Universidade do Estado do Rio de Janeiro

Centro de Tecnologia e Ciência

Faculdade de Engenharia

Roberto Claudio Rodrigues Machado

Algoritmo adaptativo de *backoff* baseado em controle de vizinhos ativos para redes IEEE 802.11

Rio de Janeiro

2016

Roberto Claudio Rodrigues Machado

Algoritmo adaptativo de *backoff* baseado em controle de vizinhos ativos para redes IEEE 802.11

Dissertação apresentada, como requisito parcial para obtenção do título de Mestre em Ciências, ao Programa de Pós-Graduação em Engenharia Eletrônica, da Universidade do Estado do Rio de Janeiro. Área de concentração: Redes de Telecomunicações

Orientador: Prof. Dr. Marcelo Gonçalves Rubinstein

Rio de Janeiro

2016

CATALOGAÇÃO NA FONTE

UERJ / REDE SIRIUS / BIBLIOTECA CTC/B

C837 Machado, Roberto Claudio Rodrigues.

Algoritmo adaptativo de backoff baseado em controle de vizinhos ativos para redes IEEE 802.11 / Roberto Claudio Rodrigues Machado. - 2016.

73 f.

Orientador: Marcelo Gonçalves Rubinstein.

Dissertação (Mestrado) – Universidade do Estado do Rio de Janeiro, Faculdade de Engenharia.

1. Engenharia Eletrônica. I. Rubinstein, Marcelo Gonçalves. II. Universidade do Estado do Rio de Janeiro. III. Título.

CDU 004.05

_____ _____

Assinatura Data

Roberto Claudio Rodrigues Machado

Algoritmo adaptativo de *backoff* baseado em controle de vizinhos ativos para redes IEEE 802.11

Dissertação apresentada, como requisito parcial para obtenção do título de Mestre em Ciências, ao Programa de Pós-Graduação em Engenharia Eletrônica, da Universidade do Estado do Rio de Janeiro. Área de concentração: Redes de Telecomunicações

Aprovado em X de X de 2016.

Banca Examinadora:

_____

Prof. Dr. Marcelo Gonçalves Rubinstein (Orientador)

Faculdade de Engenharia – UERJ

_____

Prof. Dr. Miguel Elias Mitre Campista

Universidade Federal do Rio de Janeiro – UFRJ

_____

Prof. Dr. Rodrigo de Souza Couto

Faculdade de Engenharia – UERJ

Rio de Janeiro
2016

DEDICATÓRIA

Dedico este trabalho a todas as pessoas que acreditaram na minha capacidade de alcançar meus objetivos, em exclusivo ao meu pai, João, que colocou todos os esforços durante sua vida profissional para me criar em meio às grandes dificuldades que a vida lhe proporcionou.

AGRADECIMENTOS

Agradeço ao professor Marcelo G. Rubinstein, meu orientador, pela paciência apresentada durante todo o período do Mestrado, bem como, pela experiência e conhecimentos compartilhados, além do suporte durante todo o período do curso.

Agradeço à minha família por ter aguentado dias e dias enclausurados em função da minha necessidade de estudo e foco, durante esta caminhada rumo à vitória.

Agradeço aos professores Marcelo G. Rubinstein e Alexandre Sztajnberg, representantes do programa de Pós-Graduação em Engenharia Eletrônica da UERJ, por terem me dado esta oportunidade durante a fase de avaliação e entrevistas.

Agradeço também ao amigo Carlos Bello por todo suporte, indicação e orientação, essenciais durante a fase inicial do curso.

Agradeço em especial a Deus por ter me proporcionado essa experiência junto com todo o corpo docente da UERJ, com o qual tive interação durante todo esse período.

RESUMO

MACHADO, R. C. R. *Algoritmo adaptativo de backoff baseado em controle de vizinhos ativos para redes IEEE 802.11*. 2016. 73 f. Dissertação (Mestrado em Engenharia Eletrônica) – Faculdade de Engenharia, Universidade do Estado do Rio de Janeiro, Rio de Janeiro, 2016.

Em redes sem fio padrão IEEE 802.11, o algoritmo de *backoff* da subcamada MAC desempenha um papel significativo na coordenação distribuída de estações competindo pelo acesso ao meio. Mesmo após estudos e aprimoramentos diversos, problemas de excessivas colisões e de injustiça que causa inanição definitiva ou prolongada de uma ou mais estações ainda ocorrem, principalmente em cenários de saturação. O objetivo desta dissertação é avaliar o desempenho de uma nova proposta de algoritmo de *backoff* denominado nMBEB (*Modified Binary Exponential Backoff algorithm - with node control*). O algoritmo nMBEB fornece uma melhor adaptação às variações de carga da rede através do monitoramento do número de estações ativas no alcance de uma estação remetente. O algoritmo nMBEB aplica diferentes fatores multiplicativos nas funções de incremento e decremento da janela de contenção do algoritmo de *backoff*. Os resultados das simulações, obtidos através do simulador NS-2, mostram que o algoritmo nMBEB possui melhor desempenho em termos de vazão e justiça quando comparado com o tradicional algoritmo exponencial binário de *backoff* (BEB) do IEEE 802.11 e com outros algoritmos da literatura em cenários de saturação da rede. A melhoria no desempenho também ocorre em cenários nos quais existe um grande número de estações iniciando seus respectivos tráfegos de dados de maneira assíncrona.

Keywords: Redes sem Fio, 802.11, MAC, *Backoff*, Desempenho, Justiça

ABSTRACT

MACHADO, R. C. R. *Adaptative Backoff Algorithm based on active neighbors control for IEEE 802.11 networks*. 2016. 73 f. Dissertação (Mestrado em Engenharia Eletrônica) – Faculdade de Engenharia, Universidade do Estado do Rio de Janeiro, Rio de Janeiro, 2016.

On IEEE 802.11 wireless networks, the backoff algorithm of the MAC sublayer plays a significant role on the distributed coordination of the stations competing for the media access. Even after studies and several enhancements, problems of excessive collisions and injustice that cause permanent or prolonged starvation of one or more stations still occur, especially in saturation scenarios. The purpose of this work is to evaluate the performance of a new backoff algorithm proposal named nMBEB (Modified Binary Exponential Backoff Algorithm - with node control). The nMBEB algorithm provides a better adaptation to network load variation through the monitoring of the number of active stations in range of the sender station. The nMBEB algorithm uses different multiplicative factors into contention window increment and decrement functions of the backoff algorithm. Simulation results obtained using the NS-2 simulator show that the nMBEB algorithm achieves an enhancement on throughput and fairness performance when compared with traditional Binary Exponential Backoff (BEB) algorithm of IEEE 802.11 and with other algorithms from the literature in network saturation scenarios. The performance improvement also happens in scenarios where a large number of stations asynchronously initiate their respective data flows.

Keywords: Wireless, 802.11, MAC, *Backoff*, Performance, Fairness

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

AP	*Access Point*
BEB	*Binary Exponential Backoff*
BSS	*Basic Service Set*
BSSID	*Basic Service Set ID*
CBR	*Constant Bit Rate*
CP	*Contention Period*
CPU	*Central Processing Unit*
CRC	*Cyclic Redundancy Check*
CSMA/CA	*Carrier Sense Multiple Access with Collision Avoidance*
CTS	*Clear To Send*
CW	*Contention Window*
DBA	*Dynamic Backoff Algorithm*
DCF	*Distributed Coordination Function*
DDR	*Double Data Rate*
DIDD	*Double Increment Double Decrement*
DIFS	*Distributed InterFrame Space*
DSSS	*Direct Sequence Spread Spectrum*
EIED	*Exponential Increase Exponential Decrease*
FAMA	*Floor Acquisition Multiple Access*
FCS	*Frame Check Sequence*
IBSS	*Independent Basic Service Set*
IEEE	*Institute of Electric and Electronics Engineering*
IFS	*InterFrame Space*
ISO	*International Organization for Standardization*
LLC	*Logic Link Control*
MAC	*Medium Access Control*
MACA	*Multiple Access with Collision Avoidance*
MACAW	*Multiple Access with Collision Avoidance for Wireless*
MBEB	*Modified Binary Exponential Backoff*
MILD	*Multiple Increment Linear Decrement*
NAV	*Network Allocation Vector*
NS	*Network Simulator*
NSG	*Network Scenario Generator*
OSI	*Open Systems Interconnection*
PCF	*Point Coordination Function*
PLC	*Power Line Communications*
RAM	*Random Access Memory*
RTS	*Request To Send*
SIFS	*Short InterFrame Space*
SLRC	*Station Short Retry Count*
SSD	*Solid State Drive*
SSRC	*Station Long Retry Count*
TCL	*Tool Command Language*
TCP	*Transmission Control Protocol*
UDP	*User Datagram Protocol*
Wi-Fi	*Wireless Fidelity*
WLAN	*Wireless Local Area Network*

SUMÁRIO

INTRODUÇÃO

Desde a década de 1990, quando o IEEE (*Institute of Electric and Electronics Engineering*) padronizou a comunicação local sem fio (*Wireless Local Area Network* – WLAN) comercializada como *Wi-Fi* (*Wireless Fidelity*), que a camada de enlace e suas subcamadas de acesso ao meio (*Medium Access Control* – MAC) e de controle lógico (*Logic Link Control* – LLC), são alvos de estudos diversos, a fim de melhorar a qualidade de serviço e o desempenho da rede.

Inicialmente parece ser simples aprimorar a eficiência de uma rede, agindo diretamente na camada que cuida das comunicações ponto-a-ponto ou em difusão entre as estações. No entanto, em cenários de rede sem fio, além das características de canais sem fio, como maior atenuação, e problemas de terminal escondido e exposto, existem variações do número de estações e/ou deslocamentos constantes pertinentes a mobilidade, que dificultam a comunicação entre as estações. Essas caraterísticas também criam objetos de estudo em algoritmos ou funções que integram a subcamada MAC.

Com todos estes problemas pertinentes aos cenários de rede sem fio, a subcamada MAC ainda precisa coordenar a comunicação entre estações, que transmitem simultaneamente, para minimizar colisões e garantir o envio de quadros de maneira eficiente. Para evitar colisões e fornecer um ambiente "justo", a subcamada MAC define temporizadores que são utilizados por cada estação para garantir a coordenação durante a transmissão de quadros. Ao iniciar uma tentativa de transferência de um quadro, cada estação verifica o estado da rede sem fio antes de enviá-lo. Se o meio físico estiver livre por um tempo específico, a estação inicia a transmissão. Se estiver ocupado, a estação deve esperar até o meio físico tornar-se livre durante um tempo predefinido. Após este tempo, a estação novamente espera por um intervalo de tempo aleatório, denominado tempo de *backoff*, antes de iniciar a transmissão do quadro.

Seja no modo básico ou no modo *four-way handshaking* com a adição dos quadros de controle RTS (*Request To Send*) e CTS (*Clear To Send*), o método CSMA/CA (*Carrier Sense Multiple Access with Collision Avoidance*), responsável pela coordenação do acesso de múltiplas estações a um meio físico compartilhado e concorrente, não garante a eliminação total das colisões e a não exclusividade de utilização do meio. Por esse motivo, em muitas

vertentes de estudo da subcamada MAC em redes sem fio, algoritmos diferenciados de *backoff* são propostos com o objetivo de resolver problemas de excessivas colisões e de injustiça que causa inanição[1] definitiva ou prolongada de uma ou mais estações. Ambos os problemas afetam diretamente o desempenho em termos de vazão do sistema e são agravados em cenários de saturação, ou seja, cenários onde se injeta tráfego a uma taxa superior à capacidade do canal.

Inicialmente, o objetivo deste trabalho é avaliar as funções de incremento e decremento da janela de contenção (*Contention Window* – CW) que compõem os mecanismos de *backoff* e identificar a melhor abordagem para essas funções em cenários de saturação, seguindo os critérios de avaliação de desempenho de vazão e justiça. As funções de incremento são responsáveis por aumentar o valor de CW sempre que há uma colisão na transmissão de um quadro e a função de decremento é responsável por reduzir o valor de CW sempre que há sucesso no envio de um quadro.

Em estudos anteriores (BIANCHI, 2000; WATTANAMONGKHOL et al, 2007), a variação do número de estações na rede é considerada como uma variável não-controlável diretamente. No entanto, alguns mecanismos, assim como o que será proposto neste trabalho, permitem indiretamente monitorar a variação do número de estações ao longo do tempo, a fim de realizar procedimentos diferenciados no algoritmo tradicional de *backoff*.

Este trabalho propõe um novo algoritmo de *backoff* denominado nMBEB (*Modified Binary Exponential Backoff algorithm - with node control*), que utiliza dados passados de estações vizinhas que transmitiram algum quadro em um intervalo anterior para estimar a carga na rede. Com essas informações, modificam-se as funções de incremento e decremento da janela de contenção, de forma a melhorar o desempenho da rede em termos de vazão e justiça.

Uma avaliação do desempenho de vazão e justiça foi realizada, após a aplicação da nova proposta de algoritmo de *backoff*. Resultados das simulações mostraram que a nova proposta apresenta melhores desempenhos de vazão e justiça comparada com o algoritmo tradicional (BEB) e outras propostas apresentadas em cenários de saturação.

A dissertação está organizada da seguinte maneira. O Capítulo 1 apresenta o padrão IEEE 802.11, mais especificamente sua arquitetura e sua subcamada MAC. Em seguida, no Capítulo 2, algumas propostas relacionadas de algoritmos de *backoff* são analisadas, com o objetivo de compreender seus respectivos mecanismos de incremento e decremento da janela

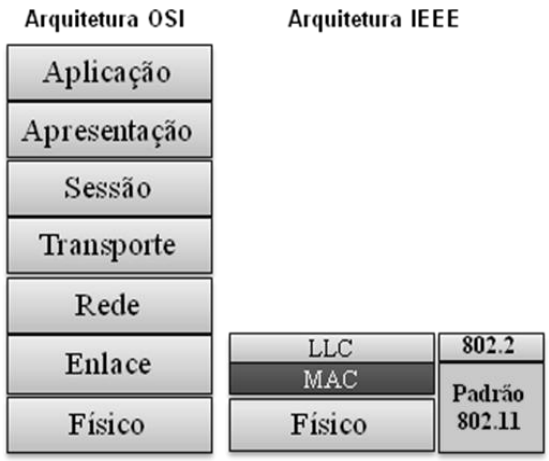
[1] **Inanição -** quando um processo, em programação concorrente, é impedido de ser executado ("morre de fome") devido a outros processos com maior prioridade.

de contenção, e avaliar quais são mais adequados em cenários de saturação. No Capítulo 3, os principais problemas encontrados em redes sem fio IEEE 802.11 são apresentados para, na sequência, introduzir a nova proposta de algoritmo de *backoff* (nMBEB) e seu mecanismo de controle de vizinhança "ativa", que significa um monitoramento ao longo do tempo das estações vizinhas concorrentes para realização de procedimentos diferenciados no algoritmo tradicional de *backoff*. O Capítulo 4 apresenta uma descrição dos cenários propostos para simulações, uma avaliação dos algoritmos, uma descrição das métricas para avaliação de desempenho e os resultados das simulações em termos de vazão e justiça da nova proposta de *backoff*. Por fim, são apresentadas as conclusões e sugestões de trabalhos futuros.

1 O Padrão IEEE 802.11

Criado pelo IEEE para especificar as camadas física e enlace, e garantir a compatibilidade entre equipamentos de diferentes fabricantes, o padrão IEEE 802.11 segue o modelo de camadas para Interconexão de Sistemas Abertos (*Open Systems Interconnection* – OSI) da Organização Internacional para Padronização (*International Organization for Standardization* – ISO) (Figura 1). Neste modelo, a camada de enlace é composta por uma subcamada de controle de enlace lógico (LLC) utilizada em diversas tecnologias dos padrões IEEE 802, e por uma subcamada de acesso ao meio (MAC), que controla o acesso de estações a um meio de transmissão compartilhado. A camada de enlace oferece uma interface para as camadas superiores da arquitetura IEEE 802.11, consolidando informações em quadros de subtipo dados ou controle, além de garantir o acesso à camada física para transmissão de quadros (TANENBAUM, 2003).

Figura 1 – Arquitetura IEEE e Subcamada MAC.



Fonte: Tanenbaum, 2003.

1.1 Arquitetura da Subcamada MAC

A arquitetura da subcamada MAC (BORGIA, 2003) é composta pela função de coordenação centralizada (*Point Coordination Function – PCF*) e pela função de coordenação

distribuída (*Distributed Coordination Function – DCF*).

A função de coordenação centralizada (PCF) é opcional e somente utilizada em redes infraestruturadas. No modo de coordenação centralizada PCF, existe um dispositivo coordenador (*Access Point – AP*) que provê autenticação aberta ou de chave compartilhada e associação de uma estação ao canal definido pelo AP e automaticamente à BSS (*Basic Service Set*), além do controle de potência e sincronização entre as estações. Este dispositivo coordenador garante o sincronismo entre às estações, sem que haja contenção, determinando assim qual estação transmite em qual intervalo de tempo.

A função coordenação distribuída DCF é o mecanismo básico que permite o compartilhamento automático do meio entre todos os nós ou estações em disputa. No modo de coordenação distribuída DCF, a subcamada MAC emprega o método de acesso baseado em contenção *CSMA/CA* que utiliza um tempo aleatório (tempo de *backoff*) para coordenar o acesso de múltiplas estações a um meio físico compartilhado e concorrente, minimizando colisões e a exclusividade do meio por uma ou mais estações.

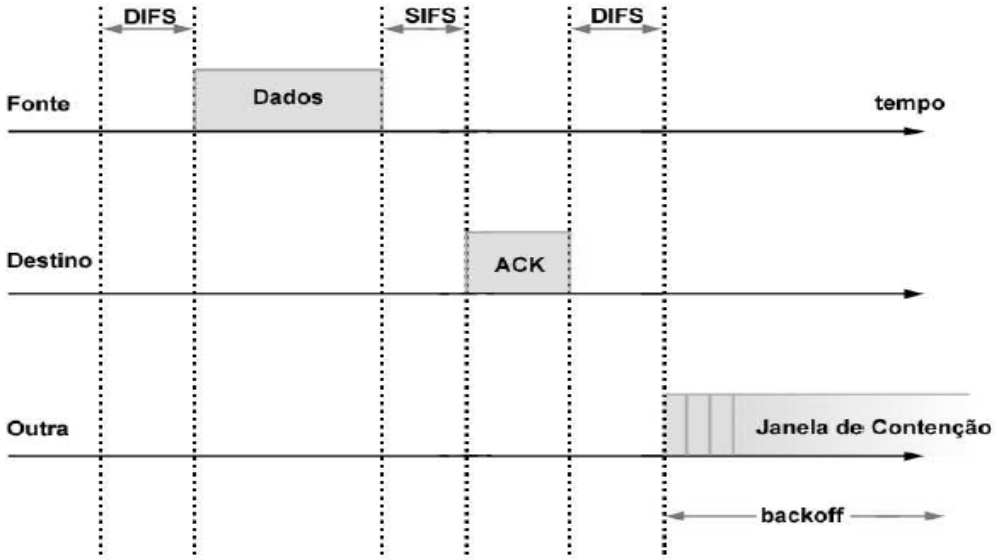
Devido ao problema de terminal escondido e a grande diferença de potência entre o sinal transmitido e o recebido, causada pela atenuação do meio, o CSMA/CA não usa quaisquer mecanismos de detecção de colisão (CAMPISTA e RUBINSTEIN, 2014). Ao invés disso, o CSMA/CA utiliza detecção de portadora virtual, reconhecimentos positivos (acknowledgments – ACKs), espaços de tempo entre quadros (*Interframe Spaces – IFSs*), e opcionalmente fornece o modo com mensagens de controle RTS e CTS (Campista e Rubinstein, 2014).

O CSMA/CA combina características dos métodos de acesso múltiplo baseados em disputa, MACA (*Multiple Access with Collision Avoidance*) (KARN, 1990), MACAW (*Multiple Access with Collision Avoidance for Wireless*) (BHARGHAVAN, 1994) e FAMA (*Floor Acquisition Multiple Access*) (FULLMER, 1995).

O método de acesso ao meio do DCF divide-se em dois modos: básico e com RTS/CTS, descritos a seguir.

1.2 O Modo Básico de Acesso

Figura 2 – Modo básico de acesso.



Fonte: Adaptada de Campista e Rubinstein, 2014.

No modo básico (Figura 2) de acesso ao meio do DCF, ao iniciar uma nova transmissão, uma estação verifica através do mecanismo de detecção de portadora, se o canal está livre ou ocupado. Se estiver livre por um intervalo entre quadros (IFS) igual a um DIFS (*Distributed InterFrame Space*), a estação transmite o quadro. Se após o tempo de um DIFS, o meio físico ainda estiver ocupado, a subcamada MAC incrementa o contador de tentativa de transmissão associado a cada quadro e inicia um procedimento de *backoff* (a ser apresentado posteriormente com detalhes), adiando assim sua transmissão por um período de tempo. O valor obtido no procedimento de *backoff* determina quantos slots de tempo que uma estação deve esperar antes de iniciar sua transmissão. O decremento do temporizador de *backoff* é realizado até alcançar o valor zero, caso o meio esteja livre; e interrompido, caso o meio torne a ser utilizado por outra estação. Quando o temporizador de backoff alcançar o valor *zero*, a estação transmite o seu quadro. Então a estação destino aplica o mecanismo de detecção de erro CRC (*Cyclic Redundancy Check*) para verificar se o quadro recebido está correto. Se estiver correto, a estação destino envia para a estação fonte, um quadro de reconhecimento positivo (ACK), após um intervalo de tempo denominado SIFS (*Short InterFrame Space*). Ao receber o quadro ACK, a estação fonte está apta a realizar uma nova tentativa de transmissão de um quadro.

1.2.1 Algoritmo Exponencial Binário de Backoff (BEB)

O tempo de *backoff*, representado pela Equação 1, pode ser interpretado como sendo uma janela temporal de adiamento da transmissão até que a estação tenha permissão para concorrer novamente ao meio; por isso é também conhecido como janela de contenção (CW). O tempo de *backoff* é dado por:

$$TempoBackoff = Random\,[\mathrm{CW}] \times SlotTime, \quad (1)$$

onde a função pseudoaleatória *Random[*CW*]* gera números inteiros uniformemente distribuídos dentro de uma faixa de [0, CW], determinando quantos slots de tempo (*SlotTime*) em microsegundos que uma estação deve esperar antes de iniciar sua transmissão. O *SlotTime* corresponde ao atraso máximo de propagação dentro de uma BSS, parametrizado conforme o meio físico (DSSS, FHSS ou Infravermelho).

O valor de CW utilizado na função *Random[*CW*]* corresponde a um número inteiro que varia entre um valor mínimo (CW_{Min}) e um valor máximo (CW_{Max}), incrementado em caso de colisão e reinicializado a CW_{Min}, em caso de sucesso na transmissão do quadro.

Ao não receber um reconhecimento positivo ACK após um curto espaço de tempo *t* (SIFS), infere-se que houve uma colisão. A estação incrementa o contador de retransmissão e aplica o algoritmo exponencial binário de *backoff* (*Binary Exponential Backoff* – BEB) para incrementar exponencialmente o valor da janela de contenção CW, conforme a Equação 2. O valor máximo de CW não ultrapassa CW_{Max}. O valor de CW é calculado da seguinte forma:

$$CW = ((CW_{Min} + 1) * 2^{i}) - 1. \quad (2)$$

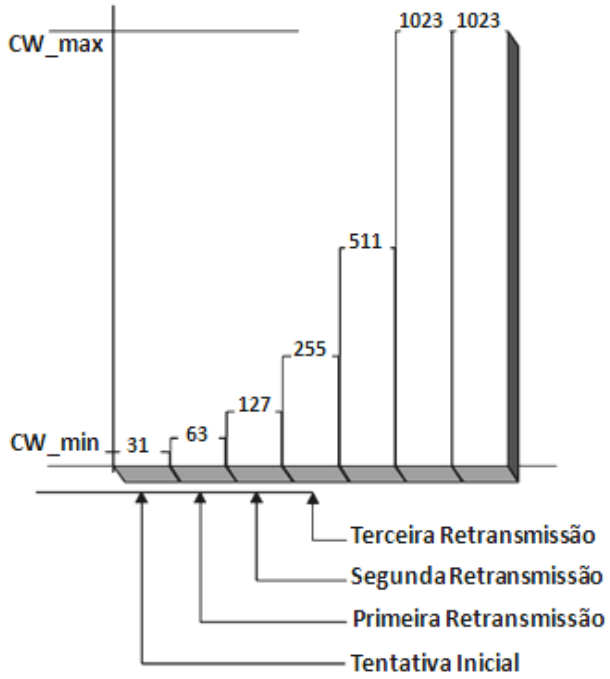
Na Equação 2, a base exponencial da função de incremento é igual a 2 (algumas propostas modificam esse valor e isso será apresentado na Seção 2.3 e *i* representa o número de tentativas de retransmissão ou estágio de transmissão, também denominado como estágio *i* da janela de contenção. A quantidade de estágios *i* da janela de contenção está diretamente associada aos valores de CW_{Min} e CW_{Max}, parametrizados de acordo com a camada física na arquitetura IEEE 802.11, com objetivo de evitar que janela de contenção cresça a valores que possam causar inanição definitiva de algumas estações ou diminua a valores que gerem números excessivos de colisões. Em seu estágio máximo ($i = m$), o valor de CW (ou W_i) é

denominado CW_{Max} (*i.e.* W_m) e no estágio mínimo ($i = 0$), é denominado CW_{Min} (ou W_0).

No caso da camada física ser o DSSS (*Direct Sequence Spread Spectrum*), os valores mínimo e máximo de CW são definidos como 31 e 1023. Dessa forma, pode-se afirmar que nesse caso o algoritmo exponencial binário de *backoff*, contém no máximo seis estágios de incremento da janela de contenção CW ou estágios de *backoff*, iniciando em $i = 0$ e terminando em $i = 5$. Ao atingir o estágio máximo de incremento em $i = 5$, novas retransmissões poderão ocorrer, porém o valor da janela de contenção CW não será incrementado e se manterá em seu valor máximo CW_{Max}.

Todos os valores de CW, em cada um dos estágios do algoritmo exponencial binário de *backoff* para o DSSS, são apresentados na Figura 3.

Figura 3 – Crescimento exponencial de CW no DSSS.



Fonte: IEEE Std 802.11, 2012.

Ao ocorrer uma colisão, o contador de retransmissão i é incrementado para $i + 1$, ocasionando uma transição do estágio i para um estágio superior $i + 1$. A transição do estágio i para $i + 1$ altera o valor da janela de contenção de W(i) para W($i+1$), ou seja, incrementará o valor de CW conforme a Equação 2.

Ao incrementar exponencialmente o valor de CW, o intervalo [0, CW] do algoritmo pseudoaleatório será ampliado, contemplando mais valores para serem sorteados e consequentemente, diminuindo a probabilidade de estações sortearem o mesmo valor de *backoff* e gerando novas colisões.

Em caso de envio de quadro com sucesso, o algoritmo exponencial binário de *backoff* realiza um "*reset*" do valor de *i*, correspondente à transição do estágio *i* para estágio inicial zero (BIANCHI, 2000). A transição do estágio *i* para zero altera o valor de W_i para W_0, ou seja, atribui o valor de CW_{Min} conforme a Equação 3.

$$CW = CW_{Min.} \quad (3)$$

Ao aplicar o valor de CW_{Min} novamente à Equação 1, o intervalo [0, CW] do algoritmo pseudoaleatório será reduzido para um intervalo mínimo [0, CW_{Min}].

Quando um dos contadores de retentativas de transmissão atinge seu valor máximo, o pacote é descartado e algoritmo exponencial binário de *backoff* realiza um *reset* de CW, atribuindo CW_{Min} à Equação 1. O contador de retentativas de transmissão curta (*Station Short Retry Count – SSRC*) é atribuído a quadros de controle e o contador de retentativas de transmissão longa (*Station Long Retry Count – SLRC*) é atribuído aos quadros de dados.

O mecanismo de *backoff* minimiza as colisões e permite que todas as estações que compartilham o mesmo meio físico sem fio, tenham chance de obter um canal para transmissão de seus quadros.

1.3 O Modo com RTS/CTS

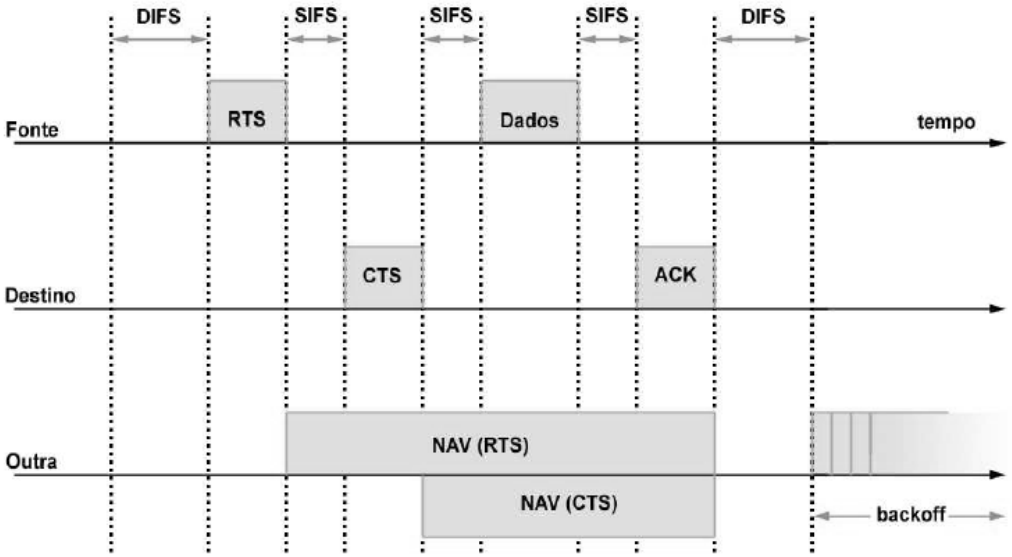
No modo com RTS/CTS (Figura 4), opcional pela norma, o DCF emprega o CSMA/CA com troca de quadros de controle RTS e CTS antes da transmissão de quadros de dados. Além disso, utiliza-se detecção virtual de portadora. No mecanismo DCF com RTS/CTS, a estação transmissora envia um quadro de controle RTS em difusão (*broadcast*) para a estação receptora, contendo o intervalo de tempo que vai entre a recepção do RTS e a recepção do ACK. Todas as estações que recebem esse quadro detectam virtualmente portadora por esse intervalo, ou seja, deixam de tentar transmitir. A estação destino responde em difusão com outro quadro de controle denominado CTS, informando a todas as estações em seu alcance que está apta a receber o quadro de dados. Essas estações que recebem o CTS não tentam acessar o meio durante o intervalo de tempo entre a recepção do CTS e a recepção do ACK.

A subcamada MAC realiza a detecção de portadora virtual através do vetor NAV

(*Network Allocation Vector*) que usa informação disponível nos cabeçalhos MAC de todos os quadros de dados, enviados durante o período de contenção (*Contetion Period - CP*) ou nos quadros RTS/CTS anteriores ao envio do quadro de dados.

A notificação do período de tempo de contenção do meio é realizada através do campo *Duration/ID* (tempo de ocupação do quadro + tempo de confirmação – ACK) contido em todos os quadros (IEEE Std 802.11, 2012) e remetido a todas as estações receptoras que estão no alcance das estações que enviaram os quadros (PERAHIA e STACEY, 2013). Cada estação ao receber quadros RTS e CTS verifica se o novo valor no campo Duration/ID é maior que o atual, e atualiza seu próprio contador. Cada estação receptora deverá considerar o meio ocupado durante o período de tempo informado no campo Duration/ID.

Figura 4 – Modo RTS/CTS.



Fonte: Adaptada de Campista e Rubinstein, 2014.

O uso ou não dos quadros de controle RTS/CTS é definido pelo parâmetro *dot11Threshold*. A subcamada MAC define um limiar de tamanho do quadro de dados para envio dos quadros de controle antes da transmissão dos dados. Caso o tamanho do quadro de dados a ser enviado seja maior que o parâmetro *dot11Threshold* a estação enviará um quadro de controle RTS antes e aguardará o quadro de CTS.

2 OUTROS ALGORITMOS DE BACKOFF

Incrementar ou decrementar o valor de CW de forma a diminuir a probabilidade de ocorrerem novas colisões ou diminuir o tempo de acesso ao meio compartilhado não parece ser uma tarefa muito difícil; no entanto, há uma lógica em qualquer algoritmo de *backoff,* representada por regras de implicação e relações entre janela de contenção-vazão, colisão-vazão e janela de contenção-colisão.

Quando a janela de contenção CW diminui, estações tornam a disputar o meio em um menor espaço de tempo, aumentando assim a probabilidade de ocorrerem colisões. Ao ocorrerem mais colisões, mais tentativas de retransmissão acontecem e obtêm-se menores vazões. Em contrapartida, quando a janela de contenção CW aumenta, maior o atraso médio e estações retardam a disputa pelo meio, reduzindo assim as colisões. Ao reduzir as colisões em uma rede saturada, as transmissões se tornam mais efetivas e obtêm-se maiores vazões. Através dessas relações conclui-se que o equilíbrio adequado entre o aumento e a diminuição do valor de CW torna-se a chave para qualquer proposta de algoritmo de *backoff*.

É com base nestas relações que diversos autores propuseram alterações nas funções de incremento e/ou decremento de CW, que compõem o algoritmo de *backoff*. Algumas propostas apresentam funções de incremento e decremento exponenciais, outras apresentam funções de incremento exponencial combinadas com decremento linear, outras propõem métodos diferenciados ou compostos por mais de uma proposta. Essas propostas incluindo uma análise de desempenho do BEB são apresentadas a seguir.

2.1 Análise Markoviana do Algoritmo de *Backoff* (BEB)

Bianchi (2000) avaliou o desempenho do sistema de coordenação distribuído DCF do IEEE 802.11 fornecendo um modelo simples, mas preciso e analítico, que calcula a vazão, levando em consideração um número finito de estações. Em seu estudo, Bianchi (2000) prova que quadros transmitidos por cada dispositivo têm probabilidade igual e independente de colisão (*p*), não levando em consideração retransmissões já ocorridas. O modelo analítico

consiste na avaliação da vazão de saturação em função da probabilidade estacionária τ (probabilidade de transmitir um quadro em um método de acesso aleatório).

Em seu modelo analítico, Bianchi (2000) mostra que a probabilidade de sucesso P_s (Equação 4) durante a transmissão do quadro em um canal é obtida em função do número de estações *n* na rede e da probabilidade estacionária τ de uma estação transmitir um quadro em um *slot* escolhido aleatoriamente:

$$P_s = n\tau\,(1-\tau)^{n-1} / 1-(1-\tau)^{n}\ . \qquad (4)$$

Como a probabilidade estacionária τ, representada pela Equação 5, é dada em função da distribuição estacionária da cadeia de *Markov* ($b_{i,k}$) em seu estágio inicial $b_{0,0}$, é possível deduzir pela Equação 6 que a quantidade de estágios *i* disponíveis no algoritmo exponencial binário de *backoff*, também influencia na probabilidade de sucesso na transmissão de um quadro, haja vista que, a função da distribuição estacionária em seu estágio inicial $b_{0,0}$ é obtida em função da variável *m*. Tem-se:

$$\boldsymbol{\tau} = \boldsymbol{b_{0,0}} / (1-p), \qquad (5)$$

$$b_{0,0} = \frac{2(1-2p)(1-p)}{(1-2p)(W+1)+pW(1-(2p)^{m})}. \qquad (6)$$

No entanto, como no padrão IEEE 802.11 o número de estações *xnodes* é uma variável não controlável e os valores de *m* e *W* são associados à camada física (*m = 6 e W = 31 e 1023*), para Bianchi (2000) a única maneira de alcançar o desempenho ideal é empregar técnicas adaptativas para ajustar tais variáveis e obter uma vazão mais próxima do valor máximo alcançável.

2.2 Algoritmo Modificado de Backoff Exponencial Binário (*MBEB*)

Wattanamongkhol et al (2007) propuseram uma modificação no algoritmo exponencial binário de *backoff*, mantendo a mesma função de incremento exponencial binário (base da função exponencial igual a 2), porém alterando a função de decremento de CW.

Wattanamongkhol et al (2007) afirmam que o mecanismo de *reset* (Equação 3) da janela de contenção CW utilizado no algoritmo exponencial binário de *Backoff* degrada o

desempenho de uma rede, uma vez que cada novo envio de quadro inicia-se com o valor mínimo de *backoff* (CW_{Min}) após obter sucesso na transmissão do quadro anterior. Esse mecanismo torna o retorno à disputa pelo meio, rápido demais para uma rede considerada saturada. Por isso, Wattanamongkhol et al (2007) propuseram uma alteração na função apenas de decremento de CW, após uma transmissão com sucesso. Ao invés de realizar um "*reset*" do valor de CW, correspondente à transição do estágio *i* para estágio inicial zero (BIANCHI, 2000), alterando o valor de *W*(*i*) para *W*(*0*), ou seja, retornando o valor de CW para CW_{Min}, o mecanismo decrementa-o de forma exponencial.

A Equação 7 representa a função decremento exponencial, elaborada para o algoritmo tradicional em função do estágio *i* de *backoff* e do estágio máximo *m*. Ao ocorrer uma transmissão com sucesso, o contador de retransmissão *i* é decrementado para *i – 1*, ocasionando uma transição do estágio (*i*) para um estágio inferior (*i - 1*). A transição do estágio (*i*) para (*i – 1*) altera o valor de W(*i*) para W(i_{-1}), ou seja, decrementa o valor de CW conforme a Equação 7 e a Figura 3:

$$CW = ((CW_{Max} + 1) / 2^{m-i}) - 1. \quad (7)$$

Como no DSSS, por exemplo, o estágio inicial é *i = 0* e o estágio final corresponde a *i = 5*, *m* é definido como 5.

Wattanamongkhol et al. (2007), assim como, Bianchi (2000) utilizam a Cadeia de Markov para analisar o efeito de uma nova função de decremento de *backoff* na obtenção da probabilidade estacionária ($\boldsymbol{\tau}$) de transmistir um quadro e da vazão normalizada *S*. A principal diferença entre as duas análises está na transição de um quadro ao obter sucesso na transmissão. Nota-se na análise de Wattanamongkhol et al. (2007) que CW retorna para um estágio *i - 1* de *backoff* imediatamente anterior, enquanto na proposta de Bianchi o estágio i retorna para zero.

2.3 Algoritmos de Aumento Múltiplo e Decremento Linear

Bharghavan (1994) apresentou um novo algoritmo denominado MILD (*Multiple Increase Linear Decrease*) que, após uma colisão, incrementa mais lentamente o valor de CW. Esse algoritmo altera a base exponencial da função de incremento de CW do BEB

(Equação 2), utilizando um fator multiplicativo *r* igual a *1,5* ao invés de *2*. O valor de CW é calculado por:

$$CW = ((CW_{Min} + 1) * r^{i}) - 1, \quad (8)$$

onde *i* representa o número de tentativas de retransmissão.

Ao utilizar o fator multiplicativo *r* igual a 1,5 na função de incremento de CW representada pela Equação 8, o algoritmo emprega uma técnica adaptativa para ajustar a variável *m*, criando estágios intermediários entre CW_{Min} (W_0, $i = 0$) e CW_{Max} (W_m, $i = m$), que são fixos pela camada física. No DSSS, existem no máximo seis estágios de transição/colisão iniciando em $i = 0$ e finalizando em $i = 5$ (Figura 3). Ao utilizar o MILD no DSSS e empregar um fator multiplicativo $r = 1,5$, se pensarmos apenas na função de incremento de CW, mais quatro estágios *i* são criados, totalizando 10 estágios de transição de $i = 0$ (CW_{Min}) a $i = m$ (CW_{Max}). Dessa forma, podemos entender que criar mais estágios intermediários entre CW_{Min} e CW_{Max} significa "tentar mais vezes", ou seja, aumentar o número de tentativas de retransmissão, sempre limitados pelos contadores de retentativas curto (SSRC) e longo (SLRC) e aumentar o número de colisões, principalmente em redes com elevada concorrência pelo acesso ao meio.

Para evitar um número excessivo de colisões ao realizar um "*reset*" do valor de CW após um envio de quadro com sucesso, que corresponde à transição do estágio *i* para estágio inicial zero (BIANCHI, 2000) ou decrementar CW de forma exponencial, correspondente à transição do estágio *i* para um estágio anterior ($i - 1$) (WATTANAMONGKHOL, 2007), o algoritmo MILD decrementa o valor de CW de forma linear (1 em 1) conforme a Equação 9. Em caso de sucesso no envio do quadro, o decremento do valor de CW é realizado de forma contínua, sendo necessários 992 envios de quadros consecutivos com sucesso para que CW seja decrementado de 1023 (CW_{Max}) até 31 (CW_{Min}) (BHARGHAVAN, 1994). O valor da janela quando há sucesso na transmissão é dado por:

$$CW = (CW_{Ant} - 1). \quad (9)$$

A função de incremento exponencial do MILD não se adapta bem quando há a transição do estado da rede de "livre" para um estado de saturação devido à quantidade de estágios intermediários. No processo inverso quando a saturação da rede é reduzida abruptamente, seu mecanismo de decremento linear também não permite um ajuste rápido de

CW para a atual carga na rede, devido à enorme quantidade de estágios *i* presentes no decremento linear.

Uma proposta semelhante foi apresentada por Mardini et al. (2011) ao também aplicar uma função de incremento exponencial lenta de CW, utilizando o fator multiplicativo *r* igual a 1,5 na base exponencial. No entanto, Mardini et al (2011) usaram uma função de decremento linear ajustável pela variável Y, ao qual inicialmente é atribuído o valor 5, diferentemente do valor 1 utilizado na função de decremento linear do algoritmo MILD (BHARGHAVAN, 1994). Dessa forma, o decremento linear de CW proposto por Mardini et al. (2011) se torna um pouco mais rápido do que no MILD.

Em estudo similar ao realizado por Bharghavan (1994), Singh et al. (2013) propuseram avaliar o efeito do fator multiplicativo *r* de *backoff*, onde $\{r \in R \mid 1 < r \leq 2$, de fração decimal 0,1$\}$ (1,1, 1,2, 1,3, 1,4, 1,5, 1,6, 1,7, 1,8, 1,9 e 2,0) na função de incremento, representada pela Equação 8. Ao substituir a base exponencial da Equação 8 por qualquer outro valor real positivo, diferente de 2, o cálculo de CW obtém valores diferentes do tradicional (BEB) para CW_{Max} e m. Por exemplo, ao utilizar o fator multiplicativo *r* igual a 3 na Equação 8, o cálculo de CW obteria o valor de CW_{Max} igual a 2591 em apenas 4 estágios, ou seja, *m* será igual a 4. Como no DSSS, CW_{Max} é predefinido com o valor 1023, mesmo que a Equação 8 gere um valor superior (2591) em $i = m$, o algoritmo utilizará 1023. No entanto, para fins de definição da equação responsável pelo decremento de CW faz-se necessário utilizar o valor de CW obtido através da Equação 8 no estágio máximo, sem considerar a restrição CW_{Max}, aqui denominado CW_{Temp}. Dessa forma, obtém-se:

$$CW = ((CW_{Temp} + 1) / r) - 1. \qquad (10)$$

O valor máximo do fator *r* (*2,0*) analisado por Singh et al. (2013) corresponde ao fator multiplicativo utilizado no algoritmo exponencial binário de *backoff* (BEB). Para Singh et al. (2013), menores fatores multiplicativos *r* geram uma maior probabilidade de colisão (*p*) por aumentarem a frequência de retentativas; logo geram uma menor probabilidade estacionária (τ) de um dispositivo transmitir um quadro em um slot escolhido aleatoriamente (Equação 5), que consequentemente ocasionará uma menor probabilidade de transmissão de quadro com sucesso (Equação 4).

Ao utilizar os fatores multiplicativos *r* inferiores a 1,5 (SINGH et al., 2013) na Equação 8, a função de incremento cria ainda mais estágios *i* de retransmissão intermediários entre CW_{Min} ($i = 0$) e CW_{Max} ($i = m$), o que torna ainda mais lento o incremento de CW.

2.4 Algoritmo Dinâmico de *Backoff*

Mardini et al. (2011) propuseram um novo algoritmo dinâmico de *backoff* denominado DBA (*Dynamic Backoff Algorithm*) semelhante ao mecanismo de "*Slow Start*" do protocolo de controle de transmissão (*Transmission Control Protocol* – TCP), que combina dinamicamente em sua função de incremento de CW, um método exponencial para melhorar a vazão da rede e outro método linear para reduzir o atraso da rede, e assim alcançar um melhor desempenho de vazão e atraso médio de pacotes. Nas colisões iniciais, a função de incremento exponencial utiliza *K* na base exponencial da função, ou seja, multiplica o valor de CW anterior por um fator multiplicativo *K*, como a seguir:

$$CW = ((CW_{Min} + 1) * K^{i}) - 1. \quad (11)$$

Após melhorar a vazão na rede, em colisões subsequentes, o algoritmo DBA aplica um fator de incremento linear T:

$$CW = (CW_{Ant} + T). \quad (12)$$

A aplicação das funções (exponencial ou linear) de incremento é determinada por *thresholds* que são associados à variável "*NumberOfBackoffs*" ou número de *backoffs*. Um *threshold* define a alteração da função exponencial para a função linear. Como a variável *NumberOfBackoffs* está associada ao estágio *i* de *backoff* ou tentativa de retransmissão, interpreta-se que após uma quantidade específica de transmissões sem sucesso, o mecanismo de incremento de CW seria alterado de exponencial para linear, ou de linear para exponencial. Por exemplo, se o primeiro *threshold* for 4, em qualquer estágio de retransmissão *i* entre 1 e 4, o incremento de CW será na forma exponencial. Se o próximo *threshold* for definido como 9, qualquer retransmissão de quadro entre o estágio 5 e 9 será realizada de forma linear. Para Mardini et al. (2011) ambas as variáveis *K* e *T* são ajustáveis para se encontrar o melhor desempenho de vazão.

Para função de decremento, Mardini et al. (2011) propuseram a redução da janela de contenção de forma linear para lidar com o problema de justiça e evitar o domínio do meio

por apenas um nó. A função de decremento de CW utiliza um parâmetro Y para decrementar o valor de CW anterior:

$$CW = (CW_{Ant} - Y), \quad (13)$$

onde Y é configurável com o valor 2, ou seja, decrementando a janela de contenção em duas unidades a cada transmissão bem sucedida.

2.5 Algoritmo de Rápida Adaptação às Variações de Carga na Rede

Ao contrário das funções de incremento exponencial lento (SINGH et al., 2013), as funções de incremento que utilizam fatores multiplicativos *r* acima de 2 diminuem a granularidade da faixa de CW; uma vez que as extremidades da faixa (CW_{Min} e CW_{Max}) se mantêm fixas pela camada física. A diminuição da granularidade dá-se pela redução da quantidade de estágios *i* existentes entre *zero* e *m*, incrementando de maneira rápida o valor de CW.

Assim como Song et al. (2013) que propuseram um novo algoritmo de *backoff* denominado EIED (*Exponential Increase Exponential Decrease*) utilizando um fator de incremento e decremento de CW igual a 2,83, Cano et al. (2013) propuseram a substituição do algoritmo exponencial binário de *backoff* por um algoritmo de *backoff* exponencial *k-ário* já atualmente utilizado nas comunicações através de rede elétrica (*Power Line Communications* – PLC) (VELLOSO, 2005; VLACHOU, 2013). Esse algoritmo apresenta ótimo desempenho em termos de vazão e justiça em condições de cargas elevadas, grandes tamanhos de pacotes e número de estações superior a 15, reduzindo a probabilidade de colisão de quadros.

O algoritmo de *backoff* exponencial *k-ário* realiza o crescimento extremamente rápido da janela de contenção CW no estágio *i* conforme as Equações 8 e 10, onde o fator multiplicativo *k* ou *r* pertence ao conjunto de inteiros positivos $Z^*_+ = \{3,4,5\}$.

As transições de um quadro no estágio *i* de um algoritmo de *backoff* exponencial *k-ário* são representadas através da cadeia de *Markov* de Wattanamongkhol et al (2007) e determinam valores de CW conforme a Tabela 1. A principal diferença está na quantidade de estágios *i* entre $i = 0$ e $i = m$. Ao utilizar o fator multiplicativo *r* igual a *3* na Equação 8, a quantidade de estágios *i* de CW será reduzida de 6 ($r = 2$) para 5 (W_0, W_1, W_2, W_3 e W_m),

conforme a Tabela 1. Da mesma forma, ao utilizar fatores multiplicativos $r = 4$ e $r = 5$, quantidade de estágios i de CW (W_i) será reduzida de 6 para 4 (W_0, W_1, W_2 e W_m).

Tabela 1 – Valores da janela de contenção CW em cada estágio *i* de *backoff* em relação ao fator multiplicativo $r \in Z^{*}+ = \{2,3,4,5\}$

Fator	**i = 0**	**i = 1**	**i = 2**	**i = 3**	**i = 4**	**i = 5**
r = 2	31	63	127	255	511	1023
r = 3	31	95	287	863	1023	--
r = 4	31	127	511	1023	--	--
r = 5	31	159	799	1023	--	--

A diminuição de estágios intermediários entre CW_{Min} e CW_{Max} para incremento ou decremento do valor da janela de contenção CW faz com que após poucas tentativas de retransmissão o quadro se mantenha em seu estágio máximo ($i = m$), e rapidamente retorne para seu estágio mínimo ($i = m$), após poucos envios de quadros com sucesso.

A probabilidade de transmitir um quadro dada uma certa probabilidade de colisão (p), está diretamente associada às variáveis W_0 e m. Reduzir a quantidade de estágios de retransmissão existentes i entre *zero* e m, impacta diretamente no aumento da probabilidade de transmissão ($\boldsymbol{\tau}$) de um quadro segundo as Equações 5 e 6.

2.6 Algoritmo Exponencial Deslizante

Na proposta de algoritmo de *backoff*, denominado DCWA (*Determinist Contention Window Algorithm*), Ksentini (2005) fornece um mecanismo determinístico de *backoff* através do uso de uma janela de contenção deslizante. Ao invés de utilizar o fator multiplicativo r de incremento igual a 2 e aumentar apenas o limite superior da faixa de [0,CW], o mecanismo também utiliza o mesmo fator multiplicativo r para aumentar o limite inferior da faixa de [0,CW]. O mecanismo é determinístico por designar uma faixa mais estreita de CW para sorteio aleatório e garantir que, após uma colisão, o valor do tempo de *backoff* seja realmente aumentado.

O algoritmo divide toda a faixa de *backoff* [0, CW_{Max}] em subfaixas menores e as associa a um estágio i. Ou seja, a cada estágio i de transmissão a faixa não será [0, CW] mas sim [$CW_{x(i)}$, $CW_{y(i)}$], onde $CW_{y\,(i)} = CW_{y\,(i\,-1)} * 2$ e $CW_{x(i)} = CW_{y\,(i)} - 32 * i$. No estágio i igual

a 1, por exemplo, a faixa de CW será definida como [32, 64]. Já no estágio *i* igual a 2, a nova faixa será definida como [64, 128].

Assim, não só o limite superior da faixa de [0, CW], conforme Figura 3, representará um estágio *i* de CW a cada retransmissão, mas toda a subfaixa. Esse mecanismo garante que, ao ocorrer uma colisão, durante o sorteio aleatório do valor de CW, a probabilidade de sorteio de números inteiros próximos à extremidade inferior da faixa (*zero*) seja nula e a estação não tente uma nova retransmissão imediatamente após a tentativa anterior. A proposta é forçar uma adaptação mais rápida ao aumento da carga na rede e conseguir uma redução da probabilidade de colisão.

Além disso, em caso de sucesso no envio de um quadro, o algoritmo DCWA não realiza o *reset* de CW para CW_{Min} a fim de mitigar a probabilidade de uma estação sortear valores baixos de *backoff* no intervalo [0,CW], monopolizando assim o meio e elevando o número de retransmissões principalmente em cenários com muitas estações em contenção.

Ao obter sucesso no envio de um quadro, o algoritmo reajusta o tamanho da faixa de *backoff* levando em consideração a carga da rede atual B(T), variável que representa a fração de slots ocupados nos últimos *n* slots observados pela estação durante a contenção (Ex.: 2/10 slots significa 20% dos slots ocupados). Como a variável B(T) varia entre 0 e 1, entende-se que se o número de slots ocupados no último intervalo é de 0,1 (10% de ocupação de slots), os novos valores da faixa [$CW_{x(i)}$, $CW_{y(i)}$] será de 10% [$CW_{x(i-1)}$, $CW_{y(i-1)}$]. Ou seja, conclui-se que com a carga da rede baixa, o algoritmo tentará uma retransmissão em um espaço de tempo mais curto. Já se a carga da rede estiver elevada e B(T) for igual a 1 (100% de ocupação de slots), os novos valores da faixa [$CW_{x(i)}$, $CW_{y(i)}$] serão equivalentes à faixa anterior.

Para Ksentini (2005) a explicação para o excelente desempenho e estabilidade de vazão no DCWA perante o algoritmo tradicional BEB, está na capacidade de resposta às flutuações de carga na rede.

2.7 Algoritmo Baseado no Controle de *Slots* Livres

No algoritmo de ajuste dinâmico da janela de contenção DCWA (*Dynamic Contention Window Adjustment*), Yu Qin et al. (2012) apresentam uma proposta de algoritmo de *backoff* que, em condições de rede saturadas, ajusta a janela de contenção baseado na observação da

média de números de *slots* livres consecutivos entre duas tentativas de transmissão (sucesso ou colisão). A variável que representa a observação média é denominada Li(t). Cada estação ativa na rede define sua faixa otimizada [Li_{min}, Li_{max}] a fim de atingir ou aproximar-se da vazão máxima. Se *Li* observado está menor que Li_{min} da faixa ideal [Li_{min}, Li_{max}], a estação aumenta seu valor de CW para retardar a tentativa de retransmissão. Ou seja, se há menos slots livres na rede que o considerado ideal, então o retorno à disputa pelo meio será postergado. Enquanto se *Li* observado for maior que Li_{max}, diminui-se o valor de CW para acelerar o retorno à disputa pelo meio.

Ao observar o valor ideal *Li*, menor que o valor mínimo da faixa Li_{min} o algoritmo incrementa o valor de CW linearmente por um fator η_{incr} (Equação 14) e se o valor ideal *Li* for maior que Li_{max} o algoritmo decrementa o valor de CW linearmente por um fator η_{decr} (Equação 15), onde CW_i significa o valor de CW atual. Tem-se:

$$\eta_{incr} = \frac{F_1}{CW_{Max} - CW_{Min}} (CW_{Max} - CW_i), \quad (14)$$

$$\eta_{decr} = \frac{F_2}{CW_{Max} - CW_{Min}} (CW_i - CW_{Min}). \quad (15)$$

Já F_1 e F_2 são números inteiros, predefinidos no algoritmo, que determinam o valor máximo de crescimento de CW. Ou seja, o incremento linear de CW será no máximo de F_1 unidades (para CW_i igual a CW_{Min}) e o decremento de CW será no máximo de F_2 unidades (para CW_i igual a CW_{Max}), porém sempre definidos em função da quantidade média de *slots* livres observados.

2.8 Resumo dos Algoritmos de *Backoff*

Para facilitar a compreensão dos próximos capítulos, a Tabela 2 a seguir consolida características importantes dos mecanismos de incremento e decremento utilizados nos trabalhos relacionados deste capítulo.

Tabela 2 – Tabela consolidada dos algoritmos relacionados

Algoritmos	**Incremento CW**	**Base de Incremento**	**Decremento CW**	**Base de Decremento**
BEB	Exp. Binário	r = 2	*Reset*	-
MBEB	Exp. Binário	r = 2	Exp. Binário	r = 2
MILD-1	Exp. Lento	r = 1.5	Linear Lento	de 1 em 1
MILD-2	Exp. Lento	r = {1,1 ... 2)	Exp. Lento	r = {1,1 ... 2)
DBA	Exp. Binário e Linear Lento	K = *r* = 2 e T = 5	Linear Lento	de 2 em 2
MBEB_r3	Exp. Rápido	r = 3	Exp. Rápido	r = 3
MBEB_r4	Exp. Rápido	r = 4	Exp. Rápido	r = 4
MBEB_r5	Exp. Rápido	r = 5	Exp. Rápido	r = 5
DWCA (Deterministic)	Exp. Binário	r = 2	Exp. Binário	r = 2
DWCA (Dynamic)	Linear Lento	(0 <= n_{decr}<= F1)	Linear Lento	(0 <= n_{decr}<= F2)

2.9 Principais Problemas dos Algoritmos de *Backoff*

2.9.1 Excessivo número de colisões

Em cenários de saturação, em geral muitas estações participam do processo de "disputa" pelo acesso ao meio. Nestes cenários, é muito comum ocorrer um número excessivo de colisões, pois muitas estações possuem pelo menos um quadro a ser transmitido (Bianchi, 2000) em qualquer instante *t*, além do problema do terminal escondido.

Alguns mecanismos utilizados em algoritmos de *backoff* realizam ações que aumentam significativamente o número de colisões e afetam o desempenho de vazão das estações. A primeira ação que aumenta o número de colisões é utilizar na função de decremento de CW, o mecanismo de *reset* (BIANCHI, 2000). Uma vez que cada novo quadro inicia com probabilidade de sorteio entre 0 e CW_{Min}, o mecanismo de *reset* torna seu retorno à disputa pelo meio rápido demais para um rede supostamente carregada. A segunda ação que

aumenta o número de colisões é insistir nas tentativas de retransmissão de um quadro em uma rede que transitou do estado “livre” para o estado “saturada”. Ao realizar o incremento de CW de forma “lenta”, ou seja, da forma linear (QIN, Y. et al., 2012) ou utilizando fatores multiplicativos *r* inferiores a 2 (BHARGHAVAN, 1994; SINGH et al., 2013), mais estágios intermediários de *backoff* são criados implicando na tentativa de transmissão de um quadro em um curto espaço de tempo em um meio saturado gerando mais colisões.

2.9.2 Injustiça

Em cenários de saturação, o acesso ao meio se torna um desafio para a maioria das estações. Mesmo existindo um mecanismo de coordenação distribuída, o sorteio aleatório pode ocasionar um certo “monopólio” por uma ou algumas estações, causando a inanição em outras. A falta de justiça pode ser explicada se dividirmos, conforme Tabela 3, a faixa entre CW_{Min} e CW_{Max}, e definirmos subfaixas menores com suas respectivas probabilidades no sorteio aleatório. Por exemplo, a probabilidade de uma estação *A* sortear um valor de CW entre 16 e 31 é de 50% no primeiro estágio *i* de retransmissão, quando o valor máximo para CW pode ser 31. Já uma estação *B* que está no sexto estágio *i* de retransmissão ($i = 5$), quando o máximo valor de CW pode ser 1023, há uma probabilidade de 1,5625% de sortear um valor entre 16 e 31. Ou seja, a menos que a estação *B* não tenha sucessivas transmissões de quadro com sucesso, a estação *A* terá maior prioridade na concorrência ao meio. Ao mesmo tempo, a mesma estação B terá mais chances (96,875%) de sortear um valor de CW superior a 31, enquanto que antes, no primeiro estágio *i* (máximo de CW igual a 31) essa chance era nula.

Estações que estão nos primeiros estágios *i* de retransmissão têm uma maior probabilidade de sortear valores inferiores na faixa [0, CW]; enquanto estações que já estão em estágios *i* de retransmissão mais próximos a *m* terão maior probabilidade de sortear um número aleatório mais próximo a CW_{Max}, gerando um aumento no atraso do acesso ao meio.

Tabela 3 – Probabilidade de sorteio aleatório do valor inteiro de CW em cada subfaixa com 6 estágios *i* de *backoff* (r = 2)

Estágio (i)	**CW_i**	**Subfaixas de CW (probabilidades de sorteio aleatório)**						
0	**31**	0-15	16-31	32-63	64-127	128-255	256-511	512-1023
		50%	50%	0%	0%	0%	0%	0%
1	**63**	0-15	16-31	32-63	64-127	128-255	256-511	512-1023
		25%	25%	50%	0%	0%	0%	0%
2	**127**	0-15	16-31	32-63	64-127	128-255	256-511	512-1023
		12,5%	12,5%	25%	50%	0%	0%	0%
3	**255**	0-15	16-31	32-63	64-127	128-255	256-511	512-1023
		6,25%	6,25%	12,5%	25%	50%	0%	0%
4	**511**	0-15	16-31	32-63	64-127	128-255	256-511	512-1023
		3,125%	3,125%	6,25%	12,5%	25%	50%	0%
5	**1023**	0-15	16-31	32-63	64-127	128-255	256-511	512-1023
		1,5625%	1,5625%	3,125%	6,25%	12,5%	25%	50%

Muitos algoritmos se preocupam corretamente em reduzir o número de colisões aumentando rapidamente o valor de CW. Todavia, se tais algoritmos aplicarem métodos de decremento de CW que retardem a adaptação da estação à transição de uma rede no estado "saturada" para o estado "livre", pode haver inanição de uma ou mais estações. Em muitos casos, uma estação alcança o estágio $i = m$ de *backoff*, onde CW é igual a CW_{Max}, porém, ao aplicar funções de decremento linear (BHARGHAVAN, 1994) ou exponencial (SINGH et al., 2013) com fatores multiplicativos *r* menores que 2, CW se mantém com valores próximos a CW_{Max}, mesmo após algumas transmissões com sucesso. A utilização dessas funções aumenta a granularidade entre CW_{Min} e CW_{Max} e, somente quando a rede estiver livre, que subsequentes transmissões com sucesso irão acontecer e CW retornará ao patamar de CW_{Min}.

2.9.3 Adaptação à carga na rede

A existência de uma rede com diversas estações associadas, não significa que a rede está no estado de saturação e que estações sempre têm algo para transmitir em qualquer instante *t*. Uma estação pode ficar por um longo período de tempo inativa, sem transmitir, como também pode ficar por vários instantes de tempo enviando quadros de dados ou controle.

Um melhor desempenho poderia ser alcançado se cada estação, em um sistema de coordenação distribuído, fosse capaz de detectar quantas estações estão concorrendo no momento que a estação tentasse transmitir um quadro, e assim conhecer a probabilidade de colisão (*p*) em um determinado instante *t*; uma vez que, quanto maior é o número de estações concorrentes na rede maior será a disputa pelo meio e maior a probabilidade de colisão. Cada estação poderia empregar técnicas adaptativas para ajustar variáveis (m, W_0) existentes no algoritmo de *backoff* para obter uma vazão mais próxima do valor máximo alcançável (Bianchi, 2000).

Na proposta de Bianchi (2000), a quantidade de estações *n* concorrendo ao meio é determinante para conhecer a probabilidade P_s de uma transmissão ocorrer com sucesso no canal físico (Equação 4). A lógica é simples; quanto maior o número de estações concorrentes, maior será a probabilidade de colisão e consequentemente menor será a probabilidade P_s de uma estação transmitir com sucesso. Conhecer o número de estações vizinhas ativas é ter como aplicar o valor de CW (W) otimizado para a carga na rede em dado instante *t*.

Ksentini (2005) indiretamente se preocupou com as flutuações de carga na rede para adaptar a função de incremento de CW e obter uma redução da probabilidade de colisão de um pacote (*p*). Ao realizar o incremento do limite inferior da faixa de CW [$CW_{inferior}$, $CW_{superior}$], o algoritmo de Ksentini (2005) indiretamente impõe que durante o sorteio do *TempoBackoff* (Equação 1)**,** valores mais altos no intervalo [0,CW] sejam sorteados e garante que uma estação aguarde um tempo mais adequado à redução da carga na rede, antes de enviar um novo quadro de dados ou controle. Dessa forma, o algoritmo de Ksentini (2005) diminui a probabilidade de quadros colidirem em um cenário de saturação. Yu Qin et al. (2012) indiretamente também se preocuparam com o monitoramento da carga na rede (estações e tráfego) ao realizarem uma contagem do número de *slots* livres, ou seja, determinarem qual o percentual de rede que está ociosa.

Em redes saturadas, a concorrência aumenta de forma que pode haver inanição de algumas estações. Por isso, é de extrema importância que todas as estações conheçam seus concorrentes, realizando o controle de seus vizinhos ativos (efetivamente transmitindo) e conseguindo assim ajustar de maneira individual e independente a frequência com que as tentativas de acesso ao meio são realizadas.

3 ALGORITMO PROPOSTO DE BACKOFF nMBEB

Nas últimas duas décadas, diversas propostas foram apresentadas para adaptar o algoritmo exponencial binário de *backoff* com a finalidade de reduzir o número de colisões na rede, melhorar o desempenho em termos de vazão do sistema e fornecer um algoritmo distribuído justo em que todas as estações possam transmitir e receber em algum instante *t*. A seguir, detalharemos as alterações propostas para alcançar uma melhor adaptação da estação às variações de carga na rede e apresentaremos a nova proposta de algoritmo de *backoff*, bem como, seu mecanismo de controle de estações vizinhas ativas.

3.1 MBEB como base para o nMBEB

Estudos anteriores de Wattanamongkhol et al (2007) e Haitao et al. (2002) comprovaram que em redes *ad hoc*, o desempenho de vazão do sistema é superior quando a função de decremento de CW é apresentada na forma exponencial, não retornando diretamente a CW_{Min}, reduzindo assim o número excessivo de colisões. Da mesma forma que o decremento linear não é indicado por causar inanição nas estações. Por esse motivo usaremos o MBEB, como algoritmo base para aplicar as alterações propostas no novo algoritmo de *backoff*. Como no MBEB o fator multiplicativo *r* é utilizado nas funções de incremento e decremento, em redes saturadas, um ajuste ideal no fator multiplicativo pode determinar qual seria a frequência ideal para tentar o acesso ao meio.

3.2 Alterações propostas para melhor adaptação às variações de carga na rede

Pelas descrições dos problemas existentes nas propostas de algoritmo de *backoff*, podemos concluir que a aplicação da função de incremento exponencial rápido ($r > 3$), descrita na Seção 2.5, melhora a adaptação das estações às mudanças bruscas e repentinas de saturação na rede. Usando um menor número de estágios intermediários *i* entre

zero e *m*, e valores W_i (CW) mais elevados, uma estação reconhece uma rede no estado "saturada" com poucas retransmissões e reconhece rapidamente uma rede no estado "livre" em poucas transmissões com sucesso. Para comprovar tal proposição, resultados apresentados pelo algoritmo de Cano et al. (2013) mostram que as funções de incremento e decremento que utilizam fatores multiplicativos *r* iguais a *4* e *5* (ambos com quatro estágios *i*) apresentam melhores resultados de vazão que fatores *r* iguais a *2* (com seis estágios *i*) e *3* (com cinco estágios *i*).

Como Cano et al. (2013) somente avaliaram o desempenho do algoritmo exponencial *k-ário* utilizando fatores multiplicativos $r \in Z^*_+ = \{3,4,5\}$, a proposta inicial desse trabalho avaliará também o desempenho de fatores multiplicativos $r \in Z^*_+ = \{6,7,8,9,10\}$. Tais fatores possuem a mesma característica de reduzir mais um estágio *i* existente entre *zero* e *m* (CW_{Min} e CW_{Max}). Ao aplicarmos na Equação 8 os fatores multiplicativos *r* maiores que *5*, a quantidade de estágios *i* de *backoff* será reduzida de 6 (W_0, W_1, W_2, W_3, W_4 e W_m), com o tradicional fator *r* igual a *2*, para 3 (W_0, W_1 e W_m), conforme a Tabela 4. E, como Cano et al. (2013) utilizaram valores *default* da função de coordenação do *Homeplug 1.0* nos parâmetros, como: *SlotTime* (tempo de um *timeslot*), *DataRate* (taxa de dados) e SIFS, então propomos avaliar o desempenho do novo algoritmo de *backoff* mantendo os parâmetros originais do DCF.

Adicionalmente, esse trabalho avaliará o desempenho do fator *r* igual a 33. Esse seria o caso mais extremo de redução de estágios *i* de *backoff*. Tal fator possui a característica de remover todos os estágios *i* intermediários entre *zero* e *m*. Ao aplicarmos na Equação 8 o fator multiplicativo *r* igual a 33, a quantidade de estágios *i* de *backoff* será reduzida de 6 (W_0, W_1, W_2, W_3, W_4 e W_m), com o tradicional fator *r* igual a *2*, para 2 (W_0 e W_m). Dessa forma, conforme Tabela 4, teríamos apenas dois valores possíveis para CW, 31 e 1023.

Tabela 4 – Valores da janela de contenção CW em cada estágio i de backoff em relação ao fator multiplicativo $r \in Z^*+ = \{2,3,4,5,6,7,8,9,10,33\}$

Fator	i = 0	i = 1	i = 2	i = 3	i = 4	i = 5
r = 2	31	63	127	255	511	1023
r = 3	31	95	287	863	1023	--
r = 4	31	127	511	1023	--	--
r = 5	31	159	799	1023	--	--
r = 6	31	191	1023	--	--	--
r = 7	31	223	1023	--	--	--
r = 8	31	255	1023	--	--	--
r = 9	31	287	1023	--	--	--
r = 10	31	319	1023	--	--	--
r = 33	31	1023	--	--	--	--

Como as melhores abordagens de algoritmo de *backoff* são aquelas que realizam o controle do número de estações "ativas" a fim de responder mais rapidamente a flutuações na rede, a nova proposta de algoritmo de *backoff* aplica o fator multiplicativo *r* de Cano et al. (2013) de melhor desempenho para uma dada quantidade de estações "ativas" na rede nas funções de incremento e decremento do valor de CW.

3.3 Nova proposta de Algoritmo de *Backoff* com controle de nós (nMBEB)

A nova proposta denominada nMBEB (*Modified Binary Exponential Backoff Algorithm - with node control*) apresenta um controle individual e distribuído da quantidade de vizinhos "ativos" que cada estação possui em um período de observação predeterminado. Cada estação faz um controle individual dos vizinhos que enviaram quaisquer quadros durante um intervalo *t* (em milissegundos) anterior à próxima tentativa de acesso ao meio. Como o algoritmo realiza a leitura do endereço "destino" de cada quadro enviado dentro de seu alcance, mesmo que uma estação A não esteja recebendo quadros diretamente de uma estação B, a mesma incluirá a estação B em sua lista de vizinhos "ativos" em função das mensagens ACKs recebidas em B, e incluirá também as estações que estão recebendo quadros de B. Ao utilizar o endereço de destino do quadro, indiretamente também estamos lidando com o problema de terminal escondido e com a mobibilidade de estações, entrando e saindo do alcance umas das outras.

A proposta do algoritmo nMBEB poderia utilizar informações de uma camada superior à subcamada MAC para realizar o controle das estações vizinhas. No entanto, ser considerada como uma estação vizinha não significa ser concorrente na tentativa de acesso ao meio, uma vez que a estação vizinha pode não ter transmitido nenhum quadro durante o último intervalo de tempo analisado. Neste algoritmo, a estação vizinha precisa estar realmente ativa para ser considerada como concorrente.

Por isso, o algoritmo determina que cada estação verifique a cada intervalo de tempo (*timer_interv*), sua própria lista dinâmica de controle de estações vizinhas "ativas", e remova da lista todas as estações que não enviaram quadros durante o último ciclo de tempo *t*. Dessa forma, cada estação visualiza quantas estações vizinhas ativas estão em seu alcance e consequentemente estima como está a saturação da rede através do histórico de envio de quadros, dentro de um dado intervalo de tempo. Quanto menor o ciclo de verificação de

estações (*timer_interv*), mais precisão teremos na verificação da carga (saturação) da rede. Estações que não enviaram quadros no intervalo não estão usufruindo da infraestrutura de rede sem fio e podem ser removidas da lista de controle de estações ativas.

Em uma análise individual de cada estação, estimar a saturação da rede durante o último ciclo de tempo *t*, entre as estações vizinhas, é conhecer o estado de disputa pelo acesso ao meio e aplicar alguma técnica adaptativa em busca de uma melhoria no desempenho do sistema distribuído (BIANCHI, 2000).

Além da lista de identificação das estações vizinhas, o algoritmo mantém uma variável *xnodes* que armazena a quantidade de estações que efetivamente transmitiram no último intervalo de tempo *t*. O objetivo principal é utilizar a variável *xnodes* para determinar o fator multiplicativo *r* mais adequado, após uma colisão inferida pelo não recebimento do ACK. O fator multiplicativo *r* mais adequado para cada quantidade de estações (*n*) será avaliado através de simulações, de forma a fechar a proposta do algoritmo de *backoff* (nMBEB).

O proposto algoritmo de *backoff* nMBEB determinará o valor de CW, após uma colisão, através da alteração da base da função exponencial de incremento ($CW = ((CW_{Min} + 1) * r^{i}) - 1$), ou seja, o valor do fator multiplicativo *r* mais adequado, em função do número de estações que estão ativas (*xnodes*) durante o último intervalo de tempo (*timer_interv*). Como o nMBEB baseia-se no MBEB, em caso de transmissão de um quadro com sucesso, o valor de CW será determinado pela alteração da base da função exponencial de decremento (Equação 10), ou seja, o valor do fator multiplicativo *r* mais adequado, em função de número de estações que estão ativas (*xnodes*) durante o intervalo *timer_interv* anterior à transmissão do quadro com sucesso.

4 A IMPLEMENTAÇÃO DO ALGORITMO nMBEB NO NS-2

Devido à sua consolidação em diversas pesquisas tecnológicas para diferentes tipos de redes de computadores e por ser um simulador orientado a objetos e de código aberto, a nova proposta de algoritmo de *backoff* foi avaliada utilizando o simulador de redes NS-2 (*Network Simulator*) (NS-2, 2015).

Para facilitar a modelagem no simulador NS-2 e a realização dos testes comparativos (YANG, 2015), um novo protocolo MAC (IEEE 802.11t) foi criado mantendo o protocolo MAC original (IEEE 802.11). Durante a modelagem, os códigos C MAC-*802_11t.cc* e MAC-*802_11t.h* foram alterados para que incluíssem os novos códigos objetos para controle das estações "ativas". Mais detalhes sobre a implementação são apresentados a seguir.

Cada estação precisa processar todos os quadros recebidos, interpretar certos campos do cabeçalho MAC, como o BSSID, e passar para camadas superiores somente os quadros que são destinados à estação ou em difusão (GAST, 2002). A informação de duração, disponível nos cabeçalhos de todos os quadros enviados, também é obtida de todos os quadros recebidos pela estação (IEEE Std 802.11, 2012), sendo utilizada diretamente no mecanismo de contenção (*backoff*).

Os conceitos supracitados são importantes para a implementação na prática do algoritmo nMBEB, uma vez que faz-se necessário encontrar uma função do simulador NS-2 (NS-2, 2015) que faça a leitura de todos os pacotes recebidos por todas as estações presentes em uma rede *ad hoc*. No código em C, *mac-802_11t.cc*, responsável pelas definições da subcamada MAC do *IEEE 802.11* no simulador NS-2, podemos identificar a função *recv_timer*() (ANEXO II), como sendo a função responsável por tratar todos os pacotes recebidos e analisar suas respectivas temporizações (LIU, 2016).

Inicialmente, no momento em que o pacote é recebido, a função *recv_timer*() verifica se a interface da estação está no modo *TRANSMIT* (transmitindo), para descartá-lo sem ajuste do NAV (*Network Allocation Vector*). Em caso do pacote ser recebido sem estar no modo *TRANSMIT*, a função verifica o FCS (*Frame Check Sequence*), presente no cabeçalho do quadro, para se certificar que o quadro não foi recebido com erro.

Ao analisar o cabeçalho, a função *recv_timer*() faz a leitura, na linha 47 do ANEXO II,

do endereço de destino do pacote, representado pela variável *dst* (*ETHER_ADDR*(*mh-dh_ra*)) no código em C (MAC-*802_11t.cc*) e verifica se o endereço é o mesmo endereço da própria estação que está recebendo o quadro. Se não for o mesmo endereço, a função *recv_timer*(), não processa o quadro e atualiza o vetor NAV (conforme o código a seguir) para tentar o acesso ao meio somente ao término do tempo definido pelo campo de duração (vetor NAV) no quadro recebido. Após esse tempo, o mecanismo de *backoff* será aplicado.

Através de uma chamada de função no código C, a função *recv_timer*() (ANEXO II) faz a conexão com a nova função adicionada, denominada *add_list_Node* (ANEXO I). A função é responsável por controlar o número de estações "ativas" na vizinhança da estação em análise.

A função *add_list_Node* é responsável por adicionar uma estação à lista de controle de estações vizinhas que enviaram quadros no último ciclo de tempo e o *timestamp* de quando a estação recebeu o quadro da estação vizinha. A cada chamada da função, ou seja, a cada quadro recebido e verificado na função *recv_timer(),* o código armazena o endereço de destino do quadro (*dst*) e, após armazenar a variável, a função *add_list_Node* faz uma varredura na lista onde estão armazenados os nós que enviaram quadros no último ciclo de tempo *t*. Caso não esteja na lista, a função armazena o endereço da estação destino e caso já esteja a função apenas atualiza o novo *timestamp* da estação que enviou o quadro.

Periodicamente, o mecanismo de controle de vizinhas chama a função *erase_nodes()* (ANEXO I) e verifica a lista com índice das estações que transmitiram durante o último ciclo de tempo. Caso a estação não tenha transmitido, a função *erase_nodes()* (ANEXO I) remove o índice da estação e atualiza a contagem de estações vizinhas. A função *erase_nodes()* é responsável por remover as estações vizinhas que não enviaram quadros (controle ou dados) dentro do último ciclo (*timer_interv*). Nesta função, uma *thread* é chamada para monitorar, de 500 em 500 ms, a lista de estações vizinhas que foram armazenadas a medida que os quadros são recebidos na subcamada MAC. Ao realizar a varredura na lista, a função erase_nodes remove todos os endereços (índices) de estações que não enviaram quadros no último ciclo de tempo *t* (*timer_interv*).

As funções *add_list_Nodes()* e *erase_nodes()* são as responsáveis por manter a lista de estações vizinhas ativas, atualizada conforme variações de carga na rede e fornecem dados atualizados para a função *calc_factor(),* cuja principal função é calcular o fator multiplicativo *r* que é utilizado pelas estações no momento de incrementar ou decrementar a janela de contenção CW.

5 SIMULAÇÕES

Inicialmente é avaliado, através de simulações, o desempenho de cada um dos fatores multiplicativos $r \in Z^*_+ = \{2,3,4,5,6,7,8,9,10,33\}$, apresentados na Tabela 4. Cada um dos fatores multiplicativos foi testado com os parâmetros originais utilizados no MAC 802.11b, conforme a Tabela 5.

Após a avaliação do desempenho de cada um dos fatores, o fator multiplicativo *r* mais adequado para a quantidade de estações ativas (*xnodes*) foi selecionado e utilizado na nova proposta de algoritmo de *backoff* nMBEB, que dinamicamente ajusta *r* e *xnodes* durante todo o tempo de simulação.

Com o objetivo de servir de base para a modificação do algoritmo MBEB e implementação do novo modelo nMBEB, antes de realizar as simulações para avaliar o desempenho da nova proposta de *backoff*, algumas simulações foram realizadas para comparação do desempenho entre os algoritmos de *backoff* BEB (BIANCHI, 2000) e MBEB (WATTANAMONGKHOL et al, 2007). Para finalizar, o desempenho do nMBEB foi comparado com ambos algoritmos já existentes BEB (BIANCHI, 2000) e MBEB (WATTANAMONGKHOL et al, 2007).

Parâmetros (Tabela 5) do simulador NS-2 (NS-2, 2015) foram alterados conforme o padrão IEEE 802.11b, que usa a taxa de dados de 11Mbps na banda de 2,4 GHz.

As simulações foram realizadas em um notebook com as seguintes configurações:

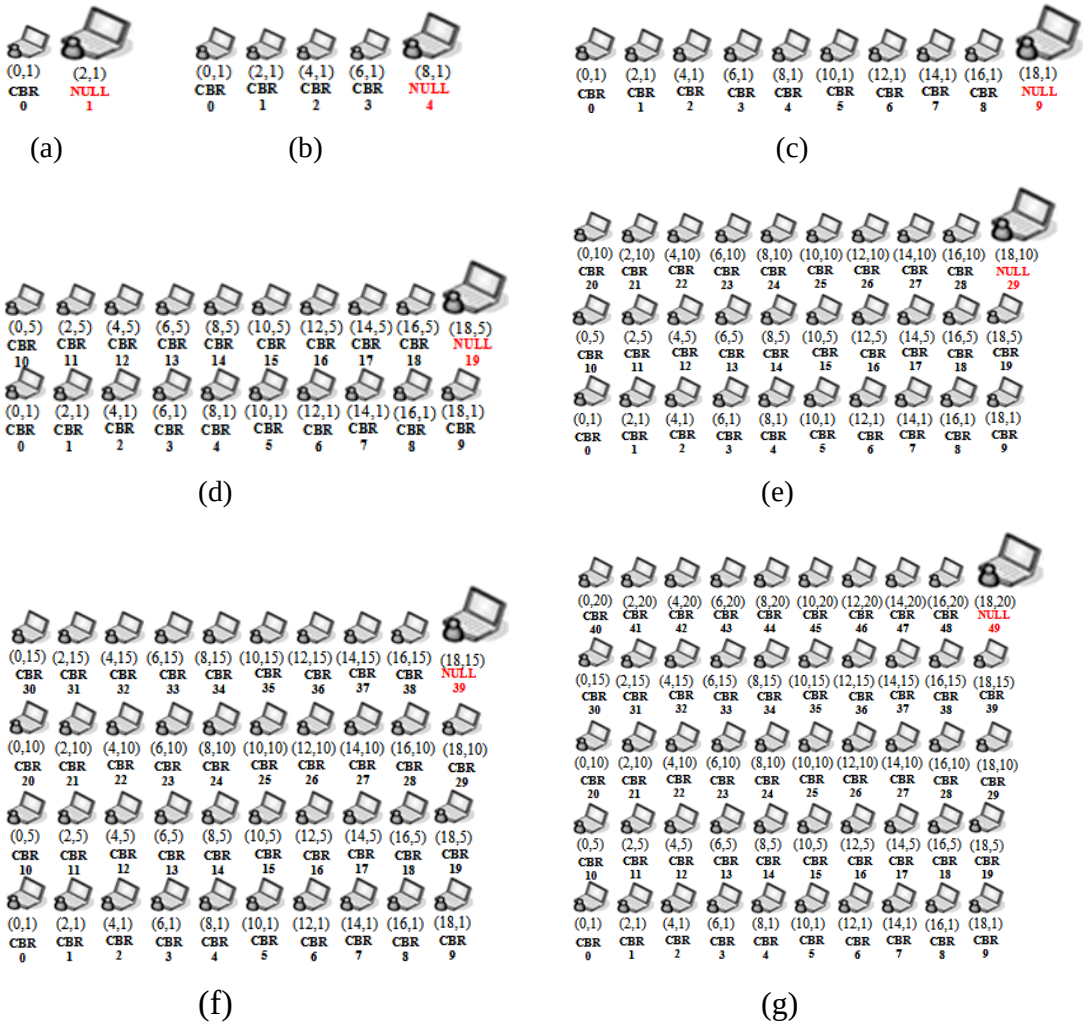
- CPU Intel Core i7-2630 Quad-core @ 2 GHz;
- Memória RAM 8 Gbytes – DDR3;
- Disco SSD 680 Gbytes;
- Sistema operacional Ubuntu 14.04.2 LTS;
- Pacote de instalação do NS-2 versão 2-35 (*ns-allinone-2.35*).

Mais detalhes do cenário, lógica, medidas e resultados são abordados nas próximas seções.

5.1 Descrição dos Cenários de Simulação

Todos os cenários foram modelados no NSG (NS-2 *Scenario Generator*) de Peng-Jung, 2016 e executados no NS-2 (NS-2, 2015). Uma única estação central em cada topologia (Figura 5) foi configurada com o agente *Null*, cuja principal função é receber todo o tráfego de dados CBR (*Constant Bit Rate*) enviado por cada cada fluxo UDP-CBR individual das estações presentes na rede. Em cada fluxo, uma estação envia quadros a uma taxa de dados constante de 11 Mbps. O objetivo principal é garantir que sempre há estações com algum quadro em sua fila de saída para ser transmitido, permitindo assim uma avaliação do comportamento do algoritmo proposto quando a rede está saturada. Nos cenários propostos, conforme a Figura 5, todas as estações estão próximas umas das outras e são capazes de captar o sinal de todas as outras.

Figura 5 – Cenários modelados no NSG (PENG-JUNG, 2016)



Legenda: (a) Cenário modelado com duas estações (1 CBR, 1 Null); (b) Cenário modelado com cinco estações CBR e um agente Null; (c) Cenário modelado com nove estações CBR e uma estação com agente Null; (d) Cenário com 20 estações (19 CBR, 1 Null); (e) Cenário com 30 estações (29 CBR, 1 Null); (f) Cenário com 40 estações (39 CBR, 1 Null); (g) Cenário com 50 estações (49 CBR, 1 Null).

A estação que contém o agente *Null* é sempre a estação de número ($n_{max} - 1$), onde n_{max} é o número máximo de estações avaliadas (2, 5, 10, 20, 30, 40 e 50). No exemplo da Figura 5c, nove estações (n_0 a n_8) enviam fluxos CBR simultaneamente para uma única estação receptora (n_9) localizada nas coordenadas (18,1); totalizando 10 estações ativas durante todo o tempo simulação. Nas simulações executadas com a quantidade de estações concorrentes *xnodes* = {20, 30, 40, 50} (Figuras 5d-5g), os cenários foram com as estações posicionadas horizontalmente e verticalmente, formando matrizes. Na Figura 5g podemos perceber a modelagem de uma matriz 5 x 10 para comportar todas as 50 estações.

As avaliações de desempenho foram realizadas no modo básico de coordenação do *CSMA/CA*, pois todas as estações estão no alcance umas das outras, não ocorrendo o problema do terminal escondido. A utilização do modo básico em redes sem fio IEEE 802.11 é definida através do parâmetro *dot11RTSThreshold.* Se o valor do parâmetro *dot11RTSThreshold* for maior que o tamanho dos pacotes UDP e CBR, significa que mensagens de controle RTS/CTS não serão enviadas.

Como na simulação os pacotes UDP e CBR foram definidos em 1500 bytes e o parâmetro *dot11RTSThreshold* foi configurado no *script* TCL com o valor 3000, mensagens de controle RTS/CTS não serão utilizadas.

Todos os cenários detalhados a seguir foram modelos utilizando os parâmetros do NS-2 (NS-2, 2015) descritos na Tabela 5.

Tabela 5 – Parâmetros do NS-2 utilizados nas simulações.

Parâmetro	**Valor**
Área de Simulação	1300 x 1300m
Duração da Simulação	60s
Tipo de tráfego	CBR
Protocolo de Roteamento	DSR
Limite de retentativas RTS	7
CW_{Max}	1023
CW_{Min}	31
RTSThreshold	3000
Taxa básica (MAC)	1 Mbps
Taxa de dados (MAC)	11 Mbps
Taxa de fluxo CBR	11 Mbps
Tamanho do Pacote CBR	1500 bytes

As temporizações e demais medidas seguem o padrão IEEE 802.11b, sendo definidas como 14 μs para o ACK, 10 μs para SIFS, tempo de slot de 20 μs e consequentemente 50 μs

para DIFS (DIFS = 2 * tempo de slot + SIFS). O tempo de transmissão do preâmbulo e cabeçalho na camada física foi definido em 192 μs. O valor de CW_{Min} definido pela camada física é de 31. As taxas de dados e de quadros de controle foram definidas em 11 Mbps e 1 Mbps, respectivamente.

São utilizados dois tipos de cenários. Um cenário denominado síncrono e outro cenário assíncrono. Em cada um dos cenários, é mantido o valor de fator multiplicativo *r* e variamos a quantidade de estações *xnodes* competindo pelo meio em *xnodes* = {2, 5, 10, 20, 30, 40 e 50}. Foram executadas 10 rodadas de simulação para cada combinação: cenário, fator e quantidade de estações; e utilizaram-se intervalos de confiança de 95%, apresentados como barras de erro nos gráficos. Foram avaliadas as seguintes métricas de desempenho:

- Vazão agregada do sistema (em kbps) – corresponde ao somatório de todas as vazões individuais dos fluxos CBR existentes na simulação;
- Índice de Justiça – corresponde a valores que variam entre 0 e 1, que determinam o grau de justiça durante o compartilhamento de um canal comum entre todas as estações.

Para avaliar a métrica de desempenho de vazão em cada um dos cenários, utilizaremos o somatório de todos os fluxos como um único fluxo, destinado à estação central, chamada de vazão agregada.

Caso os resultados obtidos de vazão agregada apresentem desempenhos semelhantes, ou seja, visualmente pelos intervalos de confiança não podemos afirmar que são diferentes, será necessária a aplicação do teste *t* (ou *t-student*) de King (2003), para avaliação da hipótese H_0: $\mu_X = \mu_Y$. A hipótese H_0 determina se existe ou não diferença entre as amostras de vazão dos dois algoritmos avaliados, para um determinado nível de significância. Em nossa análise utilizamos nível de significância igual a 95% (0,05).

Ao comparar o fator multiplicativo A com B, se o módulo do valor calculado de *t* (t_{calc}) for maior que zero e inferior ao módulo de *t* crítico retirado da tabela ***t***, devemos aceitar a hipótese H_0 (KING, 2003). Logo, devemos aceitar que os valores se encontram dentro da região H_0, desconsiderando os limites superiores e inferiores. Ou seja, conclui-se que não há diferença significativa entre as amostras apresentadas do fator multiplicativo A comparado com B, para o nível de significancia de 95%. Caso o valor calculado seja negativo, significa que ao comparar o fator multiplicativo A com B, o fator multiplicativo B apresentou melhores resultados de vazão que A.

Inicialmente, compararemos o desempenho de vazão entre o algoritmo exponencial binário de *backoff* (BEB) e o algoritmo exponencial binário modificado de *backoff* (MBEB),

com objetivo de confirmar o melhor desempenho obtido com a utilização de uma função de decremento exponencial, ao invés do *reset* a CW_{Min}. Em seguida, faremos uma avaliação do desempenho de vazão para cada um dos fatores multiplicativos $r \in Z^*_+ = \{3,4,5\}$, propostos por Cano et al. (2013), porém utilizando as configurações padrão do 802.11b, ao invés dos parâmetros de uma rede PLC. A ideia é avaliar o impacto de se utilizar apenas 4 ou 5 estágios *i* de *backoff* no MBEB, ao invés do MBEB tradicional que possui 6 estágios *i*. Após a análise com fatores multiplicativos $r \in Z^*_+ = \{3,4,5\}$, propomos a avaliação também do desempenho de vazão para outros fatores multiplicativos $r \in Z^*_+ = \{6,7,8,9,10\}$ de forma que existam apenas 3 estágios *i* de *backoff* nas funções de incremento e decremento. E pra finalizar, propomos uma avaliação também do desempenho de vazão do caso mais extremo utilizando o fator multiplicativos *r* igual a 33, que fornece apenas 2 estágios *i* de *backoff*.

Para obter o índice de justiça (IJ), é necessário analisar o fluxo CBR de cada estação individualmente a fim de identificar quanto cada estação transmitiu durante todo o período de simulação. Cada estação tem uma identificação (fid) única e exclusiva do fluxo de pacotes CBR enviado da estação origem (agente CBR) para a estação destino (agente Null). O índice de justiça determina o quão justo um cenário ambiente foi perante todas as estações ativas e em disputa pelo meio durante o tempo de simulação. O Índice de Justiça é calculado conforme a Equação 16:

$$IJ = \frac{(\Sigma_f V_f)^2}{numfluxos * \Sigma_f (V_f)^2}, \quad (16)$$

onde *numfluxos* é o número de fluxos CBR e V_f é a vazão do fluxo de uma estação específica (JAIN, 1991). No cálculo, se o índice de justiça se aproxima de 1, o compartilhamento do canal se torna mais justo perante às estações concorrentes. Já se o índice se aproxima de 0, poucas estações conseguiram utilizar o meio. Fluxos CBR com tamanho de pacotes igual a 1500 *bytes* foram aplicados. Todas as estações possuem uma mesma taxa de fluxo CBR de 11 Mbps. A comunicação entre as estações terá a duração de 60 s.

Como não há disputa pelo acesso ao meio no cenário com duas estações (Figura 5a), uma estação gerando fluxo CBR e outra recebendo todo o tráfego, avaliamos o índice de justiça apenas para as demais quantidades de estações iguais a 5, 10, 20, 30, 40 e 50.

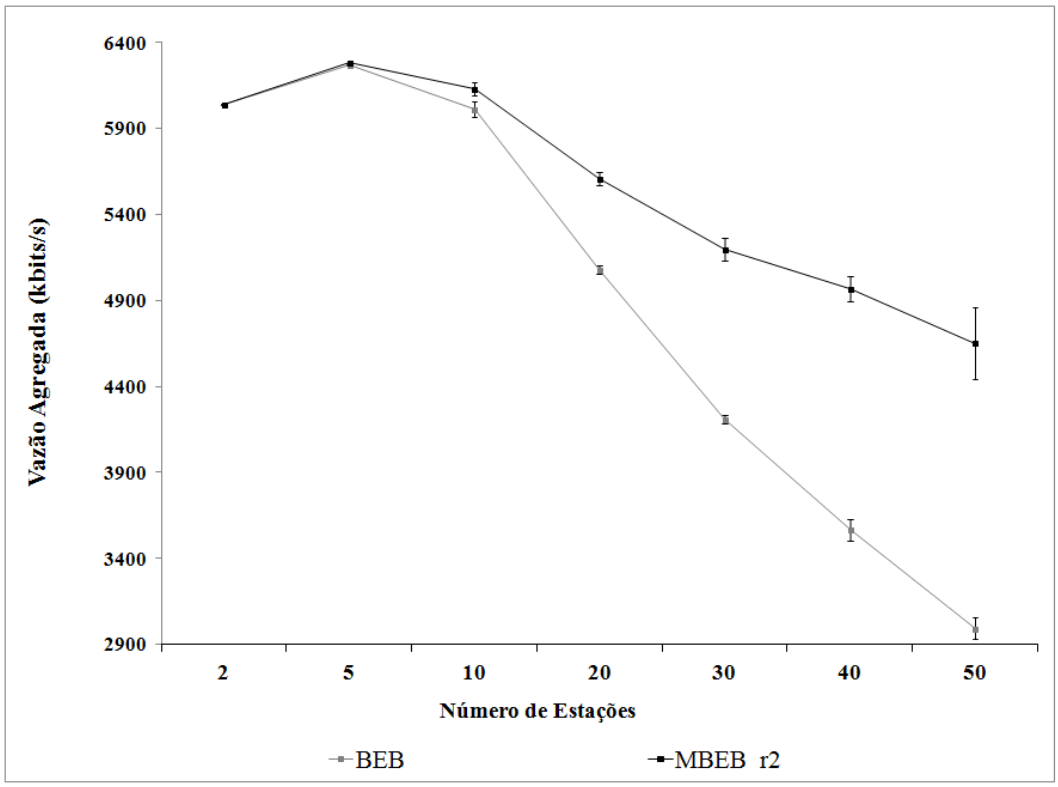
5.2 Cenário Síncrono

Inicialmente, o desempenho de vazão do sistema foi avaliado considerando que todas as estações iniciam simultaneamente seus respectivos fluxos individuais UDP-CBR no instante t = 0 s; por isso foi denominado cenário síncrono. O objetivo era modelar os cenários apresentados conforme a Figura 5, de forma que todas as estações responsáveis por enviar pacotes para a estação central iniciassem a disputa pelo acesso no início da simulação. Assim, seria possível avaliar o desempenho de vazão e justiça dos algoritmos de *backoff* em um cenário de saturação, comum em avaliações de desempenho de métodos de acesso ao meio.

5.2.1 Resultados de Vazão

Na Figura 6 é apresentado o resultado de vazão em *kbps*, no cenário síncrono, dos algoritmos de *backoff* BEB e MBEB, em função da quantidade de estações concorrentes na rede.

Figura 6 – Cenário Síncrono: Vazão Agregada BEB x MBEB



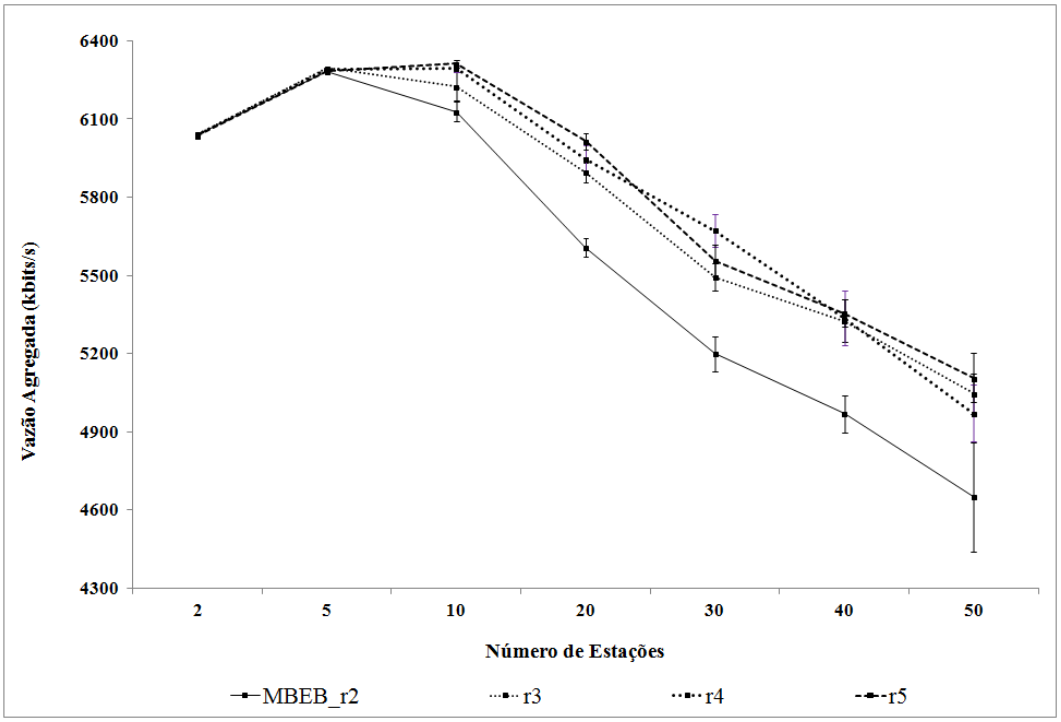
Legenda: Média dos resultados de vazão agregada dos algoritmos de *backoff* BEB e MBEB em função da quantidade de estações concorrentes(taxa de dados de 11 Mbps e tamanho de quadro igual a 1500 bytes)

Podemos observar na Figura 6 que os algoritmos BEB e MBEB apresentam resultados de desempenho de vazão muito semelhantes para uma pequena quantidade (2 e 5) de estações concorrentes ao meio. No entanto, à medida que se eleva a quantidade de estações (10, 20, 30, 40 e 50) concorrentes, o desempenho do MBEB é bem superior ao desempenho do BEB. Ambos os algoritmos apresentam pior desempenho conforme o número de estações aumenta. Esses resultados suportam as teorias de Wattanamongkhol et al. (2007) e Haitao et al. (2002), e oferecem base para as alterações e elaboração da nova proposta de algoritmo de *backoff* (nMBEB).

Em seguida, avaliamos o desempenho de vazão para cada um dos fatores multiplicativos $r \in Z^{*+} = \{3,4,5\}$, propostos por Cano et al. (2013) e o comparamos com o desempenho do MBEB que utiliza o fator multiplicativo *r* igual a 2. Podemos observar na Figura 7 que o desempenho dos fatores multiplicativos $r \in Z^{*+} = \{3,4,5\}$ é superior ao MBEB para a quantidade de estações (10, 20, 30, 40 e 50). Pela Figura 7, observamos que apesar das médias amostrais de vazão para $r \in Z^{*+} = \{3,4,5\}$ parecerem semelhantes, existe uma pequena diferença no desempenho de vazão utilizando o fator multiplicativo *r* igual a 4 em

relação ao fator multiplicativo *r* igual a 3. Da mesma forma, é possível observar que existe uma melhoria no desempenho do fator multiplicativo *r* igual a 5 em relação ao fator multiplicativo *r* igual a 4, para a maioria das diferentes quantidades de estações.

Figura 7 – Cenário Síncrono: Vazão MBEB [r2] x MBEB [r3] [r4] e [r5]



Legenda: Média dos resultados de vazão agregada dos algoritmos de *backoff* MBEB utilizando fator multiplicativo *r* = 2 e MBEB utillizando fatores multiplicativos *r* = {3, 4, e 5}, em função da quantidade de estações concorrentes (taxa de dados de 11 Mbps e tamanho de quadro igual a 1500 bytes)

Como para os fatores multiplicativos *r* iguais a 4 e 5 existem quatro estágios *i* de *backoff*, selecionamos o fator multiplicativo *r* igual a 5 e realizamos o teste *t-student* entre suas amostras de vazão e as geradas utilizando o fator multiplicativo *r* igual a 3. O objetivo é comparar o desempenho do algoritmo MBEB modificado utilizando 5 e 4 estágios *i* de *backoff*.

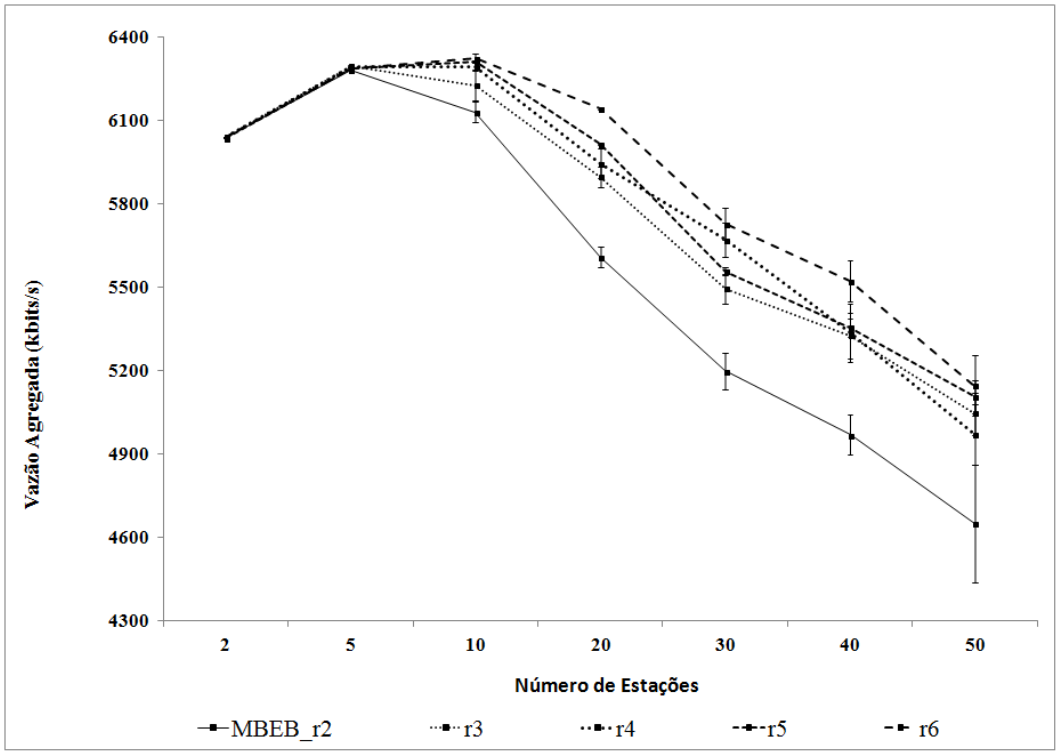
Tabela 6 – Cenário Síncrono: Testes t-student entre MBEB [r5] e MBEB [r3]

	r5 x r3	***tcrit***
#nós	***tcalc***	
2	-3,9246	2,0017
5	-4,6680	2,0017
10	3,0511	2,0017
20	4,5958	2,0017
30	1,5204	2,0017
40	0,5706	2,0017
50	0,9737	2,0017

Conforme a Tabela 6, comparamos os resultados de média de vazão entre o fator multiplicativo *r* igual a 5 e o fator multiplicativo *r* igual a 3. Se o valor de *t* calculado do fator multiplicativo *r* igual a 5 for maior que *t* crítico, significa que o desempenho de vazão do fator *r* igual a 5 é superior ao fator *r* igual a 3. Conforme a Tabela 6 o valor de *t* calculado (*tcalc*) é superior ao *t* crítico para *xnodes* igual a 10 e 20 estações. Assim, podemos concluir que para essas quantidades de estações, a média do desempenho de vazão obtida nas amostras com fator $r = 5$, foi significante superior ao desempenho de vazão obtido nas amostras do algoritmo de *backoff* utilizando fator *r* igual a 3. Já para *xnodes* igual a 30, 40 e 50, o valor de *t* calculado (*tcalc*) é inferior ao *t* crítico, portanto, não podemos afirmar que o desempenho é superior com o nível de significância de 95%. A hipótese é que com essas quantidades de estações, a melhoria no desempenho entre os fatores multiplicativos *r* não é tão significativa; e, pelos resultados encontrados, o desempenho de vazão varia bastante durante as rodadas de simulação, produzindo grandes intervalos de confiança. Por isso pelos testes t-student não podemos aceitar a hipótese H_0 de que os algoritmos são diferentes para *xnodes* igual a 30, 40 e 50. Para *xnodes* igual a 2 e 5, o *t* calculado (*tcalc*) apresentou valores negativos, portanto a média dos resultados de vazão do fator multiplicativo *r* igual a 5 foi inferior ao fator *r* igual a 3.

Mesmo havendo ajustes nos parâmetros do NS-2 (NS-2, 2015) utilizados durante as simulações, até aqui, a análise do desempenho dos fatores multiplicativos $r \in Z^{*+} = \{3,4,5\}$ baseia-se na proposta original de Cano et al. (2013). Como já avaliamos o desempenho do MBEB, utilizando 4, 5 e 6 estágios *i* de *backoff*, resolvemos aprofundar o estudo e fazer uma avaliação com apenas 3 estágios de *backoff*. Conforme a Tabela 4, o primeiro fator multiplicativo *r* a gerar apenas 3 estágios *i* de *backoff* é o 6. Por isso, adicionamos seus resultados de desempenho de vazão. Na Figura 8, observa-se que o desempenho de vazão do MBEB utilizando fator *r* igual 6 é superior ao fator *r* igual a 5 e principalmente ao fator *r* igual a 3.

Figura 8 – Cenário Síncrono: Vazão MBEB [r2] [r3] [r4] [r5] x MBEB [r6]



Legenda: Média dos resultados de vazão agregada dos algoritmos de backoff: MBEB tradicional, MBEB utilizando fatores multiplicativos *r* = {3, 4, e 5} e MBEB utilizando fator *r* igual a 6, em função da quantidade de estações concorrentes (taxa de dados de 11 Mbps e tamanho de quadro igual a 1500 bytes).

Conforme a Tabela 7 a seguir, o valor de *t* calculado (*tcalc*) na comparação entre *r* iguais a 6 e 5, é superior ao *t* crítico para *xnodes* igual a 20, 30 e 40 estações concorrentes.

Tabela 7 – Cenário Síncrono: Testes t-student entre MBEB [r6] e MBEB [r5]

	r6 x r5	***t-crit***
#nós	***tcalc***	
2	-0,6184	2,0017
5	-0,3374	2,0017
10	0,8879	2,0017
20	7,6457	2,0017
30	3,9432	2,0017
40	3,2567	2,0017
50	0,5363	2,0017

Assim, podemos concluir que para essas quantidades de estações, o desempenho de vazão obtida nas amostras com fator *r* igual a 6, foi superior ao desempenho de vazão obtido nas amostras do algoritmo de *backoff* utilizando fator *r* igual a 5. Já para *xnodes* igual a 10 e 50, o valor de *t* calculado (*tcalc*) é inferior ao *t* crítico, portanto, não podemos afirmar que o desempenho é superior com o nível de significância de 95%. Para uma quantidade pequena de estações menor ou igual a 5, já podemos observar que para qualquer fator multiplicativo,

provavelmente obteremos desempenhos muito semelhantes ou até inferiores. Já para uma grande quantidade de estações, como 50 por exemplo, a obtenção de grandes intervalos de confiança apresentados aumentou a incerteza das conclusões.

No entanto, quando comparamos o desempenho entre o fator multiplicativo *r* igual a 6 com o fator 3 e aplicamos os testes *t-student*, fica evidente o ganho no desempenho de vazão quando reduzimos a quantidade de estágios *i* de *backoff*, de 5 para 3. Através da Tabela 8, podemos observar que o *t* calculado foi superior ao *t* crítico para quase todas as quantidades de estações, exceto para *xnodes* igual a 2 e 5.

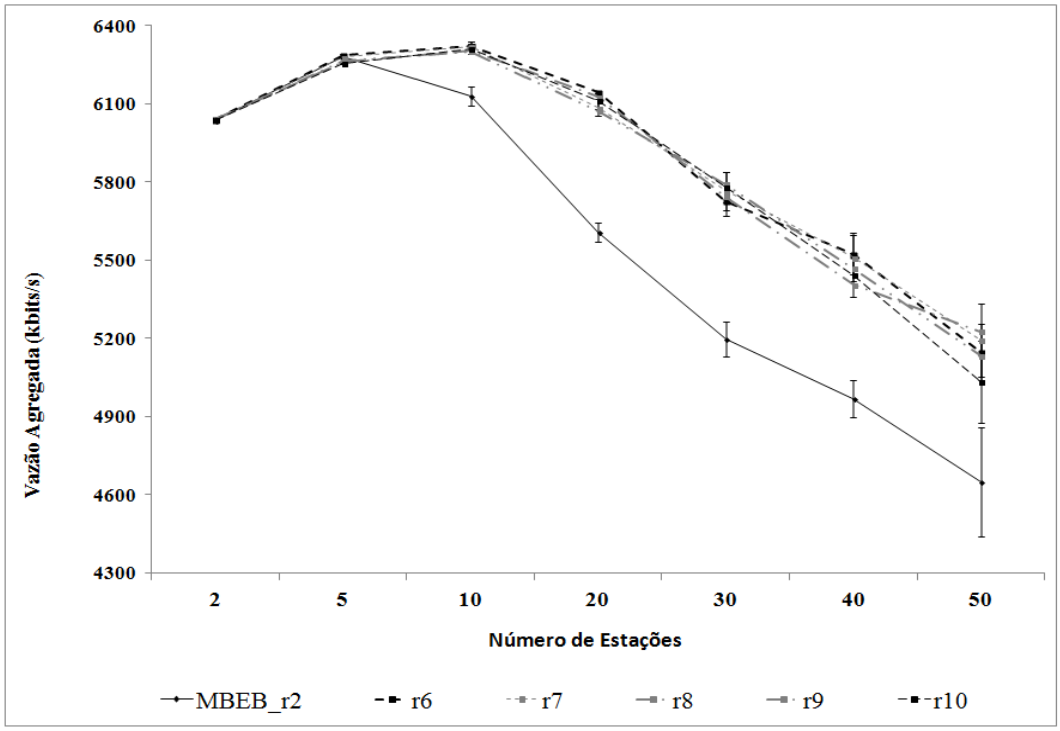
Tabela 8 – Cenário Síncrono: Testes *t-student* entre MBEB [r6] x MBEB [r3]

	r6 x r3	***tcrit***
#nós	***Tcalc***	
2	-0,6184	2,0017
5	-3,8770	2,0017
10	3,3706	2,0017
20	12,5202	2,0017
30	5,7182	2,0017
40	3,4266	2,0017
50	4,5578	2,0017

Considerando a melhoria no desempenho ao utilizar o fator *r* igual a 6 perante a utilização dos fatores 5 e 3, que apresentam quantidade de estágios *i* de *backoff* iguais a 4 e 5 respectivamente, resolvemos avaliar a possibilidade de alguma melhoria de desempenho ao utilizar fatores *r* iguais a 7, 8, 9 e 10; os quais assim como o fator 6 apresentam apenas 3 estágios *i* de *backoff*.

Na Figura 9, vemos o gráfico comparativo com os resultados da média de desempenho de cada um dos fatores *r* iguais a 6, 7, 8, 9 e 10. Podemos perceber uma semelhança muito grande entre os resultados, porém com pequenas diferenças de desempenho entre os fatores para diferentes quantidades de estações.

Figura 9 – Cenário Síncrono: Vazão MBEB [r2] x MBEB [r6] [r7] [r8] [r9] e [r10]



Legenda: Média dos resultados de vazão agregada do algoritmo de *backoff* MBEB tradicional e MBEB utilizando os fatores *r* = {6, 7, 8, 9 e 10} em função da quantidade de estações concorrentes (taxa de dados de 11 Mbps e tamanho de quadro igual a 1500 bytes)

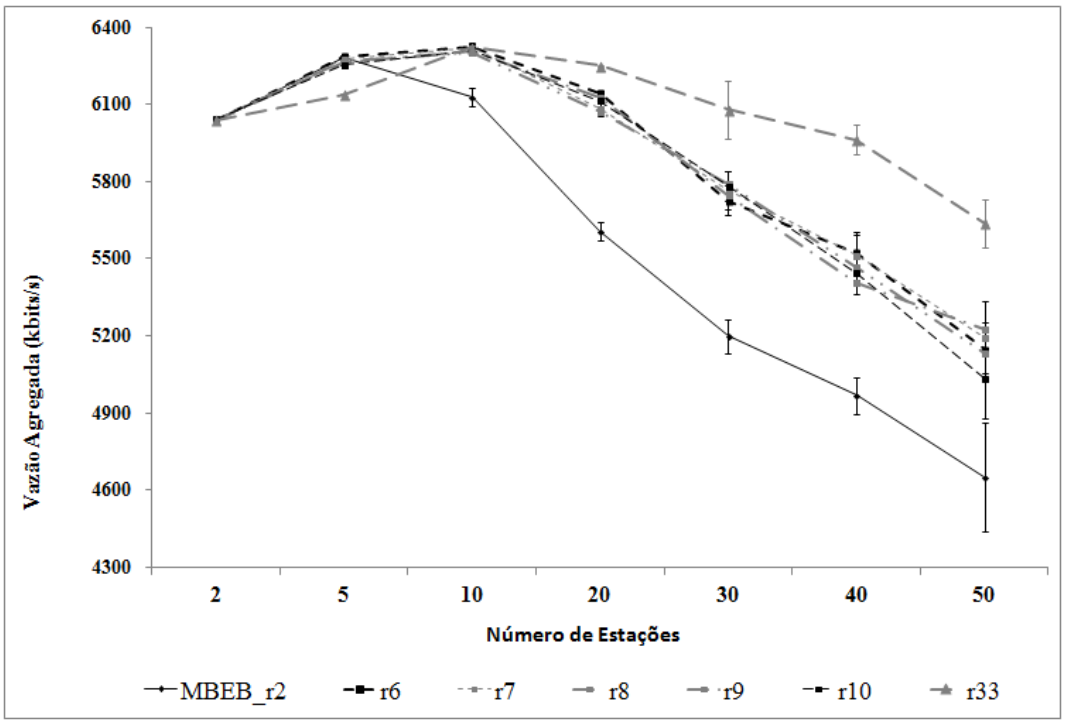
Escolhemos o último fator multiplicativo *r* igual a 10, como sendo referência para realização do teste *t-student* com os demais fatores que possuem apenas 3 estágios *i* de *backoff* (6, 7, 8 e 9). Os resultados dos testes apresentados na Tabela 9 mostram valores de *t* calculado (*tcalc*) inferiores a *t* crítico, demonstrando que as médias de desempenho do fator multiplicativo *r* igual a 10 são semelhantes às médias dos demais fatores multiplicativos para o grau de significância de 95%. Na Tabela 4, podemos perceber que os valores de CW utilizados no segundo estágio *i* de *backoff*, estão muito próximos, demonstrando assim coerência e semelhança com os resultados de desempenho encontrados e com os testes *t-student* apresentados na Tabela 9.

Tabela 9 – Cenário Síncrono: Testes *t-student* entre MBEB [r10] x MBEB [r6] [r7] [r8] [r9]

	r10 x r6	**r10 x r7**	**r10 x r8**	**r10 x r9**	***t-crit***
#nós	***tcalc***	***tcalc***	***tcalc***	***tcalc***	
2	-0,0208	0,0000	-0,2746	0,8375	2,0017
5	-14,2205	-12,4472	-4,4977	-4,1298	2,0017
10	-0,8163	-0,7254	0,6391	0,4313	2,0017
20	-1,7145	1,1505	1,7214	-0,5166	2,0017
30	1,2359	0,3097	-0,2465	-1,1386	2,0017
40	-1,0463	-1,0463	-0,4612	0,7279	2,0017
50	-1,1381	-1,4691	-1,0085	-1,6766	2,0017

Em seguida, faz necessário comparar os resultados de vazão do fator multiplicativo *r* iguais a 33, que representa apenas dois estágios *i* de *backoff*, com o MBEB tradicional e com os fatores r que representam 3 estágios i de backoff.

Figura 10 – Cenário Síncrono: Vazão MBEB [r2] [r6] [r7] [r8] [r9] e [r10] x MBEB [r33]



Legenda: Média dos resultados de vazão agregada do algoritmo de *backoff* MBEB tradicional, MBEB utilizando os fatores *r* = {6, 7, 8, 9, 10} comparados com o MBEB utilizando o fator r igual a 33, em função da quantidade de estações concorrentes (taxa de dados de 11 Mbps e tamanho de quadro igual a 1500 bytes)

Observa-se pela Figura 10, que o desempenho de vazão do MBEB utilizando fator *r* igual 33 é superior aos fatores multiplicativos *r* que representam três estágios *i* de *backoff* (6, 7, 8, 9 e 10), principalmente para uma quantidade de estações *xnodes* superior a 10 estações concorrentes. Para comprovar os resultados, realizamos o teste t-student entre o fator r igual a 33 e o fator r igual a 10. Pela Tabela 10, podemos observar e comprovar tais resultados já obeservados no teste visual. O desempenho de vazão do fator *r* igual a 33 é inferior ao fator *r* igual a 10 quando a quantidade de estações concorrentes é inferior a 10 estações.

Tabela 10 – Cenário Síncrono: Testes *t-student* entre MBEB [r33] e MBEB [r10]

#nós	**r33 x r10** ***tcalc***	***t-crit***
2	0.6600	2.0017
5	-55.3451	2.0017
10	1.1012	2.0017
20	7.7919	2.0017
30	8.1215	2.0017
40	9.9296	2.0017
50	6.3707	2.0017

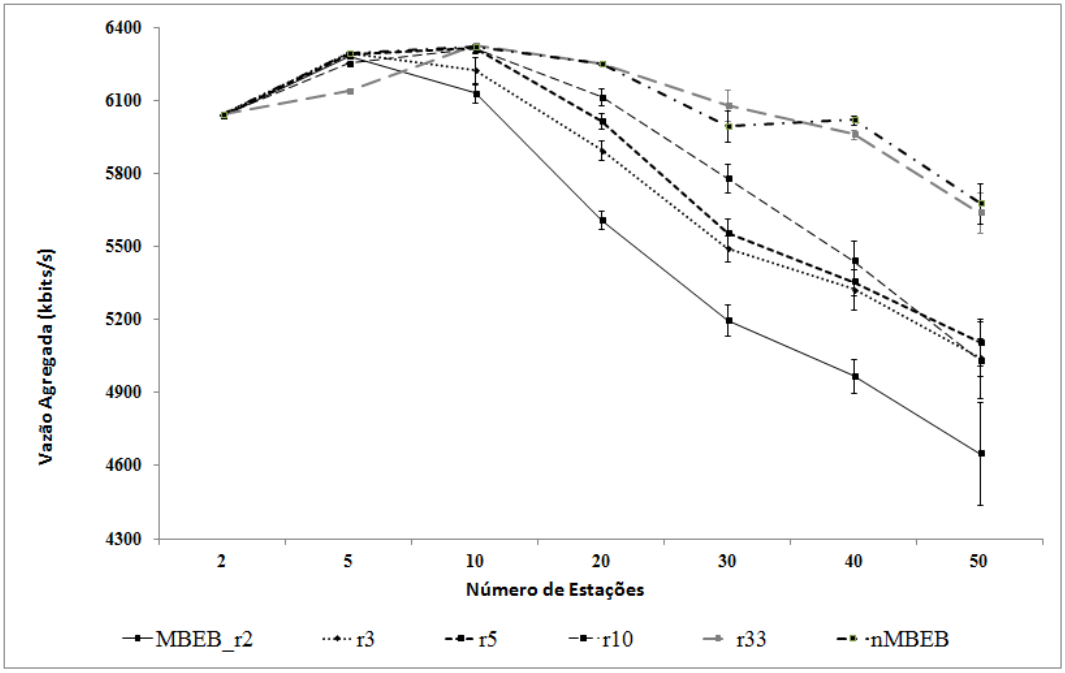
Pelos resultados encontrados na avaliação do cenário síncrono, escolhemos os cinco fatores multiplicativos que representam cada quantidade de estágio *i* de *backoff*. Fator *r* igual a 2, representante de 6 estágios; fator *r* igual a 3, representante de 5 estágios; fator *r* igual a 5, representante de 4 estágios; fator *r* igual a 10, representante de 3 estágios; e fator *r* igual a 33, representante de 2 estágios. Selecionamos dentre os cinco, os fatores com melhores desempenhos para cada quantidade de estações concorrentes (*n*) e os aplicamos na modelagem da nova proposta de algoritmo de *backoff* nMBEB, da seguinte forma:

- Se $1 \leq xnodes < 10$ então o *r* utilizado será igual a 5;
- Se $xnodes > 10$ então o *r* utilizado será igual a 33;

Aplicamos os fatores multiplicativos 5 e 33, conforme a quantidade de estações ativas no último intervalo de tempo (*timer_interv*), durante todo o período de uma simulação. O ciclo (*timer_interv*) de verificação das estações vizinhas foi definido como 500 ms, ou seja, a cada 500ms, o algoritmo nMBEB verifica quais estações enviaram quadros (controle ou dados) e remove da lista os índices das estações que ficaram inativas durante o último ciclo.

Na Figura 11 apresentamos o gráfico comparativo entre as médias de vazão obtidas com a nova proposta de algoritmo nMBEB e as médias de vazão dos demais fatores escolhidos para representar cada quantidade de estágios *i* de *backoff*.

Figura 11 – Cenário Síncrono: MBEB [r2] [r3] [r5] [r10] [r33] x nMBEB



Legenda: Gráfico comparativo da média dos resultados de vazão agregada entre a nova proposta de *backoff* nMBEB e os principais fatores multiplicativos *r* que representam às quantidades de estágios *i* de retransmissão, Gráfico dado em função da quantidade de estações concorrentes (taxa de dados de 11 Mbps e tamanho de quadro igual a 1500 bytes)

No teste visual, nota-se que os resultados encontrados de médias de vazão da nova proposta de *backoff* nMBEB aprimora o desempenho para uma quantidade de estações concorrentes inferior a 10 e se mantém semelhante aos resultados do fator *r* igual a 33.

Tabela 11 – Cenário Síncrono: Testes *t-student* entre o nMBEB e MBEB [r33]

#nós	**nMBEB x r33** *tcalc*	***t-crit***
2	-0.7652	2.0017
5	67.2405	2.0017
10	-2.0576	2.0017
20	0.3575	2.0017
30	-2.1846	2.0017
40	1.8151	2.0017
50	0.5996	2.0017

Ao realizar o teste t-student entre a nova proposta e o fator *r* igual a 33, conforme Tabela 11, pode-se perceber que o nMBEB aprimora o desempenho do MBEB, utilizando o fator r igual a 33, no cenário com 5 estações. Nos demais cenários com 2, 10, 20, 30, 40 e 50 estações, como os *ts* calculados ficaram abaixo do *t* crítico, os testes não são conclusivos; pelos resultados do teste visual podemos concluir que são semelhantes.

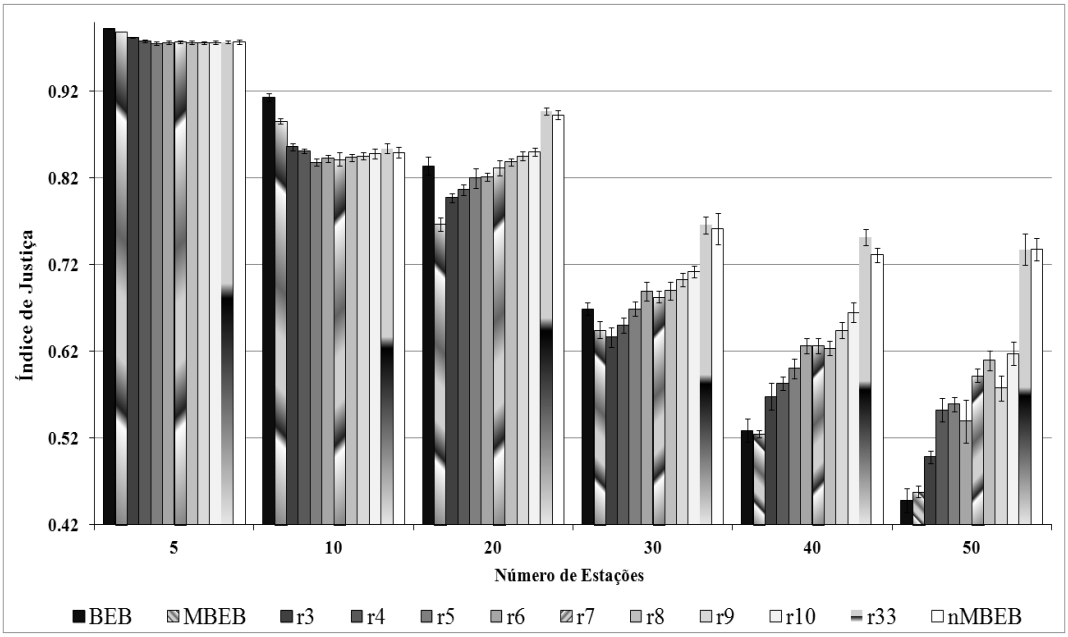
5.2.2 Resultados de Justiça

Após avaliação de vazão, iniciamos a análise da métrica relativa ao índice de justiça conforme Equação 16 da Seção 5.1, para comparar o desempenho dos algoritmos. Não avaliamos o índice de justiça para duas estações pelo fato de se tratar de um cenário onde há somente uma estação remetente e outra destinatária.

Conforme a Figura 12, percebe-se uma diminuição no índice de justiça à medida que se aumenta a quantidade de estações concorrentes na rede e um aumento conforme o fator multiplicativo *r* cresce. Nota-se que, com 5 e 10 estações concorrentes, a rede apresenta um índice de justiça mais alto utilizando os algoritmos BEB e MBEB. Acima de 10 estações concorrentes, os resultados se invertem e os maiores índices de justiça são obtidos com fatores multiplicativos *r* mais elevados.

Valores de índice de justiça próximos a 1 demonstram uma rede bastante justa. Já com valores iguais a 0,5, o sistema se torna 50% injusto e 50% justo (JAIN, 1999). Com valores de índice de justiça abaixo de 0,5, o sistema se torna majoritariamente injusto (JAIN, 1999). Pela Figura 12, nota-se acima de 40 estações concorrentes, a rede começa apresentar características de um sistema que começa a tender para um sistema injusto, utilizando os algoritmos tradicionais de *backoff* (BEB e MBEB). Com 50 estações, a rede se torna mais injusta que justa, apresentando valores de IJ inferiores a 0,5 para os algoritmos mais tradicionais BEB e MBEB, e para o fator multiplicativo r igual a 3.

Figura 12 – Cenário Síncrono: Análise de Justiça



Legenda: Análise de justiça de todos os fatores multiplicativos de *backoff*, incluindo o algoritmo tradicional BEB e a nova proposta nMBEB, em função da quantidade de estações concorrentes (taxa de dados de 11 Mbps e tamanho de quadro igual a 1500 bytes)

5.3 Cenário Assíncrono

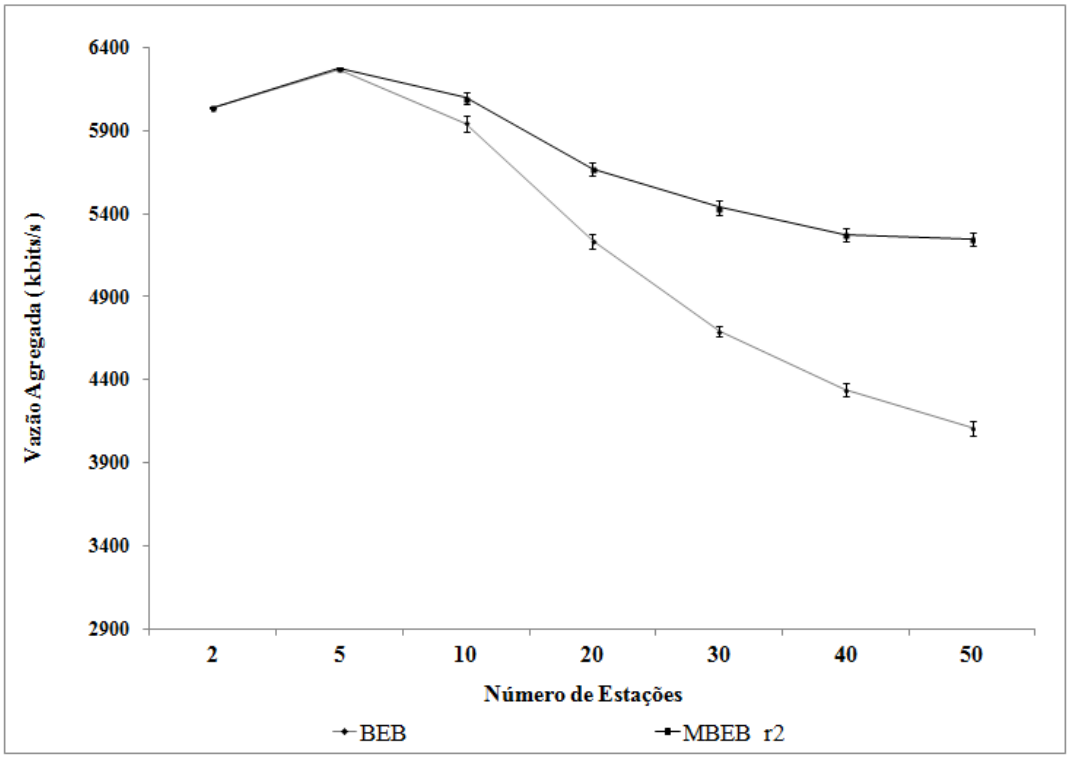
No cenário síncrono apresentado anteriormente já podemos perceber a influência da quantidade de estágios *i* de *backoff* nas métricas de desempenho avaliadas. No entanto, como todas as estações iniciam seus fluxos UDP-CBR no mesmo instante t = 0 s e terminam em t = 60s, a assincronicidade ocorre mais pela contenção das estações do que pela aleatoriedade nos fluxos CBR gerados pelas estações.

Para avaliar as métricas de desempenho da nova proposta de algoritmo de *backoff* (nMBEB), seria importante realizar uma variação na quantidade de estações durante a simulação. Por isso, propomos um cenário onde estações iniciam seus tráfegos de maneira gradativa durante as simulações. Neste cenário, todas as estações iniciam seus respectivos fluxos individuais UDP-CBR em instantes *t* distintos e terminam ao mesmo tempo em *t* igual a 60s. Não se trata de uma aleatoriedade, pois cada estação iniciará seus fluxos CBR sequencialmente em instantes predefinidos. A estação n1 iniciará seu fluxo no instante t = 1, a estação n2 no instante t = 2, e assim sucessivamente.

5.3.1 *Resultados de Vazão*

Na Figura 13, é apresentado o resultado de vazão em *kbps*, no cenário assíncrono, dos algoritmos de *backoff* BEB e MBEB, em função da quantidade de estações concorrentes.

Figura 13 – Cenário Assíncrono: Vazão Agregada BEB x MBEB



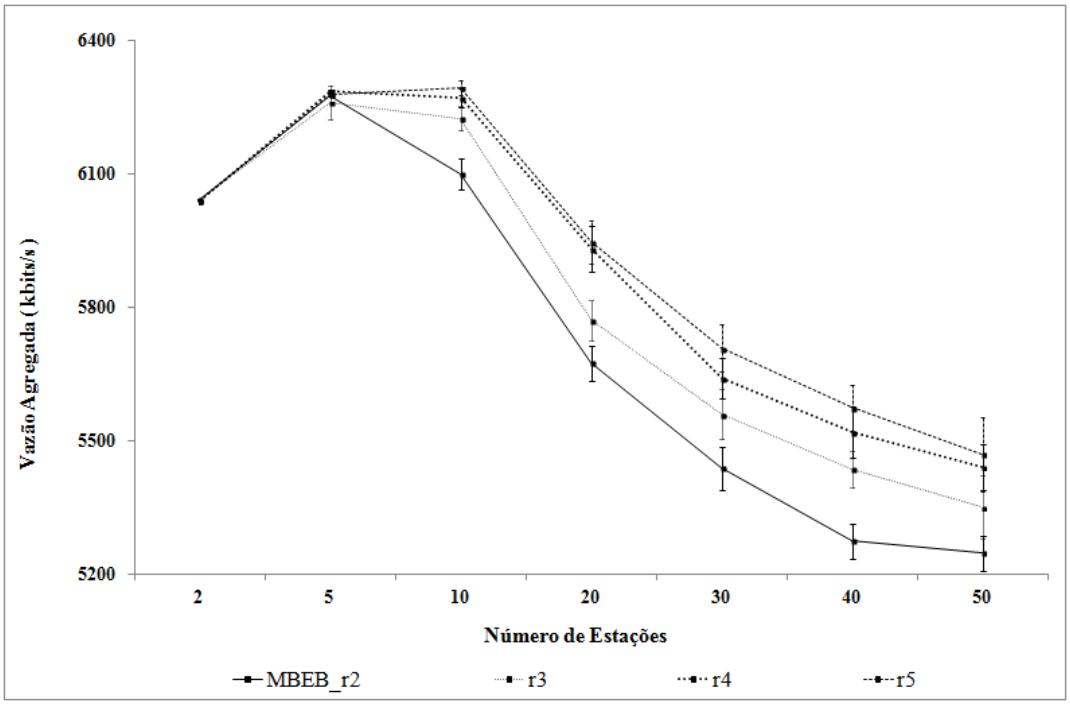
Legenda: Média dos resultados de vazão agregada dos algoritmos de *backoff* BEB e MBEB em função da quantidade de estações concorrentes (taxa de dados de 11 Mbps e tamanho de quadro igual a 1500 bytes)

Podemos observar na Figura 13 que, assim como no cenário síncrono, os algoritmos BEB e MBEB apresentam resultados de desempenho de vazão praticamente idênticos para pequenas quantidades (2 e 5) de estações, em disputa pelo meio. Já nas simulações com mais estações (10, 20, 30, 40 e 50), o desempenho do MBEB é bem superior ao desempenho do BEB. Cabe destacar que os valores das médias de vazão no cenário assíncrono foram superiores aos obtidos no cenário síncrono (Figura 6).

Na sequência, avaliamos o desempenho de vazão no cenário assíncrono para cada um dos fatores multiplicativos $r \in Z^{*+} = \{3,4,5\}$, propostos por Cano et al. (2013). Pela Figura 14, podemos observar que o desempenho dos fatores multiplicativos $r \in Z^{*+} = \{3,4,5\}$ se manteve superior ao MBEB, exceto em cenários com 2 e 5 estações. É possível visualmente

identificar uma diferença de desempenho nas médias amostrais de vazão para $r \in Z^{*+} = \{3,4,5\}$, principalmente entre o fator 5 e 3, além de todos perante ao MBEB.

Figura 14 – Cenário Assíncrono: Vazão MBEB [r2] x MBEB [r3] [r4] e [r5]



Legenda: Média dos resultados de vazão agregada dos algoritmos de *backoff* MBEB utilizando o fator multiplicativo $r = 2$ e MBEB utilizando os fatores $r = \{3, 4, e\ 5\}$, em função da quantidade de estações concorrentes (taxa de dados de 11 Mbps e tamanho de quadro igual a 1500 bytes)

No entanto, por apresentar alguns intervalos de confiança sobrepostos, principalmente entre os fatores 4 e 5, resolvemos realizar o teste *t-student* entre eles para melhor análise dos resultados. Pelos dados apresentados na Tabela 12, não existe diferença significativa nos resultados de média de vazão entre os fatores *r* iguais a 5 e 4 para o grau de significância de 95%, principalmente porque ambos os fatores representam a mesma quantidade de estágios *i* de *backoff*.

Tabela 12 – Cenário Assíncrono: Testes *t-student* entre MBEB [r5] e MBEB [r4]

#nós	**r5 x r4** ***tcalc***	***t-crit***
2	-1,3921	2,0017
5	-4,5207	2,0017
10	1,4969	2,0017
20	0,4204	2,0017
30	1,8426	2,0017
40	1,2293	2,0017
50	0,5864	2,0017

Como as simulações, no cenário assíncrono, apresentaram resultados muito próximos

entre os fatores *r* iguais a 4 e 5, ambos com 4 estágios *i* de *backoff*, selecionamos o fator multiplicativo *r* igual a 5 e realizamos o teste *t-student* entre suas amostras de vazão e as geradas pelo fator *r* igual a 3, que representa 5 estágios *i* de *backoff*.

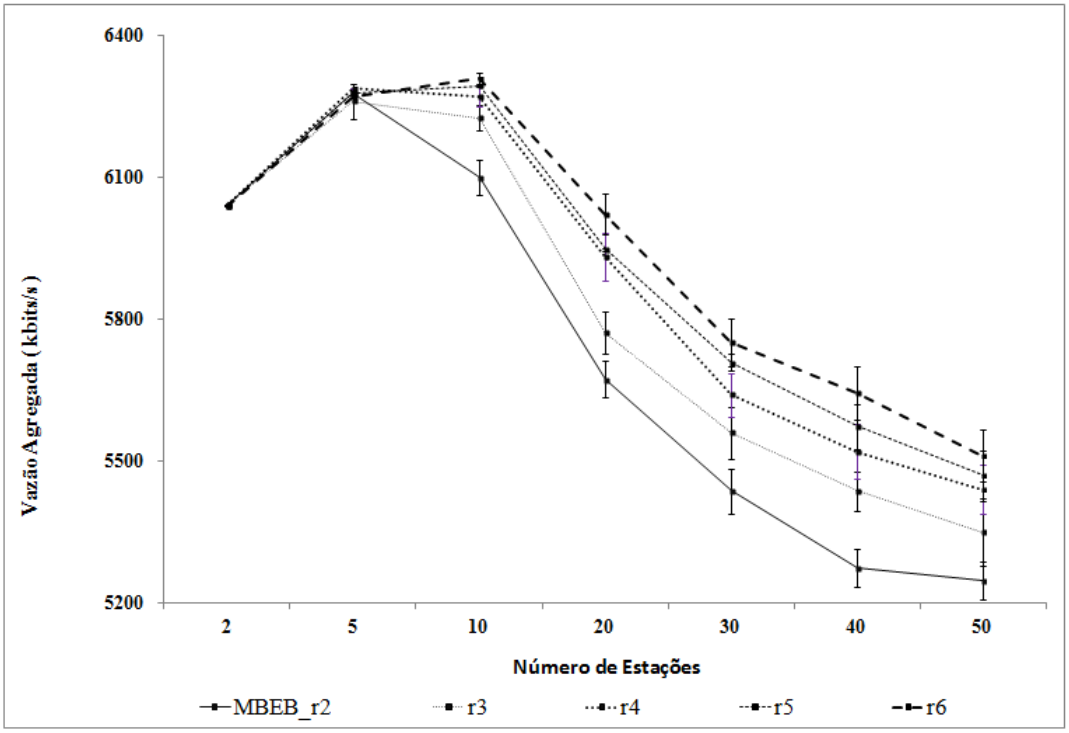
Tabela 13 – Cenário Assíncrono: Testes *t-student* entre MBEB [r5] e MBEB [r3]

	r5 x r3	
#nós	***tcalc***	***t-crit***
2	-11,274	2,0017
5	0,9773	2,0017
10	4,1070	2,0017
20	5,1636	2,0017
30	3,7191	2,0017
40	3,5607	2,0017
50	2,1029	2,0017

Conforme a Tabela 13, o valor de *t* calculado (*tcalc*) é superior ao *t* crítico para *xnodes* igual a 10, 20, 30, 40 e 50 estações concorrentes. Logo, podemos concluir que a média do desempenho de vazão obtida nas amostras com fator *r* igual a 5, foi superior ao obtido nas amostras com fator *r* igual a 3. Já para *xnodes* igual a 5 estações, o valor de *t* calculado (*tcalc*) é inferior ao *t* crítico, portanto, não podemos afirmar que o desempenho é superior com o grau de signifiância de 95%. Para *xnodes* igual a 2 estações o desempenho do fator *r* igual a 5 foi inferior ao fator *r* igual a 3.

O próximo passo da avaliação é analisar o desempenho, em cenários assíncronos, de fatores multiplicativos *r* iguais a 6, 7, 8, 9 e 10, que representam apenas três estágios *i* de *backoff*. Como pela Tabela 4, o primeiro fator multiplicativo *r* a gerar apenas 3 estágios *i* de *backoff* é o 6, adicionamos seus resultados de desempenho de vazão. Na Figura 15 a seguir, observa-se que o desempenho de vazão do MBEB utilizando fator *r* igual 6 é superior ao fator *r* igual a 5 e principalmente ao fator *r* igual a 3.

Figura 15 – Cenário Assíncrono: Vazão MBEB [r2] [r3] [r4] [r5] x MBEB [r6]



Legenda: Média dos resultados de vazão agregada dos algoritmos de backoff: MBEB tradicional, MBEB utilizando fatores multiplicativos *r* = {3, 4, e 5} e o MBEB utilizando fator *r* igual a 6, em função da quantidade de estações concorrentes (taxa de dados de 11 Mbps e tamanho de quadro igual a 1500 bytes).

Conforme a Tabela 14 a seguir, o valor de *t* calculado (*tcalc*) na comparação entre os fatores *r* iguais a 6 e 5, é superior ao *t* crítico somente para *xnodes* igual a 20 estações concorrentes.

Tabela 14 – Cenário Assíncrono: Testes *t-student* entre MBEB [r6] x [r5]

	r6 x r5	
#nós	***tcalc***	***tcrit***
2	-0,6875	2,0017
5	-4,5248	2,0017
10	1,2602	2,0017
20	2,2515	2,0017
30	1,1185	2,0017
40	1,5965	2,0017
50	0,8110	2,0017

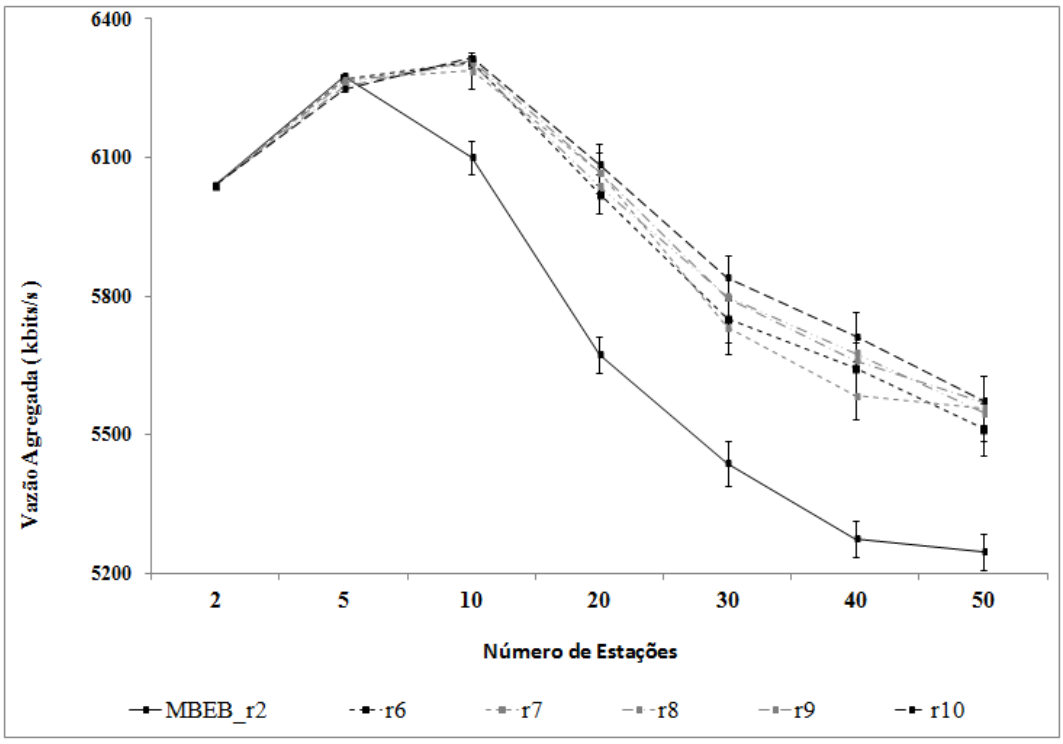
Pelos resultados do test *t-student* apresentados na Tabela 14, não podemos concluir que o fator *r* igual a 6 apresenta alguma melhoria de desempenho em relação ao fator *r* igual a 5. No entanto, conforme a Tabela 15, podemos afirmar que fator *r* igual a 6 possui um melhor desempenho quando comparado com o fator *r* igual a 3, exceto para *xnodes* = 2 e 5 estações concorrentes.

Tabela 15 – Cenário Assíncrono: Testes *t-student* entre MBEB [r6] x [r3]

#nós	r6 x r3 *tcalc*	*tcrit*
2	-1,8201	2,0017
5	0,5239	2,0017
10	5,3493	2,0017
20	7,7607	2,0017
30	4,9130	2,0017
40	5,6973	2,0017
50	3,4324	2,0017

Porém ao avaliar a média de desempenho de vazão dos demais fatores *r* iguais a 7, 8, 9 e 10, que também representam apenas 3 estágios *i* de *backoff*, podemos perceber através da Figura 16 que existem muitas semelhanças nos resultados. No entanto, para escolher um fator que representasse apenas 3 estágios *i* de *backoff*, selecionamos novamente o fator multiplicativo *r* igual a 10 por apresentar uma mínima diferença na média de resultados de vazão em relação aos demais fatores que representam os mesmos 3 estágios *i* de *backoff*.

Figura 16 – Cenário Assíncrono: Vazão MBEB [r2] x MBEB [r6] [r7] [r8] [r9] e [r10]



Legenda: Média dos resultados de vazão agregada do algoritmo de *backoff* MBEB tradicional e MBEB utilizando os fatores *r* = {6, 7, 8, 9 e 10}, em função da quantidade de estações concorrentes (taxa de dados de 11 Mbps e tamanho de quadro igual a 1500 bytes)

Para avaliar o ganho no desempenho de vazão ao reduzir o número de estágios *i* de *backoff* de 4 pra 3, em um cenário assíncrono, escolhemos o último fator multiplicativo *r* igual a 10, para realização do teste *t-student* com o fator multiplicativo *r* igual a 5. Através da

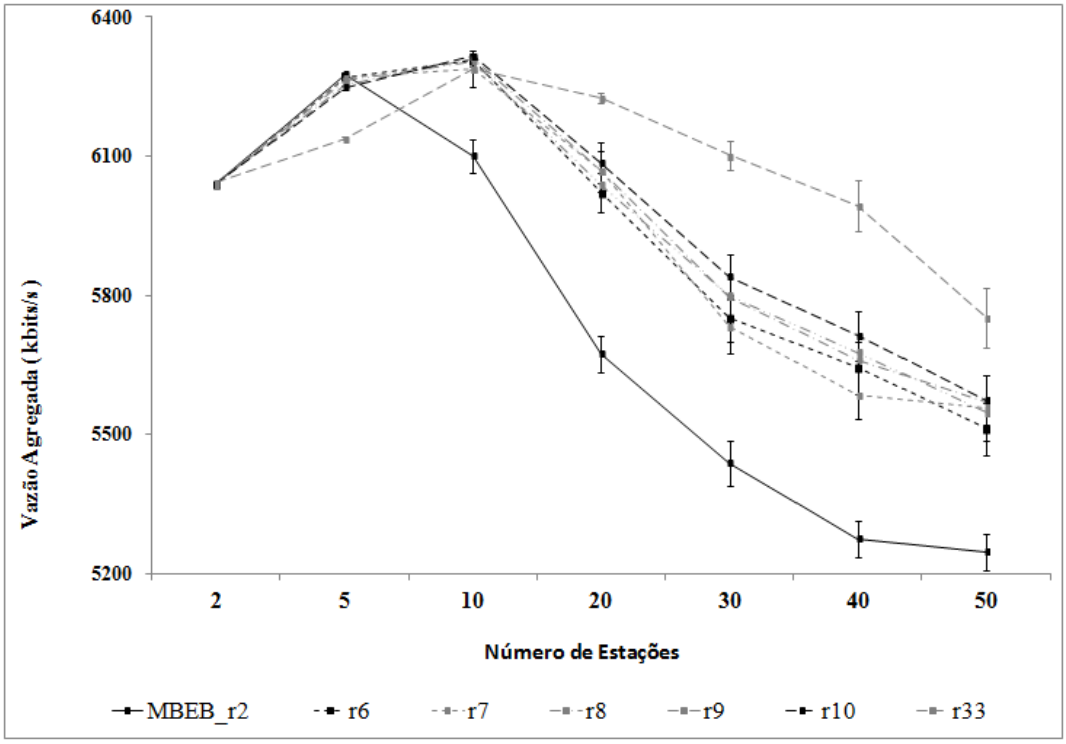
Tabela 16 prova-se que os resultados apresentados pelo fator multiplicativo *r* igual a 10 são superiores aos resultados do fator multiplicativo *r* igual a 5, exceto para *xnodes* = 2, concluindo-se que há uma melhoria quando da redução da quantidade de estágios *i* de *backoff* de 4 para 3.

Tabela 16 – Cenário Assíncrono: Testes *t-student* entre MBEB [r10] e MBEB [r5]

	r10 x r5	
#nós	***Tcalc***	***tcrit***
2	0,3432	2,0017
5	12,4975	2,0017
10	2,3333	2,0017
20	4,1180	2,0017
30	3,5534	2,0017
40	3,2673	2,0017
50	2,0315	2,0017

O passo seguinte da avaliação é analisar o desempenho, em cenários assíncronos, do fator multiplicativo *r* iguais a 33, que representa apenas dois estágios *i* de *backoff*. Para isso adicionamos na Figura 17, seus resultados de desempenho de vazão.

Figura 17 – Cenário Assíncrono: Vazão MBEB [r2] [r6] [r7] [r8] [r9] e [r10] x MBEB [r33]



Legenda: Média dos resultados de vazão agregada do algoritmo de *backoff* MBEB utilizando os fatores *r* = {2, 6, 7, 8, 9, 10 e 33}, em função da quantidade de estações concorrentes (taxa de dados de 11 Mbps e tamanho de quadro igual a 1500 bytes)

Observa-se que o desempenho de vazão do MBEB utilizando fator *r* igual 33 é superior aos

fatores multiplicativos *r* que representam três estágios *i* de *backoff*, principalmente para uma quantidade de estações *xnodes* superior a 10 estações concorrentes. Para comprovar os resultados, realizamos o teste t-student entre o fator *r* igual a 33 e o fator *r* igual a 10. Pela Tabela 17, podemos observar e comprovar tais resultados já obeservados no teste visual. O desempenho de vazão do fator *r* igual a 33 é inferior ao fator *r* igual a 10 quando a quantidade de estações concorrentes é inferior a 10 estações.

Tabela 17 – Cenário Assíncrono: Testes *t-student* entre MBEB [r33] e MBEB [r10]

#nós	r33 x r10 *tcalc*	*t-crit*
2	2.0793	2.0017
5	-44.8690	2.0017
10	-13.0529	2.0017
20	6.1365	2.0017
30	9.0535	2.0017
40	7.2256	2.0017
50	4.1107	2.0017

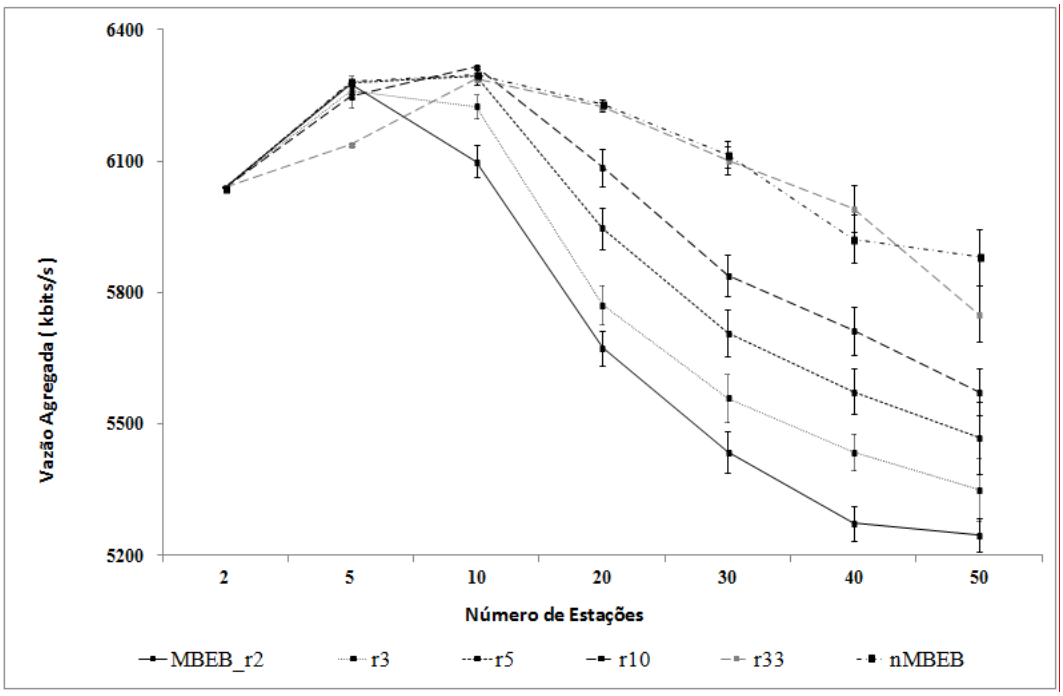
Pelos resultados encontrados na avaliação do cenário assíncrono, novamente escolhemos os cinco fatores multiplicativos que representam cada quantidade de estágio *i* de *backoff*. Fator *r* igual a 2, representante de 6 estágios; fator *r* igual a 3, representante de 5 estágios; fator *r* igual a 5, representante de 4 estágios; fator *r* igual a 10, representante de 3 estágios; e fator *r* igual a 33, representante de 2 estágios. Selecionamos dentre os cinco, os fatores com melhores desempenhos para cada quantidade de estações concorrentes (*n*) e os aplicamos na modelagem da nova proposta de algoritmo de *backoff* nMBEB, da seguinte forma:

- Se $1 \leq xnodes < 10$ então o *r* utilizado será igual a 5;
- Se $xnodes > 10$ então o *r* utilizado será igual a 33;

Aplicamos os fatores multiplicativos 5 e 33, conforme a quantidade de estações ativas no último intervalo de tempo (*timer_interv*), durante todo o período de uma simulação. O ciclo (*timer_interv*) de verificação das estações vizinhas foi definido como 500 ms, ou seja, a cada 500ms, o algoritmo nMBEB verifica quais estações enviaram quadros (controle ou dados) e remove da lista os índices das estações que ficaram inativas durante o último ciclo.

Na Figura 18 apresentamos o gráfico comparativo entre as médias de vazão obtidas com a nova proposta de algoritmo nMBEB e as médias de vazão dos demais fatores escolhidos para representar cada quantidade de estágios *i* de *backoff*.

Figura 18 – Cenário Assíncrono: MBEB [r2] [r3] [r5] [r10] [r33] x nMBEB



Legenda: Gráfico comparativo da média dos resultados de vazão agregada entre a nova proposta de *backoff* nMBEB e os principais fatores multiplicativos *r* que representam às quantidades de estágios *i* de retransmissão, Gráfico dado em função da quantidade de estações concorrentes (taxa de dados de 11 Mbps e tamanho de quadro igual a 1500 bytes)

Nota-se, pelos resultados encontrados de médias de vazão, que a nova proposta de *backoff* nMBEB melhora o desempenho de vazão com a quantidade de estações concorrentes inferior a 10 e com a quantidade de estações igual a 50. Para comprovar realizamos o teste *t-student* entre o fator r igual a 33, que apresentou anteriormente o melhor desempenho entre os fatores, e a nova proposta nMBEB.

Tabela 18 – Cenário Assíncrono: Testes *t-student* entre o nMBEB e MBEB [r33]

#nós	nMBEB x r33 *tcalc*	*t-crit*
2	-3.6425	2.0017
5	79.2404	2.0017
10	4.7169	2.0017
20	1.1153	2.0017
30	0.6114	2.0017
40	-1.7963	2.0017
50	2.7286	2.0017

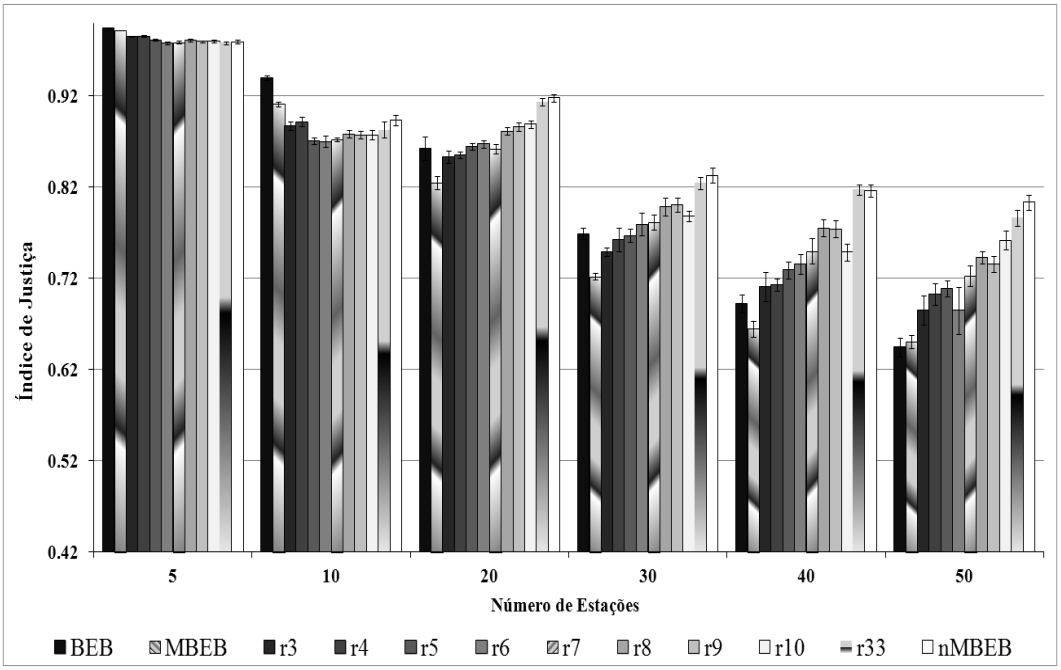
Pela Tabela 18, pode-se perceber que o nMBEB possui um desempenho superior ao MBEB, utilizando o fator r igual a 33, no cenário com 5, 10 e 50 estações. Já no cenário com 20, 30 e 40 estações, o desempenho do nMBEB foi semelhante ao desempenho do MBEB com fator *r* igual a 33.

5.3.2 Resultados de Justiça

Na análise do índice de justiça realizada no cenário assíncrono, conforme a Figura 19, é possível perceber uma certa similaridade com os resultados encontrados no cenário síncrono (Figura 12). À medida que se aumenta a quantidade estações concorrentes na rede, melhores resultados são encontrados com a utilização de fatores multiplicativos r iguais a 6, 7, 8, 9, 10 e 33. A utilização do fator multiplicativos r igual a 2, somente apresenta melhores resultados quando exisitir um número de estações inferior a 10.

Pelas características do cenário assíncrono, compreendemos que no cenário com 30 estações, por exemplo, somente após 30 segundos de simulação que todas as 30 estações iniciaram seus fluxos CBR e experimentam assim, a mesma concorrência encontrada no cenário síncrono, já com 1 segundo de simulação. Como o início dos fluxos CBR das estações acontece de forma gradual, percebe-se que as primeiras estações que iniciaram o envio de seus quadros antes das demais, têm uma maior quantidade de quadros enviados ao final da simulação, apresentando assim maiores valores de vazão individual (V_f) na Equação 16. O comportamento se aplica a todos os fatores *r* e para todas as quantidades de estações. Por apresentarem melhores fluxos de vazão é possível perceber que, no geral, todos os fatores apresentaram melhores índices de justiça ao comparar com o cenário síncrono.

Figura 19 – Cenário Assíncrono: Análise de Justiça



Legenda: Análise de justiça de todos os fatores multiplicativos de *backoff*, incluindo o algoritmo tradicional BEB e a nova proposta nMBEB, em função da quantidade de estações concorrentes (taxa de dados de 11 Mbps e tamanho de quadro igual a 1500 bytes)

Com melhores índices de justiça, entende-se que estações ficam menos tempo em contenção. Na proposta do nMBEB, menos estações em contenção, implica em menos estações removidas da lista de estações vizinhas ativas pela função *erase_nodes()* (ANEXO I). Como o nMBEB foi modelado para utilizar o fator *r* igual a 5 com menos de 10 estações concorrentes e utilizar o fator *r* igual a 33 com mais de 10 estações concorrentes, remover menos estações da lista significa que mais estações estarão ativas na rede e que durante quase todo o tempo de simulação o fator *r* igual a 33 será utilizado. Isso explica o porquê das médias de desempenho de vazão e justiça, no nMBEB, serem semelhantes ao MBEB utilizando o fator *r* igual a 33.

CONCLUSÕES

Este trabalho assumiu como objetivos avaliar a aplicação de funções não tradicionais de incremento e decremento pertencentes ao algoritmo de *backoff* que se adaptam mais rapidamente às variações de carga na rede, e avaliar o desempenho da nova proposta de algoritmo de *backoff* denominado nMBEB (*Modified Binary Exponential Backoff Algorithm - with node control*) utilizando tais funções supracitadas. Para tal, esta avaliação apoiou-se na análise dos resultados de vazão e justiça.

Ao analisar os resultados de vazão e justiça no cenário síncrono (cenário tradicional de saturação da rede), onde todas as estações iniciavam seus fluxos ao mesmo tempo em t = 0 s, constatou-se que fatores multiplicativos *r* superiores e distintos do tradicional utilizado nos algoritmos de *backoff* BEB e MBEB (*r = 2*), produzem um ganho de desempenho, principalmente quando testados em cenários com mais de 10 estações vizinhas ativas.

Fica evidente pelas Seções 5.2.1 e 5.3.1, que em simulações com número de estações acima de 10 que, ao reduzir somente um estágio *i* de *backoff*, há um ganho considerável na média de resultados de vazão agregada do sistema. Pelos resultados dos testes *t-student* apresentados no cenário síncrono, na comparação entre MBEB tradicional (fator *r* igual a 2) e MBEB utilizando o fator 3, podemos perceber que o ganho é evidente quando se reduz a quantidade de estágios *i* de *backoff* de 6 para 5. Em cenários com 10 estações, o ganho é em torno de 100 kbps. Já em cenários acima de 20 estações, o ganho aumenta para 300 kbps. O ganho adicional, com a redução da quantidade de estágios *i* de *backoff* de 5 para 4 e de 4 para 3, é um pouco menor quando comparamos com o ganho da redução de estágios *i* de *backoff* de 6 para 5. Já a redução de estágios *i* de *backoff* de 3 para 2, trouxe um ganho significativo de desempenho de vazão. Quando comparamos os resultados apresentados do algoritmo tradicional MBEB (fator *r* igual a 2) que possui seis estágios *i* de *backoff* com o algoritmo MBEB, utilizando o fator multiplicativo *r* que gera somente dois estágios *i* de *backoff*, temos um ganho acima de 600 kbps para número de estações acima de 20, podendo obter até 1Mbps de ganho com 50 estações.

Ao avaliar os resultados da nova proposta nMBEB, podemos concluir que há também um ganho em gerenciar vizinhos ativos e aplicar um melhor fator multiplicativo para a quantidade de estações concorrentes ao meio. As avaliações foram realizadas considerando

limiares de quantidade de estações que consideramos como ideais para aplicar fatores r iguais a 5 ou 33. No entanto, tais limiares podem ser ajustados.

Os resultados dos índices de justiça obtidos em cenários síncronos demonstram que à medida que se reduz a quantidade de estágios *i* de *backoff* entre CW_{Min} e CW_{Max}, melhores desempenhos são obtidos. É possível perceber que à medida que se aumenta a quantidade estações concorrentes na rede, algoritmos como os tradicionais BEB e MBEB (fator *r* igual a 2) apresentam mais rapidamente baixos índices de justiça. Alcançando patamares abaixo de 0.5, algumas estações podem iniciar processo de inanição prolongada.

Para complementar a avaliação, realizamos os mesmos testes de vazão e justiça, porém no cenário assíncrono, onde estações iniciavam seus respectivos fluxos em instantes *t* distintos, porém determinísticos e terminam ao mesmo tempo em *t* igual a 60 s. Na comparação entre os resultados de vazão no cenário síncrono e assíncrono, nota-se uma diferença na escala dos resultados de média de vazão para um número de estações acima de 10. No cenário síncrono, o resultado da média de vazão do fator multiplicativo *r* igual a 3 com 30 estações, foi de 5,493 Mbps. Já no cenário assíncrono, o resultado da média de vazão foi de 5,559 Mbps.

A diferença entre as escalas dos resultados de desempenho de vazão dos cenários, síncrono e assíncrono, pode ser explicada pelos melhores resultados de justiça encontrados no cenário assíncrono. Por experimentar melhores índices de justiça durante o tempo de simulação, estações ficam menos tempo em contenção tendo mais ocorrências de envio de quadros com sucesso, obtendo assim melhores resultados de vazão.

Em resumo, nos cenários síncronos e assíncronos obtivemos melhores resultados de vazão e justiça, utilizando a nova proposta nMBEB e fatores multiplicativos *r* maiores que 2, comparados com os tradicionais BEB e MBEB.

Trabalhos Futuros

Como sugestão inicial de para trabalho futuro, seria importante avaliar o desempenho de vazão e justiça do algoritmo de *backoff* MBEB em um cenário que apresentasse uma certa aleatoriedade não só na entrada de novos dispositivos na rede, mas também na saída dos dispositivos antes do término da simulação. No cenário assíncrono avaliado, todas as estações terminam seus fluxos CBR ao mesmo tempo e não há aleatoriedade no término dos fluxos

CBR. Dessa forma, seria possível avaliar o comportamento individual de cada fator multiplicativo r ao variar a diminuição de carga na rede. Na avaliação dos fatores multipicativos r, no cenário assíncrono, não foi feita a análise da diminuição de carga na rede como foi realizada no nMBEB.

Avaliar uma possível utilização do mecanismo de controle de estações vizinhas ativas na rede e aplicar diferentes tamanhos e métodos de fila de recepção dos pacotes (ifqType), seria uma ótima sugestão de trabalho futuro. Tornar o tamanho da fila variável conforme a carga na rede reduz a quantidade de pacotes descartados por falta de *buffer* no receptor; e aplicar o tipo de fila conforme a carga na rede poderia trazer benefícios para a qualidade de serviço (*Quality of Service* – QoS) no DCF 802.11 ao implementar diferentes prioridades para alguns tipos de serviços em redes saturadas.

REFERÊNCIAS

BHARGHAVAN, V.; DEMERS A.J., SHENKER S., et al. MACAW: a media access protocol for wireless LAN's. *ACM Special Interest Group on Data Communication (SIGCOMM), London, UK*, pp. 212–225, August 1994.

BIANCHI, G. Performance Analysis of the IEEE 802.11 DCF. *Selected Areas in Communications, IEEE Journal on*, v. 18, n. 3, p. 535-547, March 2000.

BORGIA, E.; ANASTASI G.; CONTI M.; GREGORI E. IEEE 802.11 Ad Hoc Networks: Protocols, Performance and Open Issues. *Proceedings of the Workshop on Mobile and Wireless Networks (MWN 2003), Providence (Rhode Island)*, May 2003.

CAMPISTA, M.; RUBINSTEIN, M. Advanced Routing Protocols for Wireless Networks, Wiley-IEEE Press, July 2014.

CANO, C.; MALONE, D. Evaluation of the Backoff procedure of Homeplug MAC vs. DCF. In: *Personal Indoor and Mobile Radio Communications (PIMRC). IEEE 24th International Symposium o*n, p. 1846-1850, September 2013.

FULLMER, C.L.; GARCIA, J.J. Floor acquisition multiple access (FAMA) for packet-radio networks. *ACM Special Interest Group on Data Communication (SIGCOMM), Massachusetts, MA*, p. 262–273, August 1995.

GAST, M. 802.11® Wireless Networks: The Definitive Guide - Creating and Administering Wireless Networks, O'Reilly, April 2002.

HAITAO, WU; CHENG, S.; PENG, Y.; LONG, K.; MA, J. IEEE 802.11 Distributed Coordination Function (DCF): Analysis and Enhancement. In: *Proc. ICC2002, New York*, p. 605–609, May 2002.

IEEE Std 802.11. IEEE Standard for Information technology — Telecommunications and information exchange between systems local and metropolitan area networks — Specific requirements – *IEEE Std 802.11™ - 2012*, June 2012.

Jain, R.; Durresi, A.; Babic, G. Throughput fairness index: An explanation. *ATM Forum/99-0045*, February 1999.

KARN, P. MACA - a new channel access protocol for packet radio. *ARRL/CRRL Amateur Radio Computer Networking Conference, Ontario, Canada*, p. 134–140, September 1990.

KING, B.M.; MINIUM, E.M. (2003). Statistical Reasoning in Psychology and Education. *4th Ed. New Jersey: John Wiley & Sons, Inc*, October 2003.

KSENTINI, A.; NAFAA, A.; GUEROUI, A.; NAIMI, M. Determinist contention window algorithm for IEEE 802.11. In: *Personal, Indoor and Mobile Radio Communications, 2005. PIMRC 2005. IEEE 16th International Symposium*, v. 4, p. 2712-2716, September 2005.

LIU, KE. Understanding the implementation of IEEE 802.11 MAC Standard in ns2. <http://www.cs.binghamton.edu/~kliu/research/ns2code/note.pdf>. Acesso em: Jan, 2016.

MARDINI, W. et al. An Adaptive Backoff Algorithm for Mobile Ad-Hoc Networks. *Department of Computer Science Jordan University of Science and Technology International Journal of Mobile Computing and Multimedia Communications*, v. 3, p.1-19, July 2011.

NS-2, 2015. Network Simulator - *USC University of Southern California* <http://isi.edu/nsnam/ns/>. Acesso em: December, 2015.

PERAHIA, E.; STACEY, R. Next Generation Wireless LANs 802.11n and 802.11ac. *Cambridge University Press*, v. 2, May 2013.

PENG-JUNG, WU. NS-2 Scenarios Generator (NSG). <https://sites.google.com/site/pengjungwu/nsg>. Acesso em: February, 2016.

QIN, Y.; ZHUANG, Y.; LIXIANG, M. Dynamic contention window adjustment scheme for improving throughput and fairness in IEEE 802.11 wireless LANs. *Global Communications Conference (GLOBECOM), 2012 IEEE,* p. 5074-5080, December 2012.

RAJ, JAIN. The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling. *Wiley-Interscience, New York, NY*, April 1991.

RUBINSTEIN, M. et al. Analysis of Medium Access Control Protocols for Home Networks. *Journal of Communication and Information Systems*, v. 22, p. 10-23, January 2007

SINGH, D.; PANDEY, B.; TOMAR, G.S.; SARKAR, B.K. Performance Evaluation of Backoff Method – Effect of Backoff Factor Algorithm. *Computational Intelligence and Communication Networks (CICN), 2013 5th International Conference*, p. 82-86, September 2013.

SONG, N.; KWAK, B.; SONG, J., MILLER, E. L. Enhancement of IEEE 802.11 DCF with EIED. *Vehicular Technology Conference, 2003. VTC 2003-Spring. The 57th IEEE Semiannual*, v. 4, p. 2775-2778, April 2003.

TANENBAUM, A. Redes de Computadores; Editora Campus; 4a edição, 2003.

TIAN, X.; CHEN X.; IDEGUCHI, T.; FANG, Y. Improving Throughput and Fairness in WLANs through Dynamically Optimizing *Backoff, IEICE Transactions and Communications*, p. 4328-4338, November 2005.

VELLOSO P. B.; ELIAS, M.; CAMPISTA, M.; CUNHA, D. D. O.; COSTA, L. H. M. K.; DUARTE, O. C. M. B. Analyzing the performance of wireless local area networks with an improved collision avoidance mechanism. *Revista da Sociedade Brasileira de Telecomunicações*, v. 19, n. 3, p. 53-60, December 2004.

VLACHOU, C.; HERZEN, J.; THIRAN, P. Fairness of MAC protocols: IEEE 1901 vs. 802.11. In: *Power Line Communications and its Applications (ISPLC) 2013 17th IEEE International Symposium on*, p. 58-63, March 2013.

WATTANAMONGKHOL, N. et al. Performance Analysis of Modified Backoff Algorithm in IEEE 802.11 Networks. *Internet, 2007. ICI 2007. 3rd IEEE/IFIP International Conference in Central Asia on*, p. 1-5, September 2007.

YANG, Y. Cloning a New IEEE 802.11 MAC Protocol in ns-2. <http://csie.nqu.edu.tw/smallko/ns2_old/MAC2.pdf>. Acesso em: 24 dez. 2015.

ANEXO I

O anexo I contém as três funções que foram adicionadas no arquivo mac-802_11t.cc, que contém as funções da subcamada MAC: uma função responsável pela inclusão de estações na lista de vizinhas ativas, outra função que calcula o fator multiplicativo baseado no número de estações ativas e uma terceira função que remove uma estação da lista de estações vizinhas.

```
void Mac802_11t::add_list_Node(int node)
{
	bool add = true;
	std::list<nodes>::iterator it;                                  // variavel para percorrer a lista: it
	list_nodes = (struct nodes*)malloc(sizeof(struct nodes));// variavel para guardar informações sobre o Noh na lista

	for (it = nodes_NUM.begin(); it != nodes_NUM.end(); it++) { //percorre a lista
		if((*it).num_node == node) { // verifica se Noh ja existe na lista
		 add = false;
		pthread_mutex_lock(&lock1); // protege a var (*it).t1 contra outras threads
		clock_gettime(CLOCK_PROCESS_CPUTIME_ID, &t1);
		(*it).t1 = t1.tv_nsec;                          // guarda timestamp do Noh
		pthread_mutex_unlock(&lock1);           // libera a variavel
			break;                  // sai porque ja achou o noh procurado
	}
	}
	if( add ) {                             // o Noh ainda nao existe na lista ou foi removido
		add = false;

		pthread_mutex_lock(&lock1); //protege a var list_nodes->t1 contra outras threads
		clock_gettime(CLOCK_PROCESS_CPUTIME_ID, &t1);
		list_nodes->t1 = t1.tv_nsec;                            // guarda timestamp do Noh
		pthread_mutex_unlock(&lock1);                   // libera a variavel

		list_nodes->num_node = node;            // guarda o numero do Noh
		nodes_NUM.push_back(*list_nodes);       // adiciona o Noh a lista

		std::cout << "No " << list_nodes->num_node <<
			" adicionado" //<< endl;
			<< " N:" << nodes_NUM.size() //<< endl;
		<< " Fator atual: " << calc_factor() << endl;
	}
	free(list_nodes);               // libera variavel
}
```

```
double Mac802_11t::calc_factor()
{
        double p;
        double f=1;
        double alfa;
        int n;

        n = nodes_NUM.size();                           // total de nós da lista RTS
        if (n > 0){
        if (n < 10)  f = 5;                             // fator r igual a 5
        if (n >= 10) f = 10;                            // fator r igual a 10
        }
        return f;
}


void *erase_nodes( void *ptr )
{
   struct timespec deltaT;       // diferenca entre o timestamp atual e anterior
   std::list<nodes>::iterator it; // variavel para percorrer a lista

   while(1) {
        usleep(142500); // delay de mais ou menos 1 segundo, depende do processador
        clock_gettime(CLOCK_PROCESS_CPUTIME_ID, &t2);  // pega timestamp atual

                for (it = nodes_NUM.begin(); it != nodes_NUM.end(); it++) { // percorre a
lista
                   pthread_mutex_lock(&lock2); // protege a var t1.tv_sec contra outras threads
                   t1.tv_nsec = (*it).t1;
                   pthread_mutex_unlock(&lock2);   // libera a variavel

                        deltaT.tv_nsec = diff(t1, t2).tv_nsec;// armazena o delta dos timestamps
            if(deltaT.tv_nsec > timer_interv)   // nenhum pacote enviado nos ultimos
timer_interv segundos
                   {
                        std::cout << "No rm:" << (*it).num_node << endl;
                        it = nodes_NUM.erase(it); // remove Noh
                   }
                }
   }
}
```

ANEXO II

O ANEXO II apresenta o código do NS-2 onde se encontra a função *recv_timer* que analisa todos os quadros recebidos por uma estação antes da verificação do endereço de destino do quadro.

```
void
Mac802_11t::recv_timer()
{
        u_int32_t src;
        hdr_cmn *ch = HDR_CMN(pktRx_);
        hdr_mac802_11 *mh = HDR_MAC802_11(pktRx_);
        u_int32_t dst = ETHER_ADDR(mh->dh_ra);
        u_int32_t ap_dst = ETHER_ADDR(mh->dh_3a);
        u_int8_t  type = mh->dh_fc.fc_type;
        u_int8_t  subtype = mh->dh_fc.fc_subtype;
        assert(pktRx_);
        assert(rx_state_ == MAC_RECV || rx_state_ == MAC_COLL);

      /*
       *  If the interface is in TRANSMIT mode when this packet
       *  "arrives", then I would never have seen it and should
       *  do a silent discard without adjusting the NAV.
       */
      if(tx_active_) {
            Packet::free(pktRx_);
            goto done;
      }
         /*
          * Handle collisions.
          */
         if(rx_state_ == MAC_COLL) {
                  discard(pktRx_, DROP_MAC_COLLISION);
                  set_nav(usec(phymib_.getEIFS()));
                  goto done;
         }
         /*
          * Check to see if this packet was received with enough
          * bit errors that the current level of FEC still could not
          * fix all of the problems - ie; after FEC, the checksum still
          * failed.
          */
         if( ch->error() ) {
```

```
                Packet::free(pktRx_);
                set_nav(usec(phymib_.getEIFS()));
                goto done;
        }

        /*
         * IEEE 802.11 specs, section 9.2.5.6
         *      - update the NAV (Network Allocation Vector)
         */
        if(dst != (u_int32_t)index_) {
                set_nav(mh->dh_duration);
                //std::cout << "Noh Destino do Quadro(mh->dh_ra): " << dst << " Noh Atual
(pktrx_): " <<  (u_int32_t)index_ << endl;
                add_list_Node(dst);
        }


        …

        …


        }
```