Integração e Validação de um Módulo Realista para simulação de WiMAX no Network Simulator 2 com Modulação e Codificação Adaptativa

Tiago Pedroso da Cruz de Andrade

Projeto Final de Graduação apresentada
ao
Instituto de Computação
da
Universidade de Campinas
para
a conclusão da Graduação.

Orientador: Prof. Dr. Nelson Luís Saldanha da Fonseca

Durante o desenvolvimento deste trabalho o autor recebeu auxílio financeiro do PIBIC

Campinas, 04 de dezembro de 2009

Integração e Validação de um Módulo Realista para simulação de WiMAX no Network Simulator 2 com Modulação e Codificação Adaptativa

Este exemplar corresponde à redação
final do Projeto Final de Graduação devidamente corrigida
por Tiago Pedroso da Cruz de Andrade.

Orientador e Revisor:

- Prof. Dr. Nelson Luís Saldanha da Fonseca - IC/Unicamp
- Prof. Dr. Edmundo R. M. Madeira - IC/Unicamp

Agradecimentos

Eu gostaria de agradecer muitas pessoas sem as quais esse trabalho não seria possível. Primeiro, gostaria de mostras meus profundos agradecimentos ao meu orientador Prof. Dr. Nelson Luis Saldanha da Fonseca o qual tornou possível a realização desse trabalho. Também gostaria de agradecer aos alunos de mestrado Flávio Adalberto Kubota e Pedro Henrique Gomes por suas constantes ajudas nos mais variados impecílhos que encontrei pelo caminho.

Depois, gostaria de agradecer aos meus amigos e colegas do Laboratório de Redes de Computador e da graduação, com os quais convivi durante quatro anos.

Finalmente, mais não menos importante. gostaria de agradecer aos meus pais que possibilitaram que eu pudesse estudar em uma universidade de alto padrão como a Unicamp e que me incentivaram desde o início para eu dar sempre o meu melhor.

Resumo

As simulações de ambientes bem comportado não reflete o meio ruidoso de uma rede sem fio. Diversos fenômenos físicos influênciam a transferência de pacotes e quando esses fatores não são considerados numa simulação, os resultados podem não representar um cenário real.

Esse trabalho apresenta uma implementação para trazer mais realismo numa simulação de redes IEEE 802.16 com o auxílio da Modulação e Codificação Adaptativa, para o qual um algoritmo foi implementado.

Com essa implementação, poderão ser feitos novos estudos que levam em consideração a qualidade do canal e a perda de pacotes.

Palavras-chave: WiMAX, AMC, Modelo de Propagão, ns-2.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

O acesso sem fio em banda larga (*Broadband Wireless Access* - BWA) tem recebido grande destaque tanto no meio comercial como no meio acadêmico. A tecnologia BWA é capaz de prover altas taxas de transferência de dados, alto nível de escalabilidade e baixo custo de instalação e manutenção para o acesso residencial e comercial à Internet.

O padrão IEEE 802.16 [2], frequentemente referenciado como WiMAX(*Worldwide Interoperability for Microwave Access Forum*) vem sendo desenvolvido com a finalidade de padronizar a tecnologia BWA. O padrão define a interface aérea e o protocolo de acesso ao meio para redes metropolitanas sem fio fornecendo altas taxas de transmissão para o acesso em banda larga à Internet.

Se por um lado o canal sem fio permite uma difusão maior do acesso à Internet do que as redes cabeadas, por outro lado está sujeito a uma série de fenômenos que prejudicam a transmissão e a a recepção de conteúdo. Fenômenos físicos como o *path loss* ou *multipath* têm grande influência na qualidade de acesso. Desta forma, é importante levar em consideração as imperfeições do meio.

A simulação é uma ferramenta essencial na pesquisa de redes de computadores, pois permite o desenvolvimento e a análise de novos protocolos e mecanismos em topologias complexas, sem a necessidade de implementá-las fisicamente. Dentre as ferramentas disponíveis para simulação de redes de computadores, o *Network Simulator* (ns-2) [1] é a ferramenta de maior popularidade entre os pesquisadores.

Diversos módulos [5] [7] [8] foram propostos para simulação de redes IEEE 802.16 usando o *ns-2*. Entre eles está o módulo implementado pelo *National Institute of Standards and Technology* (NIST) [8].

Este trabalho apresenta uma extensão para o módulo desenvolvido pelo NIST com uma camada física OFDM (*Orthogonal Frequency Division Multiplexing*) e um modelo de canal realista. A extensão do módulo implementa um modelo de canal que reflete o ambiente suburbano das redes WiMAX. A implementação conta com dois modelos de erros que simula a perda da qualidade da conexão assim que o a distância entre duas estações aumenta. Além disso, vários mecânismos de Modulação e Codificação Adaptativa foram estudados, levando à implementação de um algoritmo para melhorar a eficiência das simulações.

A implementação tem seu foco na representação mais detalhada da realidade, permitindo que diversos estudos sobre a camada de acesso ao meio seja realizada. Acredita-se que a extensão apresentada é de extrema valia para pesquisadores da área e que deverá ter um impacto positivo

em investigações de novos problemas em redes WiMAX.

Capítulo 2

Visão Geral do WiMAX

Antes de começarmos a falar sobre o trabalho realizado, devemos entender como a tecnologia WiMAX funciona.

Nesse capítulo, explicaremos os conceitos do WiMAX.

Primeiro, resumiremos as atividades do grupo IEEE 802.16 e sua relação com o WiMAX. Depois, discutiremos as características principais do WiMAX e descreveremos brevemente as camadas física e MAC.

2.1 IEEE 802.16 e WiMAX

O grupo IEEE 802.16 foi formado em 1998 para desenvolver um padrão para a interface aérea do wireless broadband. O foco inicial do grupo era o desenvolvimento de um sistema wireless broadband ponto-multiponto baseado em LOS para operações na frequência de 10GHz - 66GHz. O padrão resultante (o padrão original 802.16 terminado em dezembro de 2001) foi baseado na camada física single-carrier com uma camada MAC com multiplexação por divisão de tempo. Muitos dos conceitos relacionados para a camada MAC foram adotados do padrão DOCSIS.

O grupo IEEE 802.16 subsequentemente produziu o 802.16a, uma alteração do padrão, para incluir aplicações NLOS na frequência 2GHz - 11GHz, usando uma camada física baseada no OFDM. Adições na camada MAC, tais como suporte para OFDMA (*Orthogonal Frequency Division Multiplexing Access*) também foi incluido. Revisões adicionais resultaram em um novo padrão em 2004, chamado IEEE 802.16-2004, no qual substituiu todas as versões anteriores e formou a base para a primeira solução WiMAX. Esse padrão apenas solucionava as questões de aplicações fixas. Em dezembro de 2005, o grupo IEEE 802.16 completou e aprovou o padrão IEEE 802.16e-2005, uma alteração do padrão IEEE 802.16-2004 que adicionou suporte à mobilidade.

As características basicas dos vários padrões IEEE 802.16 estão resumidos na Tabela 2.1.

2.1.1 IEEE 802.16-2004

A arquitetura de uma rede que utiliza o padrão IEEE 802.16-2004 [2] possui dois elementos principais: *Base Station* (BS) e *Subscriber Station* (SS). A BS realiza a comunicação entre a rede sem fio e a rede núcleo e suporta interfaces IP, ATM, Ethernet e E1/T1. A SS fornece ao usuário acesso à rede núcleo através do estabelecimento de conexões com a BS em uma topologia

	802.16	**802.16-2004**	**802.16e-2005**
Status	Completado em Dezembro de 2001	Completado em Junho de 2004	Completado em Dezembro de 2005
Frequência	10GHz - 66GHz	2GHz - 11GHz	2GHZ - 11GHz para fixo 2GHz - 6GHz para móvel
Aplicação	LOS fixo	NLOS fixo	NLOS fixo e móvel
Arquitetura MAC	Ponto-Multiponto, Mesh	Ponto-Multiponto, Mesh	Ponto-Multiponto, Mesh
Esquema de Transmissão	Somente single-carrier	single-carrier, 256 OFDM ou 2048 OFDM	single-carrier, 256 OFDM ou OFDM escalável
Modulação	QPSK, 16QAM, 64 QAM	QPSK, 16QAM, 64 QAM	QPSK, 16QAM, 64 QAM
Taxa de Transmissão	32Mbps - 134Mbps	1Mbps - 75Mbps	1Mbps - 75Mbps
Multiplexação	Rajada TDM/TDMA	Rajada TDM/TDMA/ OFDMA	Rajada TDM/TDMA/ OFDMA
Duplexação	TDD e FDD	TDD e FDD	TDD e FDD

Tabela 2.1: Resumo dos padrões IEEE 802.16

Ponto-Multiponto (PMP). O padrão ainda permite a implementação de uma topologia *Mesh* (opcional). A principal diferença entre as topologias PMP e *Mesh* está no fato de que em uma rede PMP o tráfego flui apenas entre a BS e as SSs, enquanto que no modo *Mesh*, o tráfego pode ser roteado através das SSs e pode ocorrer diretamente entre duas SSs.

O padrão IEEE 802.16-2004 [2] define as especificações das camadas MAC (*Medium Access Control*) e física para redes BWA. A camada física opera em um formato de *frames*, os quais são subdivididos em intervalos de tempo chamados *slots físicos*. Em cada *frame* há um *subframe downlink* e um *subframe uplink*. O *subframe downlink* é utilizado pela BS para a transmissão de dados e de informações de controle para as SSs. O *subframe uplink* é compartilhado entre todas as SSs para transmissões que têm como destino a BS.

O padrão [2] define várias camadas físicas para os diferentes ambientes. Entre elas esta a camada física WirelessMAN-OFDM (Orthogonal Frequency Division Multiplexing). A idéia básica do OFDM consiste em dividir os bits em diversos streams de taxas menores que serão transmitidos por subcanais paralelos. Como consequência, temos que a duração de cada símbolo é maior, tornando o sinal menos sinsível a ruídos, à multiplicidade de caminhos e a interferência intersimbólica.

Para a camada física WirelessMAN-OFDM, o padrão [2] define sete esquemas de modulação e codificação como mostrado na Tabela 2.2. Cada modulação consegue transportar uma quantidade de símbolos e cada tipo de codificação consegue corrigir uma certa quantidade de bits. Dessa forma, cada esquema de modulação e codificação consegue oferecer uma taxa de dados diferente.

O AMC é uma técnica utilizada em diversas redes celulares e em redes sem fio convencionais, como o WI-FI, que visa adaptar a modulação e a codificação de acordo com as condições do canal, para cada usuário obter uma maior taxa de transferência de dados e menor taxa de erro de pacotes. Nesse mecanismo, o transmissor transmite os dados na maior taxa possível quando as condições do canal são boas e transmite em taxas menores quando as condições estão ruins, evitando as perdas de pacotes.

MCS	FEC code rate
BPSK	1/2
QPSK	1/2 ou 3/4
16QAM	1/2 ou 3/4
64QAM	2/3 ou 3/4

Tabela 2.2: Modulações e Codificações

O padrão [2] recomenda utilizar como medida de qualidade de sinal a relação sinal-ruído (Signal to Noise Ratio - SNR), mas deixa em aberto os valores ideais para a troca de modulação.

O padrão [2] permite dois modos de acesso ao meio físico: duplexação por divisão de freqüência (*Frequency Division Duplexing* - FDD) e duplexação por divisão de tempo (*Time Division Duplexing* - TDD). No modo FDD, os canais *downlink* e *uplink* operam simultaneamente em freqüências diferentes. No modo TDD, os *subframes uplink* e *downlink* compartilham a mesma freqüência, logo, não é possível realizar transmissões simultâneas nos dois sentidos. Cada *frame* TDD tem um *subframe downlink* seguido por um *subframe uplink*.

Na camada física, para verificar se um pacote foi recebido com uma potência suficiente, utiliza-se o RSSI (*Received Signal Strengh Indicator*) e o SNR (*Signal to Noise Ratio*). O SNR é a relação entre a potência recebida e o ruído mais a interferência do canal.

2.1.2 IEEE 802.16-2005

O adendo IEEE 802.16-2005 trouxe a característica, talvez a mais atrativa da tecnologia WiMAX, mobilidade. Essa mobilidade deve vir associada a grande largura de banda e vasta área de cobertura, para isto o padrão utiliza tecnologias como a SOFDMA, AAS (Adaptive Antenna System) e MIMO (Multiple Input Multiple Output). Porém, há um novo problema a ser enfrentado: o processo de handover.

O handover é o processo no qual uma estação móvel (MS - mobile station) está se deslocando para uma célula próxima, então ela deixa a conexão com a BS da antiga célula e estabelece uma nova conexão com a BS da nova célula. Todo este processo deve ser imperceptível para o usuário, a conexão com a Internet deve ser contínua, ou seja, o usuário deve ser capaz de se movimentar ser perder a conexão. Existem três tipos de handover: HHO (Hard Handover), MDHO (Macro Diversity Handover) e FBSS (Fast Base Station Switching).

O HHO é definido como obrigatório, enquanto os outros são opcionais. Ele funciona da seguinte maneira, primeiramente a MS tenta descobrir a topologia da rede onde está e procura por BSs próximas que ofereçam sinal de qualidade, e irá trocar essas informações com a sua atual BS. Dentre essas BSs a BS atual irá escolher uma que possua a melhor qualidade de sinal, que será chamada de BS alvo. A MS irá enviar uma mensagem à atual BS para informar que deseja iniciar o processo de handover. A antiga BS irá responder sua requisição, e então a MS deixa a conexão com a antiga BS e faz conexão com a BS alvo. Todo esse processo foi apenas para reconhecimento da rede, o processo de handover será agora iniciado e ele possui 6 fases: re-seleção de células, decisão e iniciação do handover, sincronização, ranging, término do contexto da MS e cancelamento do handover.

As fases de reseleção de célula, decisão e iniciação de handover podem ser encurtadas, já que

consiste nos mesmos procedimentos realizados pela etapa de reconhecimento da rede. Porém a decisão de qual BS será a BS alvo pode vir também da MS. Depois, ocorre a sincronização com o canal de downlink e uplink da nova BS, na fase de sincronização. Na fase ranging ocorre autenticação, autorização e todos os processos para se registrar a MS em sua nova BS, a fim de evitar problemas de segurança. E depois, na fase de término do contexto da MS, a antiga BS elimina todos os registros da MS. A fase de cancelamento é opcional, a MS tem liberdade de cancelar o processo de handover a qualquer momento.

2.2 A Camada Física do WiMAX

A camada física do WiMAX é baseada nos padrões IEEE 802.16-2004 e IEEE 802.16-2005 e foi concebida com muita influência do WI-FI, especialmente do padrão IEEE 802.11a.

O conjunto de padrões IEEE 802.16 define quatro camadas físicas, nas quais podem ser usadas com a camada MAC para desenvolver um sistema broadband wireless.

As camadas físicas definidas são:

- WirelessMAN-SC
- WirelessMAN-SCa
- WirelessMAN-OFDM
- WirelessMAN-OFDMA

2.2.1 A base do OFDM

O OFDM pertence a família de esquemas de transmissão chamada de *modulação por multi-carriers*, no qual é baseado na idéia de dividir um dado fluxo de dados em muitos fluxos menores paralelos e modular cada um em carriers separadas - chamadas de subcarriers. O esquema de modulação por multicarriers elimina ou minimiza a interferância intersimbólica (ISI) marcando o tempo do símbolo grande o suficiente para que o atrazo induzido no canal seja uma fração insignificante da duração do mesmo. Portanto, nos sistemas de alta taxa de dados em que a duração do símbolo é pequena, dividir o fluxo de dados em nuitos fluxos paralelos aumenta a duração do símbolo de cada fluxo tal que a amplitude do atrazo é somente uma pequena fração da duração do símbolo.

Para eliminar completamente a ISI, intervalos guarda são usados entre os símbolos OFDM. Fazendo o intervalo guarda maior que o atrazo de multipath, a ISI pode ser completamente eliminada. Adicionar um intervalo guarda, no entanto, implica em gasto de energia e uma degradação na eficiência do comprimento da banda.

2.2.2 Estrutura do Quadro e Slot Físico

A camada física também é responsável pela alocação dos slots físicos e enquadramento. O tempo de frequência mínimo que pode ser alocado por um sistema WiMAX para um dado canal

é chamada de slot. Cada slot consite de um subcanal sobre um, dois ou três símbolos OFDM, dependendo do esquema de divisão de subcanais utilizado. Uma série contínua de slots atribuída a um usuário é chamada de região de dados do usuário; algoritmos de escalonamento poderia alocar regiões de dados para diferentes usuários, baseado na demanda, QoS e condições do canal.

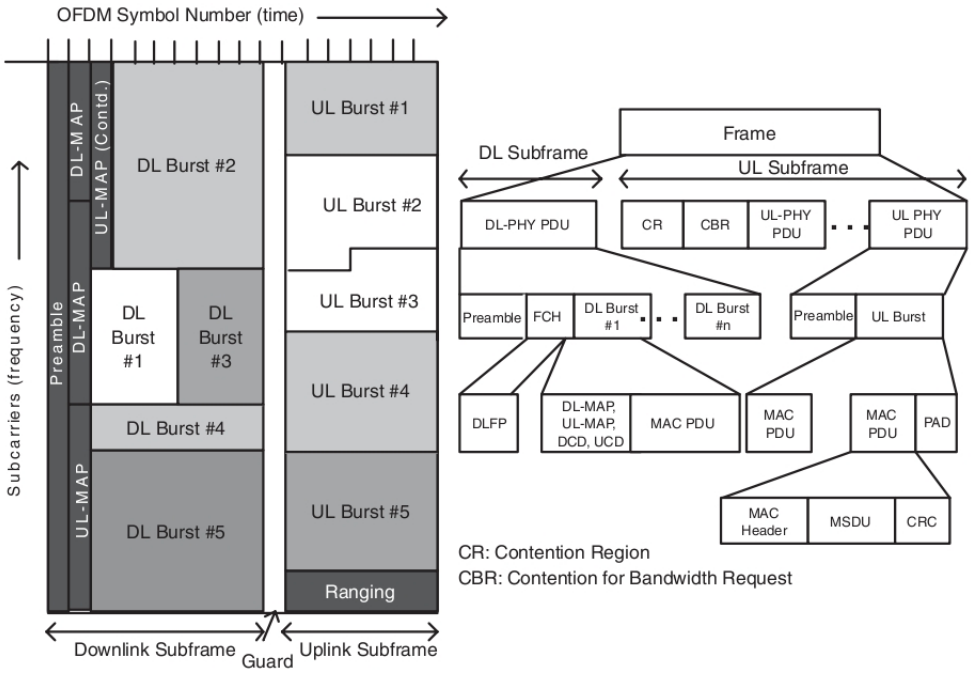

Figura 2.1: Estrutura do Quadro usando TDD para WiMAX móvel

A figura 2.1 mostra um quadro OFDM e OFDMA operando no modo TDD. O quadro é dividido em dois subframes: um subframe de downlink seguido por um de uplink depois de um pequeno intervalo de guarda. A relação de subframes downlink e uplink deve variar de 3:1 ou para 1:1 para suportar diferentes tipos de profile.

O WiMAX também suporta FDD. Nesse caso, a estrutura do quadro é a mesma exceto pelo fato que ambos, downlink e uplink, são transmitidos simultaneamente sobre diferentes carriers. Alguns dos sistemas WiMax fixos atuais utilizam o FDD.

A maioria dos desenvolvimentos WiMAX, no entanto, utilizam o modo TDD por causa de suas vantagens. Ele permite uma maior flexibilidade no compartilhamento da banda entre o uplink e o downlink, não requer paridade espectral e tem um projeto de transmissão simples.

O WiMAX é bastante flexível em termos de como multiplos usuários e pacotes são multiplexados em um único quadro. Um único quadro downlink pode contêr multiplos bursts de vários tamanhos e tipos de dados de muitos usuários.

O subframe uplink é feito de muitos bursts de uplinks de diferentes usuários.

2.3 A Camada MAC do WiMAX

A tarefa principal da camada MAC (*Medium Access Control layer*) é prover uma interface entre a camada de transporte e a camada física. A camada física pega os pacotes da camada acima (esses pacotes são chamados de MAC service data units (MSDUs)) e os organiza dentro

do MAC protocol data units (MPDUs) para a transmissão. No recebimento da transmissão, a camada MAC faz o reverso.

Ela é orientada a conexão. Cada conexão é identificada por um identificador CID (*Connection Identifier*) de 16 bits e cada SS tem um endereço MAC único que a identifica e é utilizado para registrá-la e autenticá-la na rede. Todo o tráfego, incluindo o tráfego não orientado a conexão, é mapeado para uma conexão. Além do gerenciamento das conexões, a camada MAC é responsável pelo controle de acesso ao meio e pela alocação de banda.

O projeto da camada MAC dos padrões IEEE 802.16-2004 e IEEE 802.16-2005 inclui uma subcamada de convergência que pode comunicar-se com uma variedade de protocolos de camadas mais altas, como ATM, TDM Voice, Ethernet, IP, etc. Dada a predominância dos protocolos IP e Ethernet na indústria, o WiMAX Forum decidiu suportar somente esses dois protocolos por agora.

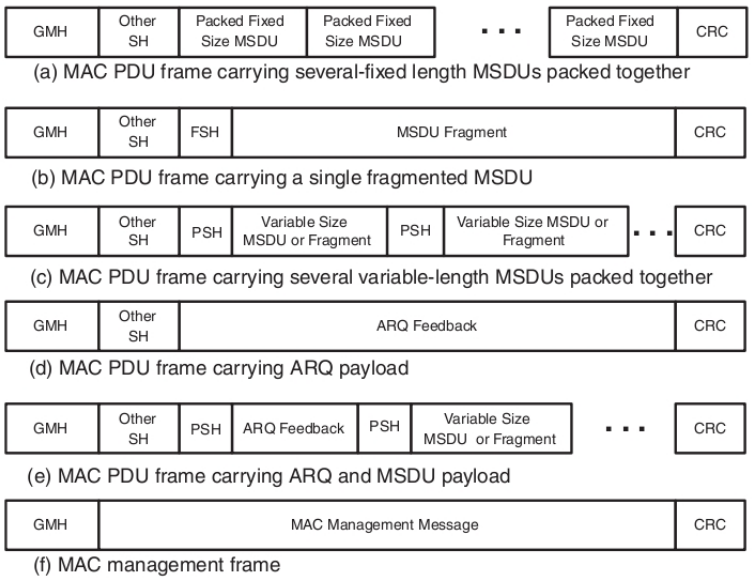

Figura 2.2: Quadros do MAC PDU

A figura 2.2 mostra vários tipos de quadros MAC PDU (packet data unit). Cada quadro MAC é pré-fixado com um cabeçalho MAC genérico (GMH) que contêm o identificador da conexão (CID), o tamanho do quadro, e bits para qualificar a presença de CRC.

2.3.1 Qualidade de Serviço

Suporte para QoS é uma das partes fundamentais do projeto da camada MAC. O WiMAX empresta algumas das idéias básicas por trás do seu projeto de QoS do padrão DOCSIS.

Um controle forte de QoS é conseguido pelo uso de uma arquitetura MAC orientada à conexão, onde todas as conexões downlink e uplink são controladas pela BS.

Antes de ocorrer alguma transferência de dados, a BS e a MS devem estabelecer uma conexão entre os dois peers da camada MAC. Cada conexão é identificada por um identificador de conexão (CID), que servem como um endereço temporário para a transmissão de dados sobre um canal em particular.

O WiMAX também define um conceito de fluxo de serviço. Um fluxo de serviço é um fluxo unidirecional de pacotes com um conjunto de parâmetros de QoS e é identificado por um identificador de fluxo de serviço (SFID).

Para suportar uma grande variedade de aplicações, o WiMAX define cinco serviços de escalonamento que devem ser suportados pelo escalonador da camada MAC da BS:

1. Serviços de Concessão não Autorizados
2. Serviços de Sondagem em Tempo Real
3. Serviços de Sondagem em Tempo não Real
4. Serviço do Melhor-Esforço
5. Serviço de Taxa Variável em Tempo Real extendido

Capítulo 3

Ferramenta Network Simulation 2

O Network Simulator 2 (*ns-2*) [1] é um simulador de eventos discretos de domínio público que contém grande parte dos protocolos utilizados na Internet, incluindo redes cabeadas e sem fio.

O *ns-2* é um dos simuladores mais populares entre os pesquisadores e em conjunto com a ferramenta de visualização Network Animator (*nam*) constitui uma plataforma didática para o ensino de redes de computadores.

O desenvolvimento do *Network Simulator* se iniciou em 1989 a partir de uma variação do simulador de redes REAL e, desde então, tem sido constantemente aperfeiçoado.

A implementação do simulador é orientada a objetos, escrita em C++ com um interpretador OTcl como frontend. O simulador possui uma hierarquia de classes C++, chamada hierarquia compilada, e uma hierarquia de classes similar dentro do interpretador OTcl, chamada hierarquia interpretada. As duas hierarquias são fortemente relacionadas. Do ponto de vista do usuário, há uma correspondência um-para-um entre uma classe na hierarquia interpretada e uma classe na hierarquia compilada. O usuário cria um novo objeto através do interpretador, este objeto é instanciado dentro do interpretador e um objeto correspondente é criado na hierarquia compilada.

Para realizar a simulação de uma rede, o usuário deve escrever um script em oTcl que inicializa um escalonador de eventos, configura a topologia da rede utilizando objetos e funções disponíveis na hierarquia e cria a lista de eventos que serão escalonados durante o período de simulação.

Capítulo 4

Módulo WiMAX

O desenvolvimento do trabalho foi feito sobre o módulo WiMAX desenvolvido pelo NIST, que implementa a camada física OFDM (*Orthogonal Frequency Division Multiplexing*) baseado na especificação *WirelessMAN OFDM PHY*.

A camada física do módulo conta com diversos parâmetros configuráveis, tais como potência de transmissão, prefixo cíclico, faixa de frequência e MCS(*Modulation and Coding Scheme*). O MCS conta com 7 diferentes perfis de modulação e codificação: BPSK 1/2, QPSK 1/2, QPSK 3/4, 16QAM 1/2, 16QAM 3/4, 64QAM 2/3 e 64QAM 3/4.

A camada MAC contém algumas mensagens de gerenciamento, tais como *Downlink Channel Descriptor* (DCD), *Uplink Channel Descriptor* (UCD), *Downlink MAP* (DL-MAP), *Uplink MAP* (UL-MAP), *ranging request*, *ranging response*, *registration request* e *registration response*.

Capítulo 5

Modelos de Propagação

5.1 Introdução

O modelo de propagação consiste em prever a potência do sinal no receptor, considerando os diversos fenômenos físicos, como *perda de percurso*, *multi-path delay spread*, *fading*, etc. O modelo de canal é amplamente utilizado no planejamento de área de cobertura das células.

O planejamento das áreas de cobertura das estações requer a estimativa dos níveis de sinal dessas estações, bem como das outras operando nas mesmas frequências adjacentes. É essencial prever as zonas limites (onde o nível de sinal é mínimo) e as zonas onde pode haver interferência.

A previsão do sinal envolve a estimativa do valor médio. As coberturas são estabelecidas para determinadas percentagens de locais e de tempo. É muito importante que os modelos de propagação se aproximem da realidade o melhor possível de forma que possam se tornar ferramentas fidedignas e indispensáveis no planejamento celular de uma rede de comuicação móvel.

A maioria dos modelos fornece o valor médio do sinal recebido no ponto de recepção. Torna-se assim necessário conhecer a estatística do sinal para determinar a sua variância. A abordagem do problema da estimativa do sinal não pode ser feita de modo exclusivamente determinístico. A estimatica correta e o desenvolvimento de modelos implicam no conhecimento de todos os que influenciam a propagação em comunicações móveis.

5.2 Classificação dos Modelos de Propagação

Para cálculos de atenuação em enlaces são utilizados em sua grande maioria modelos de propagação. Todos os modelos devem levar em consideração o comportamento aleatório do sinal propagado, que é principalmente relacionado com os instantes nos quais os diferentes sinais vindos de diferentes caminhos irão atingir o receptor.

Os modelos de propagação podem ser empregados para predizer qual o comportamento dentro de um ambiente. Essas ferramentas são bastante úteis, pois podem ajudar na localização dos pontos de acesso para reduzir os efeitos de atenuação ocasionados principalmente por multipercursos.

Os modelos de propagação dividem-se em duas grandes categorias:

- **Empíricos:** Os modelos empíricos conduzem as curvas e equações que melhor se ajustam às medidas e têm a vantagem de contabilizar todos os fatores que afetam a propagação.

Necessitam ser sujeitos a validação para locais, frequências e condições diferentes dos ambientes de medida.

- **Teóricos:** Os modelos teóricos não contabilizam todos os fatores e não considera o ambiente em que o móvel se desloca. Permitem uma fácil alteração para outros valores dos parâmetros e dependem da definição de bases de dados geográficas.

Os modelos de propagação que vêm sendo utilizados nas últimas décadas pelos conceptores de sistemas são modelos híbridos e contemplam as perspectivas empírica e teórica. Estes modelos têm alguma flexibilidade podendo ser aferidos com medidas reais realizadas nos ambientes de propagação específicos onde vão ser utilizados. Dessa forma, minimiza-se o erro entre a estimativa do sinal previsto pelo modelo de propagação e a posterior realidade.

5.3 Classificação de Ambientes

A aplicação de modelos com uma componente empírica requer a classificação de ambientes, onde é usual distiguir três grandes categorias: Rural, Suburbana e Urbana. Existem vários tipos de classificação, geralmente associadas a modelos de propagação distintos.

A classificação de ambientes considera, entre outros, alguns parâmetros tais como: ondulação do terreno, densidade da vegetação, densidade e altura dos edifícios, existência de áreas abertas e existência de superfícies aquáticas.

5.4 Exame dos Modelos de Propagação

Na literatura encontra-se uma infinidade de modelos de propagação aplicáveis em ambientes outdoor, onde a estação móvel encontra-se do lado externo de construções, prédios e casas, e ambietes indoor, onde a estação móvel encontra-se interna as construções, bem como em pico células, micro células e macro células.

5.4.1 Modelo Shadowing

Os modelos de propagação FreeSpace e TwoRay representam um cenário real, onde a potência recebida depende apenas da distância entre o transmissor e o receptor. Na realidade, porém, a potência recebida é uma variável aleatória, já que os sinais percorrem múltiplos caminhos e sofrem diversos problemas de propagação. Os modelos FreeSpace e TwoRay na verdade calculam apenas a potência média recebida. É preciso introduzir uma variável aleatória no cálculo de potência de recepção para se ter um cenário mais real.

O modelo Shadowing consite de duas partes:

1. a primeira parte é a potência média, obtida a partir de um coeficiente de perda de percursso β e uma medida padrão de potência recebida em uma distância curta d_0
2. a segunda parte corresponde a variação do sinal

5.4.2 Modelo TwoRay

Esse modelo considera tanto o caminho direto entre o transmissor e receptor quanto o caminho refletido. O modelo apresenta maior confiabilidade do que o modelo FreeSpace para longas distâncias.

Eles considera elementos como altura das antenas e desconsidera a frequência de operação da rede.

5.4.3 Modelo Nakagami

Esse modelo apresenta uma modelagem estatística para canais com *fading* com diversos parâmetros de configuração, que permitem simulações de canais ideais até ambientes metropolitanos. O modelo utiliza a distribuição de probabilidade de Nakagami, que está relacionada com a distribuição *Gamma*.

5.4.4 Modelo de Okumura

O Modelo de Okumura é amplamente usada para áreas urbanas, sendo um modelo empírico, desenvolvido com base em expereimentos de medidas.

O modelo pode ser expresso pela equação:

$$PL = PL_f + A(f,d) - G(ht) - G(hr) - G(area) \tag{5.1}$$

Onde:

PL é a atenuação média em dB

L_f é a atenuação média no Modelo FreeSpace em dB

$A(f,d)$ é o valor encontrado em curvas empíricas em dB

$G(ht)$ é o fator de ganho da estação transmissora em dB

$G(hr)$ é o fator de ganho da estação receptora em dB

$G(area)$ é o valor encontrado em curvas empíricas; expressa o ganho devido ao ambiente em que o sistema está operando em dB

ht é a altura efetiva da antena transmissora em metros

hr é a altura efetiva da antena receptora em metros

f é a frequência em MHz

5.4.5 Modelo de Hata

O Modelo de Okumura foi tomado como referência para o desenvolvimento do Modelo de Hata, que na verdade, apresenta uma formulação prática do modelo de Okumura.

O modelo de Hata, de grande aceitação, leva em consideração a morfologia sem detalhamento, isto é, utiliza 'manchas' morfológicas para diferentes ambientes de propagação como área suburbana, área urbana, área urbuna densa, área rural, etc.

O modelo de Hata é uma formulação empírica do modelo de Okumura. Aplica-se a uma faixa de frequências entre 150 e 1500MHz.

Esse modelo é melhor adequado para macro células por cobrir áreas maiores que 1 km. Como as cobertas por uma micro célula são inferiores a esta distância o mesmo então não é recomendado.

O valor médio da atenuação em área urbana é:

$$PL_u = 69,55 + 26,16log(f) - 13,82log(ht) - a(hr) + (44,90 - 6,55log(ht))log(d) \tag{5.2}$$

O valor médio da atenuação em área suburbana é:

$$PL = PL_u - 2(log(f/28))^2 - 5,40 \tag{5.3}$$

O valor médio da atenuação em área rural é:

$$PL = L_u - 4,78log(f)^2 - 18,33log(f) - 40,98 \tag{5.4}$$

Onde:
PL_u é a atenuação média em dB
ht é a altura efetiva da antena transmissora em metros
hr é a altura efetiva da antena receptora em metros
d é a distância da estação móvel até a estação base em quilômetros
f é a frequência em MHz
$a(hr)$ é o fator de correção da altura afetiva da antena receptora da unidade móvel, a qual é uma função do tamanho da área de cobertura em dB

Esse fator pode ser calculado do modo que segue:

- **Para uma região urbana**:

$$a(hr) = (1,10log(f) - 0,70)hr - (1,56log(f) - 0,80) \tag{5.5}$$

- **Para áreas urbanas densas e $f \leq 300MHz$:**

$$a(hr) = 8,29(log(1,54hr))^2 - 1,10)hr \tag{5.6}$$

- **Para áreas urbanas densas e $f > 300MHz$:**

$$a(hr) = 3,20(log(11,75hr))^2 - 4,97)hr \tag{5.7}$$

5.4.6 Modelo COST 231

Esse modelo, desenvolvido a partir dos modelos de Walfish-Bertoni e Ikegami, leva em consideração a morfologia detalhada do terreno, com informações de altura de prédios, distância entre predios, lagura média das ruas e orientação destas com relação à direção de propagação.

O Comitê de Pesquisas Europeu COST 231 desenvolveu este modelo para propagação em ambientes urbanos para utilização na faixa de frequência compreendida entre 800 e 2000 MHz, e aplica-se tanto a projetos de sistemas macro celulares, como micro celulares, podendo as antenas das estações rádio base estar situadas abaixo das alturas dos prédios.

A grande inovação do modelo do COST 231 está relacionada com o fenômeno da propagação guiada quando existe linha de visada entre a estação-base e a móvel na direção de uma rua cercada por edifícios. A propagação desta forma é diferente da propagação FreeSpace

O valor médio da atenuação deste modelo é dada por:

Para d $\geq$ 20 metros:

$$PL = 42,6 + 26log(d) + 20log(f) \tag{5.8}$$

Para d $<$ 20 metros:

$$PL = PL_f \tag{5.9}$$

Onde:

PL é a atenuação total em dB

PL_f é a atenuação no modelo FreeSpace em dB

d é a distância da estação móvel até a estação base em quilômetros

f é a frequência em MHz

5.4.7 Modelo FreeSpace

O Modelo FreeSpace é o modelo de propagação que tem a situação mais otimista de perda de percuso, onde o transmissor e o receptor estão imersos em um espaço livre de obstruções e de reflexões em qualquer direção. O mecanismo de propagação envolvido é o de linha-de-visada, onde não há nenhum obstáculo entre o transmissor e o receptor. Embora a propagação FreeSpace seja uma situação bastante particular, este modelo é bastante utilizado devido sua simplicidade.

O seu entendimento e cálculo é útil para o estudo da propagação em diferentes ambientes e para diferentes sistemas.

A atenuação do sinal de rádio é determinada pela relação entre a potência recebida e a potência transmitida. No modelo de propagação FreeSpace, calcula-se a propagação entre antenas isotrópicas (irradiação uniforme em todas as direções) e os ganhos de antenas do transmissor e do receptor.

O valor médio da atenuação para o modelo é dada por:

$$PL = 92,44 + 20log(f) + 20log(d) - 10log(G_T) - 10log(G_R) \tag{5.10}$$

Onde:

PL é a atenuação total em dB

d é a distância da estação móvel até a estação base em quilômetros

f é a frequência em MHz

5.4.8 Modelo SUI

O Modelo SUI (*Stanford University Interim*) [6], também conhecido como Modelo de Erceg, é um modelo de propagação empírico que se baseia em dados recolhidos em campo de forma experimental na faixa de 1,9 GHz em 95 macrocélulas em todo Estados Unidos. As medições foram feitas, em sua maioria, em áreas suburbanas de New Jersey, Seattle, Chicago, Atlanta e Dallas. Esse modelo também inclui o fenômeno físico *fading*, que é passagem da onda por obstáculos.

Uma vez que os sistemas WiMAX operam numa faixa de frequência de 2-11GHz e é capaz de transmitir sem linha-de-visada, o modelo é recomendado pelo IEEE 802.16 *working group* para ser utilizado em simulações de sistemas WiMAX.

O Modelo SUI tem três variantes, com base no tipo de terreno:

- Categoria A é aplicável ao terreno montanhoso com forte densidade urbana
- Categoria B é aplicável ao terreno montanhoso com baixa densidade urbana ou terreno plano
- Categoria C é aplicável ao terreno plano com baixa densidade urbana

O valor médio da atenuação quando ($d \geq d_0$) é dada por:

$$A = 20log(\frac{4\pi d_0}{\gamma}) \tag{5.11}$$

$$PL = A + 10\gamma log(\frac{d}{d0}) + s \tag{5.12}$$

Onde:

PL é a atenuação total em dB

d é a distância da estação móvel até a estação base em metros

A é o ponto de intercessão em dB

d_0 é a distância do ponto de intercessão até a estação base igual a 100 metros

γ é expoente de perda de percurso que varia de acordo com o tipo de terreno

s é o componente *fading* que é calculado com uma variável Gaussiana aleatória de média zero

Parâmetro	**Terreno A**	**Terreno B**	**Terreno C**
a	$4,6$	$4,0$	$3,6$
b	$0,0075$	$0,0065$	0.0050
c	$12,6$	$17,1$	$20,0$
σ_γ	$0,57$	$0,75$	$0,59$
μ_σ	$10,6$	$9,6$	$8,2$
σ_σ	$2,3$	$3,0$	$1,6$

Tabela 5.1: Parâmetros do Modelo SUI

Capítulo 6

Modulação e Codificação Adaptativa

6.1 Introdução

A Modulação e Codificação Adaptativa visa adaptar a modulação e a codificação do canal de acordo com a qualidade do mesmo, ou seja, de acordo com a qualidade do sinal recebido a modulação é adaptada para uma que garanta uma boa taxa de comunicação. Isso garante que os usuários obterão uma boa taxa de transferência de dados com uma menor taxa de erro dos pacotes.

Como a tecnologia WiMAX esta sujeita a diversos tipos de interferências no sinal, faz-se necessário o uso desse mecanismo para melhorar a comunicação entre a BS e a SS.

É importante entender que o padrão IEEE 802.16 [2] apenas definiu o framework de modulação adaptativa, permitindo a um sistema WiMAX comunicar-se utilizando vários bursts profiles de acordo com a qualidade do canal. Ele não especificou os valores de threshold para a troca de modulação nem o algoritmo de adaptação.

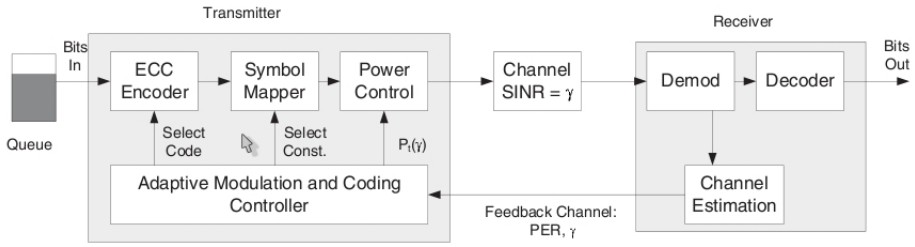

Figura 6.1: Diagrama da Modulação e Codificação Adaptativa

6.2 Fundamentos da Adaptação do Canal

A idéia básica por trás do emprego de técnicas de adaptação de canal é utilizar o canal da forma mais eficiente possível dentro das suas condições prevalecentes. Tudo é feito para sustentar uma conexão, mesmo se algum comprometimento da qualidade da conexão tenha que ser empregado. Na maioria dos sistemas que empregam adaptação de canal isso resulta em um equilíbrio entre a robustez do canal e eficiência de largura de banda.

Existem três mecanismos principais pelos quais um sistema poderia melhorar a robustez da conexão em detrimento da eficiência de largura de banda total:

- Modulação Adaptativa
- Forward Error Correction Adaptativo (FEC)

- Automatic Repeat Request (ARQ)

6.3 Funcionamento

A codificação dos dados é divida em 5 etapas:

- Aleatorização de dados (data randomization)
- Codificação do canal (channel coding)
- Ajuste de taxa (rate matching)
- H-ARQ (hybrid automatic Repeat request)
- Entrelaçamento (interleaving)

Na ***aleatorização*** de dados os bits são reorganizados de maneira aleatória, com objetivo de fazer uma encriptação e prevenir que um receptor 'não desejado' receba os dados. Depois, são formados blocos chamados FEC (forward error correction) nos quais os dados são divididos. Cada bloco FEC é formado por um número de subcanais (contendo assim várias subportadoras de dados e subportadoras piloto).

Podem ser usados vários tipos de ***codificação*** para os dados:

- Codificação Turbo
- Codificação Reed-Solomon
- Codificação Convolucional

A codificação Convolucional é a mais usada dentre as outras e é a que foi usada no padrão 802.16d-2004(em conjunto com a Reed-Solomon). A codificação Convolucional consiste em transformar uma palavra de m bits em uma palavra de n bits, de acordo com uma função definida pelos últimos k bits a serem codificados, e sua taxa de codificação é dada por m/n. No padrão 802.16 [2] é usado k igual a 7 e m/n, geralmente, igual a 1/2. Essa taxa significa que em cada palavra a ser codificada serão inseridos bits redundantes de tal forma que após a codificação teremos uma palavra com o dobro do tamanho da palavra original, ou seja, metade da palavra que realmente será transmitida é informação redundante. Assim teremos apenas a metade de nossa banda para transmitir informação útil.

Depois, na fase de ajuste de taxa, são adicionados ou retirados bits. A fase ***H-ARQ***, que pode ser ou não usada, consiste em dar aos blocos que precisam ser reenviados uma redundância a mais para que tenham uma chance maior de serem recebidos no receptor. O próximo passo é o ***entrelaçamento***, onde os bits são reposicionados em subportadoras diferentes para espalhar a informação, e assim evitar que algum erro atinja toda a informação original, lembremos que o processo de codificação inclui muitos bits redundantes.

Por fim, ocorre a modulação.

As Modulações mais utilizadas são apresentadas na Tabela 2.2.

6.4 Mecanismos Implementados na Literatura

6.4.1 QoS-Aware Link Rate Adaptation (Q-LRA)

Esse algoritmo proposto em [11] leva em consideração, para determinar a melhor modulação, a demanda de QoS e a qualidade do sinal no canal. A demanda de QoS de uma conexão é representada como a taxa de pacotes perdidos (PER).

Com base no PER, a BS calcula o threshold do SNR (signal-to-noise rate) de cada modulação.

Pela comparação do SNR calculado com o threshold do SNR de todas as modulações, a BS sugere à SS a melhor modulação.

O algoritmo Q-LRA consiste de duas fases:

1. ***Derivação da faixa de SNR aceitável:***

   Para derivar o SNR threshold de cada modulação, nós assumimos um pacote de tamanho fixo B. Assim, dado o PER, a correspondente taxa de erro de bit P_{be} pode ser calculada desse modo:

   $$P_{be} = 1 - \sqrt[B]{1 - PER} \tag{6.1}$$

2. ***Seleção da Modulação:***

   (a) A SS informa à sua BS a condição do PER do novo fluxo de serviço i.

   (b) Com o modelo de erro de canal adotado e a condição do PER especificado, a BS calcula a correspondente faixa de SNR aceitável (denotada como E_b/N_0) de todas as modulações.

   (c) Comparando o E_b/N_0 calculado com a faixa SNR obtida de cada modulação, a BS sugere à SS a melhor modulação, digamos que seja a modulação m.

   A BS aloca $Num\_slots[i]$ slots para a SS de acordo com a taxa de dados suportada pela modulação m (denotada como R_m) e do tamanho da fila de dados do fluxo de serviço i. $Num\_slots[i]$ é calculado com a seguir:

   $$Num\_slot[i] = \frac{data\_queue[i]}{R_m \times T_{slot}} \tag{6.2}$$

6.4.2 Dynamic Threshold Link Adaptation (DTLA)

Esse algoritmo proposto em [10] foi desenvolvido para ser utilizado em redes WiMAX que utilizam a tecnologia Multiple-Input-Multiple-Output (MIMO).

Essa tecnologia, MIMO, provê uma solução promissora no aumento da robustes dos dados transmitidos enquanto não sacrifica as taxas de transmissão através da diversidade do meio.

Esse algoritmo foi proposto para a mudança de modulação do canal downlink da SS, enquanto o canal uplink adota o BPSK 1/2, o burst profile mais robusto, para que mensagens sejam transmitidas para a BS.

A seleção do burst profile do canal downlink é baseado nos SNRs thresholds na camada física através de um mecanismo dinâmico de sondagem para lidar com diferentes condições do canal.

O algoritmo DTLA consiste de quatro fases:

1. ***Atualização das Estatísctica:*** O algoritmo mantém estatísticas do número de pacotes corrompidos, pacotes corrompidos sucessivos e pacotes recebidos com sucesso. para cada MCS. O SNR estimado e o cálculo do número de Demmel são mantidos para cada burst. O SNR é atualizado pela média do SNR atual com os valores anteriores de acordo com a equação:

   $$SNR_{est} = (1 - \alpha) * SNR_{est} + \alpha * SNR_{cur} \tag{6.3}$$

   , where α = [0,1]

2. ***Atualização dos Thresholds:*** Dois tipos de thresholds são introduzidos no algoritmo: o burst profile enter threshold (BPET) e o burst profile probe threshold (BPPT).

   (a) **BPET**: Cada MCS mantém um BPET no qual é o SNR inicial que o MCS esta habilitado a usar. Um MCS que tem um valor de BPET menor do que o valor de SNR atual será um forte candidato para a adaptação de taxa. As regras de adaptação são baseadas em estatísticas colhidas como o PER e BER do MCS.

   (b) **BPPT**: O BPET do MCS atualmente em uso é atualizado para o valor do SNR estimado, SNR_{est}, somente se SNR_{est} cair abaixo do valor de SNR_{BPET} e o pacote ser recebido com sucesso, porque isso mostra que o MCS pode ser usado em um SNR menor. O BPET é incrementado linearmente por $2dB$ se o pacote recebido estiver corrompido e SNR_{est} estiver acima do BPET do MCS atualmente em uso, porque isso mostra que o MCS pode não funcionar bem com esse BPET.

3. ***Escolher um MCS:*** Um MCS onde $SNR_{est} > SNR_{BPET}$ torna-se um candidato. A ordem dos MCSs é obtida do resultados das simulações na camada física.

   A Tabela 6.1 mostra a ordem das taxas adotadas pelo algoritmo.

   O algoritmo primeiro inicia com uma taxa maior do que a taxa atual e passa através de todas as taxas onde o $SNR_{est} > SNR_{BPET}$. Se o MCS falhar no recebimento de cinco pacotes consecutivos ou a condição do número de Demmel aumenta ao longo das últimas amostras com a diminuição do SNR, o MCS não é selecionado. Se o MCS satisfez o critério, será contado e o MCS com o maior número de contêiners satisfeitos será escolhida como o melhor.

Rate ID	MCS	FEC code rate	MIMO mode
0	*QPSK*	1/2	STBC
1	*QPSK*	2/3	STBC
2	*QPSK*	1/2	SM
3	*16QAM*	1/2	STBC
4	*QPSK*	3/4	SM
5	*16QAM*	3/4	STBC
6	*16QAM*	1/2	SM
7	*64QAM*	2/3	STBC
8	*64QAM*	3/4	STBC
9	*16QAM*	3/4	SM
10	*64QAM*	2/3	SM
11	*64QAM*	3/4	SM

Tabela 6.1: Ordem das Taxas

4. ***Ajuste da escolha de acordo com as condições do canal:*** Se o MCS escolhido é menor do que a taxa atual recebida, a escolha deve ser ajustada. Se a condição do número de Demmel estiver aumentando em todo o número armazenado com um decremento do SNR_{est}, o MCS que esta uma posição abaixo do rate ID atual será selecionado. Caso contrário, o MCS escolhido será enviado para a BS através do DBPC-REQ se ele for diferente do atual.

6.4.3 Modelo de Ramachandran, Bostian e Midkiff

Esse modelo proposto em [9] utiliza o SNR como métrica para a escolha da melhor MCS para a comunicação entre a BS e a SS.

O objetivo desse trabalho foi desenvolver um algoritmo simples para implementar a adaptação de canal usando o framework definido no padrão [2].

Nesse modelo, a SS calcula o SNR e verifica o melhor MCS para SNR calculado, como é recomendado pelo padrão [2]. A SS fica constantemente calculando o SNR e quando seu valor excede ou cai abaixo do nível de threshold do MCS atual, a SS requisita a troca do burst profile do canal downlink para um que seja melhor para a situação de comunicação atual.

Essa requisição de troca de modulação é feita utilizando a mensagem Range Request (RNG-REQ) definida no padrão [2]. A SS utiliza o intervalo da manutenção inicial ou o intervalo da manutenção da estação para transmitir essa mensagem para a BS com o DIUC do burst profile desejado.

Capítulo 7

Modelo de Erros

7.1 Modelo de Bustamante

Para encontrar o valor de PER, [4] propôs um modelo que usa um simulador OFMA desenvolvido em *MatLab*, configurando um perfil MCS, informação do canal e tamanho do pacote. O simulador conta com os seguintes componentes: gerador de pacotes aleatório, *Reed Solomon Coder*, *Convolutional Coder*, *Scrambler* e *Interleaver*, Modulador, Equalizador e uma modelo de canal *multipath*.

Executando diversas simulações, os valores de BER para cada perfil de modulação e codificação foi encontrado, para um intervalo de SNR, como mostra o gráfico. Para obter os valores de PER, o pacote era codificado e adicionado um número aleatório de erros, para depois ser decodificado e analizado se estava correto ou não. O valor do PER para cada número de erro pode ser visualizado na figura.

7.2 Modelo ICRFE

Um simulador WiMAX que implementa a camada física OFDM do padrão IEEE 802.16-2004 foi proposto pelo ICRFE da Vienna University of Technology. O simulador foi desenvolvido em *Matlab* e conta com três elementos básicos (Transmissor, Receptor e Modelo de Canal) para realizar a avaliação de desempenho das redes WiMAX.

Esse modelo apresenta apenas os valores de BER, e as estimativas são feitas no receptor. O receptor é composto pelos seguintes componentes: Conversor FFT, filtros, *Disassembler*, *Demapper* e *Decoder*.

Geralmente, o número de erros em uma sequência de *bits* é modelado através de uma distribuição binomial, supondo que os erros são independentes e ocorrem com a mesma probabilidade.

O valor de PER pode ser obtido a partir do BER com a equação:

$$PER = 1 - (1 - BER)^L \tag{7.1}$$

Onde L é o tamanho do pacote.

7.3 Modelo no *ns-2*

Por conta da complexidade nos cálculos, é inviável realizar uma simulação *bit* a *bit* na ferramenta *ns-2*. Por causa disto, as simulações são executadas em um simulador desenvolvido em *MatLab*. Para incluir esses modelos na ferramenta *ns-2*, as simulações *bit* a *bit* são executadas no *MatLab* e os resultados são importados em forma de tabela para o módulo WiMAX. Desta maneira, os valores de *BER* e de *PER* são retornados de forma rápida, porém, perde-se precisão, uma vez que não é possível armazenar os infinitos pontos da curva.

Para contornar o problema da precisão, ao consultar a tabela de valores, os dois valores mais próximos são recuperados e então é feita uma interpolação linear. A interpolação é o método mais viável a ser aplicado, visto que as outras alternativas, como utilizar uma interpolação com uma *Q-function* ou uma função *Arc-Tangente*, tem uma alta complexidade de implementação.

Os valores são consultados na tabela da seguinte forma. Com o *SNR* medido do pacote, os dois valores mais próximos do *SNR* são consultados na tabela de *BER*. É feita uma interpolação com esses dois valores e é encontrado um valor de *BER*. Se estiver sendo utilizado o Modelo de Erros - Bustamante, a consulta é repetida na tabela de *PER*. No caso do Modelo de Erros - ICRFE, o valor é obtido analiticamente com a equação.

Depois de feita a estimativa do *PER*, considera-se que o valor de *PER* é a probabilidade de perda de pacote. Uma variável uniforme aleatória com intervalo de 0 a 1 é utilizada. Se o valor da variável for menor ou igual ao valor da probabilidade de perda de pacote, o pacote é descartado, senão, o pacote é recebido pela camada física OFDM.

Capítulo 8

Trabalho Realizado

Esse capítulo mostra o trabalho realizado ao longo dessa pesquisa. Como foi dito anteriormente nesse documento, esse pesquisa foi focada no desenvolvimento de um modelo realista para simulações de WiMAX no ns-2.

8.1 Objetivos

Foi verificado que o módulo WiMAX que integraríamos com o simulador ns-2 não implementava modelos de erros na camada física nem modelos de propagação realistas, ou seja, modelos de propagação que se adequam aos ambientes reais onde o WiMAX poderá ser implantado. Então, para uma primeira etapa do projeto, modelos de canais realistas e modelos de erros foram implementados no simulador *ns-2*.

Um dos modelos de propagação mais utilizados em simulações é o FreeSpace, no qual já esta implementado no *ns-2*, porém, ele não providência uma medição realista.

Já o Modelo SUI, é um modelo empírico (como vismos no Capítulo 5), ou seja, um modelo mais realista, porém ele não esta implemtentado no *ns-2*, assim sendo necessário sua implementação e validação.

Para implementar o Modelo SUI, foi procurado na literatua [3] [6] as informações necessárias para o funcionamento desse modelo, como as váriaveis, as equações, etc., para que o modelo implementado fosse o mais realista possível. Em posse dessas informações, um algoritmo, apresentado no Apêndice A, foi implementado.

Além disso, modelos de erros foram estudados na literatura [4] para serem incorporados na camada física do módulo WiMAX utilizado, assim chegando a um modelo mais realista possível.

Como o padrão [2] apenas define um framework para a modulação adaptativa sem definir nenhum algoritmo como padrão, resolvemos estudar na literatura [9] [11] [10] formas de se fazer essa implementação. Depois desse estudo, um algoritmo, apresentado no Apêndice A, foi implementado e validado, deixando o nosso módulo com a capacidade de efetuar a adaptação dinamicamente.

8.2 Modelo proposto

O modelo proposto utiliza a topologia Ponto-Multiponto, acesso ao meio físico no modo TDD e implementa um modelo de propagação mais realista para os ambientes com uma camada física realista WirelessMAN-OFDM que leva em conta as imperfeições do meio.

A camada física, em conjunto com o modelo de propagação, determina se um pacote foi recebido corretamente.

O modelo de propagação retorna um valor de SNR (*Signal to Noise Ratio*), que indica a qualidade do sinal em relação ao ruído no receptor. Com o valor de SNR, a camada física calcula o BER (*Bit Error Rate*) e, de acordo com o tamanho do pacote, calcula o PER (*Packet Error Rate*). O valor do PER indica a probabilidade do pacote conter um erro.

Além disso, um algoritmo de AMC (*Adaptive Modulation and Coding*) foi implementado no nosso modelo para resolver o problema da adaptação dinâmica da modulação.

Esse algoritmo esta de acordo com o padrão [2], ou seja, leva em consideração a qualidade do sinal medido pelo SNR para saber quando a modulação deve ser trocada. Como o padrão [2] não especifica qual o valor de SNR para cada modulação, um estudo específico para determinar esse valor foi feito levando em conta somente as interferências no sinal.

Para que esse algoritmo servisse para os modelos de propagação mais utilizados, um estudo mais detalhado desse algoritmo sobre o modelo SUI foi feito, pois como mostram as figuras 8.2, 8.3 e 8.4, o valor do SNR instantâneo tem uma grande variação em comparação com o SNR instantâneo do modelo FreeSpace na figura 8.1. Por esse motivo, o SNR usado foi o médio, que é calculado com a equação 6.3, eliminando assim grande parte da variação como mostram as figuras 8.5, 8.6 e 8.7.

8.3 Simulações

Diversos cenários foram projetados para avaliar o modelo de camada física implementado. As topologias e os parâmetros de redes foram implementados em scripts *TCL* e simulados no módulo WiMAX proposto no *ns-2*.

Apenas uma célula foi modelada com uma configuração PMP (*Point Multi-Point*), utilizando uma estação base (BS) e uma estação cliente (SS). A estação cliente se inicia na mesma localização da estação base e vai se distanciando com velocidade constante. Foram executadas simulações para diferentes MCS, desde a mais robusta BPSK 1/2 até a mais eficiente 64QAM 3/4

Com potência de 0.01 watt para o modelo FreeSpace e 0.1 watt para o modelo SUI, a estação base transmite para uma estação cliente que se distancia com velocidade constante. A fonte de tráfego utilizado foi um agente CBR, com pacotes de 100 *bytes*, a uma taxa de 10Mbps sobre o protocolo UDP. Foi escolhido o protocolo UDP para que o mecanismos de controle de congestionamento não influenciasse a taxa de vazão da rede.

Os parâmetros de configuração dos dispositivos da simulação foram definidos de acordo com a especificação do documento desenvolvido pelo WiMAX Forum, que descreve a metodologia para avaliação de sistemas WiMAX.

Os principais parâmetros estão descritos na tabela 8.1

Parâmetro	**Valor**
Ganho de antena(BS)	16 dBi
Ganho de antena(SS)	0 dBi
Tamanho da antena(BS)	32 m
Tamanho da antena(SS)	1.5 m
Figura de ruído(BS)	4 dB
Figura de ruído(SS)	8 dB
Total Bandwidth	10 MHz
Frequência de operação	2.5 GHz
Potência de transmissão	0.1 watt / 0.01 watt

Tabela 8.1: Parâmetros usados nas simulações

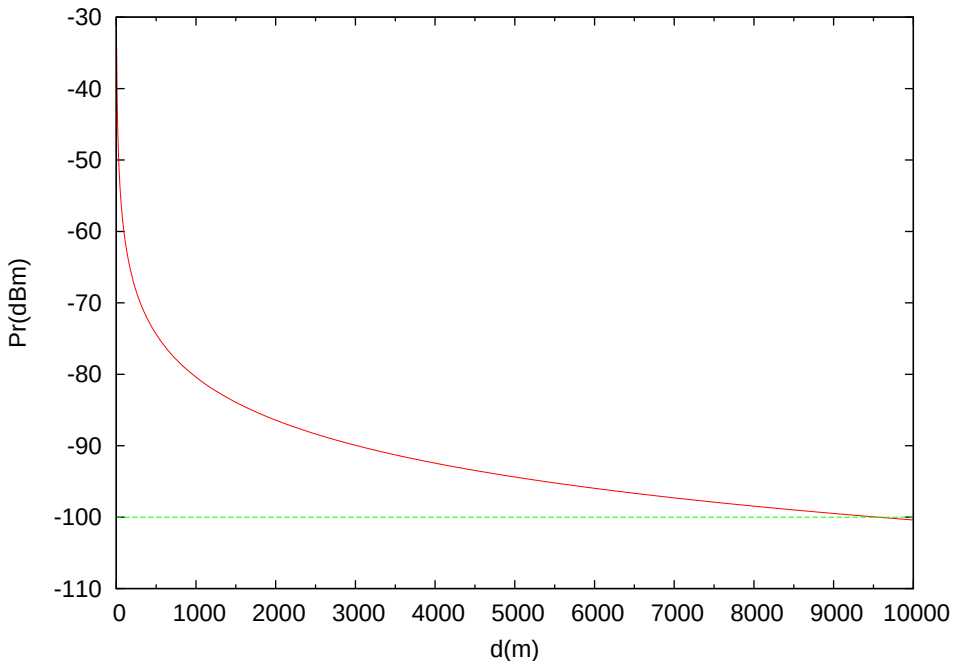

Figura 8.1: Potência instantânea recebida no Modelo de Propagação FreeSpace

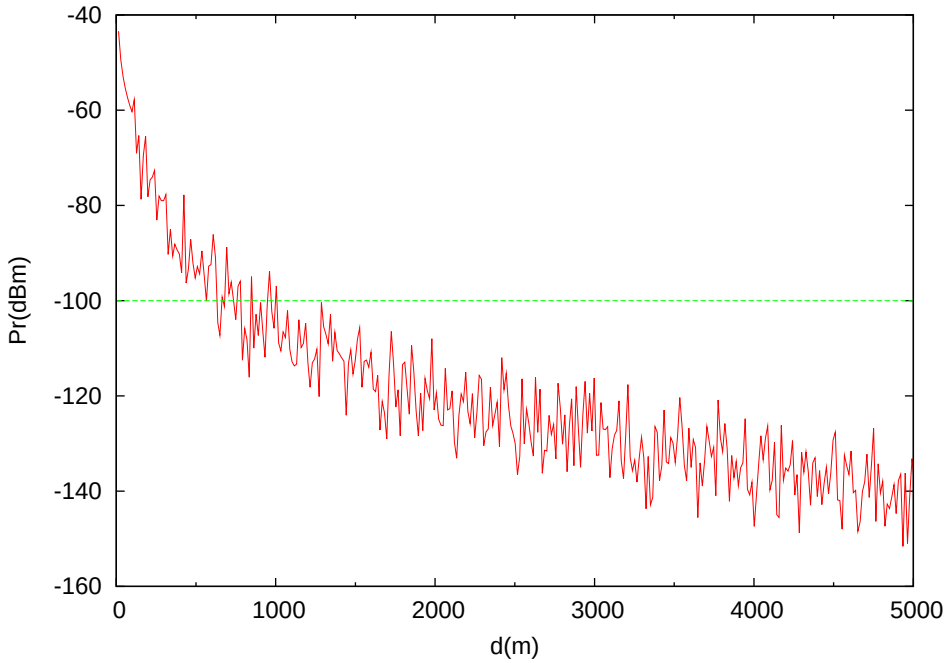

Figura 8.2: Potência instantânea recebida no Modelo de Propagação SUI - Terreno A

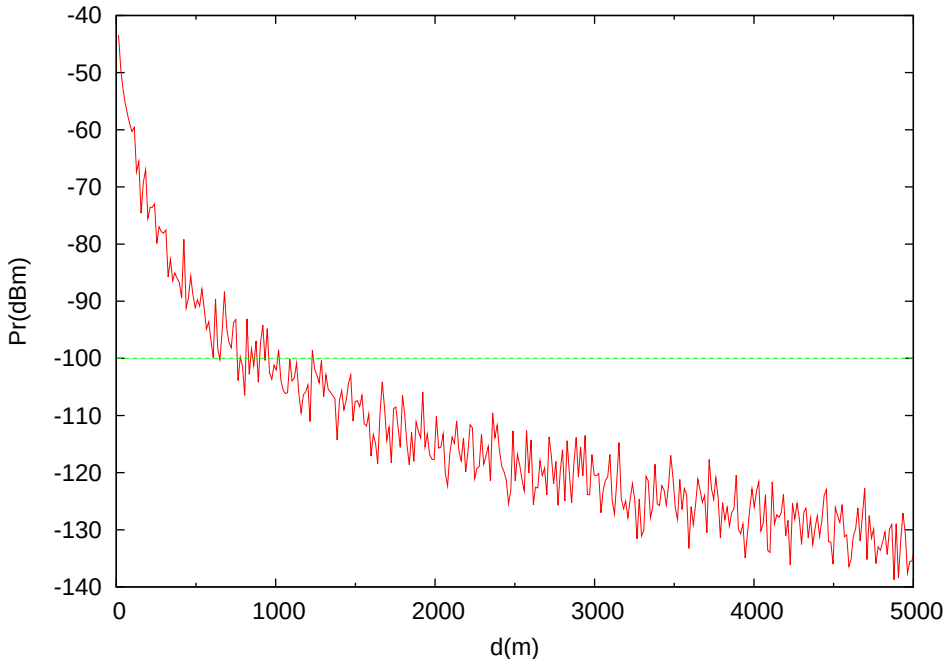

Figura 8.3: Potência instantânea recebida no Modelo de Propagação SUI - Terreno B

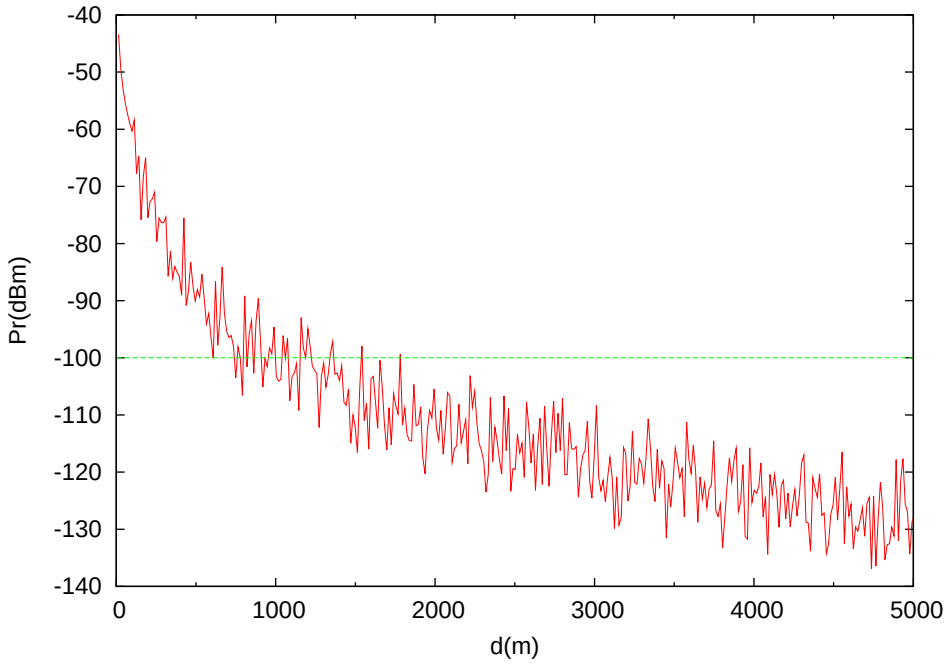

Figura 8.4: Potência instantânea recebida no Modelo de Propagação SUI - Terreno C

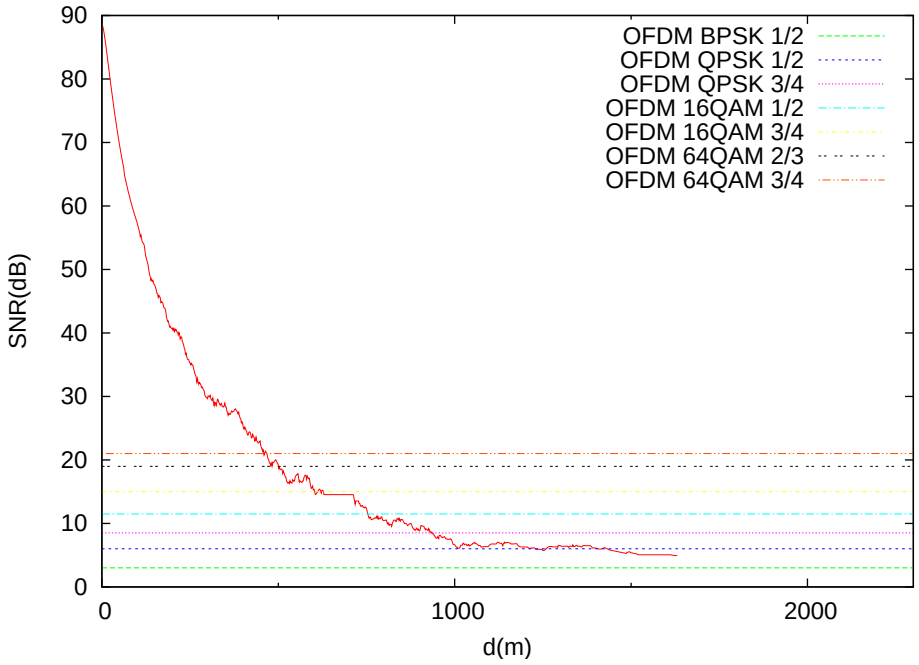

Figura 8.5: Potência média recebida no no Modelo SUI - Terreno A

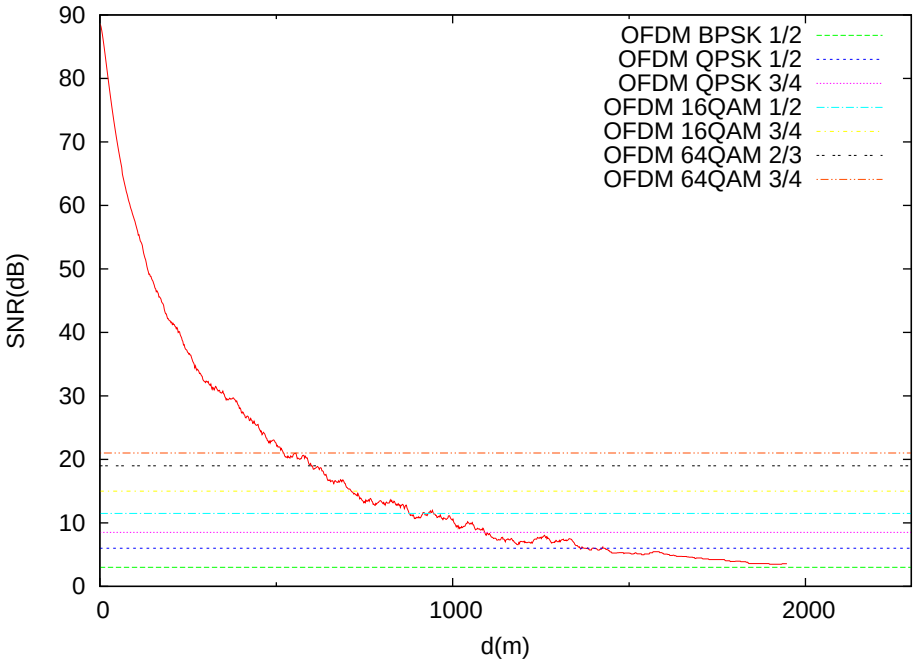

Figura 8.6: Potência média recebida no Modelo SUI - Terreno A

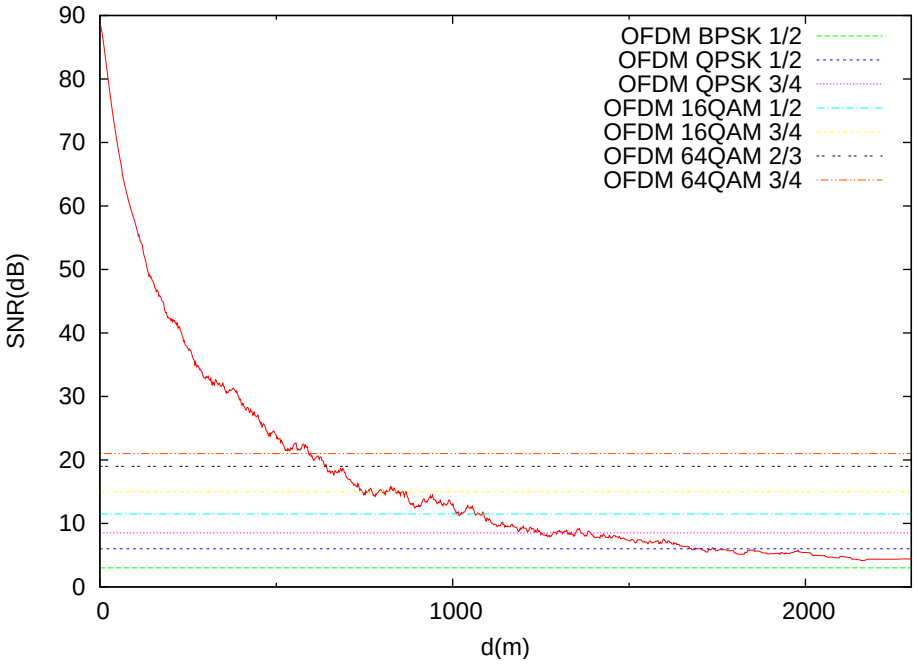

Figura 8.7: Potência média recebida no Modelo SUI - Terreno A

Capítulo 9

Resultados

9.1 Análise de cobertura

Para a avaliar a degradação do sinal com o modelo de propagação, foi feita uma análise de cobertura da célula WiMAX com uma antena isotrópica.

No primeiro cenário, o modelo de propagação utilizado foi o FreeSpace. A figura 8.1 mostra a cobertura do sinal num cenário ideal. Considerando um *threshold* de referência de -100 dBm, a área de cobertura tem um raio de aproximadamente 9,5 quilômetros.

Foram simulado os diversos tipos de terrenos do modelo SUI descritos anteriormente. As figuras 8.2, 8.3 e 8.4 mostram a cobertura do sinal em ambientes suburbanos de diferentes características. No terreno A, a área de cobertura é de aproximadamente 800 metros, no terreno B, o sinal chega a aproximadamente 1000 metros e no terreno C, a 1300 metros.

Comparando os modelos simulados (figuras 8.1, 8.2, 8.3, 8.4, 9.3, 9.4, 9.1, 9.2 e 9.5) observa-se que o modelo FreeSpace possui um raio de cobertura irreal, enquanto que o modelo SUI simula três tipos de ambientes mais realista.

Em um ambiente real, para conseguir um alcance maior, diversas técnicas são utilizadas, como por exemplo a utilização de várias antenas *(MIMO - Multiple In Multiple Out)*.

Uma experimentação de Milanovic realiza um estudo comparando modelos de propagação com algumas medidas realizadas nas áreas urbanas e suburbanas de Osijek, Croácia. O modelo de Erceg apresentou os resultados mais precisos quando comparados com os modelos de COST 231, Hata, etc.

9.2 Modelo de Erros

Para a avaliar a desempenho da rede através da AMC e do modelo de erros, simulações foram realizadas utilizando o throughput e o SNR como métricas.

O parâmetros desta simulação está descrito na seção 8.

As figuras 9.6 e 9.7 mostram a taxa de vazão de cada MCS de acordo com a distância no Modelo FreeSpace, mostrando que o modelo proposto melhora a implementação quando comparado com uma implementação sem a modelagem de erros. O modelo é uma representação mais detalhada da realidade, visto que quando um *link* é estabelecido utilizando um certo MCS, a qualidade da conexão degrada progressivamente quando a distância entre a estação base e a estação cliente aumenta.

Pelas figuras 9.6 e 9.7 observa-se que ambos os modelos trazem um maior realismo nas

simulações, entretanto, o Modelo de Erros - Bustamante apresenta um melhor resultado pois as curvas têm uma queda na taxa de vazão mais suave, enquanto que o Modelo de Erros - ICRFE apresenta um resultado mais pessimista.

Isso é devido ao fato de que o modelo proposto em [4] realiza uma simulação para encontrar o BER e realiza outra simulação para encontrar o PER, enquanto que o modelo proposto pelo ICRFE realiza simulação apenas para o encontar o valor de BER. Neste caso, o PER é estimando estatisticamente, encontrando a probabilidade mais pessimista de perder o pacote.

9.3 Modulação e Codificação Adaptativa

Pela figura 9.8 observamos o mecanismo AMC implementado em ação, adaptando a modulação para obter o melhor throughput de acordo com a variação do SNR com o aumento da distância da SS em relação a BS. Nas figuras 9.9, 9.10 e 9.11 observamos as trocas de modulação de acordo com o SNR para cada tipo de terreno do modelo SUI.

As figuras 9.12 e 9.13 mostram a taxa de vazão de cada MCS de acordo com a distância num ambiente suburbano, com os fenômenos físicos *perda de percurso* e *fading*.

Com o Modelo SUI, nota-se um aumento da aleatoriedade na rede devido ao componente de aleatoriedade.

Devido à esse comportamento mais aleatório, acredita-se que as simulações com o Modelo SUI se aproximam mais da realidade. Entretanto, esse comportamento torna o estudo da rede mais difícil de ser realizado.

Os modelos implementados trazem uma grande contribuição para as pesquisas de redes WiMAX, pois permitem que estudos de desempenho ou de algoritmos que levam em conta o canal sejam realizados.

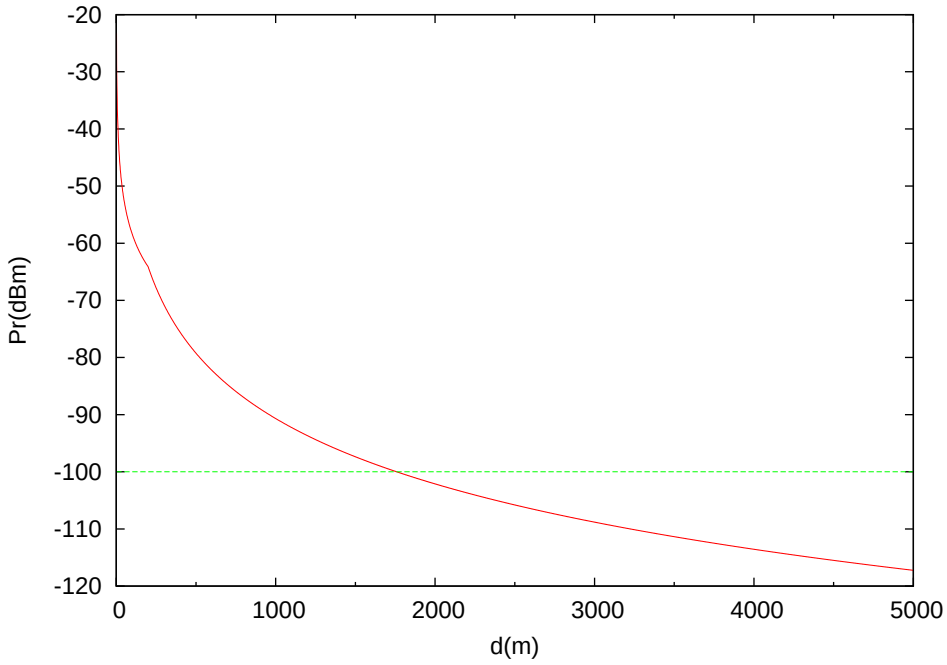

Figura 9.1: Potência recebida no Modelo de Propagação Nakagami (Determinístico)

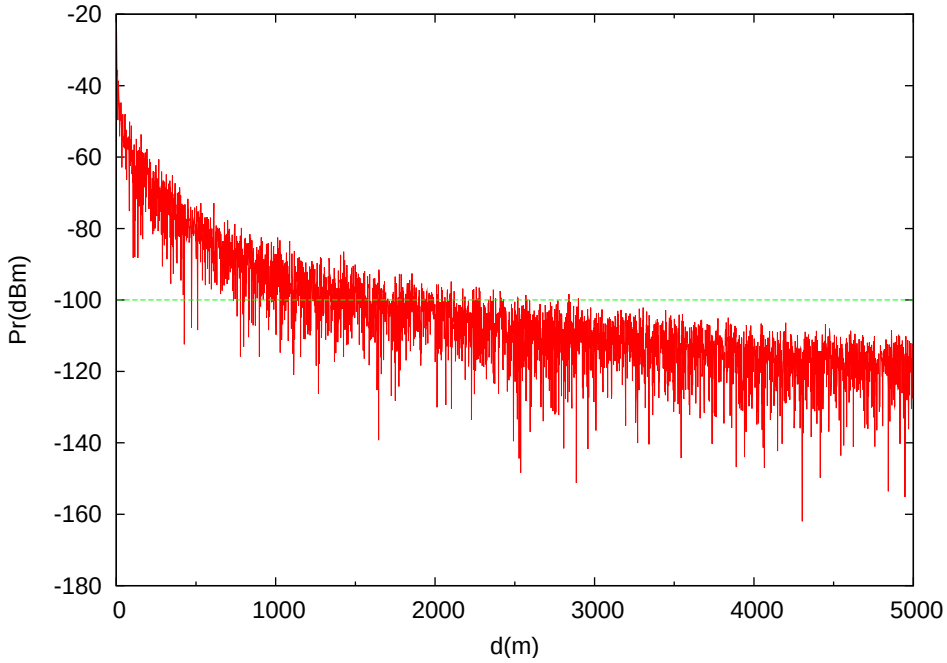

Figura 9.2: Potência recebida no Modelo de Propagação Nakagami (Probabilístico)

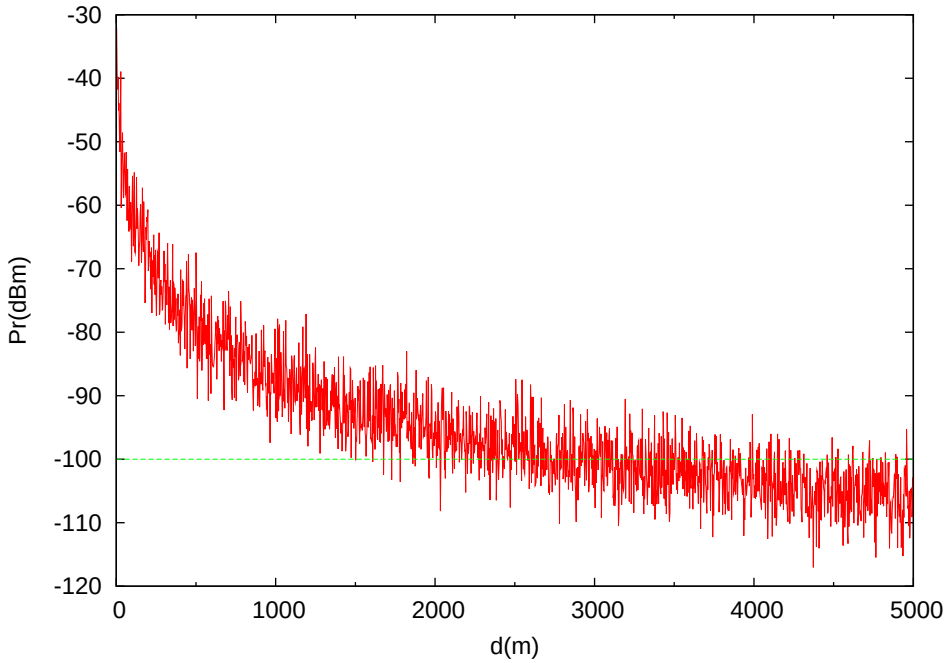

Figura 9.3: Potência recebida no Modelo de Propagação Shadowing ($\beta = 2.7$ e $\sigma_{dB} = 4.0$)

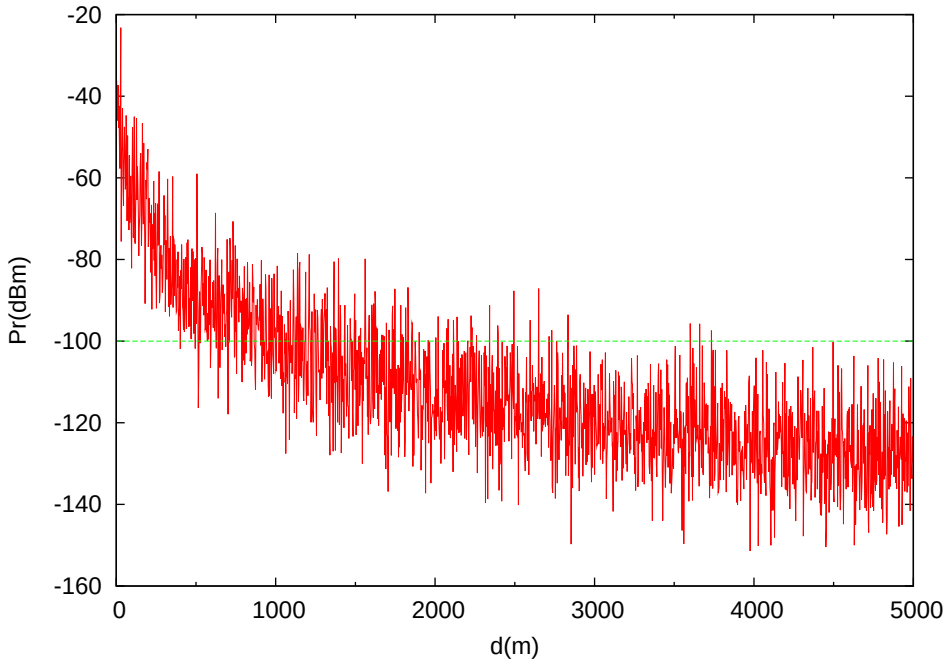

Figura 9.4: Potência recebida no Modelo de Propagação Shadowing ($\beta = 4.0$ e $\sigma_{dB} = 10.0$)

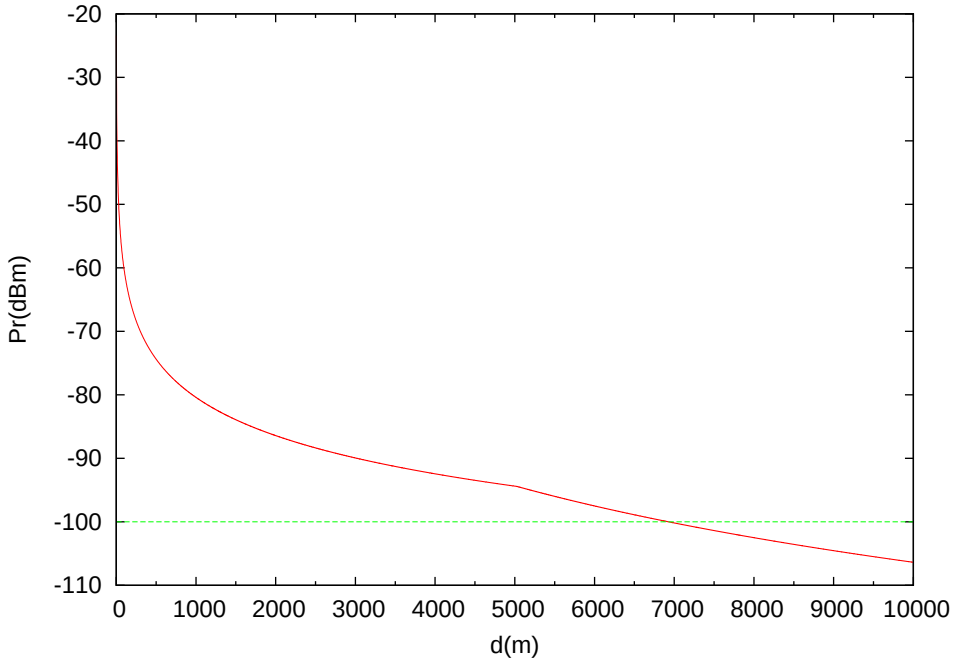

Figura 9.5: Potência recebida no Modelo de Propagação TwoRay

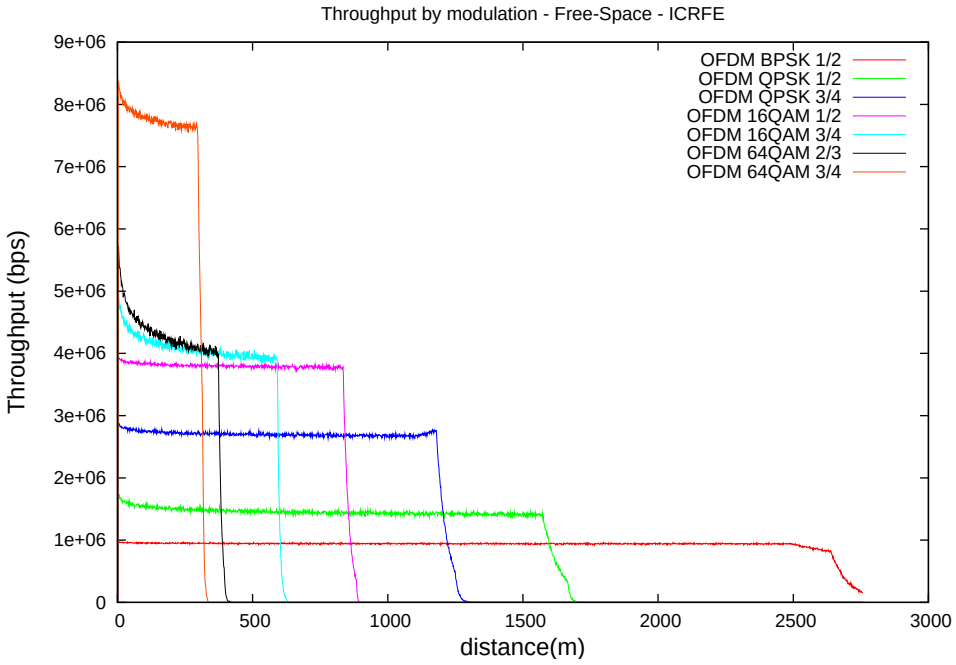

Figura 9.6: Throughput no Modelo ICRFE - FreeSpace

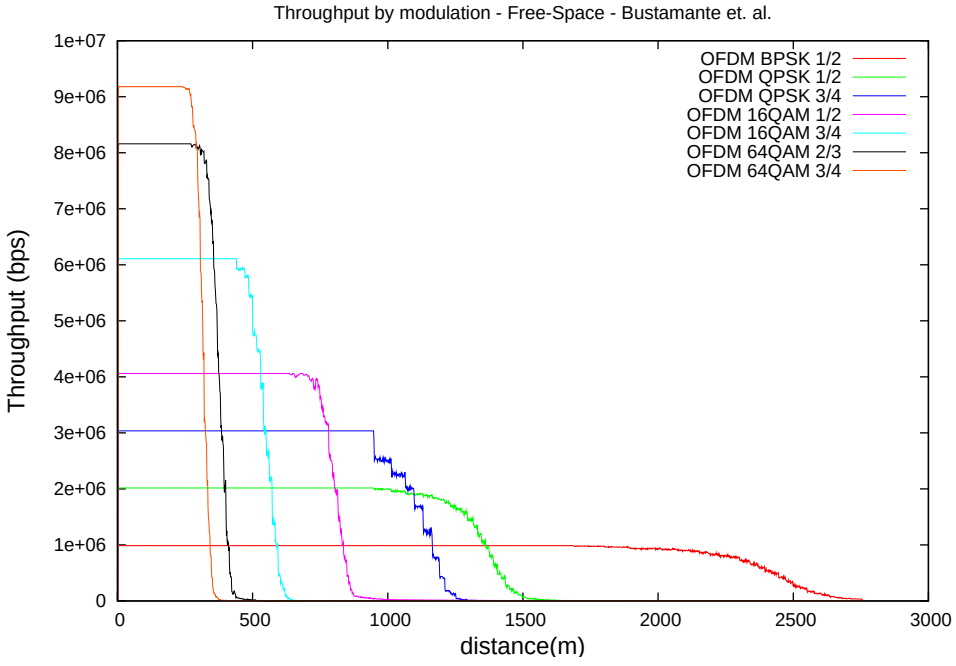

Figura 9.7: Throughput no Modelo Bustamante - FreeSpace

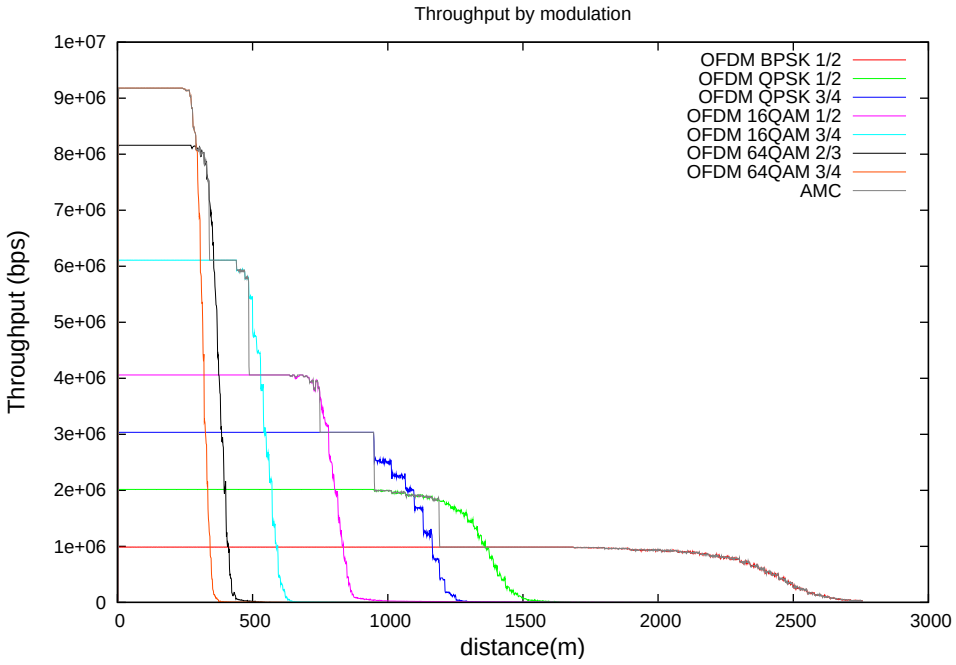

Figura 9.8: Throughput com o mecanismo de AMC no Modelo Bustamante - FreeSpace

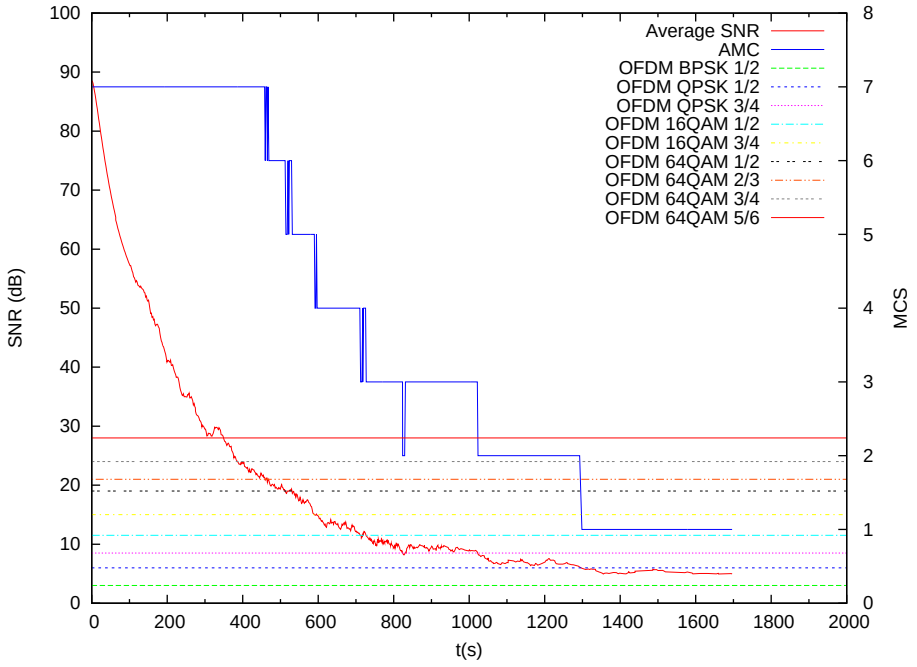

Figura 9.9: Potência recebida e AMC no Modelo SUI - Terreno A

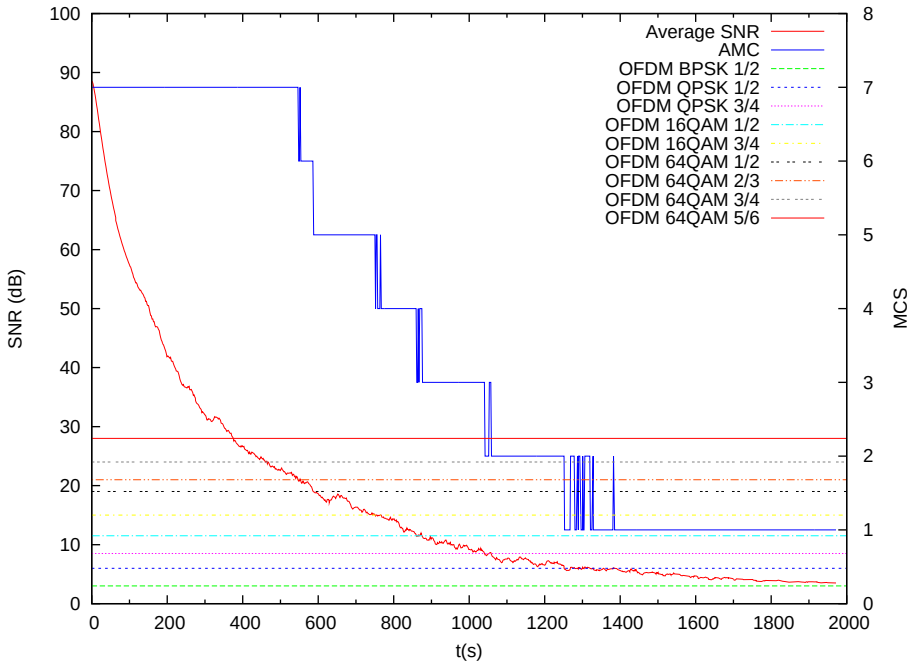

Figura 9.10: Potência recebida e AMC no Modelo SUI - Terreno B

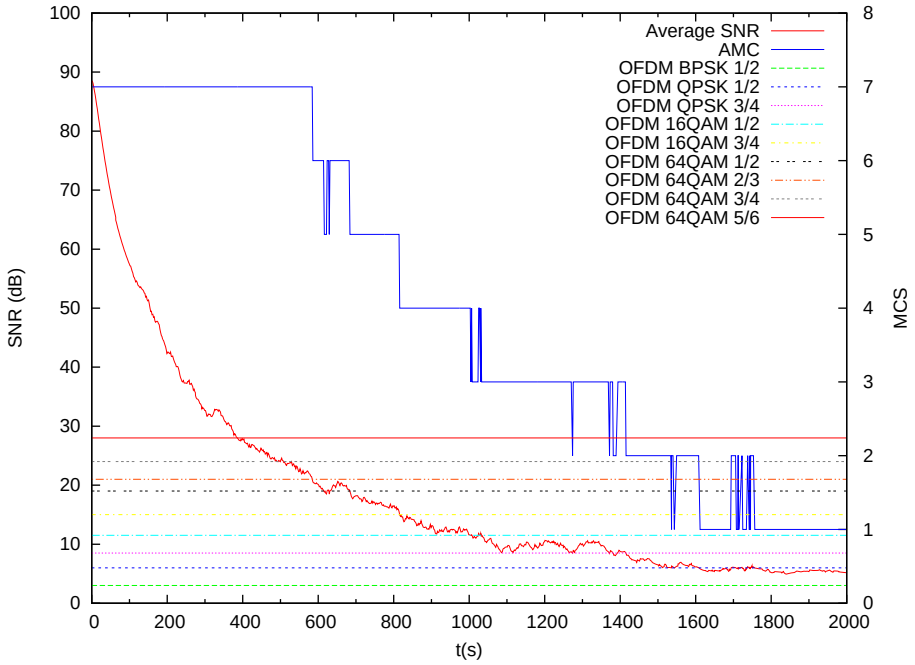

Figura 9.11: Potência recebida e AMC no Modelo SUI - Terreno C

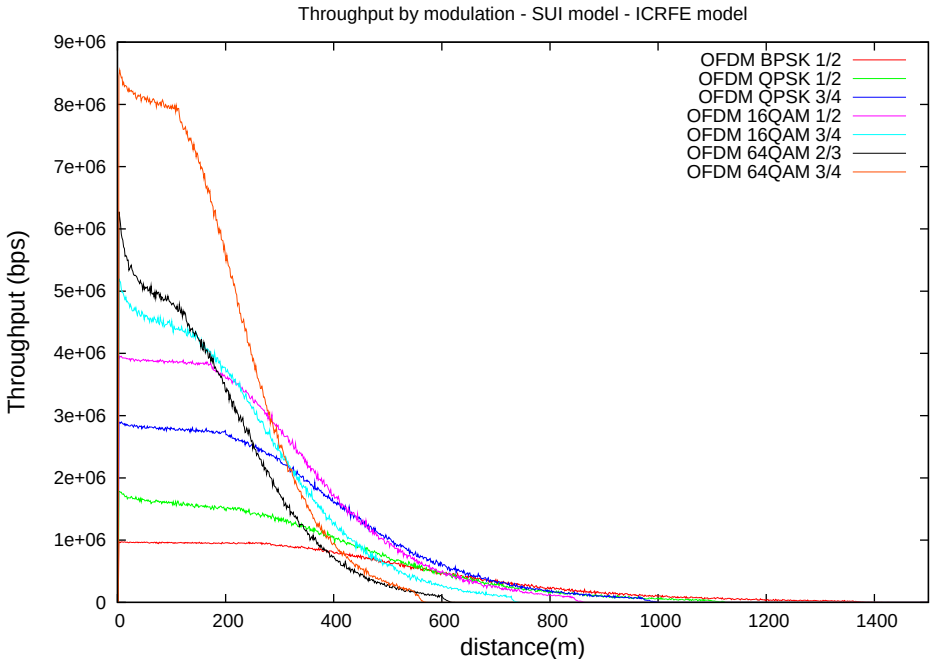

Figura 9.12: Throughput no Modelo ICRFE - modelo SUI

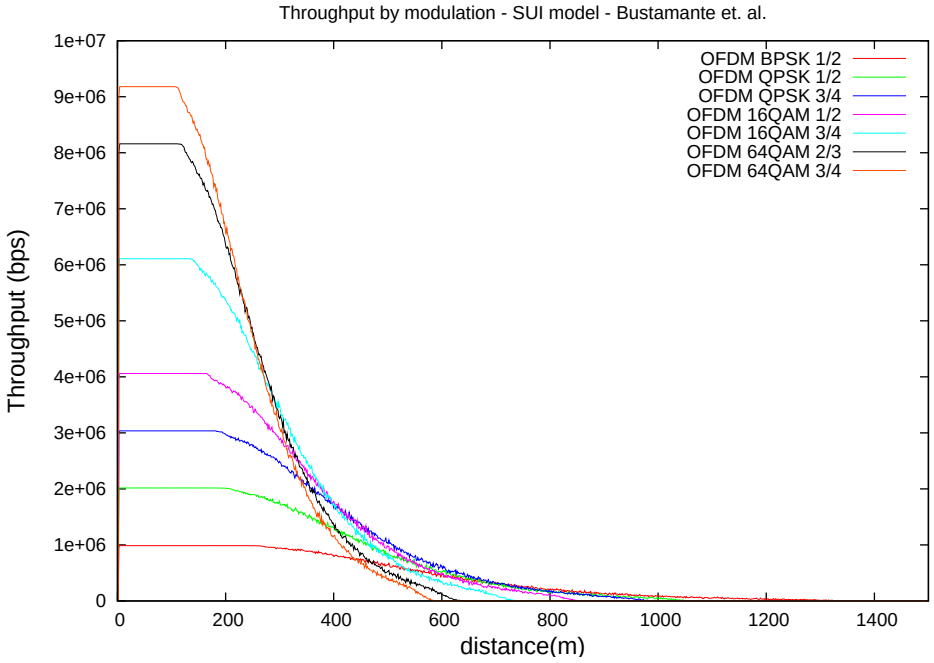

Figura 9.13: Throughput no Modelo Bustamante - modelo SUI

Capítulo 10

Conclusão

Este trabalho apresentou a implementação de uma extensão para um módulo de simulação de redes WiMAX.

A extensão proposta neste trabalho é capaz de realizar simulações de nível de *link* sem a complexidade das simulações *bit* a *bit*.

Os resultados apresentados mostram que os modelos implementado trazem mais realismo nas simulações de redes WiMAX e levam em conta os efeitos adversos que um canal sem fio sofre.

Com o modelo de propagação mais apropriado, o raio de cobertura da célula é mais realista. Já com o modelo de erros, a qualidade da rede cai progressivamente assim que a distância entre a BS e a SS aumenta.

Com o mecanismo de AMC, a BS é capaz de mudar automaticamente a modulação do canal com base na qualidade do canal, assim providenciando uma maneira de sempre manter a comunicação entre a BS e as SSs mesmo que a vazão sofra alguma perda. O importante não é ter uma comunicação excelente em grandes distâncias e sim ter alguma comunicação, mesmo que mínima, entre os nós comunicantes.

Mostramos também que o módulo WiMAX, com a implementação da camada MAC, da camada física realista e do modelo de canal, abre uma grande variedade de estudos em sistemas WiMAX, tornando-se uma importante contribuição no estudos de redes sem fio.

Apêndice A

Classes implementadas no ns-2

A linguagem utilizada para implementar os códigos foi a C++.

A.1 Classe erceg.cc

Classe que implementa o modelo de propagação SUI

```
#include <math.h>
#include <delay.h>
#include <packet.h>
#include <packet-stamp.h>
#include <antenna.h>
#include <mobilenode.h>
#include <propagation.h>
#include <wireless-phy.h>
#include <erceg.h>

const double Erceg::terrain_a[] = {4.6, 0.0075, 12.6, 0.57, 10.6, 2.3};
const double Erceg::terrain_b[] = {4.0, 0.0065, 17.1, 0.75, 9.6, 3.0};
const double Erceg::terrain_c[] = {3.6, 0.0050, 20.0, 0.59, 8.2, 1.6};
const double Erceg::terrain_d[] = {2.358, 0.00145, 0.45, 7.6, 0.35, 6.4};

int Erceg::init_x_z; // Indicates if X and Z has already been calculated
int Erceg::init_ranvar; // Indicates if RNG has already been initialized
float Erceg::x;
float Erceg::z;

static class ErcegClass: public TclClass
{
  public:
  ErcegClass() : TclClass("Propagation/Erceg") {}
  TclObject* create(int, const char*const*)
  {
     return (new Erceg);
```

```
  }
} class_erceg;

Erceg::Erceg()
{
  bind("terrain_type_", &terrain_type_); /* Terrain type - A, B, C, D or E */
  bind("dist0_", &dist0_); /* Distance 0 */
  bind("hb_", &hb_); /* Base Station height */
  bind("hm_", &hm_); /* Mobile Station height */
  bind("seed_", &seed_); /* Ranvar Seed */
  init_ranvar = 0; /* Initialization of Ranvar */
  init_x_z = 0; /* X and Y must be calculated only once */
  gama = 0.0; /* Path Loss exponent */
  s = 0.0; /* Shadow fading */
  old_dist = -1; /* Shadow fading must be calculated only for different distances */
  x = y = z = 0.0;
  ranVar = new RNG;
}

Erceg::~Erceg()
{
  delete ranVar;
}

double Erceg::Pr(PacketStamp *t, PacketStamp *r, WirelessPhy *ifp)
{
  double lambda_ = ifp->getLambda();    /* Wavelength */
  double freq_ = ifp->getFreq(); /* Frequency */
  double Xt, Yt, Zt; /* Location of transmitter */
  double Xr, Yr, Zr; /* Location of receiver */
  double A = 0.0; /* Free Space Path Loss */
  double PLf = 0.0; /* Frequency Path Loss */
  double PLh = 0.0; /* Antenna height Path Loss */
  double PL = 0.0; /* Total Path Loss */
  double Pr = 0.0; /* Total Power Received (in Watt) */
  /* Terrain type verification */
  if((terrain_type_ < 1) || (terrain_type_ > 4))
  {
     printf("Tipo de Terreno incompativel.\n");
     exit (1);
  }
  /* Get transmitter and receiver location */
```

```
t->getNode()->getLoc(&Xt, &Yt, &Zt);
r->getNode()->getLoc(&Xr, &Yr, &Zr);
Xr += r->getAntenna()->getX();
Yr += r->getAntenna()->getY();
Zr += r->getAntenna()->getZ();
Xt += t->getAntenna()->getX();
Yt += t->getAntenna()->getY();
Zt += t->getAntenna()->getZ();
/* Get distance */
double dX = Xr - Xt;
double dY = Yr - Yt;
double dZ = Zr - Zt;
double dist = sqrt(dX * dX + dY * dY + dZ * dZ);
/* Antenna gains */
double Gt = t->getAntenna()->getTxGain(dX, dY, dZ, lambda_);
double Gr = r->getAntenna()->getRxGain(dX, dY, dZ, lambda_);
/* For 0 distance return the transmitted power multplied by the gains */
if (dist == 0.0)
   return (t->getTxPr() * Gt * Gr);
/* Giving the seed to the RNG in the first iteration */
if (!Erceg::init_ranvar)
{
   ranVar->set_seed(RNG::PREDEF_SEED_SOURCE, seed_);
   init_ranvar = 1;
}
/* If the station has moved since the last packet, recalculate the random var y */
if (old_dist != dist)
{
   y = ranVar->normal(0.0, 1.0);
   // Limiting y between +-2
   if (y < -2.0) y = -2.0;
   if (y > 2.0) y = 2.0;
   old_dist = dist;
}
/* Path loss exponent must be calculated only once for a given macrocell */
if (!init_x_z)
{
   x = ranVar->normal(0.0, 1.0);
   z = ranVar->normal(0.0, 1.0);
   // Limiting z between +-2.0
   if (z < -2.0) z = -2.0;
   if (z > 2.0) z = 2.0;
```

```
    init_x_z = 1;
  }
  switch (terrain_type_)
  {
     case 1 :
     {
        if((hb_ < 10) || (hb_ > 80))
        {
           printf("Altura da Antena da Base Station incompativel.\n");
           exit (1);
        }
        if((hm_ < 0) || (hm_ > 10))
        {
           printf("Altura da Antena de Recepcao incompativel.\n");
           exit (1);
        }
        float x_calc = x * terrain_a[3];
        // Limiting x_calc between +-1.5
        if (x_calc < -1.5) x_calc = 1.5; if (x_calc > 1.5) x_calc = 1.5;
        /* Path Loss Exponent */
        gama = (terrain_a[0] - terrain_a[1]*hb_ + terrain_a[2]/hb_) +  x_calc;
        /* Shadow fading */
        s = y * (terrain_a[4] + z * terrain_a[5]);
        /* Path loss by frequency */
        PLf = 6 * log10(freq_ / 2.0e9);
        /* Path loss by height */
        if ((hm_ >= 2) && (hm_ <= 10))
           PLh = -10.8 * log10(hm_ / 2.0);
        break;
     }
     case 2 :
     {
        if((hb_ < 10) || (hb_ > 80))
        {
           printf("Altura da Antena da Base Station incompativel.\n");
           exit (1);
        }
        if((hm_ < 0) || (hm_ > 10))
        {
           printf("Altura da Antena de Recepcao incompativel.\n");
           exit (1);
        }
```

```
      float x_calc = x * terrain_b[3];
      // Limiting x_calc between +-1.5
      if (x_calc < -1.5) x_calc = 1.5;
      if (x_calc > 1.5) x_calc = 1.5;
      /* Path Loss Exponent */
      gama = (terrain_b[0] - terrain_b[1]*hb_ + terrain_b[2]/hb_) +  x_calc;
      /* Shadow fading */
      s = y * (terrain_b[4] + z * terrain_b[5]);
      /* Path loss by frequency */
      PLf = 6 * log10(freq_ / 2.0e9);
      /* Path loss by height */
      if ((hm_ >= 2) && (hm_ <= 10))
         PLh = -10.8 * log10(hm_ / 2.0);
      break;
   }
   case 3 :
   {
      if((hb_ < 10) || (hb_ > 80))
      {
         printf("Altura da Antena da Base Station incompativel.\n");
         exit (1);
      }
      if((hm_ < 0) || (hm_ > 10))
      {
         printf("Altura da Antena de Recepcao incompativel.\n");
         exit (1);
      }
      float x_calc = x * terrain_c[3];
      // Limiting x_calc between +-1.5
      if (x_calc < -1.5) x_calc = 1.5;
      if (x_calc > 1.5) x_calc = 1.5;
      /* Path Loss Exponent */
      gama = (terrain_c[0] - terrain_c[1]*hb_ + terrain_c[2]/hb_) +  x_calc;
      /* Shadow fading */
      s = y * (terrain_c[4] + z * terrain_c[5]);
      /* Path loss by frequency */
      PLf = 6 * log10(freq_ / 2.0e9);
      /* Path loss by height */
      if ((hm_ >= 2) && (hm_ <= 10))
         PLh = -10.8 * log10(hm_ / 2.0);
      break;
   }
```

```
     case 4 :
     {
        if((hb_ < 4) || (hb_ > 185))
        {
           printf("Altura da Antena da Base Station incompativel.\n");
            exit (1);
        }
        gama = (terrain_d[0] - terrain_d[1]*hb_ + terrain_d[2]/hb_);
        float std_dev_s = (terrain_d[3] - terrain_d[4]*pow(hb_,0.5) + terrain_d[5]
/pow(hb_,0.5));
        s = ranVar->normal(0.0, std_dev_s);
        /* Path loss by frequency */
        PLf = 6 * log10(freq_ / 2.0e9);
        break;
     }
  }
  /* Free Space Path Loss */
  A = FreeSpacePathLoss(lambda_, dist);
  /* If distance is smaller than d0, only FreeSpace path loss will be considered */
  if (dist < dist0_)
  {
     Pr = t->getTxPr() / pow(10.0, A / 10.0);
  }
  /* If distance is larger than d0, all path losses will be considered */
  else {
     PL = A + 10 * gama * log10((double)dist/dist0_) + s + PLf + PLh;
     Pr = t->getTxPr() / pow(10.0, PL / 10.0);
  }
  #ifdef DEBUG_ERCEG
     fprintf(stderr, "Packet received at (%f,%f). Dist = %e. Pr = %e. A = %e.
Gama = %e. s = %e. PLf = %e. PLh = %e.\n", Xr, Yr, dist, Pr, A, gama, s, PLf, PLh);
  #endif
  /* Return the power loss multiplied by the gains */
  return (Pr * Gt * Gr);
}

double Erceg::Pr_Dt(PacketStamp *t, PacketStamp *r, WirelessPhy *ifp)
{
  double lambda_ = ifp->getLambda();    /* Wavelength */
  double freq_ = ifp->getFreq(); /* Frequency */
  double Xt, Yt, Zt; /* Location of transmitter */
  double Xr, Yr, Zr; /* Location of receiver */
```

```
double A = 0.0; /* Free Space Path Loss */
double PLf = 0.0; /* Frequency Path Loss */
double PLh = 0.0; /* Antenna height Path Loss */
double PL = 0.0; /* Total Path Loss */
double Pr = 0.0; /* Total Power Received (in Watt) */
/* Terrain type verification */
if((terrain_type_ < 1) || (terrain_type_ > 4))
{
   printf("Tipo de Terreno incompativel.\n");
   exit (1);
}
/* Get transmitter and receiver location */
t->getNode()->getLoc(&Xt, &Yt, &Zt);
r->getNode()->getLoc(&Xr, &Yr, &Zr);
Xr += r->getAntenna()->getX();
Yr += r->getAntenna()->getY();
Zr += r->getAntenna()->getZ();
Xt += t->getAntenna()->getX();
Yt += t->getAntenna()->getY();
Zt += t->getAntenna()->getZ();
/* Get distance */
double dX = Xr - Xt;
double dY = Yr - Yt;
double dZ = Zr - Zt;
double dist = sqrt(dX * dX + dY * dY + dZ * dZ);
/* Antenna gains */
double Gt = t->getAntenna()->getTxGain(dX, dY, dZ, lambda_);
double Gr = r->getAntenna()->getRxGain(dX, dY, dZ, lambda_);
/* For 0 distance return the transmitted power multplied by the gains */
if (dist == 0.0)
   return t->getTxPr() * Gt * Gr;
/* Giving the seed to the RNG in the first iteration */
if (!Erceg::init_ranvar)
{
   ranVar->set_seed(RNG::PREDEF_SEED_SOURCE, seed_);
   init_ranvar = 1;
}
/* If the station has moved since the last packet, recalculate the random var y */
if (old_dist != dist)
{
   y = ranVar->normal(0.0, 1.0);
   // Limiting y between +-2
```

```
   if (y < -2.0) y = -2.0;
   if (y > 2.0) y = 2.0;
   old_dist = dist;
}
/* Path loss exponent must be calculated only once for a given macrocell */
if (!init_x_z)
{
   x = ranVar->normal(0.0, 1.0);
   z = ranVar->normal(0.0, 1.0);
   // Limiting z between +-2.0
   if (z < -2.0) z = -2.0;
   if (z > 2.0) z = 2.0;
   init_x_z = 1;
}
switch (terrain_type_)
{
   case 1 :
   {
      if((hb_ < 10) || (hb_ > 80))
      {
         printf("Altura da Antena da Base Station incompativel.\n");
         exit (1);
      }
      if((hm_ < 0) || (hm_ > 10))
      {
         printf("Altura da Antena de Recepcao incompativel.\n");
         exit (1);
      }
      float x_calc = x * terrain_a[3];
      // Limiting x_calc between +-1.5
      if (x_calc < -1.5) x_calc = 1.5; if (x_calc > 1.5) x_calc = 1.5;
      /* Path Loss Exponent */
      gama = (terrain_a[0] - terrain_a[1]*hb_ + terrain_a[2]/hb_) +  x_calc;
      /* Path loss by frequency */
      PLf = 6 * log10(freq_ / 2.0e9);
      /* Path loss by height */
      if ((hm_ >= 2) && (hm_ <= 10))
      {
         PLh = -10.8 * log10(hm_ / 2.0);
      }
      break;
   }
```

```
case 2 :
{
   if((hb_ < 10) || (hb_ > 80))
   {
      printf("Altura da Antena da Base Station incompativel.\n");
      exit (1);
   }
   if((hm_ < 0) || (hm_ > 10))
   {
      printf("Altura da Antena de Recepcao incompativel.\n");
      exit (1);
   }
   float x_calc = x * terrain_b[3];
   // Limiting x_calc between +-1.5
   if (x_calc < -1.5) x_calc = 1.5; if (x_calc > 1.5) x_calc = 1.5;
   /* Path Loss Exponent */
   gama = (terrain_b[0] - terrain_b[1]*hb_ + terrain_b[2]/hb_) +  x_calc;
   /* Shadow fading */
   /* Path loss by frequency */
   PLf = 6 * log10(freq_ / 2.0e9);
   /* Path loss by height */
   if ((hm_ >= 2) && (hm_ <= 10))
   {
      PLh = -10.8 * log10(hm_ / 2.0);
   }
   break;
}
case 3 :
{
   if((hb_ < 10) || (hb_ > 80))
   {
      printf("Altura da Antena da Base Station incompativel.\n");
      exit (1);
   }
   if((hm_ < 0) || (hm_ > 10))
   {
      printf("Altura da Antena de Recepcao incompativel.\n");
      exit (1);
   }
   float x_calc = x * terrain_c[3];
   // Limiting x_calc between +-1.5
   if (x_calc < -1.5) x_calc = 1.5; if (x_calc > 1.5) x_calc = 1.5;
```

```
        /* Path Loss Exponent */
        gama = (terrain_c[0] - terrain_c[1]*hb_ + terrain_c[2]/hb_) +  x_calc;
        /* Path loss by frequency */
        PLf = 6 * log10(freq_ / 2.0e9);
        /* Path loss by height */
        if ((hm_ >= 2) && (hm_ <= 10))
        {
           PLh = -10.8 * log10(hm_ / 2.0);
        }
        break;
     }
     case 4 :
     {
        if((hb_ < 4) || (hb_ > 185))
        {
           printf("Altura da Antena da Base Station incompativel.\n");
           exit (1);
        }
        gama = (terrain_d[0] - terrain_d[1]*hb_ + terrain_d[2]/hb_);
        /* Path loss by frequency */
        PLf = 6 * log10(freq_ / 2.0e9);
        break;
     }
  }
  /* Free Space Path Loss */
  A = FreeSpacePathLoss(lambda_, dist);
  /* If distance is smaller than d0, only FreeSpace path loss will be considered */
  if (dist < dist0_)
  {
     Pr = t->getTxPr() / pow(10.0, A / 10.0);
  }
  else
  {
     PL = A + 10 * gama * log10((double)dist/dist0_) + PLf + PLh;
     Pr = t->getTxPr() / pow(10.0, PL / 10.0);
  }
  #ifdef DEBUG_ERCEG
     printf("Packet received at (%f,%f). Dist = %e. Pr = %e. A = %e.
Gama = %e. s = %e. PLf = %e. PLh = %e.\n", Xr, Yr, dist, Pr, A, gama, s, PLf, PLh);
  #endif
  /* Return the power loss multiplied by the gains */
  return Pr * Gt * Gr;
```

```
}

int Erceg::command(int argc, const char* const* argv)
{
  if (argc == 4)
  {
     if (strcmp(argv[1], "seed") == 0)
     {
        int s = atoi(argv[3]);
        if (strcmp(argv[2], "raw") == 0)
        {
           ranVar->set_seed(RNG::RAW_SEED_SOURCE, s);
        }
        else if (strcmp(argv[2], "predef") == 0)
        {
           ranVar->set_seed(RNG::PREDEF_SEED_SOURCE, s);
        }
        else if (strcmp(argv[2], "heuristic") == 0)
        {
           ranVar->set_seed(RNG::HEURISTIC_SEED_SOURCE, 0);
        }
        return(TCL_OK);
     }
  }
  return Propagation::command(argc, argv);
}

double Erceg::FreeSpacePathLoss(double lambda, double d)
{
  if (d > dist0_)
  {
     // Erceg model A loss is for d <= 100.0
     d = 100.0;
  }
  double freePL = (4 * PI * d) / lambda;
  return (10 * log10(freePL*freePL));
}
```

A.2 Classe thresholdamccontroller.cc

Classe que implementa o AMC

```
#include "thresholdamccontroller.h"
#include "mac802_16.h"
```

```
#define N_PROFILES 9

/* Threshold table
 * UP_Threshold, Threshold, Down_Threshold, Modulation, UIUC Profile, DIUC Profile
 */

/* IEEE 802.16e standard recommendation */
const AMC_t SingleThresholdAMCController::Amc_Ofdm_Table_Standard[] = {
   {0.0, 3.0, 0.0, OFDM_BPSK_1_2,  UIUC_PROFILE_1, DIUC_PROFILE_1},
   {0.0, 6.0, 0.0, OFDM_QPSK_1_2,  UIUC_PROFILE_2, DIUC_PROFILE_2},
   {0.0, 8.5, 0.0, OFDM_QPSK_3_4,  UIUC_PROFILE_3, DIUC_PROFILE_3},
   {0.0, 11.5, 0.0, OFDM_16QAM_1_2, UIUC_PROFILE_4, DIUC_PROFILE_4},
   {0.0, 15.0, 0.0, OFDM_16QAM_3_4, UIUC_PROFILE_5, DIUC_PROFILE_5},
   {0.0, 19.0, 0.0, OFDM_64QAM_1_2, UIUC_PROFILE_6, DIUC_PROFILE_6},
   {0.0, 21.0, 0.0, OFDM_64QAM_2_3, UIUC_PROFILE_7, DIUC_PROFILE_7},
   {0.0, 24.0, 0.0, OFDM_64QAM_3_4, UIUC_PROFILE_8, DIUC_PROFILE_8},
   {0.0, 28.0, 0.0, OFDM_64QAM_5_6, UIUC_PROFILE_9, DIUC_PROFILE_9}
};

/* Vienna University - Christian Mehlfuhrer */
const AMC_t SingleThresholdAMCController::Amc_Ofdm_Table_Vienna[] = {
   {0.0, -2.0, 0.0, OFDM_BPSK_1_2,  UIUC_PROFILE_1, DIUC_PROFILE_1},
   {0.0, 3.0, 0.0, OFDM_QPSK_1_2,  UIUC_PROFILE_2, DIUC_PROFILE_2},
   {0.0, 5.5, 0.0, OFDM_QPSK_3_4,  UIUC_PROFILE_3, DIUC_PROFILE_3},
   {0.0, 8.5, 0.0, OFDM_16QAM_1_2, UIUC_PROFILE_4, DIUC_PROFILE_4},
   {0.0, 12.0, 0.0, OFDM_16QAM_3_4, UIUC_PROFILE_5, DIUC_PROFILE_5},
   {0.0, 15.5, 0.0, OFDM_64QAM_1_2, UIUC_PROFILE_6, DIUC_PROFILE_6},
   {0.0, 15.5, 0.0, OFDM_64QAM_2_3, UIUC_PROFILE_7, DIUC_PROFILE_7},
   {0.0, 17.5, 0.0, OFDM_64QAM_3_4, UIUC_PROFILE_8, DIUC_PROFILE_8},
   {0.0, 1500.0, 0.0, OFDM_64QAM_5_6, UIUC_PROFILE_9, DIUC_PROFILE_9}
};

/* Bustamante */
/* SNR para PER de 0.2 e pacotes de 150 bytes */
const AMC_t SingleThresholdAMCController::Amc_Ofdm_Table_Bustamante[] = {
   {0.0, 1.46, 0.0, OFDM_BPSK_1_2,  UIUC_PROFILE_1, DIUC_PROFILE_1},
   {0.0, 6.33, 0.0, OFDM_QPSK_1_2,  UIUC_PROFILE_2, DIUC_PROFILE_2},
   {0.0, 8.31, 0.0, OFDM_QPSK_3_4,  UIUC_PROFILE_3, DIUC_PROFILE_3},
   {0.0, 10.72, 0.0, OFDM_16QAM_1_2, UIUC_PROFILE_4, DIUC_PROFILE_4},
   {0.0, 14.46, 0.0, OFDM_16QAM_3_4, UIUC_PROFILE_5, DIUC_PROFILE_5},
   {0.0, 17.52, 0.0, OFDM_64QAM_1_2, UIUC_PROFILE_6, DIUC_PROFILE_6},
```

```
   {0.0, 17.52, 0.0, OFDM_64QAM_2_3, UIUC_PROFILE_7, DIUC_PROFILE_7},
   {0.0, 19.00, 0.0, OFDM_64QAM_3_4, UIUC_PROFILE_8, DIUC_PROFILE_8},
   {0.0, 1500.0, 0.0, OFDM_64QAM_5_6, UIUC_PROFILE_9, DIUC_PROFILE_9}
};

/* IEEE 802.16e standard recommendation */
const AMC_t MultipleThresholdAMCController::Amc_Ofdm_Table_Standard[] = {
   {4.0, 3.0, 2.0, OFDM_BPSK_1_2,  UIUC_PROFILE_1, DIUC_PROFILE_1},
   {7.0, 6.0, 5.0, OFDM_QPSK_1_2,  UIUC_PROFILE_2, DIUC_PROFILE_2},
   {9.5, 8.5, 7.5, OFDM_QPSK_3_4,  UIUC_PROFILE_3, DIUC_PROFILE_3},
   {12.5, 11.5, 10.5, OFDM_16QAM_1_2, UIUC_PROFILE_4, DIUC_PROFILE_4},
   {16.0, 15.0, 14.0, OFDM_16QAM_3_4, UIUC_PROFILE_5, DIUC_PROFILE_5},
   {19.5, 19.0, 18.0, OFDM_64QAM_1_2, UIUC_PROFILE_6, DIUC_PROFILE_6},
   {22.0, 21.0, 20.0, OFDM_64QAM_2_3, UIUC_PROFILE_7, DIUC_PROFILE_7},
   {25.0, 24.0, 23.0, OFDM_64QAM_3_4, UIUC_PROFILE_8, DIUC_PROFILE_8},
   {29.0, 28.0, 27.0, OFDM_64QAM_5_6, UIUC_PROFILE_9, DIUC_PROFILE_9}
};

/* Vienna University - Christian Mehlfuhrer */
const AMC_t MultipleThresholdAMCController::Amc_Ofdm_Table_Vienna[] = {
   {-3.0, -2.0, -1.0, OFDM_BPSK_1_2,  UIUC_PROFILE_1, DIUC_PROFILE_1},
   {2.0, 3.0, 4.0, OFDM_QPSK_1_2,  UIUC_PROFILE_2, DIUC_PROFILE_2},
   {4.5, 5.5, 6.5, OFDM_QPSK_3_4,  UIUC_PROFILE_3, DIUC_PROFILE_3},
   {7.5, 8.5, 9.5, OFDM_16QAM_1_2, UIUC_PROFILE_4, DIUC_PROFILE_4},
   {11.0, 12.0, 13.0, OFDM_16QAM_3_4, UIUC_PROFILE_5, DIUC_PROFILE_5},
   {14.5, 15.5, 16.5, OFDM_64QAM_1_2, UIUC_PROFILE_6, DIUC_PROFILE_6},
   {14.5, 15.5, 16.5, OFDM_64QAM_2_3, UIUC_PROFILE_7, DIUC_PROFILE_7},
   {16.5, 17.5, 18.5, OFDM_64QAM_3_4, UIUC_PROFILE_8, DIUC_PROFILE_8},
   {1500.0, 1500.0, 1500.0, OFDM_64QAM_5_6, UIUC_PROFILE_9, DIUC_PROFILE_9}
};

/* Bustamante */
const AMC_t MultipleThresholdAMCController::Amc_Ofdm_Table_Bustamante[] = {
   {3.0, 4.0, 5.0, OFDM_BPSK_1_2,  UIUC_PROFILE_1, DIUC_PROFILE_1},
   {8.0, 9.0, 10.0, OFDM_QPSK_1_2,  UIUC_PROFILE_2, DIUC_PROFILE_2},
   {9.0, 10.0, 11.0, OFDM_QPSK_3_4,  UIUC_PROFILE_3, DIUC_PROFILE_3},
   {13.5, 14.5, 15.5, OFDM_16QAM_1_2, UIUC_PROFILE_4, DIUC_PROFILE_4},
   {15.0, 16.0, 17.0, OFDM_16QAM_3_4, UIUC_PROFILE_5, DIUC_PROFILE_5},
   {19.5, 20.5, 21.5, OFDM_64QAM_1_2, UIUC_PROFILE_6, DIUC_PROFILE_6},
   {19.5, 20.5, 21.5, OFDM_64QAM_2_3, UIUC_PROFILE_7, DIUC_PROFILE_7},
   {20.5, 21.5, 22.5, OFDM_64QAM_3_4, UIUC_PROFILE_8, DIUC_PROFILE_8},
   {1500.0, 1500.0, 0.0, OFDM_64QAM_5_6, UIUC_PROFILE_9, DIUC_PROFILE_9}
```

```
};

/*
 ********** Single Threshold AMC Controller **********
 */
static class SingleThresholdAMCControllerClass : public TclClass {
   public:
   SingleThresholdAMCControllerClass() : TclClass("AMCController/SingleThreshold") {}
   TclObject* create(int, const char*const*)
   {
      return (new SingleThresholdAMCController());
   }
} class_singlethresholdamccontroller;

/*
 * Create a Simgle Threshold AMC Controller
 */
SingleThresholdAMCController::SingleThresholdAMCController ()
{

}

/*
 * Interface with the TCL script
 * @param argc The number of parameter
 * @param argv The list of parameters
 */
int SingleThresholdAMCController::command(int argc, const char*const* argv)
{
   return TCL_OK;
}

/**
 * Initializes the scheduler
 */
void SingleThresholdAMCController::init ()
{

}

Ofdm_mod_rate SingleThresholdAMCController::getOfdm_mod_rate(PeerNode * peer)
{
```

```
   return OFDM_BPSK_1_2;
}

uiuc_t SingleThresholdAMCController::getOfdm_uiuc(PeerNode * peer)
{
   return UIUC_PROFILE_1;
}

diuc_t SingleThresholdAMCController::getOfdm_diuc(PeerNode * peer)
{
   double sinr = peer->getSINRWatch()->average();
   diuc_t diuc_old = (diuc_t)peer->getDIUC();
   diuc_t diuc_temp = DIUC_PROFILE_1;
   /* get error model*/
   Error_model error_model_ = ErrorRateModel::instance()->getModel();
   for (int i = 0; i < N_PROFILES; i++)
   {
      if (getAMCThreshold(i, error_model_).threshold <= sinr)
      {
         diuc_temp = getAMCThreshold(i, error_model_).diuc;
      }
   }
   if (diuc_old != diuc_temp) printf("Changing modulation: %d %d at %f\n",diuc_old,
diuc_temp, NOW);
      return diuc_temp;
}

AMC_t SingleThresholdAMCController::getAMCThreshold(int index, Error_model error_model_)
{
   /* Return threshold value acording to the error model */
   if (error_model_ == STANDARD)
   {
      return Amc_Ofdm_Table_Standard[index];
   } else if(error_model_ == DEFAULT)
   {
      return Amc_Ofdm_Table_Vienna[index];
   } else if(error_model_ == BUSTAMANTE)
   {
      return Amc_Ofdm_Table_Bustamante[index];
   }
}
```

```
/**
 ********** Multiple Threshold AMC Controller **********
 */
static class MultipleThresholdAMCControllerClass : public TclClass
{
   public:
   MultipleThresholdAMCControllerClass() : TclClass("AMCController/MultipleThreshold") {}
   TclObject* create(int, const char*const*)
   {
      return (new MultipleThresholdAMCController());
   }
} class_multiplethresholdamccontroller;

/*
 * Create a Simgle Threshold AMC Controller
 */
MultipleThresholdAMCController::MultipleThresholdAMCController ()
{

}

/*
 * Interface with the TCL script
 * @param argc The number of parameter
 * @param argv The list of parameters
 */
int MultipleThresholdAMCController::command(int argc, const char*const* argv)
{
   return TCL_OK;
}

/**
 * Initializes the scheduler
 */
void MultipleThresholdAMCController::init ()
{

}

Ofdm_mod_rate MultipleThresholdAMCController::getOfdm_mod_rate(PeerNode * peer)
{
   return OFDM_BPSK_1_2;
```

```
}

uiuc_t MultipleThresholdAMCController::getOfdm_uiuc(PeerNode * peer)
{
   return UIUC_PROFILE_1;
}

diuc_t MultipleThresholdAMCController::getOfdm_diuc(PeerNode * peer)
{
   double sinr = peer->getSINRWatch()->average();
   diuc_t diuc_old = (diuc_t)peer->getDIUC();
   diuc_t diuc_temp = DIUC_PROFILE_1;
   /* get error model*/
   Error_model error_model_ = ErrorRateModel::instance()->getModel();
   int index = 0;
   for (int i = 0; i < N_PROFILES; i++)
   {
      if (getAMCThreshold(i, error_model_).threshold <= sinr)
      {
         diuc_temp = getAMCThreshold(i, error_model_).diuc;
         index = i;
      }
   }
   if (index > 0)
   {
      // Se estiver na faixa acima do Threshold e abaixo do Up_Threshold
      if (sinr <= getAMCThreshold(index, error_model_).up_threshold)
      {
         // Se vier de baixo
         if(diuc_old < getAMCThreshold(index, error_model_).diuc)
         {
            if (diuc_old != diuc_temp) printf("Changing modulation: %d %d at %f\n",diuc_old,
diuc_temp, NOW);
               return getAMCThreshold(index-1, error_model_).diuc;
         }
         else
         {
            // Se vier de cima
            if (diuc_old != diuc_temp) printf("Changing modulation: %d %d at %f\n",diuc_old,
diuc_temp, NOW);
               return diuc_temp;
         }
```

```
      }
   }
   if (index < N_PROFILES-1)
   {
      // Se estiver na faixa acima do Threshold_Down e abaixo do proximo Threshold
      if (sinr >= getAMCThreshold(index+1, error_model_).down_threshold)
      {
         // Se vier de baixo
         if(diuc_old <= getAMCThreshold(index, error_model_).diuc)
         {
            if (diuc_old != diuc_temp) printf("Changing modulation: %d %d at %f\n",diuc_old,
diuc_temp, NOW);
               return diuc_temp;
         }
         else
         {
            // Se vier de cima
            if (diuc_old != diuc_temp) printf("Changing modulation: %d %d at %f\n",diuc_old,
diuc_temp, NOW);
               return getAMCThreshold(index+1, error_model_).diuc;
         }
      }
   }
   if (diuc_old != diuc_temp) printf("Changing modulation: %d %d at %f\n",diuc_old,
diuc_temp, NOW);
      return diuc_temp;
}

AMC_t MultipleThresholdAMCController::getAMCThreshold(int index, Error_model error_model_)
{
   /* Return threshold value acording to the error model */
   if (error_model_ == STANDARD)
   {
      return Amc_Ofdm_Table_Standard[index];
   } else if(error_model_ == DEFAULT) {
      return Amc_Ofdm_Table_Vienna[index];
   } else if(error_model_ == BUSTAMANTE) {
      return Amc_Ofdm_Table_Bustamante[index];
   }
}
```

Referências Bibliográficas

[1] *The network simulator - ns-2*, http://www.isi.edu/nsnam/ns/.

[2] *Ieee standard for local and metropolitan area networks part 16: Air interface for fixed broadband wireless access systems*, IEEE Std 802.16-2004 (2004).

[3] Jeffrey G. Andrews, Arunabha Ghosh, and Tias Muhamed, *Fundamentals of wimax: Understanding broadband wireless networking*, Fevereiro 2007.

[4] L. Betancur, R. C. Hincapi, and R. Bustamante, *Wimax channel: Phy model in network simulator 2*, WNS2'06: Proceeding from the 2006 workshop on ns-2: the IP network simulator (2006), 6.

[5] J. Chen, C. C. Wang, F. C. D. Tsai, C. W. Chang, S. S. Liu, J. Guo, W. J. Lien, and J. H. Sum e C. H. Hung, *The design and implementation of wimax module for ns-2 simulator*, WNS2'06: Proceeding from the 2006 workshop on ns-2: the IP network simulator (2006), 5.

[6] V. Erceg, L. Greenstein, S. Tjandra, S. Parkoff, A. Guptam, B. Kulic, A. Julius, and R. Bianchi, *An empirically based path loss model for wireless channels in suburban environments*, Selected Areas in Communications, IEEE Journal **17** (1999), 1205–1211.

[7] J. Freitag and N. L. S. da Fonseca, *Wimax module for the ns-2 simulator*, **Set.** (2007), 1–6.

[8] National Institute of Standards and Tecnology, *The network simulator ns-2 nist add-on - ieee 802.16 model (mac+phy)*, (2007).

[9] S. Ramachandran, C. W. Bostian, and S. F. Midkiff, *A link adaptation algorithm for ieee 802.16*, IEEE Wireless Communications and Networking Conference **3** (2005), no. Mar, 1466–1471.

[10] Tsz, *A link adaptation algorithm in mimo-based wimax systems*, Journal of Communications **2** (2007), no. 5.

[11] Yu-Ting Yu and Hsi-Lu Chao, *Qos-aware link rate adaptation in ieee 802.16 broadbamd wireless access systems*, 18th Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (2007).