar 4.10 Diagram Waktu Alur Pengiriman Paket Pada CoopMAC-U

Durasi tiap paket tergantung dari panjang masing-masing paket yang telah dijelaskan pada Bab III sub bab 3.5 dan *data rate* yang digunakan. Pada penelitian ini digunakan dua model propagasi yang masing-masing model menggunakan modem yang berbeda, demikian pula dengan *data rate* modemnya pun juga berbeda. Durasi paket dari masing-masing modem dijelaskan pada Tabel 4.12 dan 4.13 berikut ini.

Tabel 4.12 Tabel Durasi Paket Pada Modem Kanal Thorp (*data rate = 4096* bps).

Tipe Paket	Panjang Paket (bit)	Durasi Paket (detik)
RTS	72	0,017578125
HTS	48	0,01171875
CTS	72	0,017578125
DATA	4096	1
ACK	48	0,01171875

Tabel 4.13 Tabel Durasi Paket Pada Modem Kanal Bellhop (*data rate = 720* bps).

Tipe Paket	Panjang Paket (bit)	Durasi Paket (detik)
RTS	144	0,2
HTS	96	0,13333
CTS	144	0,2
DATA	720	1
ACK	96	0,13333

Dari Tabel 4.13 di atas, bisa diketahui bahwa panjang paket kontrol pada model kanal bellhop adalah dua kali dari panjang paket kontrol pada model kanal thorp. Hal ini disebabkan karena pada model kanal bellhop, modem yang dipakai dalam simulasi mempunyai kemampuan *channel coding* dengan menggunakan viterbi *decoder* ½ dengan *constraint length* 9 sehingga terdapat penambahan satu bit untuk tiap bit yang dikirim.

BAB V
KESIMPULAN DAN SARAN

Pada bab ini, diuraikan beberapa kesimpulan yang dapat diambil dari pembahasan sebelumnya dan saran mengenai kelanjutan penelitian ini.

5.1 Kesimpulan

1. Pada simulasi menggunakan model kanal thorp, CoopMAC-U menghasilkan *normalized throughput* rata-rata 0.0130687127908 paket/detik yang mengungguli Aloha dan MACA-U yang masing-masing menghasilkan *normalized throughput* rata-rata berurutan 0.002363942625805 paket/detik dan 0.0079863012568 paket/detik.
2. Pada simulasi menggunakan model kanal bellhop, Aloha menghasilkan *normalized throughput* rata-rata sebesar 0.020318910256410 paket/detik dan mengungguli CoopMAC-U dan MACA-U yang yang masing-masing menghasilkan *normalized throughput* rata-rata berurutan 0.008323450854701 paket/detik dan 0.008260416666667 paket/detik.
3. CoopMAC-U menghasilkan *normalized throughput* rata-rata yang hanya lebih tinggi 0.000063034188034 dibandingkan dengan yang dihasilkan oleh MACA-U pada simulasi menggunakan kanal bellhop adalah disebabkan karena kemampuan modem menghasilkan paket error yang rendah (Pe<0.1) sehingga kinerja kooperatif untuk meningkatkan keberhasilan paket tidak signifikan.

5.2 Saran

1. Perlu dibuatkan model *physical layer* yang lebih mendekati kondisi laut yang sebenarnya namun tetap ramah terhadap kompleksitas perhitungan.
2. Perlu optimasi dari pengaturan waktu tunggu setiap pengiriman paket untuk mencegah paket gagal karena *state* berubah terlalu cepat atau sebaliknya, *node* terlalu lambat untuk menyadari bahwa paket telah gagal dan perlu melakukan transmisi ulang.

DAFTAR PUSTAKA

[1] wikipedia, "Wikipedia Ensiklopedia Bebas," 22 Januari 2015. [Online]. Available: https://id.wikipedia.org/wiki/Bumi.

[2] X. Guo, M. R. Frater dan M. J. Ryan, "Design of a Propagation-delay-tolerant MAC Protocol for Underwater Acoustic Sensor Networks," *Oceanic Engineering, IEEE Journal of,* vol. 34, no. 2, pp. 170-180, 2009.

[3] I. F. Akyildiz, D. Pompilo dan T. Melodia, "Underwater Acoustic Sensor Networks: Research Challenges.," *Ad hoc networks,* vol. 3, no. 3, pp. 257-279, 2005.

[4] F. T. Ulaby, Fundamentals of Applied Electromagnetics., Prentice Hall, 1995.

[5] P. C. Etter, Underwater Acoustic Modeling and Simulation, CRC Press, 2013.

[6] S. Milica dan J. Preisig, "Underwater Acoustic Communication Channels: Propagation Models and Statistical Characterization," *Communications Magazine, IEEE,* vol. 47, pp. 84-89, 2009.

[7] N. Chirdchoo, W.-S. Soh dan K. C. Chua, "RIPT: A Receiver-Initiated Reservation-Based Protocol for Underwater Acoustic Networks," *Selected Areas in Communications, IEEE Journal on,* vol. 26, no. 9, pp. 1744-1753, 2008.

[8] H.-H. Ng, W.-S. Soh dan M. Motani, "An Underwater Acoustic MAC Protocol Using Reverse Opportunistic Packet Appending," *Computer Networks,* vol. 57, no. 14, pp. 2733-2751, 2013.

[9] H.-H. Ng, W.-S. Soh dan M. Motani, "A Bidirectional-concurrent MAC Protocol with Packet Bursting for Underwater Acoustic Networks," *Oceanic Engineering, IEEE Journal of,* vol. 38, no. 3, pp. 547-565, 2013.

[10] M. Vajapeyam, S. Vedantam, U. Mitra, J. C. Preisig dan M. Stojanovic, "Distributed Space–Time Cooperative Schemes for Underwater Acoustic Communications," *Oceanic Engineering, IEEE Journal of,* vol. 33, no. 4, pp. 489-501, 2008.

[11] Z. Han, Y. L. Sun dan H. Shi, "Cooperative Transmission for Underwater Acoustic Communications," dalam *Communications, 2008. ICC'08. IEEE International Conference on*, 2008.

[12] Y. W. P. Hong, C. C. J. Kuo dan W.-J. Huang, Cooperative Communications and Networking, Springer, 2010.

[13] T. Cinar dan M. B. Orencik, "An Underwater Acoustic Channel Model Usin Ray Tracing in ns-2," dalam *Wireless Days (WD), 2009 2nd IFIP*, Paris, 2009.

[14] A. F. Harris III dan M. Zorzi, "Modelling the Underwater Acoustic Channel in ns2," dalam *{Proceedings of the 2nd international conference on Performance evaluation methodologies and tools*, 2007.

[15] A. Sehgal, I. Tumar dan J. Schonwalder, "Aquatools: An Underwater Acoustic Networking Simulation Toolkit," dalam *IEEE, Oceans, Sydney, Australia May*, 2010.

[16] R. J. Urick, Principles of Underwater Sound, McGraw-Hill, 1983.

[17] I. F. Akyildiz dan M. C. Vuran, Wireless Sensor Networks, John Wiley & Sons, 2010.

[18] L. Xavier, An Introduction to Underwater Acoustics: Principles and Applications, Springer Science & Business Media, 2002.

[19] P. Wang, F. Wei, L. Zhang dan V. O. Li, "Asynchronous Cooperative Transmission in Underwater Acoustic Networks," dalam *Underwater Technology (UT), 2011 IEEE Symposium on and 2011 Workshop on Scientific Use of Submarine Cables and Related Technologies (SSC)*, 2011.

[20] W. W. Dargie dan C. Poellabauer, Fundamentals of Wireless Sensor Networks, John Wiley & Sons, 2010.

[21] P. Liu, Z. Tao, S. Narayanan, T. Korakis dan S. S. Panwar, "CoopMAC : A Cooperative MAC for Wireless LANs," *Selected Areas in Communications, IEEE Journal on,* vol. 25, no. 2, pp. 340-354, 2007.

[22] "NS-3 A Discrete-Event Network Simulator," [Online]. Available: https://www.nsnam.org/. [Diakses 1 12 2015].

[23] J. G. Proakis dan M. Salehi, Digital Communication, McGraw-Hill Education, 2007.

[24] Wikipedia, "The Free Encyclopedia," 29 July 2015. [Online]. Available: https://en.wikipedia.org/wiki/Bit_error_rate. [Diakses 12 12 2015].

[25] Ocean Acoustic Library, "Ocean Acoustics Library," [Online]. Available: http://oalib.hlsresearch.com/Rays/index.html. [Diakses 31 Maret 2015].

[26] National Centers For Environmental Information, [Online]. Available: https://www.nodc.noaa.gov/OC5/woa13/woa13data.html. [Diakses 12 Desember 2015].

[27] Google Maps, "Google Maps," [Online]. Available: https://www.google.co.id/maps/place/9%C2%B007'30.0%22S+133%C2%B007'30.0%22E/@-8.1677651,133.4501884,7.5z/data=!4m2!3m1!1s0x0:0x0?hl=en. [Diakses 12 12 2015].

[28] N. Parrish, S. Roy, W. L. Fox dan P. Arabshahi, "Rate-Range for an FH-FSK Acoustic Modem," dalam *Proceedings of the second workshop on Underwater networks*, 2007.

[29] N. Parrish, L. Tracy, S. Roy, P. Arabshahi dan W. L. Fox, "System Design Considerations for Undersea Networks : Link and Multiple Access

Protocols," *Selected Areas in Communications, IEEE Journal on,* vol. 26, no. 9, pp. 1720-1730, 2008.

[30] P. Frenger, P. Orten dan T. Ottosson, "Convolutional Codes with Optimum Distance Spectrum," *Communications Letters, IEEE,* vol. 3, no. 11, pp. 317-319, 1999.

[31] H.-H. Ng, W.-S. Soh dan M. Motani, "MACA-U : A Media Access Protocol for Underwater Acoustic Networks," dalam *Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008. IEEE*, 2008.

LAMPIRAN A

1. Instalasi NS3

NS3 (*Network Simulator* 3)adalah salah satu *discrete event simulator* yang berlisensi GNU GPLv2 yang banyak dipakai oleh kalangan akademik untuk mensimulasikan protokol jaringan. *Discrete event simulator* bekerja dengan cara memodelkan kerja sistem sebagai urutan *event* diskrit dalam waktu. Setiap *event* atau kejadian yang terjadi, ditandai dengan perubahan *state* dan diantara *event* yang berbeda, simulator bisa langsung melompati waktu tanpa perlu menunggu waktu sampai *event* selanjutnya terjadi. Dengan demikian, waktu simulasi bisa lebih efisien.

Untuk mendapatkan program NS3, bisa langsung mengunduhnya dari situs web dengan *url* https://www.nsnam.org/ . Saat artikel ini diketik, versi terbaru untuk *stable release* yang tersedia adalah versi ns-3.24.1. Program NS3 tersedia dalam bentuk *source code* sehingga baru bisa digunakan kalau sudah dilakukan proses kompilasi. Untuk menginstal NS3 disarankan untuk menggunakan sistem operasi linux dengan distribusi sesuai dengan preferensi pengguna. Pada artikel ini akan dijelaskan proses instalasi NS3 menggunakan Linux Ubuntu 14.04 LTS.

Langkah pertama adalah menginstal paket-paket pendukung agar NS3 dapat berjalan. Dengan menggunakan *terminal* dan *root privelege*, ketikkan perintah berikut ini :

```
# apt-get install gcc g++ python
# apt-get install qt4-dev-tools libqt4-dev
# apt-get install mercurial
# apt-get install bzr
# apt-get install cmake libc6-dev libc6-dev-i386 g++-multilib
# apt-get install gdb valgrind
# apt-get install gsl-bin libgsl0-dev libgsl0ldbl
# apt-get install flex bison libfl-dev
# apt-get install tcpdump
# apt-get install sqlite sqlite3 libsqlite3-dev
```

```
# apt-get install libxml2 libxml2-dev
# apt-get install libgtk2.0-0 libgtk2.0-dev
# apt-get install vtun lxc
# apt-get install uncrustify
# apt-get install doxygen graphviz imagemagick
# apt-get install texlive texlive-extra-utils texlive-latex-
extra texlive-font-utils dvipng
# apt-get install python-sphinx dia
#  apt-get  install  python-pygraphviz  python-kiwi  python-
pygoocanvas libgoocanvas-dev
# apt-get install libboost-signals-dev libboost-filesystem-
dev
#  apt-get  install  openmpi-bin  openmpi-common  openmpi-doc
libopenmpi-dev
```

Setelah semua paket pendukung di atas selesai diinstal, langkah selanjutnya adalah mengekstrak ns-allinone-3.24.1.tar.bz2 dengan perintah :

```
#  tar xjf ns-allinone-3.24.1.tar.bz2
```

Kemudian masuk ke direktori `/ns-allinone-3.24.1` dengan perintah :

```
# cd /ns-allinone-3.24.1
```

Langkah yang terakhir adalah mem-*build* kode sumbernya dengan perintah :

```
# ./build.py
```

Tunggu sampai proses instalasi selesai.

2. Menjalankan Program Simulasi

Untuk menjalankan simulasi pada NS3, skenario simulasi dan topologi jaringan disimpan dalam *file* yang berlokasi di `/ns-allinone-3.24.1/ns-3.24.1/scratch/` sedangkan *file* algoritma protokol, model kanal dan

pendukung lainnya disimpan pada direktori `/ns-allinone-3.24.1/ns-3.24.1/src/uan/model/` .

Pada buku thesis ini dilakukan simulasi tiga protokol yang berbeda, yaitu Aloha, MACA-U dan CoopMAC-U. File yang digunakan dalam simulasi tersimpan dalam direktori sebagai berikut :

`/ns-allinone-3.24.1/ns-3.24.1/scratch/ :`

- `coop.cc`
- `coop.h`
- `aloha.cc`
- `aloha.h`
- `macau.cc`
- `macau.h`

`/ns-allinone-3.24.1/ns-3.24.1/src/uan/model/ :`

- `uan-mac-aloha.cc`
- `uan-mac-aloha.h`
- `uan-macau.cc`
- `uan-macau.h`
- `uan-header-macau.cc`
- `uan-header-macah.h`
- `uan-mac-coop.cc`
- `uan-mac-coop.h`
- `uan-header-addr.cc`
- `uan-header-addr.h`
- `uan-header-coop.cc`
- `uan-header-coop.h`

Dari tiga protokol yang simulasikan, hanya Aloha saja yang sudah disertakan dalam paket awal instalasi NS3. Oleh karena itu, agar NS3 bisa mengenali protokol baru yang ditambahkan, yaitu MACA-U dan CoopMAC-U maka file algoritma masing-masing protokol harus didaftarkan dalam *file* `wscript`. File ini tersimpan pada direktori `/ns-allinone-3.24.1/ns-3.24.1/src/uan/` . Untuk mengeditnya bisa dengan perintah berikut ini :

```
# gedit /ns-allinone-3.24.1/ns-3.24.1/src/uan/wscript
```

Pada bagian `module.source = [ ]`, tambahkan baris perintah berikut ini di dalam kurung kotak :

```
'model/uan-macau.cc',
'model/uan-header-macau.cc',
'model/uan-mac-coop.cc',
'model/uan-header-addr.cc',
'model/uan-header-coop.cc',
```

Dan pada bagian `headers.source = [ ]`, tambahkan baris perintah berikut ini di dalam kurung kotak :

```
'model/uan-macau.h',
'model/uan-header-macau.h',
'model/uan-mac-coop.h',
'model/uan-header-addr.h',
'model/uan-header-coop.h',
```

Kemudian simpan file wscript dan tutup program editor.

Untuk mejalankan simulasi, masuk ke folder `/ns-allinone-3.24.1/ns-3.24.1/` dimana *file* `waf` berada. Berikut ini adalah contoh perintah untuk menjalankan simulasi aloha :

```
# ./waf --run aloha
```

Atau bila dikehendaki untuk menyimpan semua tampilan teks yang dihasilkan selama proses simulasi ke dalam sebuah file, bisa ditambahkan instruksi berikut :

```
# ./waf --run aloha > namafile.txt 2>&1
```

LAMPIRAN B

1. Cara Compile Acoustic Toolbox

Acoustic Toolbox yang tersedia di http://oalib.hlsresearch.com/Rays/index.html adalah program yang ditulis dengan menggunakan bahasa FORTRAN. Oleh karena itu, untuk bisa mengkompilasi *source code* agar bisa dijalankan perlu kompiler FORTRAN terinstal di komputer. Bila belum tersedia bahasa FORTRAN di komputer, bisa dilakukan instalasi dengan perintah :

```
# apt-get install gfortran
```

Setelah `gfortran` terinstall, ekstrak file `at` dan masuk folder `/misc`. Edit *file* Makefile dengan perintah :

```
# nano Makefile
```

Tambahkan instruksi `FC=gfortran` pada baris kedua sehingga nantinya di awal baris menjadi seperti berikut ini :

```
AR = ar
FC=gfortran
# Intel compiler needed the following definition for the archiver:
# AR = xiar
```

Simpan *file* dan tutup editor. Lanjutkan dengan mengetikkan perintah

```
# make clean
# make install
```

Kembali ke folder utama (`/at`) dan masuk ke folder `/tslib`. Edit *file* Makefile dengan perintah :

```
# nano Makefile
```

Tambahkan instruksi `FC=gfortran` pada baris kedua sehingga nantinya di awal baris menjadi seperti berikut ini :

```
AR = ar
FC=gfortran
# Intel compiler needed the following definition for the archiver:
# AR = xiar
```

Kembali ke folder utama (`/at`) dan ketikkan perintah :

```
# make install
```

Tunggu proses instalasi sampai muncul tulisan `Acoustic Toolbox Installed.`

2. Cara Menjalankan Bellhop

Dari *folder* `/at`, masuk ke *folder* `/Bellhop` dengan perintah :

```
# cd Bellhop
```

Untuk menjalankan simulator bisa dengan perintah :

```
# ./Bellhop.exe MunkB_Arr
```

Dimana `MunkB_Arr` adalah contoh *environment file* dalam format `.env` yang tersimpan dan bisa di-*copy* dari folder `/at/test/Munk`.

LAMPIRAN C

1. Source Code sspmaker.m

```
clear all;
close all;
clc;
%woa13_decav_s00mn04v2 latitude=-9.125   longitude 133.125
sal=[34.198 33.751  33.752  33.751  33.75   33.754  33.752  33.754
33.961  34.083  34.087  34.103  34.141  34.237  34.312  34.385
34.405  34.433  34.466  34.489  34.499  34.5    34.54   34.565
34.573];
%woa13_decav_t00mn04v2
temp=[28.134    28.137  28.097  28.048  27.982  27.939  27.808
27.673  27.557  27.442  27.272  27.14   26.968  26.756  26.528
25.992  25.347  24.585  23.678  22.962  22.339  20.095  17.763
16.242  15.628];
depth=[0    5   10  15  20  25  30  35  40  45  50  55  60  65  70
75  80  85  90  95  100 125 150 175 200];

ssp=mackenzie(sal, temp, depth);
plot(depth,ssp,'LineWidth',2);
ylabel('Kecepatan Perambatan Suara di Air (meter/detik)');
xlabel('Kedalaman Laut (meter)');
grid on;
```

2. Source Code fungsi mackenzie.m

```
function [ssp]=mackenzie(salinity, temperature, depth)
S=salinity;
T=temperature;
D=depth;

ssp= 1448.96 + (4.591.*T)-...
(5.304 * (10^-2).*(T.^2))+...
(2.374 * (10^-4).*(T.^3))+...
(1.340 *(S-35)) +...
(1.630 * (10^-2).*D) +...
(1.675 * (10^-7).*(D.^2)) -...
(1.025 * (10^-2).*T.*(S - 35)) -...
(7.139 * (10^-13).*T.*(D.^3));
```

[*Halaman ini sengaja dikosongkan*.]

LAMPIRAN D

1. Environment File arafura.env

```
'Arafura profile'     ! TITLE
25720.0           ! FREQ (Hz)
1                 ! NMEDIA
'SVF'             ! SSPOPT (Analytic or C-linear interpolation)
51  0.0  200.0   ! DEPTH of bottom (m)
0     1540.68    /
5     1540.29    /
10    1540.29    /
15    1540.26    /
20    1540.19    /
25    1540.18    /
30    1539.97    /
35    1539.76    /
40    1539.8     /
45    1539.75    /
50    1539.45    /
55    1539.26    /
60    1538.99    /
65    1538.69    /
70    1538.33    /
75    1537.25    /
80    1535.84    /
85    1534.12    /
90    1532.01    /
95    1530.32    /
100   1528.81    /
125   1523.22    /
150   1517.04    /
175   1512.91    /
200   1511.43    /
'A' 0.0
 200.0  1600.00 0.0 1.0 /
1                 ! NSD
100.0 /           ! SD(1:NSD)  (m)
1                 ! NRD
100.0 /           ! RD(1:NRD)  (m)
1                 ! NR
0 1.2  /          ! R(1:NR )  (km)
'A'               ! 'R/C/I/S'
5500              ! NBeams
```

```
-60.0 60.0 /          ! ALPHA1,2 (degrees)
0.0  200.0 2.2        ! STEP (m), ZBOX (m), RBOX (km)
```

Dimana :

FREQZ : adalah frekuensi kerja dalam Hz

NSD : *number of source depth*

SD : *source depth* dalam meter

NRD : *number of receiver depth*

RD : *receiver depth* dalam meter

NR : *number of receiver*

R : *range*

R/C/I/S : *ray, coherent, incohorent,semi coherent*. A=*arrival*.

NBeams : *number of beams*

ALPHA : sudut pancaran

ZBOX (m) : kedalaman

RBOX(km) : *range*.

DAFTAR RIWAYAT HIDUP

Penulis bernama lengkap Muhammad Syirajuddin S., anak kedua dari tiga bersaudara. Dilahirkan oleh pasangan Sudja'i, S.Ag. dan Rosyidah di Mojokerto pada tanggal 17 Oktober 1985. Penulis lulus dari sekolah dasar MI Miftahul Ulum Pandanarum pada tahun 1997 dan meneruskan ke MTS Al-Ghozalie Pungging sampai tahun 2000. Kemudian penulis melanjutkan pendidikan di jenjang sekolah menengah atas di SMUN 1 Mojosari dan lulus pada tahun2003. Selanjutnya, penulis meneruskan pendidikan di Diploma III Politeknik Elektronika Negeri Surabaya jurusan Teknik Telekomunikasi dan lulus tahun 2006.

Penulis sempat bekerja di perusahan penyedia layanan telekomunikasi satelit dan terestrial di Cikarang mulai tahun 2006 sampai dengan 2011. Di tengah kesibukan bekerja, penulis juga berjuang melanjutkan studi di program ekstensi (lintas jalur) Teknik Elektro Universitas Indonesia mulai tahun 2007 dan lulus pada tahun 2010. Penulis memutuskan untuk kembali ke bangku kuliah untuk meneruskan studi di jenjang strata 2 di Institut Teknologi Sepuluh Nopember mulai tahun 2014.

Penulis juga pernah tinggal dan melakukan penelitian di Kumamoto University selama satu semester pada *spring semester* 2015 di bawah bimbingan Prof. Tsuyoshi Usagawa dan Dr. Yoshifumi Chisaki.