Universidade Federal de Goiás – UFG.
Escola de Engenharia Elétrica, Mecânica e de Computação
Programa de Pós-Graduação em Engenharia Elétrica e de Computação

Daniel Porto Queiroz Carneiro

Alocação de Recursos em Redes sem Fio Multiportadoras com Ondas Milimétricas Utilizando Aprendizado por Reforço Baseado em Modelo Markoviano

Goiânia

2022

Processo: 23070.032506/2022-73 Documento: 3046490

UNIVERSIDADE FEDERAL DE GOIÁS
ESCOLA DE ENGENHARIA ELÉTRICA, MECÂNICA E DE COMPUTAÇÃO

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO (TECA) PARA DISPONIBILIZAR VERSÕES ELETRÔNICAS DE TESES

E DISSERTAÇÕES NA BIBLIOTECA DIGITAL DA UFG

Na qualidade de titular dos direitos de autor, autorizo a Universidade Federal de Goiás (UFG) a disponibilizar, gratuitamente, por meio da Biblioteca Digital de Teses e Dissertações (BDTD/UFG), regulamentada pela Resolução CEPEC nº 832/2007, sem ressarcimento dos direitos autorais, de acordo com a Lei 9.610/98, o documento conforme permissões assinaladas abaixo, para fins de leitura, impressão e/ou download, a título de divulgação da produção científica brasileira, a partir desta data.

O conteúdo das Teses e Dissertações disponibilizado na BDTD/UFG é de responsabilidade exclusiva do autor. Ao encaminhar o produto final, o autor(a) e o(a) orientador(a) firmam o compromisso de que o trabalho não contém nenhuma violação de quaisquer direitos autorais ou outro direito de terceiros.

1. Identificação do material bibliográfico

[X] Dissertação [ ] Tese [ ] Outro*:_____

*No caso de mestrado/doutorado profissional, indique o formato do Trabalho de Conclusão de Curso, permitido no documento de área, correspondente ao programa de pós-graduação, orientado pela legislação vigente da CAPES.

Exemplos: Estudo de caso ou Revisão sistemática ou outros formatos.

2. Nome completo do autor

Daniel Porto Queiroz Carneiro

3. Título do trabalho

"Alocação de Recursos em Redes sem Fio Multiportadoras com Ondas Milimétricas Utilizando Aprendizado por Reforço Baseado em Modelo Markoviano"

4. Informações de acesso ao documento (este campo deve ser preenchido pelo orientador)

Concorda com a liberação total do documento [X] SIM [ ] NÃO[1]

[1] Neste caso o documento será embargado por até um ano a partir da data de defesa. Após esse período, a possível disponibilização ocorrerá apenas mediante:
a) consulta ao(à) autor(a) e ao(à) orientador(a);
b) novo Termo de Ciência e de Autorização (TECA) assinado e inserido no arquivo da tese ou dissertação.
O documento não será disponibilizado durante o período de embargo.
Casos de embargo:
- Solicitação de registro de patente;
- Submissão de artigo em revista científica;
- Publicação como capítulo de livro;
- Publicação da dissertação/tese em livro.

Obs. Este termo deverá ser assinado no SEI pelo orientador e pelo autor.

Processo: 23070.032506/2022-73 Documento: 3046490

Obs. Este termo deverá ser assinado no SEI pelo orientador e pelo autor.

Documento assinado eletronicamente por **Flavio Henrique Teles Vieira**, **Professor do Magistério Superior**, em 14/07/2022, às 16:44, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **DANIEL PÔRTO QUEIROZ CARNEIRO**, **Discente**, em 14/07/2022, às 16:47, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no site https://sei.ufg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **3046490** e o código CRC **9BBF6202**.

Daniel Porto Queiroz Carneiro

Alocação de Recursos em Redes sem Fio Multiportadoras com Ondas Milimétricas Utilizando Aprendizado por Reforço Baseado em Modelo Markoviano

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica e de Computação como requisito parcial para a obtenção do Título de Mestre em Engenharia Elétrica e de Computação.

Universidade Federal de Goiás – UFG

Escola de Engenharia Elétrica, Mecânica e de Computação

Programa de Pós-Graduação em Engenharia Elétrica e de Computação

Area de concentração: Engenharia de Computação

Orientador: Prof. Dr. Flávio Henrique Teles Vieira

Coorientador: Prof. Dr. Alisson Assis Cardoso

Goiânia

2022

Ficha de identificação da obra elaborada pelo autor, através do
Programa de Geração Automática do Sistema de Bibliotecas da UFG.

Carneiro, Daniel Porto Queiroz
Alocação de Recursos em Redes sem Fio Multiportadoras com Ondas Milimétricas Utilizando Aprendizado por Reforço Baseado em Modelo Markoviano [manuscrito] / Daniel Porto Queiroz Carneiro. - 2022.
cvi, 106 f.: il.

Orientador: Prof. Dr. Flávio Henrique Teles Vieira; co-orientador Dr. Alisson Assis Cardoso.
Dissertação (Mestrado) - Universidade Federal de Goiás, Escola de Engenharia Elétrica, Mecânica e de Computação (EMC), Programa de Pós-Graduação em Engenharia Elétrica e de Computação, Goiânia, 2022.
Bibliografia.
Inclui siglas, símbolos, gráfico, tabelas, algoritmos, lista de figuras, lista de tabelas.

1. Alocação de Recursos em rede 5G. 2. Aprendizado por Reforço. 3. Processo de decisão de Markov. 4. DQN adaptativa. I. Vieira, Flávio Henrique Teles, orient. II. Título.

CDU 621.3

Processo: 23070.032506/2022-73 Documento: 3025062

UNIVERSIDADE FEDERAL DE GOIÁS

ESCOLA DE ENGENHARIA ELÉTRICA, MECÂNICA E DE COMPUTAÇÃO

ATA DE DEFESA DE DISSERTAÇÃO

Ata nº 04 da sessão de Defesa de Dissertação de **Daniel Porto Queiroz Carneiro**, que confere o título de Mestre em **Engenharia Elétrica e de Computação**, na área de concentração em **Engenharia de Computação**.

Aos oito dias do mês de julho do ano de dois mil e vinte e dois, a partir das **14h00min**., realizou-se a sessão pública de Defesa de Dissertação intitulada **"Alocação de Recursos em Redes sem Fio Multiportadoras com Ondas Milimétricas Utilizando Aprendizado por Reforço Baseado em Modelo Markoviano"**. Os trabalhos foram instalados pelo Orientador, Professor Doutor **Flávio Henrique Teles Vieira (EMC/UFG)** com a participação dos demais membros da Banca Examinadora: Professor Doutor **Anderson da Silva Soares (INF/UFG)**, membro titular externo; Professor Doutor **Alisson Assis Cardoso (EMC/UFG)** co-orientador, membro titular interno e Professor Doutor **Rodrigo Pinto Lemos (EMC/UFG)** membro titular interno. Durante a arguição os membros da banca **não fizeram** sugestão de alteração do título do trabalho. A Banca Examinadora reuniu-se em sessão secreta a fim de concluir o julgamento da Dissertação, tendo sido o candidato **aprovado** pelos seus membros. Proclamados os resultados pelo Professor Doutor Flávio Henrique Teles Vieira, Presidente da Banca Examinadora, foram encerrados os trabalhos e, para constar, lavrou-se a presente ata que é assinada pelos Membros da Banca Examinadora, **aos oito dias do mês de julho do ano de dois mil e vinte e dois**.

TÍTULO SUGERIDO PELA BANCA

Documento assinado eletronicamente por **Rodrigo Pinto Lemos**, **Professor do Magistério Superior**, em 08/07/2022, às 15:37, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Flavio Henrique Teles Vieira**, **Professor do Magistério Superior**, em 08/07/2022, às 15:38, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Alisson Assis Cardoso**, **Professor do Magistério Superior**, em 08/07/2022, às 15:38, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Processo: 23070.032506/2022-73 Documento: 3025062

Documento assinado eletronicamente por **Anderson Da Silva Soares**, **Professor do Magistério Superior**, em 08/07/2022, às 15:42, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **DANIEL PÔRTO QUEIROZ CARNEIRO**, **Discente**, em 08/07/2022, às 16:09, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no site https://sei.ufg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **3025062** e o código CRC **84C288AF**.

Referência: Processo nº 23070.032506/2022-73 SEI nº 3025062

Agradecimentos

Agradeço primeiramente a Deus, pelo milagre que é a vida. Aos meus pais, irmão e família pelo constante apoio. Aos meus professores, por sua preocupação com o aprendizado, em especial ao meu orientador e co-orientador por compartilharem seus conhecimentos e pelos constantes exemplos de postura e didática. Aos colegas que ajudam a humanizar os nossos erros. A CAPES pelo apoio financeiro.

"Aprender é, de longe, a maior recompensa."

William Hazlitt,

1778-1830.

Resumo

Nesta dissertação, apresenta-se algoritmos de alocação de recursos baseado em aprendizado por reforço para um sistema de comunicação multiportadora considerando múltiplos usuários e efeitos de multipercurso e perda média do percurso em uma transmissão assumindo ondas milimétricas. Para tal, propõe-se que o sistema de comunicação possa ser descrito por um modelo Markoviano representado pelos estados da fila nos *buffers* e estados dos canais. Para os algoritmo de alocação de recursos deste trabalho, são introduzidas funções de recompensa a serem utilizadas no algoritmo de aprendizado por reforço *Q-learning*. Os resultados obtidos nas simulações mostram que a aplicação dos algoritmos propostos de escalonamento de recursos provê, de forma geral, melhoria nos parâmetros de desempenho do sistema de comunicação considerado, como por exemplo, aumento de vazão e diminuição de perda de pacotes. Comparações com outros algoritmos apresentados na literatura são realizadas, mostrando também que o uso da função de recompensa proposta e modelo Markoviano considerado torna o escalonamento de usuários e o compartilhamento de recursos mais eficientes. Ainda, é apresentada uma solução para alocação de recursos e potência utilizando uma *Deep Q-Network*. A modelagem de estados propostos para rede DQN soluciona algumas limitações encontradas com a representação matricial dos estados e amplia os limites para o tamanho do *buffer*.

Palavras-chave:Alocação de Recursos em rede 5G. Aprendizado por Reforço. Processo de decisão de Markov. DQN adaptativa.

Abstract

In this dissertation, we present reinforcement learning-based resource allocation algorithms for a multicarrier communication system considering multiple users and the effects of multipath and average path loss in a transmission assuming millimeter waves. To this end, it is proposed that the communication system can be described by a Markovian model represented by queue states in *buffers* and channel states. For the resource allocation algorithms of this work, we introduce reward functions to be used in the reinforcement learning algorithm *Q-learning*. The results obtained in the simulations show that the application of the proposed algorithms for resource scheduling provides, in general, an improvement in the performance parameters of the considered communication system, such as, for example, increased throughput and decreased packet loss. Comparisons with other algorithms presented in the literature are carried out, also showing that the use of the proposed reward function and considered Markovian model makes the scheduling of users and the sharing of resources more efficient. Furthermore, a solution for resource and power allocation using a *Deep Q-Network* is presented. The modeling of states proposed for the DQN network covers some limitations encountered with the matrix representation of states and extends the limits for the size of the *buffer*.

Keywords: 5G Resource allocation. Reinforcement Learning. Markov Decision Process. Adaptative DQN.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

LTE	*Long Term Evolution*
IoT	*Internet of Things*
FDM	*Frequency Multiplexing*
CP-OFDM	*Ciclic Prefix Orthogonal Frequency Multiplexing*
BER	*Bit Error Rate*
TDL	*Tapped Delay Line*
QoS	*Quality of Service*
EE	*Eficiência Energética*
KPI	*Key Performance Indicator*
5G	*Quinta geração de rede de internet móvel*
AWGN	*Additive White Gaussian Noise*
SNR	*Signal to Noise Ratio*
WN0	*Densidade de potência do ruído*
PL	*Path Loss*
BPSK	*Binary phase-shift keying*
QAM	*Quadrature amplitude modulation*
pBER	*Probabilidade de erro de bit*
TDM	*Time-division multiplexing*
f_D	*Máximo efeito Doppler*
DQN	*Deep Q-Network*
MTU	*Maximum transmission unit*
ReLU	*Rectified Linear Unit*

Trabalhos Submetidos e Publicados

Trabalhos aprovados e/ou publicados:

- CARNEIRO D. P. Q., CARDOSO A. A., Gabriel C., VIEIRA , F. H. T. Aprendizado por Reforço para Escalonamento de Recursos em Sistema sem Fio Multiportadora com Ondas Milimétricas Utilizando Modelo Markoviano In: IX Escola Regional de Informática ERI-GO, Online, 2021.
- CARNEIRO D. P. Q., CARDOSO A. A., VIEIRA , F. H. T. . Escalonamento de Recursos em Sistema Multiportadora com Ondas Milimétricas Utilizando Aprendizado por Reforço Markoviano In: SBPO Simpósio Brasileiro de Pesquisa Operacional, Online, 2021.
- CARNEIRO D. P. Q., CARDOSO A. A., VIEIRA , F. H. T. . Aprendizado por Reforço Baseado em Modelo Markoviano para Alocacação de Recursos em Sistema Multiportadora com Ondas Milimétricas In: SBAI Simpósio Brasileiro de Inteligência Artificial, Online, 2021.

Trabalhos submetidos:

- CARNEIRO D. P. Q., CARDOSO A. A., VIEIRA , F. H. T. . Adaptive Resource Allocation in 5G CP-OFDM Systems Using Markovian Model Based Reinforcement Learning Algorithm In: Neural Computing and Applications, Springer, 2022.

Sumário

Introdução

Um dos desafios em sistemas de comunicação é compartilhar recursos de forma eficiente sendo estes limitados. Além do número de frequências disponíveis de transmissão ser finito, a potência utilizada é um fator limitante especialmente em dispositivos com baterias (S; MOHAMAD, 2016). Com a demanda cada vez mais alta de qualidade, alta taxa de transmissão e com o crescimento de usuários e dispositivos, uma estratégia adequada de alocação de recursos se mostra imperativa (SANGAIAH et al., 2020).

Os sistemas de comunicação são complexos e podem priorizar um indicador de desempenho específico em detrimento a outros, por exemplo, aumentar a vazão sem se preocupar com gasto de potência, atender equipamentos de usuários mais próximos à estação rádio base e postergar o atendimento de equipamentos de usuários mais distantes. Zhu et al. (2018) propõem um algoritmo de aprendizado por reforço aplicado a um sistema IoT (*Internet of Things*) multiusuários. O agente único de controle atende um usuário de cada vez, multiplexando o atendimento no tempo. Apesar de considerar a velocidade relativa entre o transmissor e o receptor, não são abordadas as distâncias entre eles ou suas velocidades absolutas.

No artigo de Ford et al. (2017), aborda-se o desempenho de um sistema LTE OFDM (*Long Term Evolution Orthogonal Frequency Multiplexing*) onde são consideradas mais informações para os ganhos dos canais como situação de inoperância (*outage*). Além disso, são comparadas estatísticas de dados reais de um sistema de comunicação em ambiente urbano com resultados de um Modelo Markoviano finito de canal aplicados ao sistema LTE-OFDM considerado. Em sistemas OFDM, o atendimento dos usuários é feito simultaneamente em diferentes frequências. Com isso, o desempenho do sistema pode apresentar maiores valores de vazão, menores valores de tempo de espera dos pacotes na fila no *buffer* e menores valores de perda de pacotes do que sistemas que não consideram múltiplas subportadoras (PATTETI; KUMAR; KALITKAR, 2016).

No presente trabalho, considera-se um sistema CP-OFDM (*Cyclic Prefix – Orthogonal Frequency Division Multiplexing*) de comunicação para multiusuários com um agente inteligente de controle baseado em aprendizado por reforço. Para este sistema de comunicação, valores de potência são utilizados nas transmissões aos usuários de tal forma a atender um valor mínimo de BER (*Bit Error Rate* - Taxa de erro de bit) para os fluxos de tráfego dos usuários de acordo com as modulações consideradas. Assume-se um modelo TDL (*Tapped Delay Line*) para a modelagem deste canal aleatório (3GPP, 2018). Nesse trabalho, utiliza-se o algoritmo *Q-learning* com iteração de política e um modelo Markoviano para descrever os estados do sistema de comunicação. O modelo de canal utilizado segue a

configuração apresentada por Hong Shen Wang e Moayeri (1995) e 3GPP (2018). Os diferentes valores dos ganhos dos canais levam em conta perdas por múltiplo percurso e falta de linha de visada. Quanto à avaliação de QoS (*Quality of Service*), assim como em Zhu et al. (2018) avalia-se a vazão média de pacotes, a perda média de pacotes, ocupação média de pacotes na fila do *buffer* e eficiência energética média do sistema de comunicação. Diferente de Zhu et al. (2018) que considera um sistema TDM, neste trabalho o algoritmo de alocação de recursos é aplicado a um sistema multiportadora considerando-se uma modelagem de canal (TDL-*Tapped Delay Line*) mais apropriada para a tecnologia 5G (3GPP, 2018).

Como principais objetivos deste trabalho, pode-se citar:

1. Desenvolver algoritmos de alocação de recursos baseados em aprendizado por reforço adaptados para melhorar parâmetros de qualidade de serviço em um sistema de comunicação CP-OFDM multiusuários considerando diferentes cenários;

2. Desenvolver um simulador para o o sistema de comunicação proposto de modo a avaliar a performance das estrategias a serem consideradas.

3. Utilização de *Q-Learning off-policy* baseado em modelo Markoviano do sistema considerando simultaneamente os estados do *buffer* e do canal de comunicação.

4. Propor funções de utilidade robustas para o algoritmo de aprendizado por reforço levando em consideração os principais indicadores de qualidade de serviço;

5. Validar o modelo markoviano considerado simulando o sistema com dados reais de tráfego.

6. Apresentar estratégia de escalonamento adaptativo utilizando modelo de distribuição de probabilidade para dados reais aplicando aprendizado por reforço e cadeias de Markov.

7. Apresentar estratégia de escalonamento adaptativo utilizando dados reais de chegada e rede neural profunda de aprendizado por reforço (DQN - *Deep Q-Network*).

8. Comparar resultados obtidos com alocação de potência tendo como objetivo que um valor máximo de BER para o sistema não seja ultrapassado com resultados obtidos ao se treinar um agente para alocação de potência de forma mais livre das restrições de BER, ambos os casos considerando dados reais de tráfego

1 Sistema de Comunicação Multiportadora e Multiusuário

Neste Capítulo, apresenta-se o referencial teórico utilizado para a simulação de um sistema de comunicação multiportadora e multiusuário considerando uma célula conforme mostrado na Figura 1.1. Foca-se na transmissão *downlink* do modelo do sistema OFDM, ou seja, na comunicação que ocorre na direção da estação base para o equipamento do usuário. Além disso, apresenta-se neste Capítulo, como o sistema de comunicação é modelado utilizando estados finitos e Cadeias de Markov.

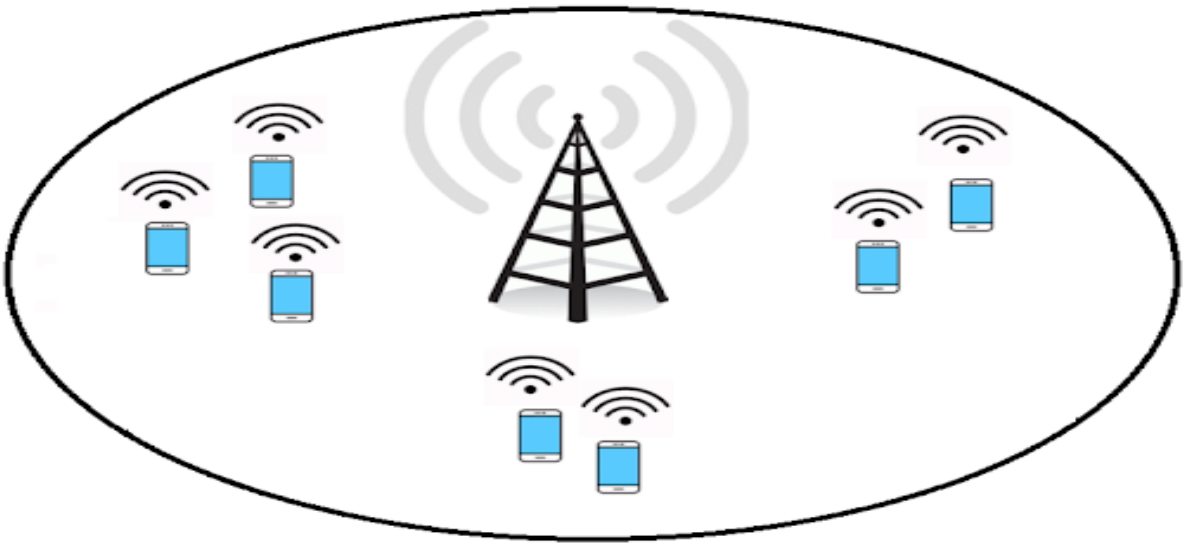

Figura 1.1 – Sistema de comunicação com uma estação base.

1.1 Modelo do Sistema OFDM

O modelo de sistema OFDM considerado neste trabalho consiste em uma estação rádio base (agente) que deve tomar decisões a cada TTI (intervalo de tempo, utilizado igual a 0.5 ms para o slot) sobre quando e como atender K equipamentos de usuários. Para isso, o sistema de comunicação utiliza M canais e J modos de transmissão através de um sistema de subportadoras (OFDM) apresentando um *buffer* de tamanho L para cada usuário.

O agente é treinado utilizando aprendizado por reforço (*Q-learning*) com base nos estados possíveis do sistema e uma função de recompensa. Para descrever os estados do sistema de comunicação, adota-se um modelo Markoviano, isto é, a mudança de estado

exige o conhecimento do estado atual, da ação escolhida e do ambiente (característica estocástica). Assim, são descritos a seguir como são estimados os estados do *buffer*, do canal e como a alocação de potência para os usuários é efetuada.

1.1.1 Estados do *Buffer*

O *buffer* possui tamanho L, ou seja, tem capacidade de armazenar no máximo L pacotes para cada usuário. A contribuição do estado de *buffer* para o estado do sistema leva em conta as $L+1$ quantidade possíveis de pacotes, com inclusão do zero, para cada um dos K usuários. Dessa forma, para K usuários cujos dispositivos apresentam *buffer* de tamanho L, tem-se $(L+1)^K$ estados de *buffer* possíveis no sistema (Zhu et al., 2018). Há mudanças no estado do *buffer* com a chegada ou saída de dados, o que pode se modificar a cada TTI.

1.1.2 Estados do Canal

Após deixar o transmissor, o sinal se propaga no ambiente e irá chegar ao receptor com características diferentes de quando partiu. A seguir são tratados dois efeitos na amplitude e potência do sinal considerando que não há um caminho direto (visada direta) entre o transmissor e receptor. Esses efeitos levam em consideração os múltiplos percursos que o sinal pode percorrer antes de chegar ao destino (desvanecimento), e perda média de percurso na intensidade do sinal causada pela distância.

1.1.2.1 Múltiplos Percursos

O sinal que chega ao receptor passa por reflexões diversas (podendo inclusive chegar sem nenhuma reflexão). Assim, existem múltiplos percursos do sinal. Considerando um ambiente de ruído AWGN (*Additive White Gaussian Noise*), a amplitude do sinal (v) pode ser modelada por uma variável aleatória caracterizada por uma distribuição de Rayleigh Matz e Hlawatsch (2011):

$$p_v(v) = \frac{v}{\sigma^2} e^{\frac{-v^2}{2\sigma^2}} \tag{1.1}$$

onde $p_v(v)$ é a densidade de probabilidade da amplitude do sinal v e σ é conhecido como fator de escala. É a moda da distribuição de Rayleigh e tem relação com o desvio padrão das distribuições normais que dão origem a distribuição de Rayleigh.

A distribuição de Rayleigh ocorre quando analisa-se o comportamento de duas variáveis independentes com distribuição gaussiana em conjunto, isso é, quando variáveis aleatórias com distribuições independentes são as componentes de um vetor ou de um número complexo de interesse.

Aqui nos interessa a amplitude v do sinal. O sinal modulado OFDM é complexo com componentes consideradas independentes e com distribuições gaussianas de média

zero e desvio padrão σ e, portanto, sujeitas a ruídos de densidade de potência igual a σ^2 cada uma.

A distribuição de probabilidade da potência $v^2 = P \cdot |h|^2$ do sinal transmitido é, portanto, exponencial Proakis e Salehi (2008).

Assumindo que a potência na estação base P e a densidade de potência do ruído WN_0 sejam constantes na duração de 1 TTI (0.5 ms), a SNR (*Signal to Noise Ratio*) é dada por:

$$SNR = |h|^2 \frac{P}{WN_0} \tag{1.2}$$

onde h representa o coeficiente de ganho do canal e WN_0 a densidade de potência do ruído.

Seja ρ_m o ganho médio de potência do canal, então a probabilidade do ganho médio do canal ser ρ é dada por:

$$p_\rho(\rho) = \frac{1}{\rho_m} e^{\frac{-\rho}{\rho_m}} \tag{1.3}$$

Para obter valores realistas para o ganho do canal, neste trabalho, foi considerado o modelo TDL-B (*Tapped Delay Line*) para caracterizar um canal com características variantes no tempo, conforme descrito em 3GPP (2018). O modelo TDL para múltiplos percursos considera que a resposta ao impulso do canal pode ser representada por (JAIN, 2007):

$$h(t, \tau) = \sum_{i=1}^{N} c_i(t)\delta(\tau - \tau_i) \tag{1.4}$$

onde $h(t, \tau)$ é a resposta ao impulso, t o tempo , τ o atraso , $c_i(t)$ a amplitude variante no tempo e δ a função impulso.

O valor de N adotado no modelo TDL-B é de 23 atrasos (τ_i's). A base utilizada para cálculo dos atrasos é de 1148 ns que corresponde ao atraso longo para frequência de 6 GHz. Portanto, tem-se uma combinação de várias cópias defasadas do sinal original. Na Tabela 1.1 podem ser visualizados os parâmetros obtidos de 3GPP (2018) e utilizados para modelar o efeito de múltiplos percursos.

Como descrito anteriormente o estado do *buffer* é representado por um conjunto discreto de valores. Para o estado do canal tem-se uma distribuição contínua, logo, se faz necessário a divisão em intervalos/classes os valores de ganho de canal para análise inteira de estados.

A quantidade de intervalos para o ganho do canal utilizada na dissertação recebe o nome de C_h. Para definir os intervalos definem-se os limiares. Com C_h estados possíveis tem-se, a rigor, $C_h - 1$ limiares. Nas simulações, adotou-se como estado 1 do canal como sendo referente ao pior estado em termos de relação sinal-ruído ou ganho para o par

Tabela 1.1 – Modelo TDL-B 3GPP (2018).

Tap #	Atraso Normalizado	Potência em [dB]	Distribuição do desvanecimento
1	0,0000	0	Rayleigh
2	0,1072	-2,2	Rayleigh
3	0,2155	-4	Rayleigh
4	0,2095	-3,2	Rayleigh
5	0,2870	-9,8	Rayleigh
6	0,2986	-1,2	Rayleigh
7	0,3752	-3,4	Rayleigh
8	0,5055	-5,2	Rayleigh
9	0,3681	-7,6	Rayleigh
10	0,3697	-3	Rayleigh
11	0,5700	-8,9	Rayleigh
12	0,5283	-9	Rayleigh
13	1,1021	-4,8	Rayleigh
14	1,2756	-5,7	Rayleigh
15	1,5474	-7,5	Rayleigh
16	1,7842	-1,9	Rayleigh
17	2,0169	-7,6	Rayleigh
18	2,8294	-12,2	Rayleigh
19	3,0219	-9,8	Rayleigh
20	3,6187	-11,4	Rayleigh
21	4,1067	-14,9	Rayleigh
22	4,2790	-9,2	Rayleigh
23	4,7834	-11,3	Rayleigh

usuário-subportadora, sendo que o agente considera que neste estado (quando há mais de 1 estado de canal) não há transmissão.

Na Figura 1.2 visualiza-se a distribuição do ganho ao considerar a amostragem de um único canal (uma única vez) a cada *TTI*. Os valores foram normalizados pelo valor médio do ganho ao se amostrar o modelo TDL ((3GPP, 2018)). Foram realizadas 90000 amostragens da função exponencial para cálculo da média e distribuição. Também é possível observar o ganho médio do canal e os limiares ($C_h - 1 = 3$) para transição de estado do canal.

Obtém-se os valores dos limiares do ganho de potência do canal utilizando os resultados da amostragem consecutiva de C_h valores da distribuição exponencial com média $|h_m|^2$. Várias repetições desse processo de amostragem são feitos seguido da ordenação dos ganhos, ou seja, 22250 grupos de 4 ganhos ordenados para $C_h = 4$. Por fim, extrai-se a média dos 22250 grupos de vetores. Assim, tem-se C_h valores médios ordenados que representam o ganho médio do canal para cada estado. Os limiares são obtidos ao extrair a média entre os ganhos médios vizinhos desses valores. Esse método realiza a amostragem de C_h ganhos de 1 canal a cada iteração.

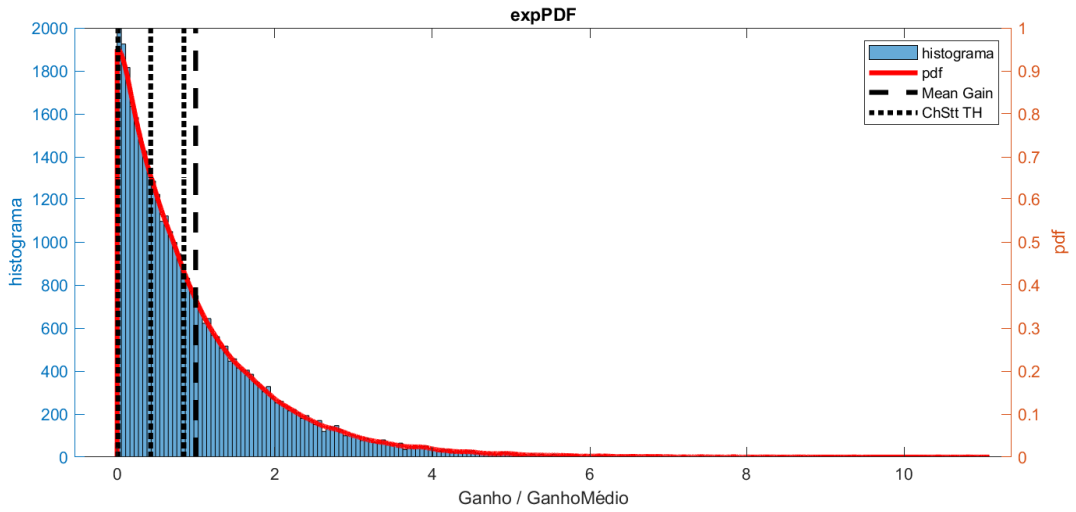

Figura 1.2 – Distribuição normalizada do ganho de 1 canal e limiares para 4 estados possíveis de canal.

Na prática, ao se utilizar mais canais (faixas de sub portadoras) é possível aproveitar a variedade de informação referente aos ganhos de cada usuário em cada canal. Pode-se obter valores maiores para a distribuição ao maximizar a soma dos ganhos de cada usuário em seu canal. Isso acontece pelo fato de termos variedade estatística ao obter $K \cdot M$ amostras da distribuição da Figura 1.2 cujo comportamento é exponencial. Uma distribuição mais realista onde leva-se em consideração a solução do problema de atribuição para o caso de $K = 2$ usuários e $M = 2$ é mostrada na Figura 1.3. Como tem-se a informação de ganho para cada par usuário-canal pode-se escolher a melhor forma de atribuir os usuários nos canais. O método utilizado para decidir os pares canal-usuário é o chamado algoritmo húngaro (KUHN, 1955) que resolve o problema de atribuição (*Assignment problem*). Destaca-se que o agente ainda é o responsável por atribuir ao usuário um canal, mas utiliza valores de ganho estatisticamente mais robustos para essa escolha. Para o caso em que se tem mais usuários que canais $K > M$, são realizadas M^2 amostragens. Os M ganhos com a maior soma são passados para o agente como os ganho de canal. Sem essa consideração seria necessário considerar as permutações das colunas da matriz de ganhos $K \cdot M$ na modelagem dos estados de canal.

De forma simplificada, deseja-se encontrar a matriz usuário-ganho do canal cuja dimensão é $K \cdot M$ com maior traço, o seja, com a maior soma dos elementos da diagonal principal. A matriz é modificada pela permutação das suas colunas. Para o caso de $K = M = 3$ a configuração inicial representa a disposição do usuário 1 no canal 1, usuário 2 no canal 2 e usuário 3 no canal 3. Com essa solução, até 3! operações são realizadas. Apesar de não ser a ótima, é uma solução rápida para $M \leq 3$. Veremos adiante que existe uma limitação quanto ao tamanho das matrizes de estado do sistema especialmente para

valores de K e M maiores que 6.

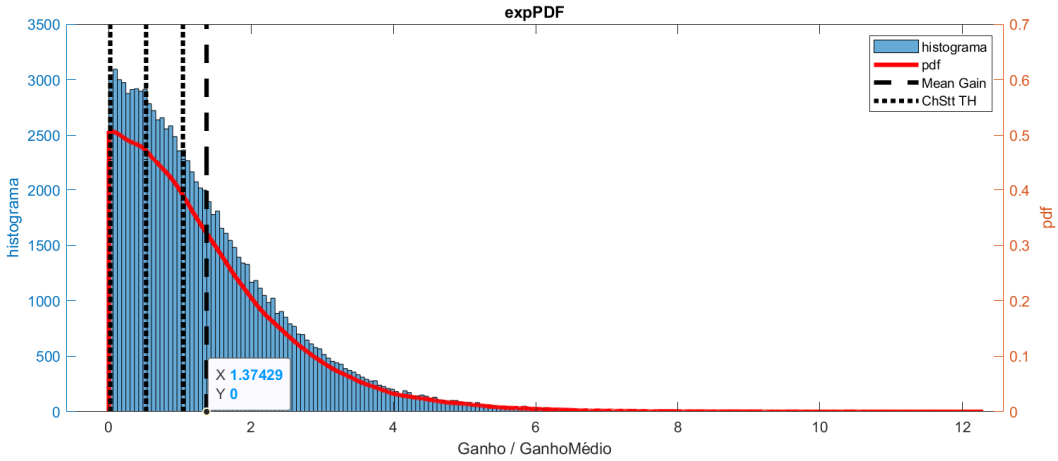

Figura 1.3 – Distribuição normalizada do ganho de 2 usuários e 2 canais utilizando algoritmo húngaro e limiares para 4 estados possíveis de canal.

O ganho médio do canal é obtido através da média de 500 iterações dos valores gerados pelas distribuições e pelo método húngaro mencionados anteriormente. Este ganho médio do canal é posteriormente utilizado no cálculo das matrizes de recompensa imediata $\mathbf{R}$ do algoritmo de aprendizado por reforço. Mais especificamente, em cada iteração obtém-se uma matriz $K \cdot M \cdot C_h$ de 3 dimensões de ganhos. Ordena-se a matriz ao longo da dimensão 3, que representa as C_h faixas de ganho. Aplica-se o método húngaro à média ao longo da dimensão 3, ou seja, em uma matriz $K \cdot M$ obtendo assim os pares usuário-canal. Com os pares obtidos, monta-se a matriz de ganhos $C_h \cdot M$ corrente e atualiza-se de forma recorrente a matriz de ganhos médio utilizando o contador de iterações. Como a entrada do algoritmo húngaro é uma matriz e a saída é um vetor, ele reduz a dimensão da matriz de K usuários e M canais para uma configuração média de usuários pareados com os canais em um vetor de tamanho M.

Foi desconsiderado o efeito da variação na frequência das sub-portadoras para a frequência da portadora central (6GHz), logo para se obter os limiares de ganho a matriz de ganhos médios $C_h \cdot M$ tem sua média extraída ao longo da dimensão 2 resultando em um vetor de dimensão C_h que representa o ganho médio de um canal nos diferentes estados de canal. Esse valor é replicado para a quantidade de canais disponíveis se tornando novamente uma matriz $C_h \cdot M$ mas agora com colunas idênticas.

Com os valores de ganho médio por estado de canal, os limiares são obtidos tomando a média dos ganhos vizinhos 2 a 2.

O processo de amostragem de $K \cdot M$ ganhos e obtenção da melhor configuração é realizado durante a simulação a cada TTI.

1.1.2.2 Perda Média do Percurso

Além do efeito dos múltiplos percursos no ganho do canal obtido do modelo TDL, a distância entre transmissor e receptor também é um fator limitante no seu valor. O modelo de perda média de percurso utilizado segue a Equação de Friis, com parâmetros obtidos de 3GPP (2018):

$$PL = 32.4 + 20log(fc) + 30log(D) \tag{1.5}$$

onde fc (GHz) é a frequência da portadora e D (m) a distância entre transmissor e o receptor. O ganho final do canal considerando os dois efeitos (perda por múltiplos percursos e perda média do percurso) pode ser representado por,

$$|h|^2 = \frac{\rho}{10^{PL/10}} \tag{1.6}$$

onde ρ é o ganho do canal pelo efeito de múltiplo percurso, sendo uma variável aleatória.

Para valores de ganho abaixo de um certo valor, o atendimento para o usuário não é realizado (estado ocioso).

1.1.3 Potência

A potência $P(c, j)$ do sinal ao deixar a estação base e utilizar o canal c e modo j é calculada explicitando o termo $P(c, j)$ da equação de BER (taxa de erro de bit) máxima conforme mostram as equações (1.8) e (1.10) que dependem do modo j e da potência do ruído WN_0. Neste trabalho, considera-se um valor de potência do usuário de tal modo a garantir uma BER igual ou menor do que 0.001. Para atendimento da demanda de tráfego dos usuários, utiliza-se 4 modos diferentes de transmissão 4QAM, 16QAM , 64QAM e 256QAM que são capazes de transmitir $j = 2$, 4, 6 e 8 bits por símbolo proporcionando diferentes taxas de atendimento de pacotes. O valor de potência a ser alocada é obtido explicitando o termo P nas equações de probabilidade de erro de bit ($pBER$) referentes às modulações consideradas, dadas a seguir (Zhu et al., 2018):

- $BPSK, j = 1$

$$p_{BER} \leq 0.5\sqrt{\text{erfc}(\rho(c)\frac{P_{BPSK}(c)}{WN_0})} \tag{1.7}$$

onde erfc é a função de erro complementar.

$$P_{BPSK}(c) = \frac{\text{inverfc}(2 \cdot p_{BER})^2}{\rho(c)/WN_0} \tag{1.8}$$

onde inverfc é a função inversa da função de erro complementar.

- $2^j - QAM, j > 1$

$$p_{BER} \leq 0.2e^{\frac{-1.6\rho(c)P_{QAM}(c)}{WN_0(2^j-1)}} \tag{1.9}$$

$$P_{QAM}(c) = \frac{(2^j - 1)ln(5 \cdot p_{BER})}{-1.6\rho(c)/WN_0} \tag{1.10}$$

Como obtém-se a potência que deve ser alocada pela equação 1.10 considerando um valor máximo de BER, a relação sinal-ruído SNR se mantém constante($\rho(c) \cdot P(c)/WNo$ pelo menos para um modo de transmissão fixo) . Além disso, como considera-se a potência do ruído constante, da Equação 1.2 o produto $|h|^2 \cdot P$ também se mantêm constante. Utiliza-se o ganho do canal como métrica para separar os estados de canal e não a potência do sinal do usuário devido a influência do modo de transmissão no valor da potência. Com o ganho do canal tem-se uma boa estimativa da qualidade do canal, independente do modo de transmissão mesmo com uma SNR constante.

1.1.4 Eficiência Energética

Utiliza-se ao longo do texto o conceito de eficiência energética. Considera-se a eficiência energética descrita por 3GPP (2020) mais especificamente na seção 4.2.1.3 *"Network Energy Efficiency quantitative KPI"*. Adota-se peso 1 para a contribuição de cada usuário:

$$EE(s,a) = \frac{1}{K}\sum_{k=1}^{K}\frac{f_k(s,a)}{P_k(s,a)} \tag{1.11}$$

onde f_k é o fluxo de pacotes, P_k a potência alocada e K é o número de usuários. Mais adiante é detalhada a forma como se obtém $f_k(s,a) = min(l_k + b_k, V \cdot j_k)$.

1.2 Tecnologia LTE-OFDM

A tecnologia LTE-OFDM utiliza um intervalo entre sub-portadoras de 15 kHz e um total de 12 subportadoras por bloco de recurso. Como o sinal é composto por componentes de frequências ortogonais, pode haver sobreposições entre frequências que compõem as bandas das subportadoras. Isso permite maior eficiência espectral, pois a largura de faixa é menor quando comparado ao FDM (*frequency division multiplexing*). Em um sistema FDM há maior distanciamento no domínio da frequência entre as subportadoras (maior largura de faixa). Não é permitida a sobreposição entre elas e, por esse motivo, são adicionadas bandas de guarda entre as frequências de transmissão. Podemos também entender que para uma mesma largura de faixa disponível, a comunicação CP-OFDM acomoda mais sub-portadoras e, portanto, leva vantagem em quantidade de símbolos.

A comunicação CP-OFDM possibilita altas taxas de transmissão com símbolos de duração mais longa no domínio do tempo (menor taxa de símbolos por subportadora mas, ao mesmo tempo, mais sub-portadoras). Isso dificulta a interferência entre símbolos provocada pelo efeito de múltiplos percursos do sinal já que o símbolo é longo. Assim, tem-se transmissões em paralelo de baixa velocidade imunes ao efeito do canal (*fast fading* - desvanecimento rápido) mas que se somam no receptor e garantem altas taxas de transmissão e menor interferência inter simbólica.

1.2.1 Blocos de recurso e modelo fluido do sistema de comunicação

A Figura 1.4 apresenta a estrutura de um *frame* para o sistema de comunicação LTE CP OFDM onde pode-se observar o tamanho do bloco de recurso para um *slot* do *subframe*. Como visualizado, para o modo normal há 7 símbolos por bloco de recurso, cada bloco de 12 subportadoras terá, portanto, $12 \times 7 = 84$ elementos de recurso. Cada elemento de recurso representa uma frequência e símbolo (que varia com o modo de transmissão). As 12 subportadoras espaçadas de 15kHz ocupam um total de $15 \times 12 =$ 180kHz de banda. Com uma largura de banda de 20 MHz (LTE release 8) tem-se um total de 100 blocos de recurso ou 8400 elementos de recurso a cada 0.5 ms, ou 8400 símbolos a cada 0.5 ms . Nos resultados das seções 3.1 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9 , 3.10, 3.11 , 3.12, e 3.13 adota-se TTI de 0.5 ms.

A tecnologia LTE utiliza os modos QPSK/4QAM, 16QAM, 64QAM e 256QAM que correspondem a transmissão de 2, 4, 6 e 8 bits por símbolo (elemento de recurso). Logo, ao utilizar toda banda disponível(20 MHz) ,estão disponíveis 8400 elementos de recurso (ou símbolos). Pode-se transmitir, portanto, 16800, 33600, 50400 e 67200 bits por TTI a depender do modo, a serem distribuídos para os usuários. Como cada TTI tem 0.5 ms isso corresponde a 33.6, 67.2, 100.8, 134.4, Mbps.

Em termos de pacotes, para um tamanho médio de pacote igual a 105 bytes (menor que MTU -*Maximum transmission unit* = 1500 bytes) e largura de banda de 20 MHz, tem-se fluxo máximo de: $16800/(8 \cdot 105) = 20$, $33600/(8 \cdot 105) = 40$, $50400/(8 \cdot 105) = 60$ e $67200/(8 \cdot 105) = 80$ pacotes por TTI. Ao dividir esses valores por 5 (quantidade de usuários das simulações) chega-se aos valores: 4, 8, 12 e 16 pacotes referentes aos modos 2^2QAM,2^4QAM, 2^6QAM, e 2^8QAM . Assim, o modo "j" terá capacidade de transmitir até "$2j$" pacotes em um modelo fluido com 5 usuários. O número 2 é chamado de taxa de codificação (*coding rate*). Em Zhu et al. (2018) o resultado $2j$ para quantidade de pacotes transmitidos é utilizado sem maior detalhamento com a diferença que usa um intervalo de decisão 4 vezes maior e igual a 2 ms, 3 usuários, os modos BPSK, 4QAM, 8QAM e 16QAM, considera taxa média de chegada de pacotes no máximo igual a 0,9 e considera atendimento TDM (*Time-Division Multiplexing*). O tamanho do pacote irá depender da natureza da informação transmitida e pode ter bastante influência na eficiência energética,

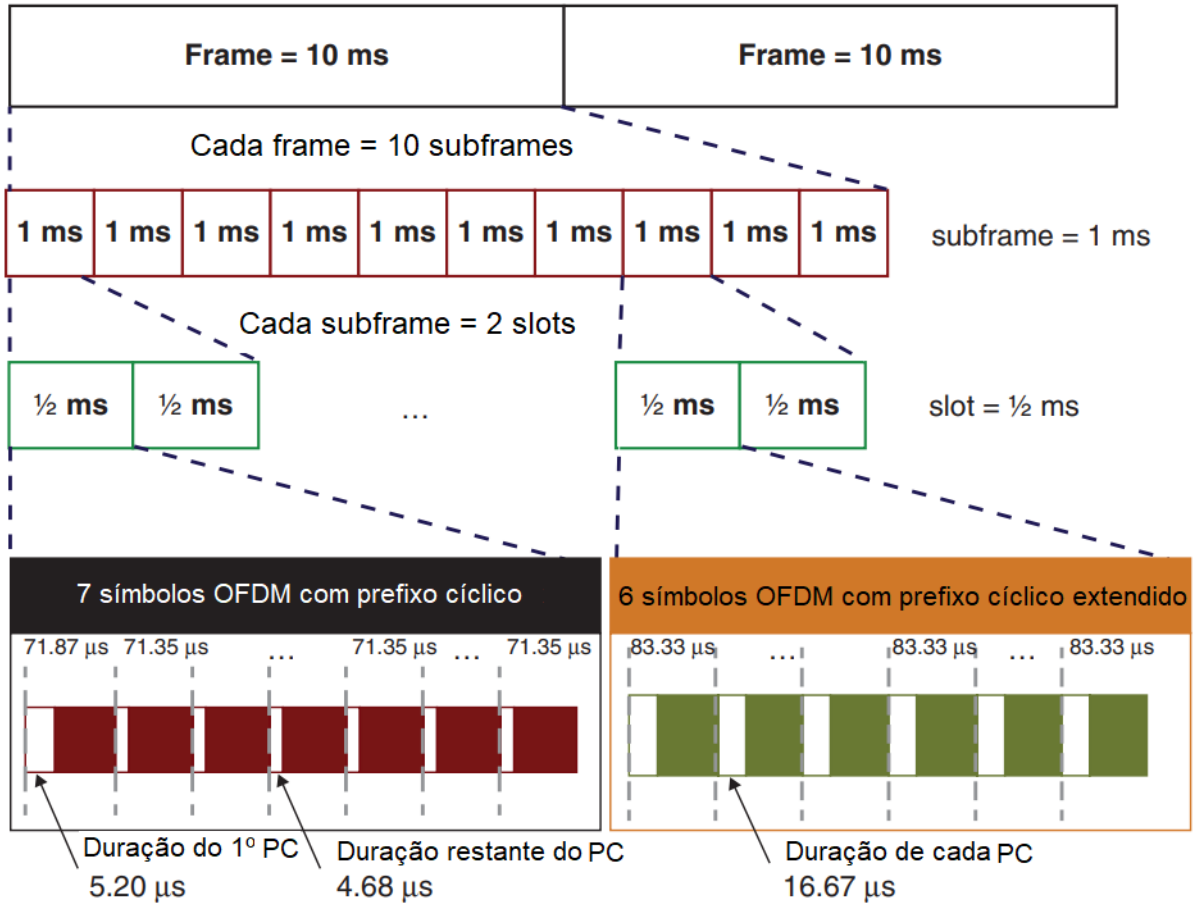

Figura 1.4 – Estrutura de um frame para o sistema de comunicação LTE-OFDM. Fonte: Zarrinkoub (2014)

especialmente para um sistema real (PUJOLLE, 2006).

A modelagem do sistema de comunicação CP-OFDM realizada nesta dissertação fixa o tamanho dos pacotes em 105 bytes (exceto quando especificado), considera uma quantidade reduzida de eventos e, portanto, a tomada de decisão restrita ao *subframe* (1 ms). O termo TTI é utilizado para definir o menor intervalo de tempo em que novos pacotes são recebidos e é possível avaliar a taxa de erro de bit BER. Considerando o padrão LTE Release 15, o *subframe* é igual a 1 ms, ou seja, e o tamanho de um *slot* de *subframe* igual a 0.5 ms conforme apresentado na Figura 1.4.

Essa consideração supõe que a comunicação é feita de forma contínua, isto é, o fluxo de pacotes ocorre gradualmente e pode ser observado a nível de bit. Essa consideração também nos permite utilizar as equações de BER para o problema. O modelo fluido reduz a quantidade de eventos simulados e agiliza a simulação (CARVALHO et al.,).

2 Aprendizado por Reforço Markoviano para Escalonamento de Recursos

No Capítulo anterior apresentou-se conceitos referentes a sistemas de comunicação multiportadora e multiusuário. Abordou-se sobre as características físicas do canal representadas através do ganho que é uma variável aleatória. Apresentou-se também a tecnologia utilizada para o sistema neste trabalho: LTE-OFDM, e ainda, são introduzidos os conceitos de estado de *buffer* e canal. Estas informações e conceitos serão úteis para a compreensão da modelagem Markoviana baseada em estados que será apresentada adiante.

Neste Capítulo é apresentado o conceito de cadeia de Markov e como a mesma pode ser utilizada em algoritmos de aprendizado por reforço. São apresentadas as definições utilizadas no Modelo Markoviano para o estado de *buffer* e canal de forma a constituir um conjunto fechado de possíveis estados do sistema e ações finitas para o agente. Além disso, propõe-se neste capítulo, 4 equações para a função de recompensa do algoritmo de aprendizagem por reforço. Por fim, apresenta-se motivações para se avaliar também o desempenho de algoritmos baseado em redes neurais profundas e ao se considerar agrupamento dos usuários em *clusters*.

O sistema CP-OFDM descrito no Capítulo anterior pode ser modelado como uma cadeia de Markov se for assumido que o estado seguinte do sistema dependa somente do estado atual e da ação escolhida pelo agente (FORD et al., 2017),(Hong Shen Wang; Moayeri, 1995) e (Zhu et al., 2018). Isso quer dizer que o agente deve tomar sua decisão com base no estado atual.

O aprendizado por reforço é uma técnica que consiste em um agente tomando decisões em diversos estados de um ambiente e recebendo recompensas ou punições pelas suas ações (SUTTON; BARTO, 2018). Utilizando uma série de exemplos baseados em tentativa e erro, o agente busca aprender a melhor política, ou seja, a melhor sequência de ações a serem tomadas em determinado ambiente de forma a obter valores de recompensas maiores. Apesar de se basear apenas no estado atual para tomar a decisão, a "memória"de reforços positivos e negativos permite ao agente ter uma projeção de longo prazo da melhor ação. Em um sistema com modelo probabilístico de transição de estados, essa memória é representada pela matriz $\mathbf{P}$ que modela a interação com o ambiente para cada possível ação do agente.

Nesta dissertação, um algoritmo de aprendizado por reforço baseado no *Q-learning* considerando um modelo Markoviano para o ambiente (sistema de comunicação CP-OFDM multiusuário) é utilizado, no qual é necessário obter as probabilidades de transição

de estados e as recompensas de cada ação possível. É possível obter essa matriz de transição de estados com a extensiva observação do sistema. Para tal, propõe-se considerar uma distribuição conhecida (de Poisson) para algumas variáveis aleatórias do modelo do ambiente, mais precisamente para a distribuição de probabilidade da chegada de pacotes.

Devido às diferentes formas de se efetuar acesso ao meio pelos usuários em um sistema de comunicação, e a título de comparar seus desempenhos, considera-se 2 tipos de configuração ou modo de operação para o conjunto de ações do sistema. A primeira configuração ou cenário chamado de ações TDM (*Time-Division Multiplexing*) atende 1 usuário de cada vez, ou a cada intervalo de tempo. Por esse motivo, os usuários estão separados entre si por intervalos de tempo diferentes. A segunda configuração ou modo de operação chamado de OFDM (*Orthogonal Frequency Multiplexing*) contém ações em que se atende mais de 1 usuário simultaneamente através de diferentes frequências. Neste caso, além de poderem ser atendidos em instantes diferentes, os usuários são atendidos simultaneamente, mas em frequências diferentes.

2.1 Modelo Markoviano e o Aprendizado por Reforço

Nessa seção são apresentados alguns conceitos importantes e as motivações para escolha da técnica de aprendizado por reforço para treinar um agente que irá alocar recursos e potência em um sistema de comunicação multiusuários.

Processos de decisão de markov são parte da base para o entendimento das soluções utilizando aprendizado por reforço. Um processo é uma sequência de estados de um sistema e ações tomadas por um agente a medida que o tempo passa. No contexto apresentado, considera-se um processo discreto onde o tempo é representado por uma sequência de passos (instantes) e estocástico pois os estados do sistema variam de forma aleatória. Assim, o processo pode ser representado pela sequência dos estados $s_0, s_1, ..., s_t$.

De maneira mais geral, o estado futuro $(t+1)$ de um sistema (causal) depende de toda a história de estados experimentados ate o instante t. Em um processo de Markov, o estado futuro depende apenas do estado presente, ou seja, $s_{t+1} = F(s_t)$. Como se conhece as probabilidades de transição entre os estados (estocástico) e os estados são finitos, pode-se modelar a função F como uma matriz $\mathbf{P}$, de modo que $P_{ij} = P[s_{t+1} = j|s_t = i]$. Mais detalhes serão apresentados sobre a representação dos estados, ações e matriz de transição do sistema nas seções futuras.

Ao agente cabe escolher as ações. E para isso, é preciso que conheça as vantagens/desvantagens de cada ação, ou a recompensa. Como a recompensa de longo prazo é de difícil modelagem (o que justifica treinar um agente), limita-se a modelar a recompensa de curto prazo e deixamos que o agente adquira experiência e encontre a melhor recompensa de longo prazo. Assim, pode-se atribuir para cada estado do sistema e ação do agente uma

recompensa $\mathbf{R} = R(s, a)$.

Como cada estado carrega informação suficiente para transição de estado, não é necessário representar a recompensa do processo inteiro $s_0, s_1, ..., s_t$, apenas s. As funções que representam as recompensas de longo prazo (V e Q) serão detalhadas nas seções futuras.

Como nosso problema situa-se em um cenário com grande quantidade de estados e ações, decisões sequencias tomadas por agentes em tempo discreto e características estocásticas do ambiente, a técnica de aprendizado por reforço com modelagem Markoviana se mostra interessante de acordo com resultados apresentados na literatura (FORD et al., 2017),(Hong Shen Wang; Moayeri, 1995),(Zhu et al., 2018),(CARNEIRO; CARDOSO; VIEIRA, 2021) (CARNEIRO et al., 2021).

2.2 Ações TDM

Neste modo de operação ou configuração para o sistema, as ações atendem um único usuário de cada vez. Escolhe-se qual usuário (k dos K possíveis), em que canal transmitir (m dos M possíveis) e qual modo de transmissão (j dos J possíveis), ou seja $K \cdot M \cdot J + 1$ ações diferentes. O $+1$ representa a ação de não agir. Atende-se um usuário de cada vez no *TTI* com ações TDM (*Time-Division Multiplexing*). Esse modo de operação ou configuração para o sistema também é denominado neste trabalho de **Cenário 1**.

2.3 Ações OFDM

Para este modo de operação do sistema, cada ação é composta pela escolha do usuário e do modo de transmissão para cada um dos M canais. Estas ações se referem ao cenário onde permite-se atendimento simultâneo através de ações OFDM. Neste trabalho, quando os dois tipos de ações (TDM e OFDM) são utilizados refere-se ao **Cenário 2**.

A multiplexação de frequência ortogonal permite transmitir simultaneamente em M canais utilizando subportadoras (frequências diferentes estrategicamente selecionadas para maior eficiência espectral). Assim, cada ação OFDM é composta de até M dos K usuários e até M dos $(J + 1)$ modos de transmissão.

Algumas considerações são necessárias sobre como escolher os usuários, canais e modos. Para o conjunto de ações, uma consideração é que a cada canal (conjunto de sub portadoras) deve ser atribuído um único usuário por TTI , e que o mesmo modo não seja escolhido para canais diferentes. A quantidade de ações N_a possíveis é a permutação M dos $(J + 1)$ modos multiplicados pela permutação M dos K usuários dada pela seguinte

equação (VASCONCELOS; CARDOSO; VIEIRA, 2020):

$$N_a = \frac{K!}{(K-M)!}\frac{(J+1)!}{(J+1-M)!} \tag{2.1}$$

onde o termo J+1 cobre também os casos em que se decide não atender naquele canal.

Na Equação 2.1, restringe-se as possibilidades de combinar os modos de transmissão, isso é, o mesmo modo não pode ser utilizado simultaneamente para diferentes usuários, tem-se o fatorial de J e não a exponencial. Há uma exceção no caso de $M = 2$, neste caso substitui-se $J + 1$ por J, uma vez que ao escolher modo j=0 para algum dos $M = 2$ canais tem-se o caso TDM.

Outro conjunto de ações pode ser obtido ao utilizar modo fixo entre os usuários. Este conjunto de ações é menor e permite um atendimento mais justo em termos de estado do *buffer* caso mais de um usuário demande a capacidade máxima do sistema (modo com maior capacidade). Assim, tem-se outro conjunto de ações OFDM:

$$Na = \frac{K!}{(K-M)!} \cdot J \cdot \sum_{j=0}^{M-2} \frac{M!}{j!(M-j)!} \tag{2.2}$$

Na Equação 2.2, nota-se que o sistema de comunicação atende pelo menos 2 usuários ($M \geq 2$) utilizando os J modos disponíveis.

Alguns resultados (seções 3.2 e 3.9) apresentam o conjunto de ações da equação (2.1) e outros utilizam o conjunto de ações da equação 2.2

2.4 Transição de Estados

A transição de estados por ação ocorre segundo um processo Markoviano. Assim, considera-se que o novo estado só depende do estado anterior da ação utilizada e do ambiente que define a chegada de dados e ganhos do canal para o TTI corrente. Os estados possíveis combinam os diferentes estados de *buffer* $S_b = (L+1)^K$ e diferentes estados de canais $S_c = C_h^M$ de forma a obter $S_b \times S_c$ estados possíveis, onde L é número de pacotes suportados pelo *buffer* e C_h é a quantidade de intervalos para os valores do ganho do canal. Por serem considerados independentes os estados do canal e do *buffer*, a probabilidade de transição é o produto das probabilidades de transição do *buffer* e do canal. O conjunto de estados do *buffer*, canal e sistema são denominados $\mathcal{L}$, $\mathcal{C}$ e $\mathcal{S}$ respectivamente.

2.4.1 Estados do *buffer*

Uma das informações de entrada para o modelo Markoviano, a taxa média de pacotes, está relacionada com a quantidade b de pacotes gerados. Assume-se que a geração de pacotes obedeça a uma distribuição de Poisson com taxa média de chegada λ para cada

um dos K usuários. Ou seja, tem-se a seguinte equação para a probabilidade de ocorrência de b pacotes:

$$Prob(b, \lambda) = \frac{e^{-\lambda}\lambda^b}{b!} \tag{2.3}$$

Durante o TTI i, o *buffer* do usuário k possui l pacotes. Se chegam b pacotes a este *buffer* e t_a pacotes são transmitidos com a ação a, o novo estado do *buffer* é:

$$l_{i+1} = min(l_i + b - t_a, L) \tag{2.4}$$

Assim, pode-se reescrever (2.3) como:

$$p_{bk}(l_i, l_{i+1}|a) = \frac{e^{-\lambda}\lambda^b}{b!} \tag{2.5}$$

Como os usuários são independentes entre si, tem-se que:

$$p_b(\mathcal{L}, \mathcal{L}^{'}|a) = \prod_{k=1}^{K} p_{bk}(l, l^{'}|a) \tag{2.6}$$

sendo $\mathcal{L}$ e $\mathcal{L}^{'}$ estados do conjunto combinado de $(L+1)^K$ estados e l e $l^{'}$ os $L+1$ estados individuais do *buffer*. A matriz p_b de transição de estados do *buffer* depende da ação escolhida (que define t_a) e, portanto, é uma matriz $S_b \cdot S_b \cdot Na$. É interessante observar que se o tamanho máximo do *buffer* é igual a zero ($L = 0$) tem-se um único estado possível para o *buffer* e $p_b(\mathcal{L}, \mathcal{L}^{'}|a) = \prod_{k=1}^{K} 1 = 1$

A obtenção dos termos $p_{bk}(l, l^{'}|a)$ na Equação 2.6 é fundamental para calcular a matriz $p_b(\mathcal{L}, \mathcal{L}^{'}|a)$. Primeiramente, se calcula as probabilidades individuais de cada usuário k, e multiplica-se no produtório. Assim, para um *buffer* de tamanho $L = 2$ os termos l e l' podem assumir 3 valores: 0, 1 ou 2. Logo, para um único usuário e ação "a" definida, a matriz $p_{bk}(l, l^{'}|a)$ possui dimensão 3x3 e pode ser representada por:

$$p_b(\mathcal{L}, \mathcal{L}^{'}|a) = p_{b1}(l, l'|a) = \begin{pmatrix} p_{00} & p_{01} & p_{02} \\ p_{10} & p_{11} & p_{12} \\ p_{20} & p_{21} & p_{22} \end{pmatrix} \tag{2.7}$$

A Equação 2.3 representa a interação do sistema/usuários com o ambiente, e a ação a o efeito do agente no sistema/usuários. A ação do agente é definir o canal e o modo de transmissão de cada usuário (considerando o Cenário 2). Portanto, é necessário observar a nível de pacotes tanto o efeito do ambiente como do agente para cada um dos estados possíveis. Os termos centrais da matriz (fora das bordas) são mais simples de obter, uma vez que estão restritos a uma única possibilidade. Já os termos das bordas podem representar probabilidade acumuladas. Por exemplo, caso a ação seja não fazer nada, o termo $p_{01} = Prob(1, \lambda) = \frac{e^{-\lambda}\lambda^1}{1!} = \lambda \cdot e^{-\lambda}$ que é a probabilidade do *buffer* passar

de vazio para 1 pacote, ou seja, como o agente não faz nada (ação a) é a probabilidade de chegar 1 pacote da distribuição de Poisson, apenas o efeito do ambiente.

Para a primeira e última coluna da matriz $p_{b1}(l, l'|a)$ da Equação (2.7) é preciso considerar todas as possibilidades, i.e, na primeira linha, para o termo p_{02} qualquer quantidade maior ou igual a 2 que chegar irá encher o *buffer* já que o agente não fará nada. Porém, para p_{00} a única possibilidade é a chegada de 0 pacotes. Mas caso a ação alocasse recursos para transmitir até 3 pacotes, por exemplo, deve-se avaliar também as probabilidades de chegar 1, 2 ou 3 pacotes para definir o valor de p_{00}. Na prática, pode-se evitar de calcular diretamente 1 elemento de cada linha e usar o complemento da soma dos outros elementos da linha para obtê-lo. A tarefa de calcular o último termo pode ser exaustiva uma vez que, a rigor, deve-se contemplar todas as possibilidades de chegada de pacotes. De maneira geral, o termo da primeira coluna costuma ser mais fácil de calcular que o da última pelo menos de forma exata.

Para obter de forma direta o termo da última coluna, sem ter que somar infinitas possibilidades, adota-se uma precisão, i.e, define-se o horizonte no valor de $b > \lambda$ em que $Prob(b, \lambda) = 0.001$, ou simplesmente utiliza-se a integral da pdf(*probability density function*), que é justamente uma soma infinita, a cdf(*cumulative distribution function*) que nos dá a probabilidade acumulada. Também pode ser feito para o cálculo dos termos da primeira coluna. O método utilizando um horizonte finito para os termos da última colunas é utilizada no cálculo da matriz $R(\mathcal{S}|a)$ para avaliar o valor médio da recompensa imediata ao se tomar a ação a quando no estado s.

Vale lembrar que as equações apresentadas para os cálculos de $P(\mathcal{S}, \mathcal{S}'|a)$ e $R(\mathcal{S}|a)$ se referem a um único usuário, único estado de canal e para uma ação simples (não fazer nada) e portanto são equivalentes a $p_b(l, l'|a = 1)$ e $R(l|a = 1)$. A combinação dos K usuários, Na ações e C_h estados de canais contribui para o crescimento rápido da matriz $\mathbf{P}$ que tem dimensão $[(L+1)^K.C_h^M] \cdot [(L+1)^K.C_h^M] \cdot [Na]$.

Para otimizar o desempenho do algoritmo utiliza-se vetorização para os cálculos no espaço de estados e aproveita-se as simetrias do problema, i.e, realiza-se os cálculos de $p_{bk}(l, l'|a)$ uma única vez por usuário e ação, ou seja, $K(L+1)^2 \cdot Na$ operações além de realizar a combinação de forma eficiente dos estados de *buffer* e canal. Por exemplo, para o caso do segundo usuário, cada termo p_{ij} da Equação 2.7 deve ser expandido por uma outra matriz de mesmo tamanho referente ao segundo usuário. O resultado será uma matriz $9 \cdot 9$ e $27 \cdot 27$ para o terceiro usuário. O mesmo ocorre ao incluir os estados dos canais, os elementos se expandem em matrizes representando a probabilidade de mudança de estado de canal $C_h^M \cdot C_h^M$. Por fim, tem-se uma matriz com as probabilidades de transição de estados quadrada com $[(L+1)^K \cdot C_h^M]^2$ elementos. É importante que a matriz de estados S e de ações A sejam calculadas uma única vez e armazenadas.

Na Figura 2.1 pode-se visualizar as cadeias referentes às matrizes de transição

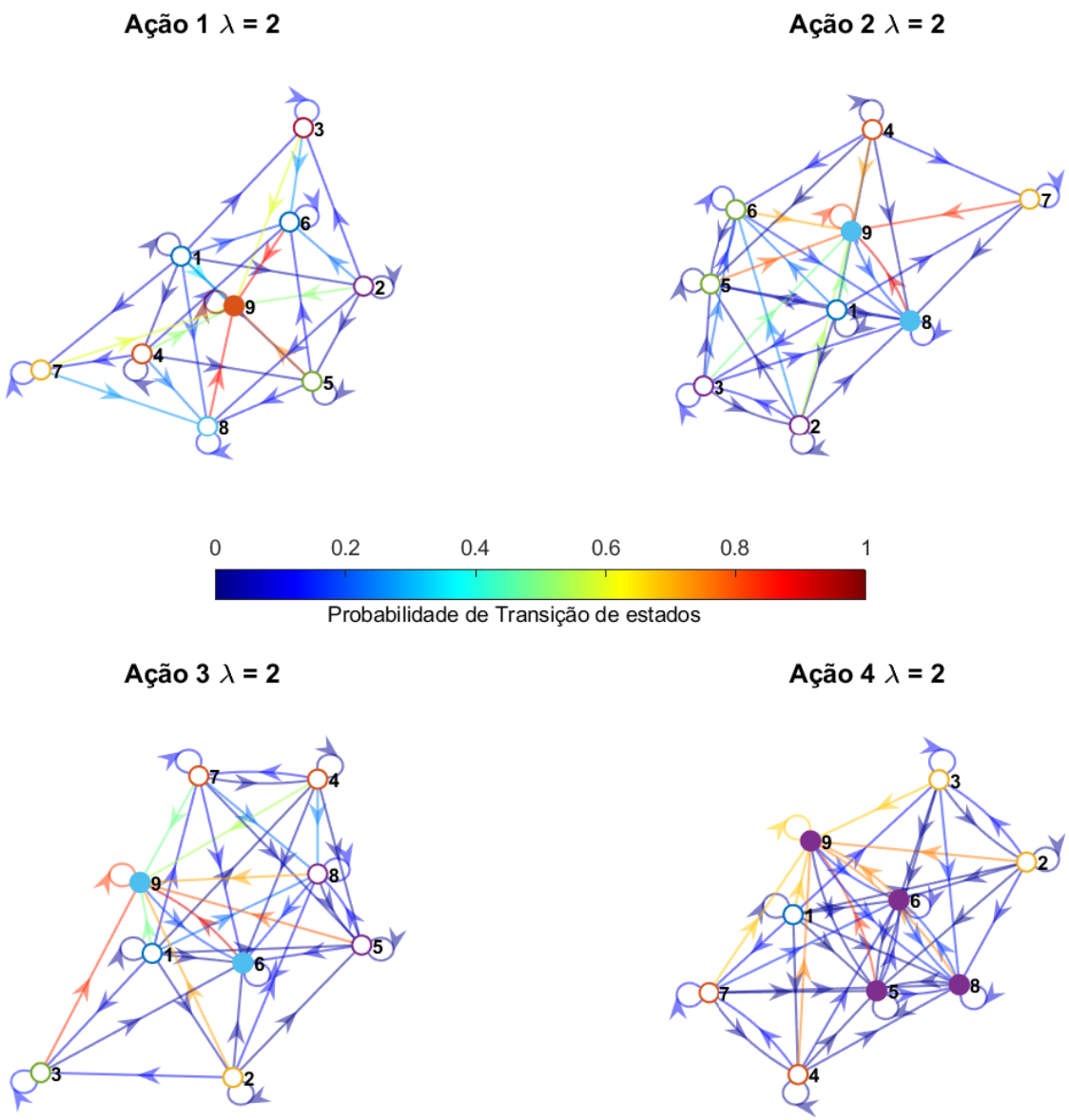

Figura 2.1 – Cadeia de Markov representando a probabilidade de transição de estados do *buffer* de tamanho máximo 2 para 2 usuários.

de estado do *buffer* para 4 ações diferentes. A primeira ação não faz nada. As ações 2 e 3 atendem um único usuário e a ação 4 atende 2 usuários simultaneamente. Os nós preenchidos representam estruturas cíclicas do estado estacionário dos estados do *buffer*.

2.4.2 Estados do canal

O ambiente de comunicação simulado neste trabalho possui C_h estados possíveis para cada um dos M canais $C = c_0, ...c_{C_h-1}$ de acordo com o ganho $|h|^2$ do canal e $C_h - 1$ limiares $\rho = \rho_1, \rho_2, ...\rho_{C_h-1}$, com $\rho_0 = 0$ e $\rho_{C_h} = \infty$. O termo ρ_n representa o limiar de transição entre os estados do canal. Representa-se os estados do canal (variável contínua) de forma finita através dos limiares de transição. Aqui, representa-se a distribuição de uma amostragem por um usuário em um canal. Na prática, com mais canais e usuários tem-se um ganho estatístico na distribuição já que observa-se uma matriz $K \cdot M$ de ganhos de usuário por canal. A probabilidade do canal n estar no estado c_n quando feita uma única observação é dada por:

$$p_{cm}(c_n) = \int_{\rho_n}^{\rho_{n+1}} pdf(\rho)d\rho \tag{2.8}$$

Da Equação (1.3), obtém-se:

$$p_{cm}(c_n) = e^{\frac{-\rho_n}{\rho_m}} - e^{\frac{-\rho_{n+1}}{\rho_m}} \tag{2.9}$$

Neste trabalho, adota-se a propriedade Markoviana também para os estados dos canais. A transição de estado dos canais se dá apenas entre estados vizinhos do intervalo, i.e, a mudança no ganho do canal ocorre entre regiões vizinhas, e apenas 1 limiar é transposto. Isso considera um modelo sem mudanças bruscas entre o estado do canal. A grau de alteração de estado será maior ou menor dependendo da quantidade de estados de canal considerados e essa consideração só pode ser observada se pelo menos 3 estados de canal são considerados, caso contrário todos os estados de canal são vizinhos entre si. Considerando um processo Markoviano de nascimento e morte, as probabilidades de transição de estado para estado melhor ou pior ocorrem de forma independente para cada canal e são dadas pelas seguintes equações:

$$p_{cm}(c_n, c_{n+1}) = \frac{N(c_{n+1})Tf}{p_c(c_n)} \tag{2.10}$$

$$p_{cm}(c_n, c_{n-1}) = \frac{N(c_n)Tf}{p_c(c_n)} \tag{2.11}$$

onde T_f é a duração do TTI em segundos e $N(c_n)$ é o número de vezes que o limiar de c_n é cruzado por segundo, ou seja, o numerador nas equações (2.10) e (2.11) são estimativas para a probabilidade do canal mudar de estado em 1 TTI. Ao se dividir os numeradores das equações (2.10) e (2.11) por $p_c(c_n)$ obtém-se probabilidades condicionais, i.e., probabilidades de determinadas transições ocorrerem dado que o estado atual (c_n) é conhecido. O valor de $N(c_n)$ é obtido utilizando a seguinte equação (RAPPAPORT et al., 1996):

$$N(c_n) = \sqrt{\frac{2\pi\rho_{c_n}}{\rho_m}} f_D e^{-\rho_{c_n}/\rho_m} \tag{2.12}$$

em que f_D é a frequência de máximo efeito Doppler.

Dado que são M canais, cada um com C_h estados possíveis e pode-se utilizar mais de um canal ao mesmo tempo (para usuários diferentes) com o atendimento OFDM, tem-se C_h^M estados possíveis e independentes. Logo,

$$p_c(\mathcal{C}, \mathcal{C}') = \prod_{m=1}^{M} p_{cm}(c, c') \tag{2.13}$$

sendo $\mathcal{C}$ e $\mathcal{C}'$, estados do conjunto combinado de C_h^M estados, e c e c' o estado individual de cada canal, do conjunto de C_h estados. É interessante notar que caso se considere um único estado para os canais ($C_h = 1$), o canal nunca muda de estado. Nesse caso considera-se o ganho médio do canal para a simulação e tem-se $p_c(\mathcal{C}, \mathcal{C}') = \prod_{m=1}^{M} 1 = 1$.

Pode-se representar os estados dos canais com uma cadeia de Markov. Na Figura 2.2 observa-se este comportamento para o caso de $M = 2$ canais e $C_h = 4$ estados de canal. Cada nó na Figura 2.2 representa um estado de canal do sistema, sendo que toda cadeia retrata todas os possíveis estados de cada canal. Como visto, essa configuração apresenta $4^2 = 16$ estados. As arestas representam transição de estados, e as cores mais quentes representam uma maior probabilidade de transição entre os estados.

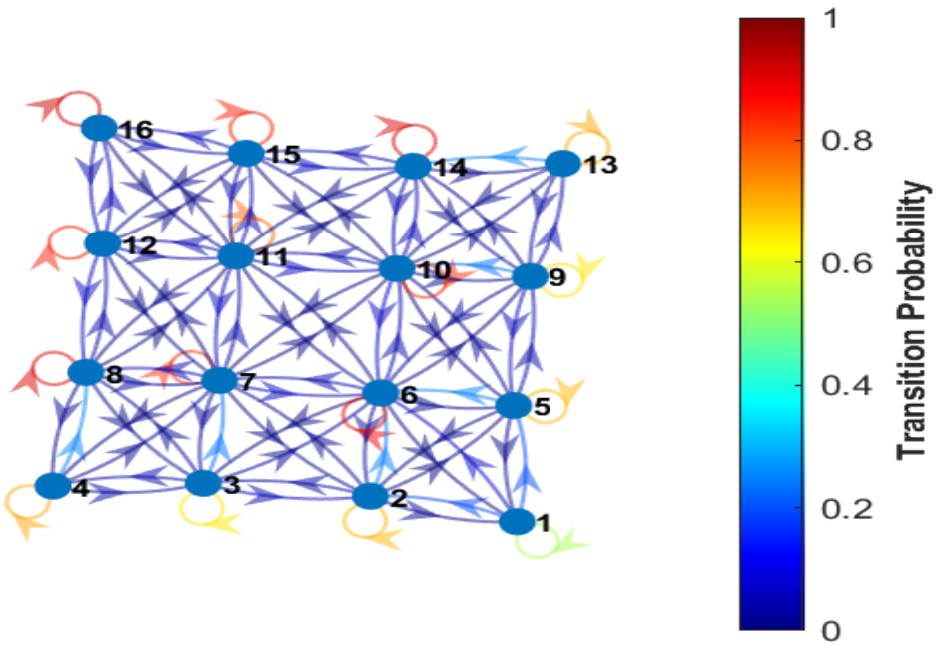

Figura 2.2 – Cadeia de Markov para os estados do canal para um sistema com de 2 usuários e 2 canais utilizando algoritmo húngaro e apresentando cada canal 4 estados possíveis.

Como pode ser notar pela Figura 2.2 os estados 4 e 1 não são vizinhos. É necessário

que estados intermediários sejam visitados para que ocorra a transição de 1 para 4. Além disso, percebe-se como há maior tendência dos estados dos canais se manterem no estado atual pela cor dos *loops*. A matriz de transição de estados $p_c(\mathcal{C}, \mathcal{C}^{'})$ para esse caso é 16x16 e contém zeros para transições não possíveis.

Como se assume que a transição de estados dos canais siga um modelo Markoviano, $pc_m(c, c^{'})$ é zero para estados não vizinhos. Assim, como a mudança de estados do *buffer* e do canal são independentes entre si, a probabilidade total de transição de estados é:

$$p_s(\mathcal{S}, \mathcal{S}^{'}|a) = \prod_{k=1}^{K} p_{bk}(l, l^{'}|a) \prod_{m=1}^{M} p_{cm}(c, c^{'}) \tag{2.14}$$

2.5 Modelagem Markoviana do Sistema de Comunicação

O conjunto de estados do sistema $\mathcal{S}$ tem $(L+1)^K \cdot C_h^M$ estados. Além disso, na Equação (2.14) tem-se as probabilidades de transição entre esses estados para cada ação. Assim, assumindo o comportamento sequencial do processo de decisão de alocação de recursos e visando otimizar o retorno de longo prazo, é natural considerar um modelo de Markov para o sistema.

Os parâmetros de QoS são equacionados para validar o uso do modelo com a simulação. Para uma ação a, a probabilidade do estado estacionário $P_\pi(\mathcal{S})$ é obtida da solução não trivial do sistema linear:

$$P_\pi(\mathcal{S}) = P_s(\mathcal{S}, \mathcal{S}^{'} \mid a).P_\pi(\mathcal{S}) \tag{2.15}$$

como $P_s(\mathcal{S}, \mathcal{S}^{'} \mid a)$ conhecido, resolvendo (2.15) obtém-se $P_\pi(\mathcal{S})$ para cada estado s_i.

Considerando o estado estacionário, a probabilidade de transições de estado dada por (2.14) pode ser reescrita como a probabilidade de estado estacionário $P_\pi(s_i)$ e o tamanho médio da fila $\hat{B}$ no *buffer* considerando estados de *buffer* $b(s_i)$ pode ser estimado pela seguinte equação (FORD et al., 2017):

$$\hat{B} = \sum_{s_i=1}^{S} P_\pi(s_i) b(s_i) \tag{2.16}$$

e sabendo que

$$P_\pi(L) = P(T).P(L \mid T) + (1 - P(T)).P(L \mid \bar{T}) \tag{2.17}$$

onde $P_\pi(L)$ é a probabilidade de ter o *buffer* cheio em regime permanente e $P(L \mid T) = 1$ pois o *buffer* sempre está cheio após transbordar. A probabilidade de perda de pacotes é dada por:

$$P(T) = \frac{P_\pi(L) - P(L \mid \bar{T})}{1 - P(L \mid \bar{T})} \tag{2.18}$$

Visto que $P(L \mid \bar{T})$ é a probabilidade de ter um *buffer* cheio, mas sem perda de pacotes, ela dependerá da taxa de chegada de Poisson e do estado atual.

Com a informação da média de pacotes que chegam λ, os números de pacotes perdidos e transmitidos por usuário podem ser calculados, respectivamente, por:

$$\text{Lost}_k^{\lambda} = \lambda P(T) \tag{2.19}$$

$$f_k = \lambda(1 - P(T)) \tag{2.20}$$

2.6 Função Utilidade

Durante o treinamento por reforço é preciso definir métricas para avaliar o desempenho do agente. Utilizam-se variáveis de interesse para uma função que irá modelar o desempenho do agente e influenciar as suas decisões. Por exemplo, se o objetivo é melhorar um certo parâmetro de desempenho de um sistema e minimizar o custo de um processo, faz sentido atribuir valor positivo para este parâmetro de desempenho e uma relação direta com a recompensa utilizando a variável "parâmetro de desempenho"no numerador. Já para o custo, faz mais sentido atribuir sinal negativo ou relação inversa por exemplo utilizando o custo no denominador. É necessário que a avaliação imediata da ação seja percebida pelo agente. A essa recompensa imediata se dá o nome de função de utilidade, função de recompensa ou função objetivo (SUTTON; BARTO, 2018).

Ao combinar diferentes objetivos da otimização trata-se um problema multi-objetivo utilizando um único agente que tem como objetivo maximizar a função de recompensa. A função utilidade do algoritmo de aprendizagem por reforço guiará o agente a optar por ações que por exemplo, minimizem perdas, maximizem ganhos ou tentem fazer os dois. Pode acontecer que ao ordenar as recompensas para os valores do domínio se encontre valores iguais de recompensa, ou seja, pode-se observar simetrias para diferentes ações. Isso representa o caráter multiobjetivo da Aprendizagem por Reforço, ao se considerar múltiplas etapas da decisão, avaliar a função de recompensa acumulada de longo prazo essa simetria pode desaparecer ou não. Não é exigido, portanto, que a função seja injetora, ou seja, que diferentes entradas ou elementos do domínio tenham valores diferentes de imagem.

A função utilidade é responsável por agrupar as variáveis do sistema e modelar uma relação entre elas que permita à solução do processo de aprendizado por reforço convergir para regiões com desempenho desejado. Neste trabalho, os parâmetros utilizados na função utilidade são o fluxo de dados $B_k(s, a)$ em pacotes do usuário k e o custo $C_k(s, a)$ do usuário k composto pelo consumo de potência e a pressão total dos usuários no *buffer* ou de pacotes perdidos.

Neste trabalho, a função utilidade (ou de recompensa) adotada como base no algoritmo de Aprendizagem por Reforço tem a seguinte forma geral:

$$R(s,a) = \sum_{k=1}^{K} \frac{B_k(s,a)}{C_k(s,a)} \tag{2.21}$$

$$B_k(s,a) = min(l_k + b_k, V \cdot j_k) \tag{2.22}$$

onde l_k é quantidade de pacotes no *buffer* do usuário k, b_k é o número de pacotes que chegaram para o usuário k, V é a taxa de código (*code rate*) e $j_k = 0...J$ o modo de transmissão para o usuário k. Cabe ressaltar que Zhu et al. (2018), por simplicidade, considera o valor máximo (capacidade) para o numerador, ou seja, $B_k(s,a) = V.j$.

A função de recompensa utiliza por (Zhu et al., 2018) leva em conta o tamanho da fila no *buffer*, a quantidade de pacote transmitidos e a potência alocada, conforme descrita a seguir.

Definição 2.1 *Sejam as variáveis V, j_k, K, $P_k(s,a)$ e l_k, a taxa de codificação, modo atribuído ao usuário k, número de usuários, potência alocada ao usuário k pela ação a quando no estado s e o número de pacotes no buffer do usuário k após a ação a, respectivamente, define-se a função de utilidade de Zhu pela seguinte equação:*

$$R_{ZhuTDM}(s,a) = \frac{\sum_{k=1}^{K} V.j_k}{\sum_{k=1}^{K} P_k(s,a) \sum_{k=1}^{K} e^{0.5.l_k}} \tag{2.23}$$

2.6.1 Função de utilidade proposta considerando a pressão da parcela de pacotes perdidos

Neste trabalho, propõe-se adotar na função utilidade, a média das razões $\sum(B/C)$ entre os usuários em vez de se considerar a razão das médias $\sum B / \sum C$ conforme feito em Zhu et al. (2018), buscando evitar que apenas alguns usuários sejam privilegiados no processo. No caso da razão das médias, cada componente k da soma recebe pesos diferentes com base no valor C_k. A Equação 2.24 mostra a diferença dos pesos ao utilizar os dois tipos de média.

$$\frac{\sum B}{\sum C} = \sum \frac{B}{\sum C} = \sum \frac{B}{C} \cdot \frac{C}{\sum C} \neq \sum \frac{B}{C} \cdot 1 = \sum \frac{B}{C} \tag{2.24}$$

Pode-se perceber que a razão das médias adiciona ao modelo de aprendizado por reforço um peso proporcional ao custo da parcela corrente k quando comparado à média das razões. O custo na função utilidade representa grandezas/funções que se deseja minimizar. Isso tem impacto na métrica da recompensa, pois boas soluções de baixo custo para um dos usuários, quando comparadas a outras soluções de alto custo de outro usuário serão ofuscadas por esse ponderamento utilizado na soma. A mesma consideração foi adotada

para o cálculo da eficiência energética descrito na seção de resultados. Assim, propõe-se que o custo $C_k(s,a)$ do usuário k seja dado por:

$$C_k(s,a) = P_k(s,a) \sum_{k=1}^{K} (fk) \tag{2.25}$$

onde f_k é a pressão no *buffer*.

$$fk = e^{0.5A_k} \tag{2.26}$$

onde A_k é a quantidade de pacotes no *buffer* Zhu et al. (2018) ou a quantidade de pacotes perdidos do usuário k após ação a. Considera-se uma função exponencial (a pressão) para evitar a divisão por zero e ajudar a diferenciar os possíveis valores do expoente. Assim, a equação para a função utilidade proposta se torna:

$$R(s,a) = \sum_{k=1}^{K} \frac{B_k(s,a)}{P_k(s,a) \sum_{k=1}^{K} e^{0.5.Lost_k}} \tag{2.27}$$

A função utilidade proposta visa contemplar com maior intensidade cenários onde o *buffer* fica cheio. Ao utilizar os pacotes perdidos como parâmetros a função de recompensa terá valores diferentes com base na quantidade de pacotes perdidos. O horizonte do denominador fica mais amplo, já que A_k não está limitado ao tamanho do *buffer*. Para tamanho de *buffer* (L) cuja ordem de grandeza é maior do que a taxa λ de chegada, tem-se da Equação (2.3) que $Prob(L,\lambda) \approx 0$, ou seja, as funções de recompensa proposta e a apresentada em Zhu et al. (2018) se aproximam.

A definição de eficiência energética é dada pela Equação (1.11). Com essa definição pode-se substituir o termo $\sum_{k=1}^{K} \frac{B_k(s,a)}{P_k(s,a)}$ da Equação 2.27 por $EE(s,a)$. Assim, como primeira proposta de função de recompensa, optou-se por substituir o tamanho da fila no *buffer* pela quantidade de pacotes perdidos, da forma descrita a seguir.

Proposição 2.1 *Sejam as variáveis $EE(s,a)$, V, j_k, K, $P_k(s,a)$, $Lost_k^{\lambda}$, l_k e b_k, a eficiência energética, a taxa de codificação, o modo atribuído ao usuário k, o número de usuários, a potência alocada ao usuário k pela ação a quando no estado s, o número de pacotes perdidos pelo usuário k após a ação a para taxa média de chegada λ, o número de pacotes no buffer do usuário k antes da ação a e o número de pacotes que chegaram para o usuário k para taxa média de chegada λ, respectivamente, propõem-se a função de utilidade proposta pela seguinte equação:*

$$R_{Pro1}(s,a) = \frac{EE(s,a)}{\sum_{k=1}^{K} e^{0.5.(Lost_k)}} = \frac{\frac{1}{K} \sum_{k=1}^{K} \frac{E[min(l_k + b_k^{\lambda}, V \cdot j_k])}{P_k(s,a)}}{\sum_{k=1}^{K} E[e^{0.5.(Lost_k)}]} \tag{2.28}$$

onde E[.] representa o operador esperança matemática.

Neste trabalho, avalia-se inicialmente o desempenho da utilização da função utilidade proposta $R_{Pro1}(s,a)$ e a de Zhu et al. (2018) como funções objetivo no algoritmo de aprendizado por reforço. No Capítulo 3, o desempenho dessa função é comparado com a de Zhu et al. (2018) em 2 cenários: o Cenário 1 considerando ações TDM e o Cenário 2 considerando ações TDM e OFDM.

2.6.2 Funções de utilidade utilizando o tamanho do *buffer* e os pacotes perdidos combinados

Nesta subseção, são propostas mais 3 funções de utilidade para o algoritmo de aprendizagem de reforço que levam em conta além da quantidade de pacotes transmitidos e da potência alocada, tanto o tamanho da fila no *buffer* como a quantidade de pacotes perdidos. As funções de recompensa propostas ainda utilizam a taxa média de chegada de pacotes para dar relevância a contribuição da quantidade de pacotes perdidos, conforme descritas a seguir.

Proposição 2.2 *Sejam as variáveis $EE(s,a)$, V, j_k, K, $P_k(s,a)$, $Lost_k$, l_k, b_k^λ e λ_k, a eficiência energética, a taxa de codificação, o modo atribuído ao usuário k, o número de usuários, a potência alocada ao usuário k pela ação a quando no estado s, o número de pacotes perdidos pelo usuário k após a ação a, o número de pacotes no buffer do usuário k antes da ação a, o número de pacotes que chegaram para o usuário k e a taxa média de chegada de pacotes do usuário k respectivamente, propõem-se as funções de utilidade dadas pelas seguintes equações:*

$$R_{Prop2}(s,a) = \frac{EE(s,a)}{\sum_{k=1}^{K}(e^{0.5Lost_k} + B_k)]} = \frac{\frac{1}{K}\sum_{k=1}^{K}\frac{E[min(l_k+b_k^\lambda, V\cdot j_k)]}{P_k(s,a)}}{\sum_{k=1}^{K}E[(e^{0.5Lost_k} + B_k)]} \tag{2.29}$$

$$R_{Prop3}(s,a) = \frac{EE(s,a)}{\sum_{k=1}^{K}\frac{e^{0.5(l_k+Lost_k)}}{B_k+1}} = \frac{\frac{1}{K}\sum_{k=1}^{K}\frac{E[min(l_k+b_k^\lambda, V\cdot j_k)]}{P_k(s,a)}}{\sum_{k=1}^{K}E[\frac{e^{0.5(l_k+Lost_k)}}{B_k+1}]} \tag{2.30}$$

$$R_{Prop4}(s,a) = \frac{EE(s,a)}{\sum_{k=1}^{K}\lambda_k.Lost_k + e^{0.5.(B_k+Lost_k)}} = \frac{\frac{1}{K}\sum_{k=1}^{K}\frac{E[min(l_k+b_k^\lambda, V\cdot j_k)]}{P_k(s,a)}}{\sum_{k=1}^{K}E[\lambda_k.Lost_k + e^{0.5.(B_k+Lost_k)}]} \tag{2.31}$$

Juntamente com a função de Zhu et al. (2018), os desempenhos do algoritmo de aprendizagem por reforço utilizando as propostas são comparados para o Cenário 2 com ações TDM + OFDM no Capítulo 3. Ainda é apresentada alguma alteração nessas funções durante a simulação da alocação de recursos para os sistemas de comunicação considerados. Essas alterações são explicadas no capítulo de Simulações e Resultados.

O algoritmo de aprendizado por reforço considerado é baseado no algoritmo *Q-Learning* com iteração de política, apresentado na seção a seguir.

2.7 Algoritmo Q-Learning

Uma política π é um vetor de ações escolhidas para cada estado s dentre os N_s estados possíveis do sistema, ou seja, é uma realização das $(N_a)^{Ns}$ possíveis políticas. Com N_a ações possíveis para cada estado e Ns estados únicos tem-se $(N_a)^{Ns}$ permutações de política. O algoritmo *Q-Learning* utiliza uma função objetivo (que serão denominadas como função 1 - Zhu (Zhu et al., 2018) e função 2 - Proposta) para calcular a recompensa imediata do estado atual s_i ao seguir determinada ação $\pi(s_i)$: r_i^π.

Para o problema proposto o horizonte é infinito, ou seja, não se tem um estado final. Por esse motivo, utiliza-se um coeficiente de desconto menor que 1 ($\gamma < 1$) de modo que as funções de valor de estado $V^\pi(s_i)$ e de ação $Q^\pi(s_i, a_i)$ possam convergir.

Para que seja possível considerar todas as recompensas futuras em cada etapa, é necessário conhecer o comportamento futuro com antecedência, ou seja, a política que será seguida. Se o objetivo é a recompensa do estado, ao seguir determinada política, a função de valor de estado $V^\pi(s_i)$ deve ser observada.

$$\begin{aligned} V^\pi(s_i) &= r_i^\pi + \gamma r_{i+1}^\pi + \gamma^2 r_{i+2}^\pi + ... \\ \gamma V^\pi(s_{i+1}) &= 0 + \gamma r_{i+1}^\pi + \gamma^2 r_{i+2}^\pi + ... \end{aligned}$$

$$V^\pi(s_i) = r_i^\pi + \gamma V^\pi(s_{i+1}) \tag{2.32}$$

Na Equação 2.32, o termo $V^\pi(s_i)$ representa o valor do estado antes de se tomar a decisão da ação atual, ou seja, é uma média do valor do estado para todas as ações. Da mesma forma pode-se chegar a uma expressão para a função de valor de ação $Q(s_i, a_i)$:

$$Q(s_i, a_i) = R(s_i, a_i) + \gamma Q(s_{i+1}, a_{i+1}) \tag{2.33}$$

Nesse caso, a ação atual é definida por a_i e o valor de $R(s_i, a_i)$ é conhecido. $Q(s_{i+1}, a_{i+1})$ é um valor médio futuro que depende principalmente do modelo do ambiente.

A Figura 2.3 mostra a diferença entre as duas funções, de valor de estado $V^\pi(s)$ (Figura 2.3.a) e a de valor de ação $Q^\pi(s, a)$ (Figura 2.3.b). A função $V^\pi(s)$ é uma média da recompensa de longo prazo ao seguir a política π partindo do estado s. A função $Q^\pi(s, a)$, por outro lado, oferece um nível a mais de detalhe pois considera a ação a. Nesse caso, tem-se a recompensa futura para o estado s ao se tomar a ação a, podendo ela estar ou não na política π. Se as ações futuras a' consideradas pelo fator de desconto, seguem a política π temos um método *on-policy*.

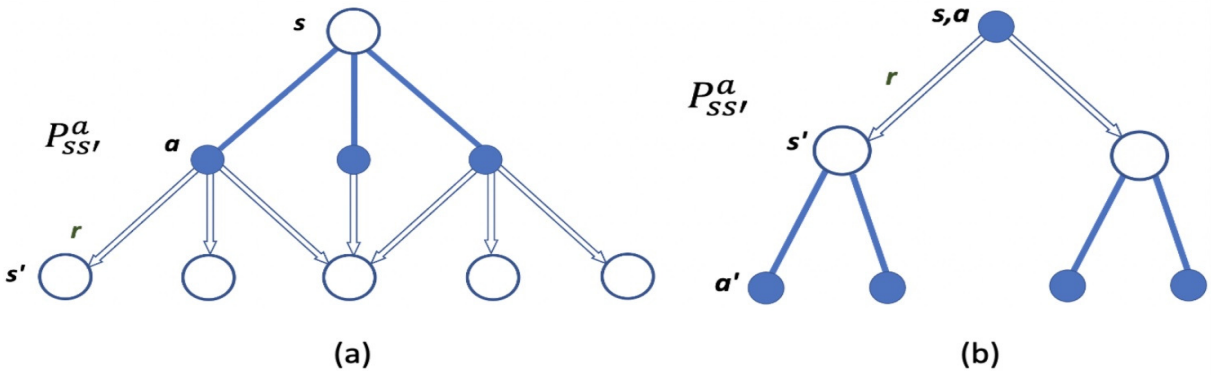

Figura 2.3 – Diferença entre função de valor de estado $V^\pi(s)$ (a) e valor de ação $Q^\pi(s,a)$ (b).

É comum que a política π não seja seguida durante o aprendizado. Nesses casos, utiliza-se o termo $\pi(s|a) \neq 1$ que representa a probabilidade do agente seguir a política ótima quando no estado s . Pode-se entender como um fator de exploração, uma vez que não se conhece π. Assim, é verdade que:

$$V^\pi(s) = \sum_{a \in A} \pi(s|a) \cdot Q^\pi(s,a) \tag{2.34}$$

Nota-se que ao se chegar ao ótimo, $\pi(s|a) = 1$ e, assim pode-se escrever:

$$V^\pi(s) = \sum_{a \in A} Q^\pi(s,a) \tag{2.35}$$

A função de valor utilizada neste trabalho é a de estado $Q(s_i, a_i)$. Primeiro, é preciso conhecer a probabilidade de se chegar ao estado $s_{i+1} = s'$ partido de s_i e tomando a ação a_i. Para isso, pode-se usar a matriz $P(s, s'|a = a_i)$. Além disso, é preciso saber qual ação a_{i+1} tomar quando no estado s_{i+1}. Bellman resolveu esse problema propondo uma solução para escolha de $a_{i+1} = a'$ através da seguinte equação (SUTTON; BARTO, 2018):

$$Q(s_i, a_i) = R(s_i, a_i) + \gamma \sum_{s'} P(s_i, s', a_i) Q(s', a') \tag{2.36}$$

onde $R(s_i, a_i)$ é a recompensa imediata dada pela função utilidade e γ é o fator de desconto.

A proposta de Bellman é utilizar o argumento a que maximiza a esperança de $Q(s', a)$ dado pela multiplicação da função $Q(s', a)$ pela probabilidade de transição de s para s' dada pelo modelo. Como a função Q é única, deve ser capaz de representar tanto o estado de partida quanto o estado de chegada. Assim,

$$a' = a_{max} = Arg_a Max[\sum_{s'} P(s_i, s', a_i) Q(s', a)] \tag{2.37}$$

Para um caso real em que não se conhece π utiliza-se os ótimos locais em a. Ou seja, ao tomar a ação ótima para cada passo espera-se obter uma função $Q(s, a)$ mais próxima de $Q^{\pi}(s, a)$ e que forneça política ótima (π). Essa consideração simplifica o problema com horizonte aparentemente infinito de passos em sub-problemas menores encadeados. Essa estratégia caracteriza uma solução conhecida como *off-policy*, pois não se utiliza uma política fixa para definir as recompensas futuras no episódio. A Equação (2.38) é conhecida como Equação de Bellman (SUTTON; BARTO, 2018).

$$Q(s_i, a_i) = R(s_i, a_i) + \gamma \sum_{s'} P(s_i, s', a_i) \max_{a'} Q(s', a') \tag{2.38}$$

A forma como a' é escolhido define o método como *on-policy* ou *off-policy*. Caso a' respeite a política π corrente tem-se um método *on-policy*, o agente aprende na política. Essa estratégia costuma ser mais lenta, mas há maior garantia de convergência para ótimos globais. Caso $a' = a_{max}$ como na Equação 2.37, tem-se um método *off-policy*, o agente opta pela melhor ação que a estimativa atual da função Q provém. Há maior viés nessa estratégia uma vez que a função Q muda constantemente até que a política corrente (que se espera seja a ótima) não mude.

O modelo Markoviano considerado para o sistema de comunicação provê a matriz $\mathbf{P}$ dada pela Equação 2.14 e assim é possível encontrar a ação a_{max} para as transições de estado $s_i \rightarrow s_{i+1}$. Basicamente, o algoritmo *Q-Learning* consiste em adaptar o valor de $\mathbf{Q}$ e a política ótima π até que π se estabilize ou que se tenha atingido o número máximo de iterações. Ao convergir, a política π composta pela ação a ser tomada em cada estado será dada por:

$$\pi(s_i) = \max_{a}[Q(s_i, a)] \tag{2.39}$$

Nas Figuras 2.4 e 2.5 pode-se visualizar a primeira estimativa da superfície Q(s,a) e o valor final ao convergir, respectivamente. As figuras são referentes aos estados e ações de um sistema de comunicação de 2 usuários, *buffer* de tamanho igual a 2, 2 canais, 4 modos de transmissão e 2 estados de canais, totalizando 36 estados e 41 ações a serem tomadas. Percebem-se os valores menores de Q(s,a) para as ações TDM que representam as 17 primeiras ações da função proposta. Além disso, nota-se a diferença entre a recompensa imediata da primeira iteração e o valor final da função Q. Outro fator interessante é a quantidade de máximos locais. Esse fenômeno ocorre pela natureza simétrica do problema e é um dos principais desafios de se utilizar um único agente e transformar um problema aparentemente multi-objetivo através de uma função que relaciona esses objetivos.

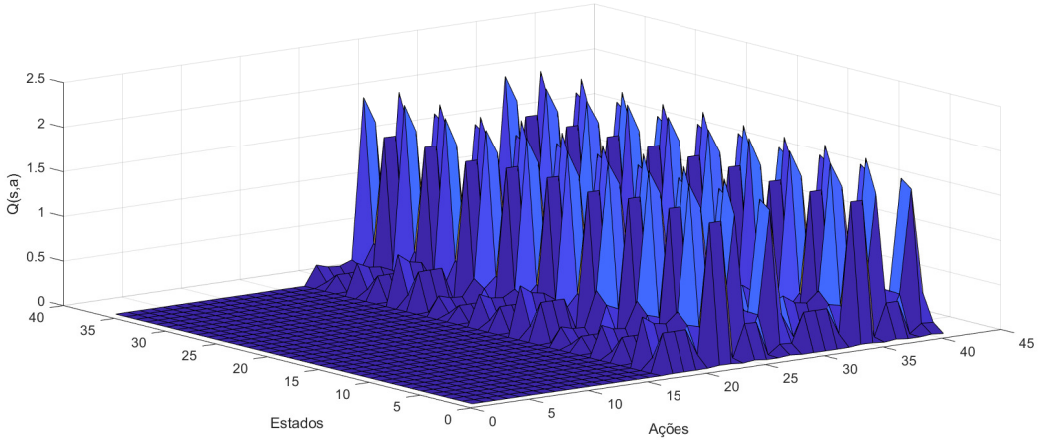

Figura 2.4 – Primeira estimativa da função Q no algoritmo de Q-learning com modelo.

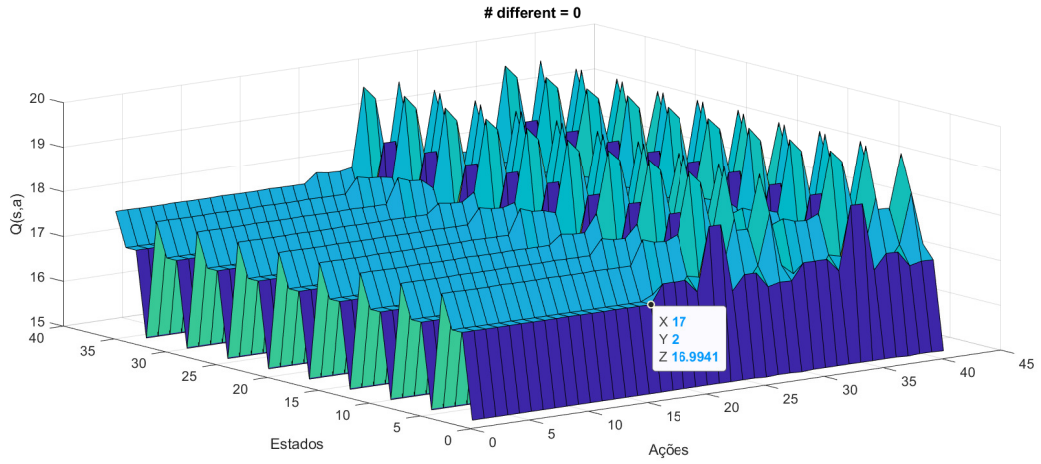

Figura 2.5 – Valor final da função Q no algoritmo de Q-learning com modelo.

2.8 Alocação de Recursos Considerando Tempo de Processamento dos Parâmetros do Modelo Markoviano

O Algoritmo 2.1 soluciona o problema de alocação de recursos e simula o atendimento síncrono dos usuários, isso é, o atendimento aguarda o cálculo da melhor política para o sistema. Nesse caso, o tempo de processamento para o cálculo da melhor política (solução do problema) não é considerado. É um algoritmo inicial e serve como referência para análise da melhor solução para o problema.

O Algoritmo 2.2 soluciona o problema de alocação de recursos e simula o atendimento assíncrono dos usuários, isso é, o atendimento não aguarda o cálculo da melhor política para o sistema. Na prática, monitora-se o tempo de processamento e impede o cálculo de nova política até que tenha sido de fato terminado o processamento para se obter uma solução para o problema de alocação de recursos. Em especial, para cenários adaptativos onde ocorre variação na taxa média de chegada de pacotes, esse algoritmo simula com mais fidelidade o que acontece na prática. Como o agente deve fornecer uma decisão em um tempo por vezes ordens de grandeza menor que o tempo necessário para se efetuar o processamento do cálculo/atualização da política, o agente deve tomar decisões com base em processamentos passados na maioria das vezes.

Esses 2 algoritmos apresentados são considerados nas simulações do Capítulo 3 para avaliar os algoritmos de alocação de recursos.

As entradas dos algoritmos podem ser visualizadas na tabela 2.1

Tabela 2.1 – Entradas dos algoritmos

K	Número de usuários
L	Tamanho máximo do *buffer*
M	Número de canais(sub-Bandas)
J	Número de Modos
C_h	Número de estados de canal
Λ	Taxas médias de chegada para dados sintéticos
γ	Fotor de desconto para recompensas futuras
$LimAdpt$	Variação mínima para recalcular política
$MinGanho$	Ganho mínimo de canal para atendimento
$MaxCluster$	Máximo número de usuários por *cluster*
BW	Largura de Banda
f_c	Frequência da portadora
BER_{max}	Mínima taxa de erro de Bit
D	Raio máximo de cobertura

- Passo 1: Define-se a matriz de estados **S** através dos estados de *buffer* e de canal.

  Os estados do *buffer* são obtidos da permutação com repetição das $L+1$ possíveis quantidades de pacotes no *buffer* de cada um dos K usuários, totalizando $(L+1)^K$

estados de *buffer*. Os estados de canal são obtidos da permutação com repetição dos C_h possíveis estados de canal de cada um dos M canais, totalizando $(C_h)^M$ estados de canal.

A matriz de estados do sistema **S** contém a combinação dos estados de *buffer* e canal totalizando $(L+1)^K(C_h)^M$ estados.

- Passo 2: Define-se a matriz de ações **A** através das ações de escolher o usuário de cada canal e o modo de cada canal.

  As ações de escolher o usuário no canal são obtidas da permutação sem repetição dos K usuários nos M canais totalizado $\frac{K!}{(K-M)!}$ para ações OFDM e $K \cdot M$ para ações TDM. Caso $M > K$, o valor do ganho médio de canal sofre um ganho estatístico, e adota-se M=K.

  As ações de escolher o modo no canal são obtidas dos J modos totalizando J para o caso TDM (não considera-se $J \cdot M$ pois com apenas um usuário/canal por ação, não justifica escolher outro canal para o modo).

  O caso OFDM pode variar segundo a definição das ações. Ao adotar o conjunto de ações da equação 2.1 sem repetição de modo, tem-se $\frac{(J+1)!}{(J+1-M)!}$ ações. Ao adotar o conjunto de ações da equação 2.2 que mantem fixo o modo escolhendo apenas o número de usuários que serão atendidos, tem-se $J \cdot \sum_{j=0}^{M-2} \frac{M!}{j!(M-j)!}$.

  A matriz de ações **A** contém a combinação das ações de escolher o usuário no canal e o modo no canal, totalizando $KMJ + 1 + \frac{K!}{(K-M)!}\frac{(J+1)!}{(J+1-M)!}$ ou $KMJ + 1 + \frac{K!}{(K-M)!}J \cdot \sum_{j=0}^{M-2} \frac{M!}{j!(M-j)!}$ ações.

- Passo 3: Define-se a taxa de codificação, potência do ruído e ganho médio do canal através do modelo TDL-B da (3GPP, 2018) e sistema LTE CP-OFDM.

  A taxa de codificação é obtida com o tamanho do pacote , tamanho do bloco de recurso, largura de banda, e número de canais (sub-bandas). No sistema LTE o tamanho do bloco de recurso é de 12 subportadoras espaçadas de 15 kHz cada e 7 símbolos. Logo, a taxa de codificação que converte a transmissão em bits para pacotes do sistema fluido é (pacotes/bit):

$$v = \frac{7BW}{15000.M.8.Pacote} \tag{2.40}$$

  onde M é o número de canais (sub-bandas), BW a largura de banda em Hz (descontada a banda de guarda), e Pacote o tamanho do pacote em bytes. Adota-se tamanho do pacote de 105 bytes para dados sintéticos e banda de 20 MHz, e 3000 bytes para simulação adaptativa de dados reais com até 640 MHz de banda. A potência do ruído é obtida considerando toda a banda. É importante salientar que o tamanho do pacote é um parâmetro variável na prática e tem relação direta com o valor do TTI.

O ganho médio do canal é obtido após simular o efeito de múltiplos percursos utilizando a tabela 1.1 de atrasos e potências do modelo TDL-B de 3GPP (2018) e o ganho médio de percurso considerando a distância máxima D. A divisão do ganho médio de cada estado de canal é obtida pela amostragem e ordenação de C_h amostras da distribuição exponencial utilizando a média obtida do modelo TDL-B distância D repetidas vezes com extração da média ao final. Os limiares adotados são calculados tomando a média dos ganhos médio de cada estado de canal.

- Passo 4:

  Obtém-se a potência média de cada estado e ação.

  Com detalhes o modo de transmissão, estado do canal, potência do ruído e BER máxima calcula-se a potência de cada par estado e ação.

- Passo 5:

  Aplica-se o algoritmo de aprendizado por reforço e simula-se os intervalos de transmissão.

  Quando permitido (restrição para o algoritmo 2.2): Calcula-se a política de atendimento através da matriz de transição de estados por ação $\mathbf{P}(s, s^{'}, a)$ e matriz de recompensas $\mathbf{R}(s, a)$, ou seja, o método de Q-learning off-policy com interação de política e com modelo markoviano é aplicado para obter a política ótima.

  Para o caso onde utiliza-se a rede DQN, as matrizes $\mathbf{P}$ e $\mathbf{R}$ não são calculadas e aplica-se quantidades variadas de episódio e passos na rede a depender da situação. Para dados sintéticos, simula-se as chegadas de pacotes em até 100 episódios de 10 passos para cada λ fixo. Para os dados reais, utiliza-se a média móvel da taxa de chegada de pacotes de cada usuário para simular menos episódios (10) e menos passos(5) com chegada respeitando a distribuição de poisson e obtém-se históricos da forma s, a, r, s' para ajuste de $a = \arg\max_a Q(s, a)$.

  Simula-se o atendimento dos usuários e observa-se os parâmetros de QoS para a plotagem.

Algoritmo 2.1 *: Simulação síncrona com estimação dos parâmetros do modelo Markoviano*

- ***Entrada:***

  $K, L, M, J, C_h, \Lambda, \gamma, LimAdpt, MinGanho, MaxCluster, BW, f_c, BER_{max}, D$

- ***Passo 1 - Definir matriz S através dos estados de buffer e de canal.***

  Estados do Buffer $\leftarrow$ *permutação*(L, K)

  Estados do Canal $\leftarrow$ *permutação*(C_h, M)

  $S \leftarrow$ *combinação*(*Estados do Buffer*, *Estados do Canal*)

- ***Passo 2 - Definir matriz A das ações de escolher o usuário no canal e o modo no canal.***

  Usuário no Canal $\leftarrow$ *permutação(K ,M)*

  $ModonoCanal \leftarrow$ *permutação(J ,M)*

  $A \leftarrow$ *combinação(Usuário no Canal ,Modo no Canal)*

- ***Passo 3 - Definir Ganho Médio através do modelo TDL-B do canal.***

  V, WN_0, *Ganho Médio(s)* $\leftarrow ModeloDoCanal(K, M, C_h, MinGanho, BW, f_c, D)$

- ***Passo 4 - Definir a Potencia Media alocada para estado e acao.***

  $PotenciaMedia(s, a) \leftarrow ObterPotencia(S, A, V, WN_0, BER_{max}$, *Ganho Médio(s)*$)$

- ***Passo 5 - Realizar o treinamento e simular o atendimento.***

  Enquanto $tti < TTI_{MAX}$

  $\lambda_i, Ganho \leftarrow Ambiente$

  $\Delta\lambda \leftarrow |\lambda_i - \lambda_0|$

  Se $\Delta\lambda > LimAdpt.\lambda_0$

  P,R $\leftarrow ModeloDeTransiçãoDeEstados(S, A, Pot.Média, \lambda_i)$

  política $\leftarrow AprendizadoPorReforçoMDP(P, R, \gamma)$

  Limpar P,R,Q

  $\lambda_0 \leftarrow \lambda_i$

  Atender$(K, L, M, J, V, C_h$, *política*, $Ganho, Amostra(\lambda_i))$

  tti++

Algoritmo 2.2 *: Simulação assíncrona com estimação do parâmetros do modelo Markoviano*

- ***Entrada:***

  $K, L, M, J, C_h, \Lambda, \gamma, LimAdpt, MinGanho, MaxCluster, BW, f_c, BER_{max}, D$

- ***Passo 1 - Definir matriz S através dos estados de buffer e de canal.***

  Estados do Buffer $\leftarrow$ *permutação*(L, K)

  Estados do Canal $\leftarrow$ *permutação*(C_h, M)

  $S \leftarrow$ *combinação*(*Estados do Buffer*, *Estados do Canal*)

- ***Passo 2 - Definir matriz A através das ações de escolher o usuário e o modo no canal.***

  Usuário no Canal $\leftarrow$ *permutação*(K, M)

  Modo no Canal $\leftarrow$ *permutação*(J, M)

  $A \leftarrow$ *combinação*(*Usuário no Canal*, *Modo no Canal*)

- ***Passo 3 - Definir Ganho Médio através do modelo TDL-B do canal.***

  $V, WN_0, GanhoMédio(s) \leftarrow ModeloDoCanal(K, M, C_h, MinGanho, BW, f_c, D)$

- ***Passo 4 - Definir a Potência Média alocada para estado e ação.***
  PotênciaMédia(s,a) $\leftarrow ObterPotência(S, A, V, WN_0, BER_{max}, GanhoMédio(s))$

- ***Passo 5 - Realizar o treinamento e simular o atendimento respeitando o tempo de treinamento.***

  Enquanto $tti < TTI_{MAX}$

  $\lambda_i, Ganho \leftarrow Ambiente$

  $\Delta\lambda \leftarrow |\lambda_i - \lambda_0|$

  Se $\Delta\lambda > LimAdpt.\lambda_0$ ***e*** $trainT * (nTrain + 1) \leq (tti - 1) * TTI$

  P,R $\leftarrow ModeloDeTransiçãoDeEstados(S, A, Pot.Média, \lambda_i)$

  política $\leftarrow AprendizadoPorReforçoMDP(P, R, \gamma)$

  Atualizar(*trainT*) , *Limpar P,R,Q* , *nTrain++*

  $\lambda_0 \leftarrow \lambda_i$

  Atender$(K, L, M, J, V, C_h, política, Ganho, Amostra(\lambda_i))$

  tti++

2.8.1 Deep Q-Learning (DQN): Solução que também oferece Modelo para o Sistema

A utilização de um modelo para um sistema pode facilitar a solução do problema. Entretanto, na prática, são comuns os problemas onde os modelos teóricos não são tão representativos ou até mesmo nem existem. Assim, é interessante abordar soluções que possam contemplar esses cenários.

Neste trabalho, para avaliar uma solução para o problema de alocação de recursos sem a utilização de um modelo da chegada de dados para o sistema de comunicação, optou-se pelo uso de redes neurais profundas por serem capazes de aprender as funções de recompensa de interesse, uma vez que são aproximadores universais e também pela sua capacidade de incorporar nos aproximadores de Q informação detalhada da dinâmica do sistema, ou seja, prover um modelo do ambiente.

O trabalho de Lee, Girnyk e Jeong (2020) propõe o uso de DQN (*Deep Q-learning Network*) e DDPG (*Deep Deterministic Policy Gradient*) para resolver o problema de *precoding* em sistema MIMO (*Multiple Input Multiple Output*) com um usuário. Nesta dissertação, uma rede DQN será utilizada para resolver o problema de alocação de recursos em cenários SISO (*Single Input Single Output*) OFDM com múltiplos usuários sujeito a restrição de BER máxima e com objetivos variados representados por uma função única de recompensa.

Algumas diferenças entre a solução utilizando rede DQN sem modelo para o sistema de comunicação e a solução com as matrizes $\mathbf{P}$ e $\mathbf{R}$ que modelam respectivamente a transição de estados e a recompensa, podem ser listadas:

1. Não se calcula a matriz $\mathbf{P}(s, s'|a)$ de probabilidade de transição de estados nem a matriz $\mathbf{R}(s|a)$ de recompensas médias. A DQN incorpora as experiências com o ambiente no aproximador da função Q(s,a) e na memória do agente. A cobertura do espaço de estados e ações irá depender da quantidade de treinamentos/exemplos na memória.

2. Não é preciso definir o tamanho do espaço de estados, apenas sua forma. No Capítulo 3 na seção de resultados utilizando DQN é discutida a forma do estado de entrada. Porém, é necessário listar o espaço de ações (discreto) pois cada ação é representada na camada de saída da rede neural. A Figura 2.6 representa uma rede neural que aprende a ação ótima para um estado s ($argmax(Q(s, a))$) com 1 camada de entrada, 3 camadas ocultas e uma camada de saída. Em uma rede DQN, utiliza-se uma rede classificadora (com várias saídas) em conjunto com o algoritmo *Q-Learning* , i.e., acopla-se o cálculo de Q com a decisão da melhor ação provida por uma rede neural. Assim, a rede aproxima a função Q e provê a ação ótima. Isso permite

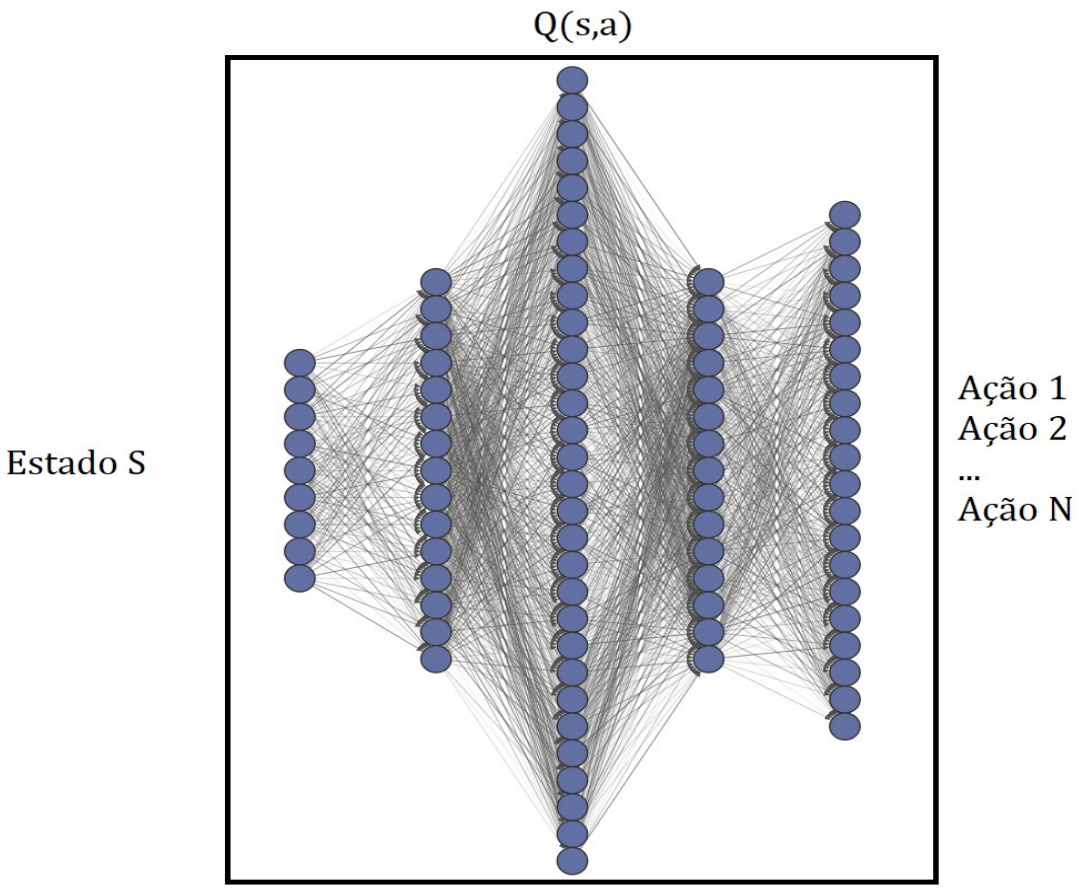

Figura 2.6 – Representação da rede neural que modela a função Q(s,a).

maior rapidez ao consultar a rede uma vez que estamos interessados na ação a que maximiza Q(s,a) e não especificamente no valor de Q.

3. A Equação de Bellman pode ser representada em sua forma alternativa, onde a função Q(s,a) é atualizada com base na taxa de aprendizagem α:

$$Q(s_i, a_i) \leftarrow (1 - \alpha) \cdot Q(s_i, a_i) + \alpha \cdot (R_i + \gamma \max_{a'} Q(s', a')) \tag{2.41}$$

   Na Equação 2.41, a cada novo passo de interação com ambiente é preciso definir o novo estado s' , a recompensa imediata R_i de tomar a ação a_i quando no estado s_i.

4. Um agente (crítico) irá escolher uma ação aleatória a_i com probabilidade ϵ (*exploration*) e com probabilidade $1 - \epsilon$ escolher a melhor ação a_i (*exploitation*) com base nas experiências passadas guardadas na memória no formato $e_i = (s_i, a_i, r_{i+1} s_{i+1})$. Essas entradas (*replay memory*) serão consultados futuramente em uma espécie de "treinamento supervisionado com rotulação dinâmica"(*experience replay*). Essa me-

mória, de tamanho finito, guarda a dinâmica do sistema modelado pelo aproximador, ou seja, tem a característica de um modelo do sistema de comunicação.

Após guardar a tupla na memória, atualiza-se os pesos da rede, ou seja, algumas (a depender do tamanho do *batch*) entradas passadas (aleatórias) da memória são revividas. Compara-se a saída do *forwarding* da rede (para ação a_i rotulada) com os rótulo da memória (r_i), obtém-se a função de perda e realiza-se o *backpropagation* para definir as modificações a serem realizadas nos pesos. É interessante notar que realiza-se o *forwarding* 2 vezes, já que também é preciso calcular ($maxQ(s^{'}, a^{'})$) no passo da Equação 2.41. Como o rótulo também guarda o valor do estado futuro s_{i+1} basta entrar com ele na rede e utilizar a saída de maior valor. Algumas aplicações utilizam duas redes independentes para essas duas tarefas (DDQN - *double deep Q-learning network*).

5. Ao utilizar entradas aleatórias para atualizar os pesos da rede, realiza-se um aprendizado menos dependente da ordem de entrada das amostras. O objetivo da rede é modelar a função Q(s,a) e não achar a política ótima. Uma vez encontrado os valores adequados para os parâmetros da função Q(s,a), a obtenção da política ótima é uma consequência.

Mais detalhes sobre a rede DQN podem ser obtidos no artigo original de Mnih et al. (2013). A solução utilizando rede DQN apresentada no presente texto também é caracterizada como *off-policy*. Como *experience replay* é usado para escolher a melhor ação, a implementação desse tipo de rede neural tem uma especialização para o problema, o que caracteriza um aprendizado profundo e justifica a nomenclatura *deep Q-Learning*.

2.9 Motivação para o agrupamento dos usuários em *clusters*

Percebe-se que, para solucionar o problema de forma analítica é preciso uma matriz **P** completa com as probabilidades de transição de todos os estados entre si para cada ação e uma matriz de recompensas **R** com as recompensas imediatas para cada estado e ação. Ocorre que, com o aumento do número de usuários e canais, o número de estados e ações crescem rapidamente e, por consequência, o tamanhos das matrizes **P** e **R**. Com a limitação de memória RAM é natural que essa memória se esgote antes de conseguir atingir a dezena de usuários.

Como proposta, para melhorar o desempenho do algoritmo proposto em relação à memória utilizada, sugere-se a redução de problemas com mais usuários em problemas menores, agrupando alguns usuários em *clusters* e resolvendo múltiplos problemas de forma desacoplada. Para esse trabalho foi escolhido um limite de até 3 usuários para cada *Cluster*. Em um primeiro cenário considerou-se 4 modos de transmissão, 4 estados de

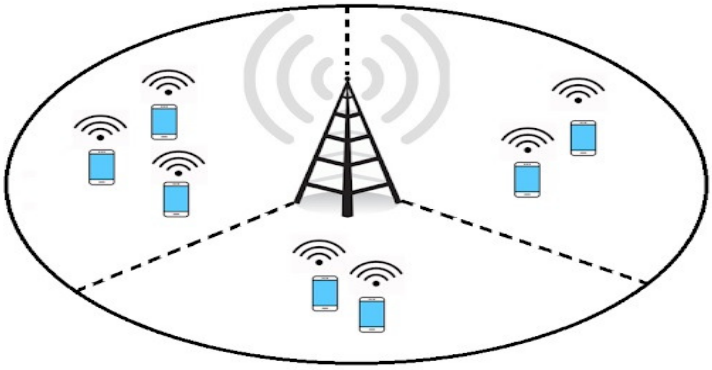

Figura 2.7 – Divisão dos usuários em *clusters* por posição geográfica.

canal e *buffer* de tamanho 2. Essa configuração é composta por $S = (2+1)^3 \cdot (4)^3 = 1728$ estados e ao utilizar o primeiro conjunto de ações (sem repetição de modos) se tem $N_a = 3 \cdot 3 \cdot 4 + 1 + 3! \cdot ((4+1)/2)! = 397$ ações. A matriz $\mathbf{P}$ terá portanto um total de 1.185.435.648 elementos. Ao dividir os usuários em *Clusters* de 3 ou 2 unidades, atende-se a uma maior quantidade de clientes, com aumento linear do custo computacional.

3 Simulações e Resultados

Neste capítulo, apresenta-se e discute-se os resultados obtidos com a aplicação dos algoritmos de alocação de recursos baseados em aprendizagem por reforço em um sistema de comunicação multiportadoras com transmissão via ondas milimétricas. Para tal, foi implementado em Matlab um simulador para o sistema de comunicação que se baseia na tecnologia LTE-OFDM. A frequência da portadora considerada é de 6 GHz e considera-se uma distância fixa de 60 metros entre transmissor e receptor. Para as primeiras simulações utiliza-se 20 MHz de banda com tamanho de pacote igual a 105 bytes. Com 5 usuários isso nos permite uma taxa de codificação (de símbolos em pacotes) de 2 e capacidade de transmissão $2i$ onde i é o número de bits transmitidos pelo modo: 2 bits/símbolo no 4-QAM, 4 bits/símbolo no 16-QAM, 6 bits/símbolo no 64-QAM e 8 bits/símbolo no 256-QAM. Quando valores diferentes dos parâmetros são adotados, cada subseção especifica os valores utilizados.

São apresentados neste capítulo os resultados de uma série de simulações da aplicação dos algoritmos de alocação de recurso ao sistema de comunicação considerado utilizando diferentes cenários com o objetivo de:

- Validar a modelagem Markoviana do sistema por meio do cálculo dos parâmetros de QoS discutidos na seção 2.5 e simulação do ambiente com K, L, M, J, V, C_h tamanho máximo do *cluster* fixados usando dados reais. Seção 3.1;
- Comparar diferentes formas de atendimento nos Cenários 1 e 2 (TDM e OFDM) e diferentes funções de recompensa para uma faixa ampla de λ e fixados K, L, M, J, V, C_h e tamanho máximo do *cluster*. Seção 3.2;
- Evidenciar a diferença no desempenho de 5 funções de recompensa diferentes, para uma faixa ampla de λ e fixados K, L, M, J, V, C_h e tamanho máximo do *cluster*. Seção 3.3;
- Evidenciar a diferença no desempenho médio de 2 funções de recompensa diferentes, para uma faixa restrita de λ, fixados L, M, J, V, C_h e variando a quantidade de usuários K e mantendo o tamanho máximo do *cluster* igual a K. Seção 3.4;
- Evidenciar a diferença no desempenho médio de 5 funções de recompensa diferentes, para uma faixa ampla de λ, fixados L, J, V, C_h, tamanho máximo do *cluster* e variando a quantidade de usuários e canais $K = M$. Seção 3.5;

- Evidenciar a diferença no desempenho médio de 5 funções de recompensa diferentes, para uma faixa ampla de λ, fixados K, M, J, V, C_h, tamanho máximo do *cluster* e variando o tamanho máximo do *buffer* L. Seção 3.6;

- Evidenciar a diferença no desempenho médio de 5 funções de recompensa diferentes, para uma faixa ampla de λ, fixados K, L, J, V, C_h, tamanho máximo do *cluster* e variando a quantidade de canais M. Seção 3.7;

- Evidenciar a diferença no desempenho médio de 5 funções de recompensa diferentes, para uma faixa ampla de $\lambda < 3L$ e $\lambda > L$, fixados K, L, M, J, V, tamanho máximo do *cluster* e variando a quantidade de estados de canais C_h. Seção 3.8;

- Evidenciar a diferença no desempenho de 5 funções de recompensa diferentes, para uma faixa ampla de λ e fixados K, L, M, J, V, C_h e tamanho máximo do *cluster* com $M = K \cdot 16$. Seção 3.9;

- Evidenciar a diferença no desempenho de 5 funções de recompensa diferentes, utilizando dados reais de 5 usuários agrupados em pacotes em uma simulação adaptativa síncrona utilizando modelo poissoniano para chegada de dados e fixados K, L, M, J, V, C_h e tamanho máximo do *cluster*. Seção 3.10;

- Evidenciar a diferença no desempenho de 5 funções de recompensa diferentes, utilizando dados reais de 5 usuários agrupados em pacotes em uma simulação adaptativa assíncrona utilizando modelo poissoniano para chegada de dados, onde se respeita o tempo de processamento necessário antes de considerar um novo resultado adapatado. Fixados K, L, M, J, V, C_h e tamanho máximo do *cluster*. Seção 3.11; e

- Evidenciar a diferença no desempenho de 5 funções de recompensa diferentes, utilizando dados reais de 5 usuários agrupados em pacotes em uma simulação adaptativa assíncrona utilizando rede DQN e considerando modelo poissoniano para chegada de dados, fixados K, L, M, J, V, C_h e tamanho máximo do *cluster*. Seção 3.12.

- Evidenciar a diferença no desempenho de diferentes técnicas de alocação de potência, utilizando dados reais de 5 usuários agrupados em pacotes em uma simulação adaptativa assíncrona utilizando rede DQN , fixados K, L, M, J, V, C_h e tamanho máximo do *cluster*. Seção 3.13.

3.1 Validação do Modelo Markoviano

Antes de se aplicar o Modelo Markoviano apresentado nas seções seguintes em um contexto de algoritmo de Aprendizado por Reforço, apresenta-se nesta seção uma avaliação do seu desempenho em descrever o sistema de comunicação considerado. Assim, para avaliar a eficiência do modelo Markoviano em descrever o comportamento de um sistema CP-OFDM com séries reais de tráfego, a formulação da teoria das filas foi utilizada para se estimar parâmetros de QoS do sistema. Os valores dos parâmetros de QoS do sistema preditos pelo modelo de Markoviano são comparados aos fornecidos via simulação do sistema de comunicação. Para capturar o comportamento do ambiente assim como validar o Modelo Markoviano para o sistema de comunicação, considera-se nesta seção que o agente não realiza nenhuma ação.

O sistema escolhido tem K=3 usuários, comprimento do *buffer* $L = 2$ e o número de estados do canal $C_h = 2$. As séries de tráfego consideradas nas simulações são referentes aos dados apresentados em (MAWI, 2019). Essas séries de tráfego MAWI representam dados diários de fluxos de tráfego de diferentes aplicativos coletados no *backbone* da Internet do grupo de pesquisa *Measurement and Analysis on the WIDE Internet* (MAWI, 2019). As taxas médias de transmissão desses traços (traces) de tráfego são calculadas adaptativamente e inseridas no modelo Markoviano proposto como a taxa média de chegada do processo de Poisson (2.5) de modo a se obter a matriz de probabilidades de transição (2.14).

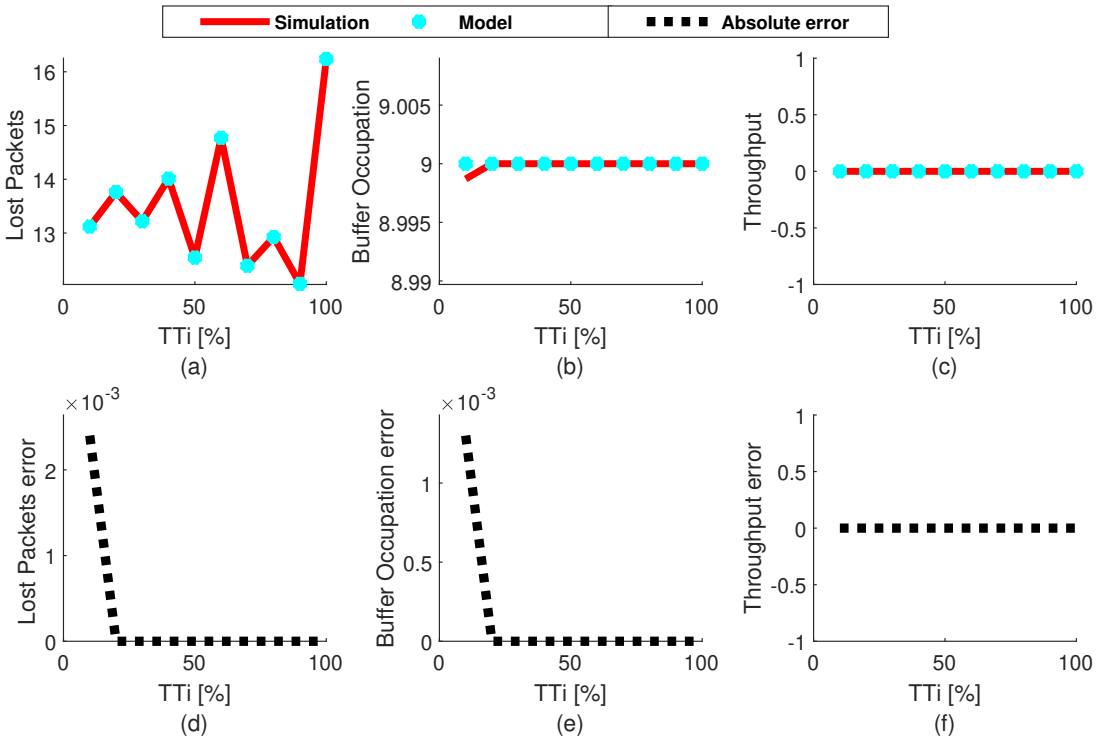

Figura 3.1 – Validação do modelo Markoviano

A Figura (3.1) mostra o número de pacotes perdidos em (a), ocupação do *buffer* em (b) e Vazão em (c). Os resultados descritos na Figura (3.1) são referentes aos valores obtidos com as expressões analíticas do modelo Markoviano (equações (2.16), (2.19), (2.20)) e via simulação do sistema de comunicação OFDM. As curvas da Figura (3.1)-d, (3.1)-e e (3.1)-f apresentam o erro absoluto. Observa-se que os resultados do erro absoluto entre os valores dos parâmetros fornecidos pelas simulações e pelas equações (2.16), (2.19), (2.20) são baixos e indicam que a escolha de um modelo Markoviano para a modelagem adaptativa do comportamento de enfileiramento de comunicação é válida, mesmo ao se considerar séries de tráfego reais nas simulações.

3.2 Comparando o desempenho utilizando as funções de recompensa e atendimento TDM e OFDM

Para realizar a comparação do desempenho obtido pelo sistema de comunicação utilizando o algoritmo de alocação de recurso baseado em aprendizado por reforço considerando as funções de recompensa 2.23 e 2.28 e as formas de atendimento TDM e OFDM foram simulados 25 segundos (equivalente a 50000 *TTI's*). O sistema de comunicação escolhido tem K=5 usuários divididos em 2 *clusters* distintos de tamanho 3 e 2 usuários, tamanho de *buffer* $L = 2$, quantidade de canais $M = 5$, quantidade de modos de transmissão $J = 4$ (4QAM, 16QAM, 64QAM e 256QAM) e quantidade de estados de canal $C_h = 4$. Para este caso, foi aplicada a estratégia em que o mesmo modo não pode ser utilizado simultaneamente para diferentes usuários por prover melhores resultados de desempenho para proposta de Zhu et al. (2018). Assim, neste caso, a quantidade de ações é dada pela Equação 2.1. A taxa de chegada de dados durante a simulação foi variada entre 0.1 e 14.5 pacotes/TTI. Para a taxa de desconto utilizou-se $\gamma = 0.9$.

O ganho médio do canal ρ_m é obtido pela média de 1000 amostras do modelo TDL-B (3GPP, 2018). Durante a simulação, para ganhos de canal menores que $0.01\frac{\rho_m}{10^{PL/10}}$ para os dispositivos de usuários, considera-se condição de inoperância, ou seja, o atendimento à demanda do usuário não é realizado. Como a largura de banda para o sistema considerado é de 20MHz, o valor da potência do ruído é de $10^{-13}W$ para uma densidade de potência do ruído igual a $-100.5dBm$. Para 6 GHz, velocidade do receptor de 1.5 m/s e transmissor com posição fixa tem-se $f_D = \frac{6.10^9}{3.10^8}.1.5 = 30Hz$ para o máximo efeito Doppler, conforme Rappaport et al. (1996).

Foram obtidos resultados para 2 cenários de simulação, conforme mencionado no Capítulo 2. O cenário 1 contempla ações individuais (TDM), onde são comparadas as duas funções de utilidade para esse cenário. O cenário 2 contempla as ações completas incluindo as ações OFDM além das ações simples (TDM) do Cenário 1. Para diferenciar os resultados dos diferentes cenários e funções são adotadas as seguintes nomenclaturas:

C1-Zhu, C1-Prop, C2-Zhu e C2-Prop.

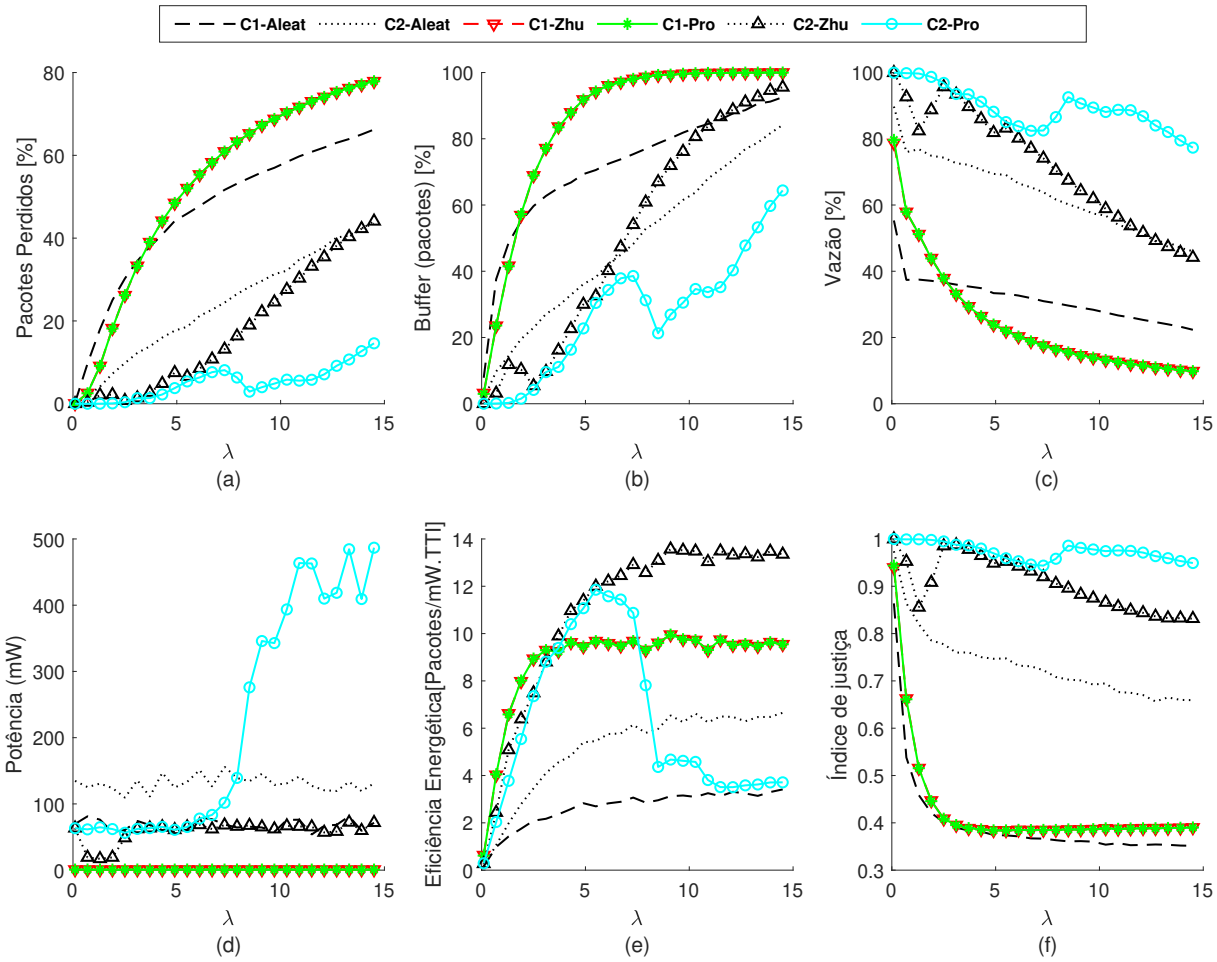

Figura 3.2 – Parâmetros de QoS versus taxa média de geração de pacotes para banda de 20 MHz e 2 cenários: (a) Pacotes Perdidos, (b) ocupação do *buffer*, (c) Fluxo de pacotes, (d) Potência, (e) Eficiência energética e (f) Índice de justiça

A legenda C1-Zhu nas figuras diz respeito à função de recompensa utilizada por Zhu et al. (2018) (Eq. 2.27) para ações que multiplexam o atendimento no tempo (Ações TDM ou Cenário 1).

A legenda C1-Prop nas figuras diz respeito à configuração do sistema que multiplexa o atendimento de usuários no tempo (Ações TDM ou Cenário 1) mas onde se considera a função proposta de recompensa dada por (Eq. 2.28). Ou seja, esta função de recompensa se difere da de Zhu et al. (2018) pela substituição do tamanho da fila no *buffer* após a ação pela quantidade de pacotes perdidos.

A legenda C2-Zhu diz respeito à função de recompensa utilizada por Zhu et al. (2018) (Eq. 2.27) para ações que multiplexam o atendimento tanto no tempo quanto na frequência (Ações TDM e OFDM ou Cenário 2).

A legenda C2-Prop diz respeito à função de recompensa proposta no trabalho (Eq.

2.28) para ações que multiplexam o atendimento tanto no tempo quanto na frequência (Ações TDM e OFDM ou Cenário 2).

Além da aplicação de escalonamento de recursos via aprendizagem por reforço, considerou-se também nas simulações uma seleção aleatória da alocação de recursos tanto para o Cenário 1 como para o Cenário 2 denominadas C1-Aleat e C2-Aleat nas figuras. Basicamente, escolhe-se uma ação aleatória do conjunto de ações TDM para o C1-Aleat e (TDM+OFDM) para o C2-Aleat.

No cenário 1, para valores de $3.7 < \lambda < 14.5$ onde λ é a taxa média de chegada de pacotes para os usuários, o algoritmo aleatório provê melhores resultados em termos de pacotes perdidos, ocupação do *buffer* e fluxo de pacotes como pode-se ver nas figuras 3.2 (a),(b) e (c) respectivamente.

Porém, piores resultados em termos de consumo de potência, eficiência energética e índice de justiça visualizados nas figuras 3.2 (d),(e) e (f) respectivamente.

Ao visualizar a Figura 3.2 (c) percebe-se como os algoritmos propostos consomem menos potência comparado ao algoritmo aleatório no Cenário 1. O cálculo da eficiência energética utilizado é dado pela Equação (1.11).

Ainda, avalia-se o índice de justiça (*fairness*) das soluções. O cálculo de índice de justiça de vazão é definido pela seguinte equação (JAIN; DURRESI; BABIC, 1999):

$$fairness = \frac{1}{K}\frac{(\sum_{k=1}^{K}||Nf_k||)^2}{\sum_{k=1}^{K}||Nf_k||^2} \tag{3.1}$$

onde Nf_k é o fluxo normalizado do usuário k e K o número de usuários. A base para normalização utilizada é a demanda de pacotes para o usuário no TTI corrente somada a quantidade de pacotes no *buffer* do usuário k.

No cenário 2, observa-se maior diferença entre os algoritmos de aprendizado por reforço e sua vantagem quando comparados com a solução aleatória. Visualiza-se uma maior diferença no comportamento das duas soluções com aprendizado por reforço a partir de um determinado λ maior que 6.1. Porém, para faixa de $0.1 < \lambda < 1.9$ percebe-se diferença nas soluções ao se monitorar o tamanho do *buffer* (proposta de Zhu et al. (2018)) e a quantidade de pacotes perdidos.

Em média, para o Cenário 2, menos pacotes são perdidos para faixa de valores de λ, há menor ocupação do *buffer* e mais pacotes são transmitidos ao utilizar a solução proposta como observado nas figura 3.2 (a), (b) e (c) respectivamente.

Para o solução proposta (curva C2-Prop) uma maior eficiência energética é obtida por TTI ao custo de alguns pacotes a mais perdidos como pode ser visualizado na Figura 3.2 (d) para faixa de λ até 6.1. Para valores de $\lambda > 6.1$, ao utilizar a solução proposta, maior potência é alocada e obtém-se maior índice de justiça como observado nas figura 3.2

(d) e (e) respectivamente.

No cenário 2 e para o intervalo de $0.1 < \lambda < 1.9$ tem-se um comportamento que motiva a utilização do *buffer* na função de recompensa. A solução proposta consume relativamente mais potência que a função de Zhu (Figura 3.2 (d)), mas obtém-se taxa de pacotes perdidos muito próximas e maiores taxas de fluxo (figuras 3.2 (a) e (c)). Atribui-se esse comportamento ao tamanho relativo do *buffer* em comparação com a taxa média de chegada de pacotes λ. A ocupação do *buffer*, portanto, é uma métrica que funciona bem para modelar o comportamento do agente já que há pouca chance de se perder pacotes.

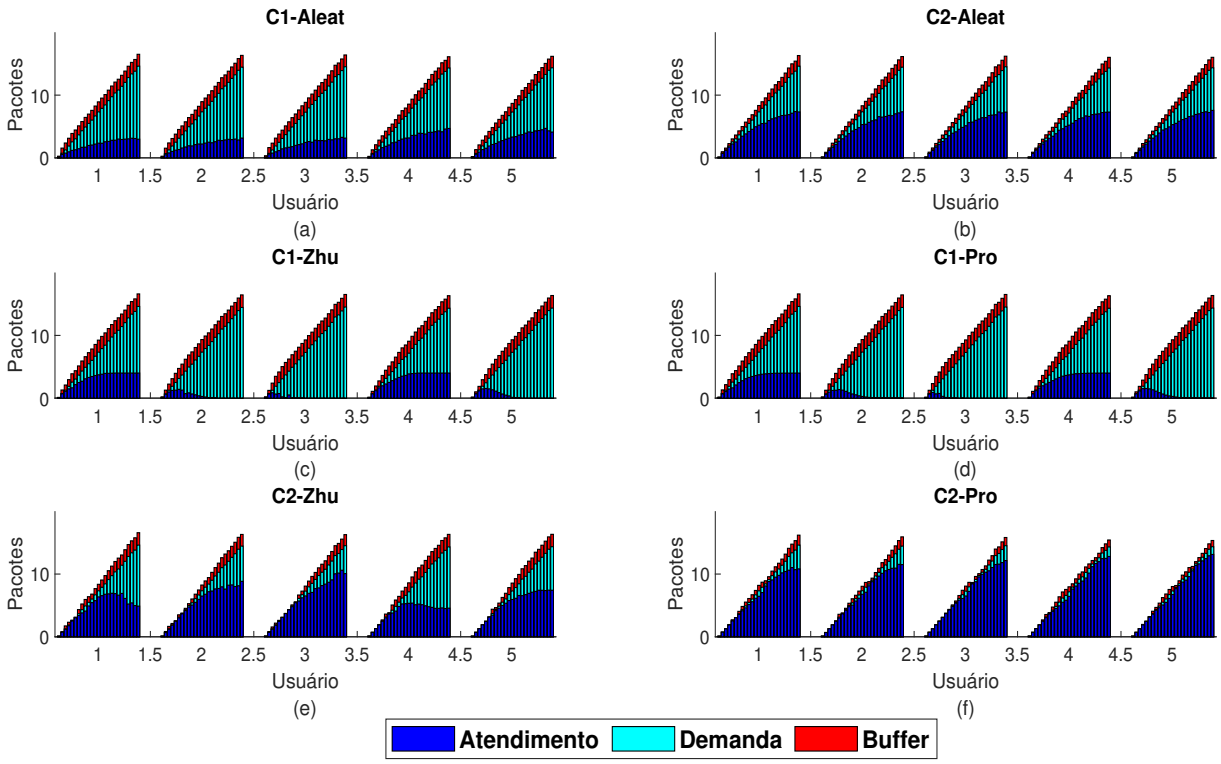

Figura 3.3 – Balanço dos pacotes para cada função de recompensa e cenário.

Na Figura 3.3, observa-se a distribuição de recursos entre os usuários. Os gráficos são separados por estratégia de escalonamento. Percebe-se a vantagem de utilizar a proposta no cenário 2 no que diz respeito a: Pacotes perdidos, pacotes transmitidos, utilização do *buffer* e índice de justiça de vazão.

Quando se deseja o melhor desempenho possível para o sistema, avalia-se questões como menor espera dos pacotes na fila (menor tamanho de fila no *buffer*), menor perda de pacotes e maiores taxas. O algoritmo proposto se mostrou superior nesses requisitos para taxa média de chegada acima de 4.1 pacotes por *TTI*. Como era de se esperar, os algoritmos baseados em aprendizagem por reforço foram mais eficientes do que a alocação aleatória de recursos para o cenário 2 em relação aos parâmetros de desempenho analisados, e a definição da função de recompensa pode mudar o comportamento do agente.

Ao se comparar os resultados de tamanho do *buffer* e quantidade de pacotes perdidos é difícil dizer qual parâmetro seria o mais adequado a ser analisado para se determinar qual algoritmo é mais eficiente. Nas seções futuras apresenta-se novas funções de recompensa que utilizam esses dois parâmetros em conjunto para melhorar o desempenho do algoritmo de alocação de recursos.

Segundo os critérios estabelecidos de eficiência energética e parâmetros como fluxo, perda de pacotes, tamanho da fila no *buffer*, o uso de OFDM+TDM (cenário 2) mostra maior vantagem quando comparado ao modo TDM simples (cenário 1), apesar da maior complexidade do cenário 2. As diferenças nos indicadores de QoS entre os 2 cenários ocorrem quando as taxas de chegada são grandes $\lambda > 0.5$ (pelo menos em comparação com o tamanho da fila no *buffer*). Para taxas $\lambda < 0.5$ a função de recompensa (de Zhu et al. (2018) e a proposta) no cenário 2 aponta para políticas que não utilizam ações OFDM. Há semelhança dos gráficos de Qos para $\lambda < 0.5$ para cenários 1 e 2. São situações em que não se perde pacotes mesmo atendendo um usuário de cada vez devido à baixa taxa de chegada de dados. À medida que a taxa de chegada aumenta, nota-se as curvas dos cenários 1 e 2 se afastarem.

Observa-se grande diferença no consumo de potência dos dois cenários. Atribui-se essa diferença ao uso de apenas um canal por TTI no cenário 1 e também ao fato de se ter 3 canais disponíveis. Ocorre que o atendimento TDM (cenário 1) irá escolher sempre o melhor canal entre 3 para o atendimento. Isso dá robustez estatística ao ganho do canal e modifica um pouco a sua distribuição exponencial durante a simulação. Obtém-se características semelhantes para o cenário 2 ao utilizar número de canais M maior que o número de usuários K. Essa estratégia é discutida na Seção 3.9.

Atribui-se aos menores valores de índice de justiça observados nos gráficos do cenário 1 (Figura 3.2 (e) e 3.3) à simetria do problema. Como o valor da função de recompensa ao atender um único usuário é o mesmo para condições iguais de usuário, nada incentiva o agente a intercalar a escolha dos usuários. O uso explícito do índice de justiça na função de recompensa pode contornar esse problema que aparece especialmente nas propostas do cenário 1 (TDM).

3.3 Resultados com diferentes funções de utilidade

Como observou-se na Seção 3.2, o desempenho da solução utilizando aprendizado por reforço é sensível à definição da função de utilidade. Mesmo a função de recompensa proposta promovendo que o algoritmo de alocação de recursos apresente melhores ou igual taxa de transmissão para valores de $\lambda > 3.1$, para valores fora dessa faixa a influência do tamanho da fila no *buffer* se mostrou mais decisiva para minimizar a quantidade de pacotes perdidos. Entende-se portanto, que este efeito não deva ser ignorado. Assim, se faz interessante a investigação de variações da função de utilidade (recompensa).

No Capítulo 2, foram apresentadas propostas para função de utilidade que consideram a combinação do tamanho do *buffer* com a quantidade de pacotes perdidos. Para conveniência, são apresentadas novamente a seguir as diferentes funções de utilidade definidas e comparadas fixando o Cenário 2, ou seja, abrangendo ações TDM e OFDM do agente. Além disso, outro conjunto de ações descrito pela Equação 2.2 é utilizado. O número de ações é menor, o que agiliza a simulação.

Sejam as variáveis $EE(s,a)$, V, j_k, K, $P_k(s,a)$, $Lost_k$, l_k, b_k e λ_k, a eficiência energética, a taxa de codificação, o modo atribuído ao usuário k, o número de usuários, a potência alocada ao usuário k pela ação a quando no estado s, o número de pacotes perdidos pelo usuário k após a ação a, o número de pacotes no *buffer* do usuário k antes da ação a, o número de pacotes que chegaram para o usuário k e a taxa média de chegada de pacotes do usuário k respectivamente. São consideradas neste trabalho, incluindo as propostas, as seguintes funções de utilidades:

$$R_{ZhuQL}(s,a) = \frac{EE(s,a)}{\sum_{k=1}^{K} e^{0.5.l_k}} \tag{3.2}$$

$$R_{Prop1}(s,a) = \frac{EE(s,a)}{\sum_{k=1}^{K} E[e^{0.5.(B_k+Lost_k)}]} \tag{3.3}$$

$$R_{Prop2}(s,a) = \frac{EE(s,a)}{\sum_{k=1}^{K} E[(e^{0.5Lost_k} + B_k)]} \tag{3.4}$$

$$R_{Prop3}(s,a) = \frac{EE(s,a)}{\sum_{k=1}^{K} E[\frac{e^{0.5(l_k+Lost_k)}}{B_k+1}]} \tag{3.5}$$

$$R_{Prop4}(s,a) = \frac{EE(s,a)}{\sum_{k=1}^{K} E[Lost_k x \lambda_k + e^{0.5.(B_k+Lost_k)}]} \tag{3.6}$$

onde $EE(s,a)$ é a eficiência energética definida na equação 1.11, $P_k(s,a)$ a parcela da potência alocada para garantir a BER mínima para o usuário k, $Lost_k$ e B_k a quantidade de pacotes perdidos e o estado do *buffer* após tomar a ação a no estado s do usuário k respectivamente e λ_k a taxa média de chegada de pacotes do usuário k.

Ainda, utiliza-se um algoritmo chamado de "Ação Fixa"como um limite superior para a potência e Fluxo de pacotes. Esse algoritmo utiliza o maior modo de transmissão (256QAM) para todos os canais e intercala de forma aleatória a seleção de usuário por canal.

Os parâmetros da simulação são: K=5 usuários, tamanho de *buffer* $L = 2$, quantidade de canais $M = 5$, quantidade de modos de transmissão $J = 4$ (4QAM, 16QAM, 64QAM e 256QAM) e quantidade de estados de canal $C_h = 2$. As taxas de chegada simuladas são: $\lambda = [0.1, 0.7, 1.3, ..., 14.5]$. Para o fator de desconto utilizou-se $\gamma = 0.9$.

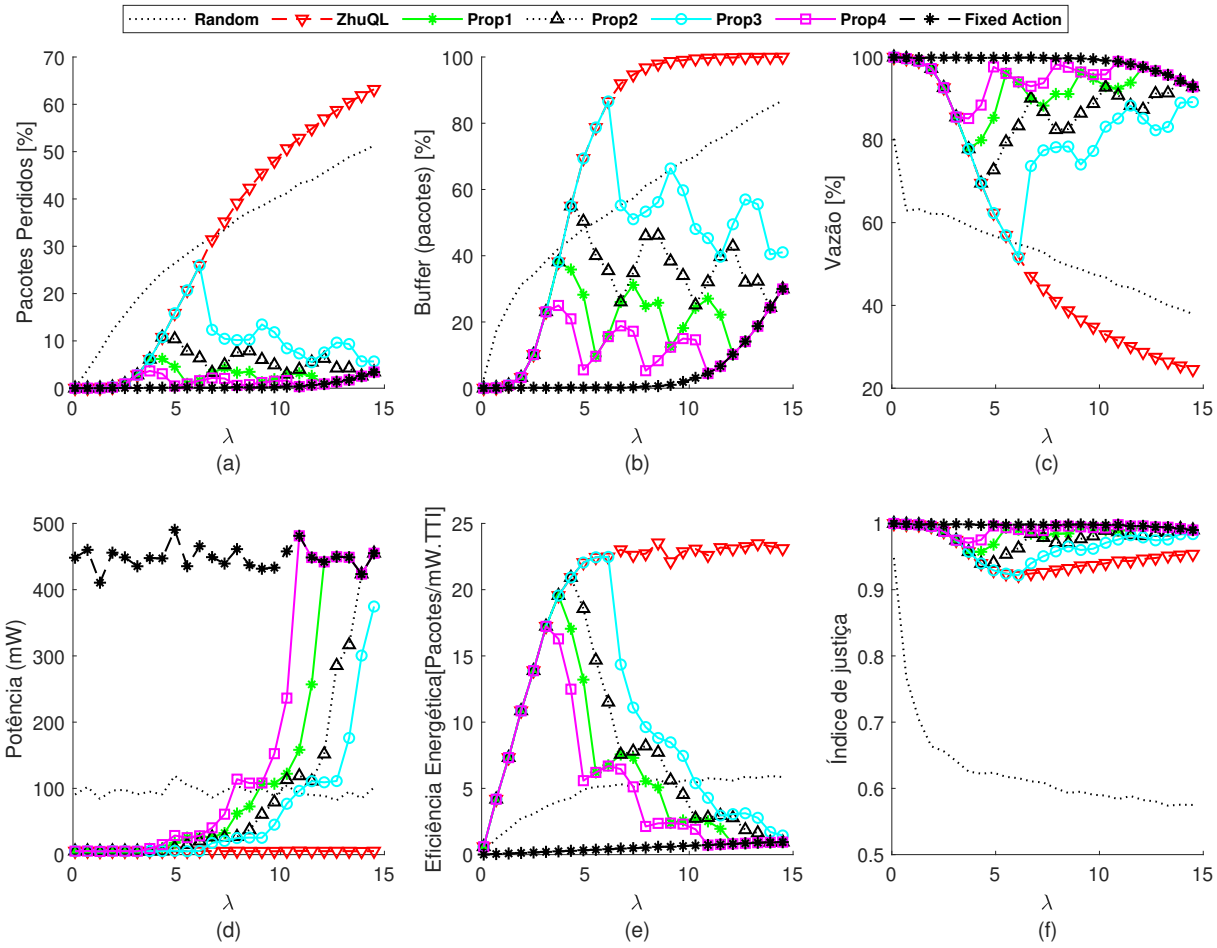

Figura 3.4 – Parâmetros de QoS versus taxa média de geração de pacotes para banda de 20 MHz e várias funções de recompensa: (a) Pacotes Perdidos, (b) ocupação do *buffer*, (c) Fluxo de pacotes, (d) Potência, (e) Eficiência energética e (f) Índice de justiça

Foi adicionada uma estratégia de atendimento que consiste em escolher o modo 256QAM para todos os usuários e alocá-los de forma aleatória nos canais disponíveis. Essa estratégia representa um limiar superior médio para o consumo de potência. Caso o canal esteja abaixo do limiar mínimo ($0.02\rho_m$) a ação é alterada para ociosa nesse canal de modo a manter os níveis de potência mais estáveis para a plotagem.

Nas figuras 3.4 (a), (b), (c), (d), (e) e (f), percebe-se como as propostas se afastam da solução de Zhu et al. (2018) para valores diferentes de λ. Essa diferença começa a ocorrer em $\lambda = 3.1$ para a proposta 4. A proposta que tem a variação em relação a de Zhu et al. (2018) mais tardia é a 3 que se distancia deste algoritmo em $\lambda = 4.3$. Resumidamente, tem-se dois interesses em conflito: a transmissão do máximo de pacotes possíveis por TTI (que está relacionada a uma menor ocupação do *buffer*, menor perda de pacote e maior índice de justiça de vazão) e a eficiência energética (que está de acordo com baixo consumo de potência durante a transmissão).

Nota-se, ao comparar as figuras desta seção com as da seção anterior para o Cenário 2 (Seção 3.2) que utiliza outro conjunto de ações, que o algoritmo de Zhu et al. (2018) apresentou pior desempenho para o conjunto de ações considerado nesta seção, em termos de pacotes perdidos, ocupação do *buffer*, fluxo de pacotes e índice de justiça com o novo conjunto de ações. Atribui-se esse comportamento a impossibilidade de escolher o modo 4QAM para mais de um usuário na Seção 3.2.

Entre os algoritmos de aprendizagem por reforço aplicados a alocação de recursos considerados nesta seção, a solução de Zhu et al. (2018) é aquela que apresenta maior quantidade de pacotes perdidos, maior ocupação do *buffer*, menor fluxo de pacotes, menor consumo de potência, maior eficiência energética e menor índice de justiça de vazão como obeserva-se na Figura 3.4.

A proposta 4 é justamente o contrário, i.e, entre as soluções que utilizam aprendizado por reforço apresenta menor quantidade de pacotes perdidos, menor ocupação do *buffer*, maior fluxo de pacotes, maior consumo de potência, menor eficiência energética e maior índice de justiça de vazão como obeserva-se na Figura 3.4.

As propostas 1, 2 e 3 apresentam resultados intermediárias entre as duas. A proposta 1 está mais próxima da proposta 4 e a proposta 3 está mais próxima da proposta de Zhu et al. (2018). A proposta 2 tem comportamento intermediário entre as propostas que utilizam aprendizado por reforço.

Nas Figuras 3.5, 3.6, 3.7 e 3.8, pode-se observar a diferença entre as funções Q (que é o espaço de busca) para os algoritmos considerados. O sistema de comunicação simulado para as figuras em questão apresenta K=3 usuários, $L = 2$ tamanhos de *buffer*, $M = 2$ canais, $J = 3$ modos e $C_h = 1$ estado de canal.

As funções Q de valor de ação considerando a solução de Zhu e a Proposta 1 se aproximam quando a taxa de chegada é cerca de 25% do tamanho do *buffer* (Figuras 3.5 e 3.6). Nesse caso, a probabilidade de perder pacotes é baixa pois exige que o *buffer* esteja cheio e a ação escolhida não seja suficiente para atender apenas os pacotes que chegaram, o que só ocorreria (em média) se o agente não escolhesse nenhum modo de transmissão. É interessante observar que a repetição de picos ao longo do eixo das ações corresponde

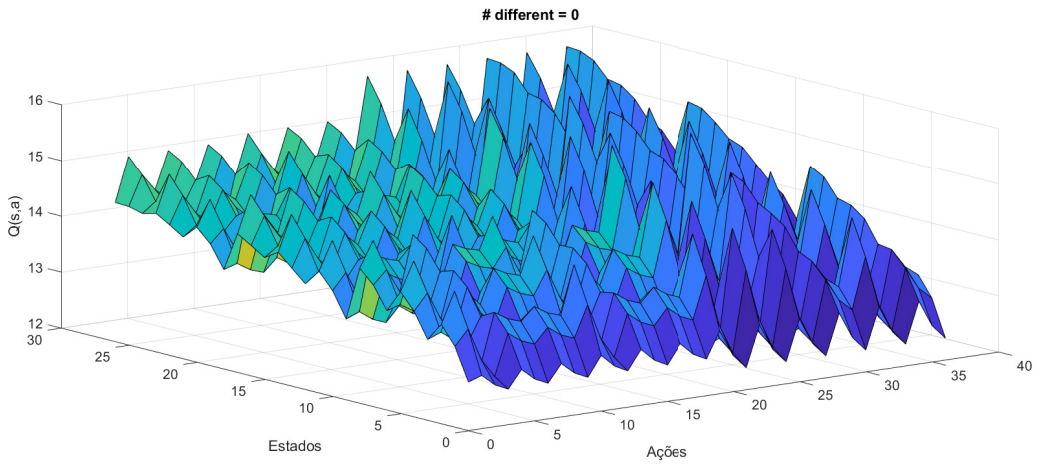

Figura 3.5 – Valor da função Q usando recompensa de Zhu et al. (2018) para taxa média de chegada de 0.5 pacotes por *TTI*

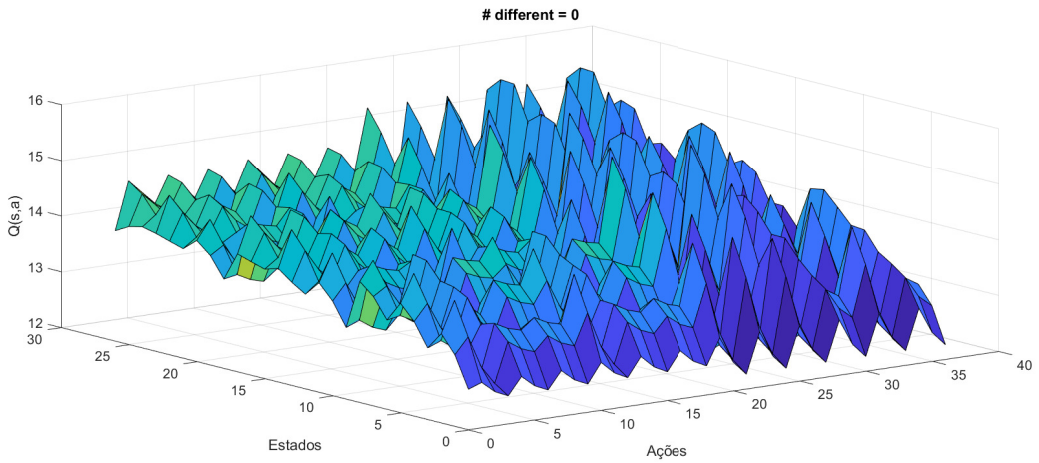

Figura 3.6 – Valor da função Q usando recompensa da Proposta 1 para taxa média de chegada de 0.5 pacotes por *TTI*

às diferentes formas de alocar 3 usuários em 2 canais, ou seja, a permutação de três dois a dois, ou 3!. Os dois grupos correspondem às ações TDM (um dos 2 modos é nulo) e OFDM (os dois modos são não nulos).

Já com uma taxa de chegada maior, por exemplo para 9 pacotes por TTI, a ocupação do *buffer* é menos crítica. Dessa forma, a granularidade das possibilidades de chegada de pacotes não é bem modelada como se observa pela simetria no gráfico da Figura 3.7. O algoritmo *Q-learning* com modelo e iteração de política, inclusive, converge mais lentamente, uma vez que a melhor ação para cada estado pode conter ambiguidades. Em outras palavras, há maior chance das soluções do algoritmo se situarem em máximos locais. Na Figura 3.8, visualiza-se uma superfície mais propícia de se buscar por máximos

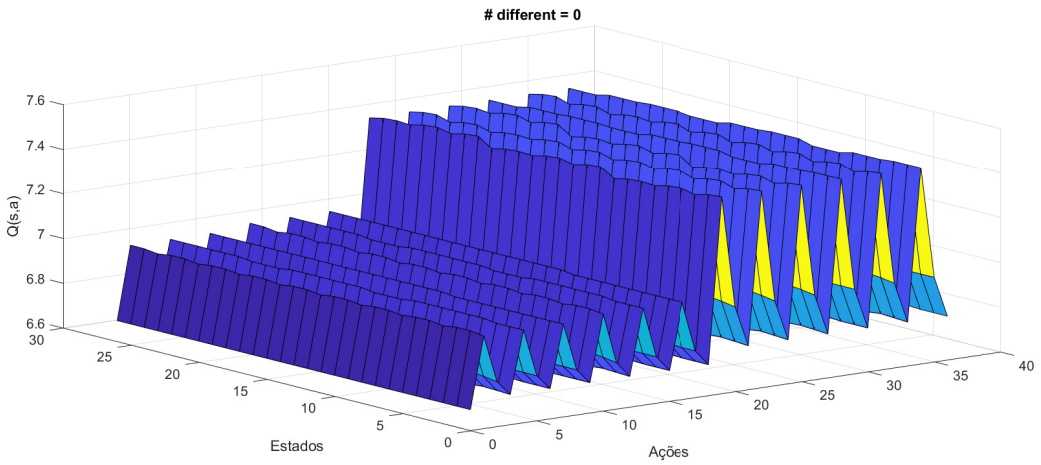

Figura 3.7 – Valor da função Q usando recompensa de Zhu et al. (2018) para taxa média de chegada de 9 pacotes por *TTI*

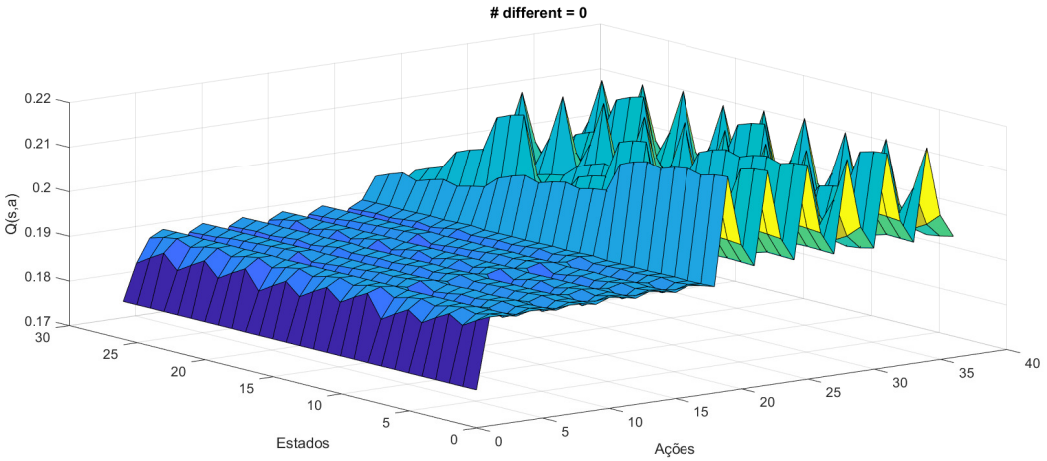

Figura 3.8 – Valor da função Q usando recompensa da Proposta 1 para taxa média de chegada de 9 pacotes por *TTI*

o que agiliza a convergência do algoritmo.

Na Figura 3.9 visualiza-se a função Q quando utilizando a proposta 4. Há uma redução no seu valor absoluto em cerca de 10 vezes em relação à Figura 3.8 devido à divisão por $\lambda = 9$ na função de recompensa da proposta 4 (Equação 3.6). A proposta 4 apresentou os melhores resultados para fluxo (taxa de transmissão) de pacotes e para a taxa média de pacotes perdidos.

Percebe-se pela Figura 3.10 como as funções de recompensa das propostas de Zhu et al. (2018), 2 e 3 permitem atendimento mais eficiente dos usuários utilizando mais etapas. O estado 1 corresponde ao estado de *buffer* vazio para todos os usuários e os

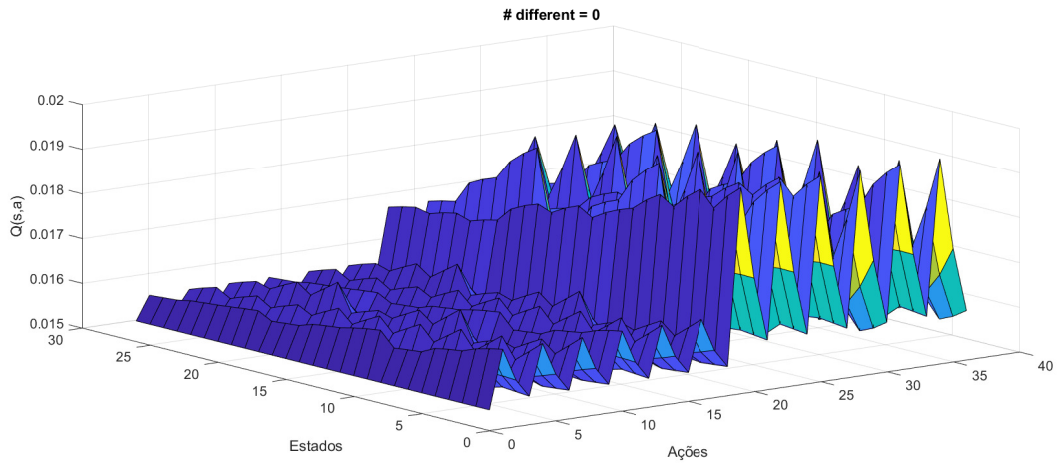

Figura 3.9 – Valor da função Q ao se utilizar a Proposta 4 para taxa média de chegada de 9 pacotes por *TTI*

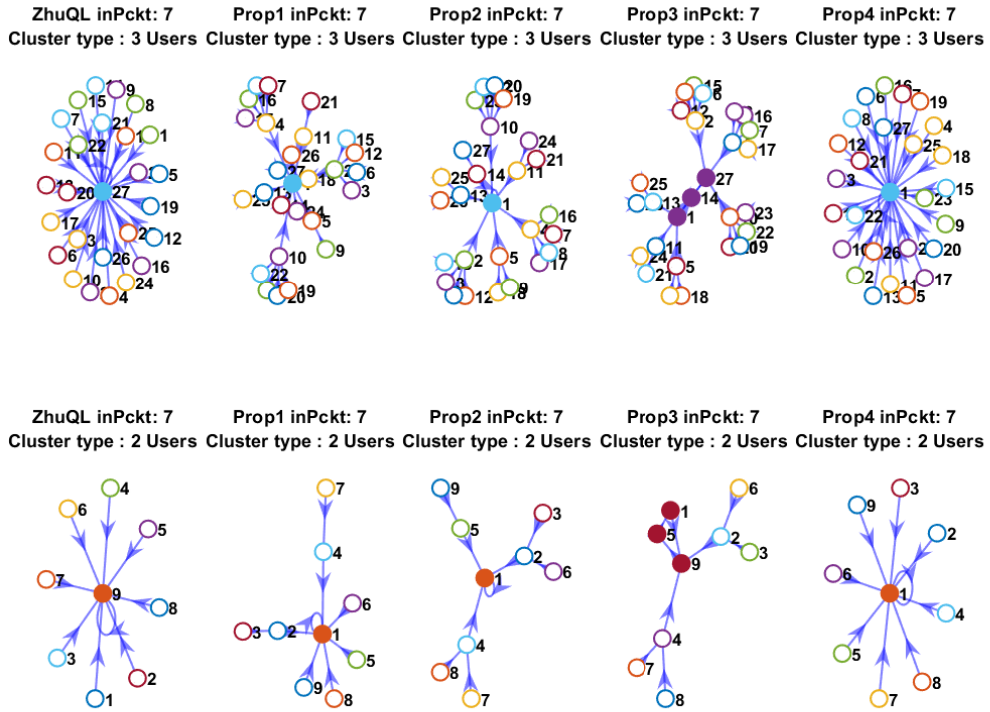

Figura 3.10 – Cadeia de Markov para os estados do *buffer* e estado fixo de canal quando o agente segue as políticas encontradas para os diferentes *cluster* e propostas e uma chegada de 7 pacotes por *TTI*

estados 9 e 27 correspondem ao estados de *buffer* cheio para o *cluster* de 2 e 3 usuários respectivamente.

Nota-se que a proposta 4 provê melhores resultados em termos de vazão do que as demais analisadas. Já em termos de eficiência energética a proposta 3 apresenta melhores resultados. Apesar da solução de (Zhu et al., 2018) ser superior para $\lambda > 5$ em termos de eficiência energética, a quantidade de pacotes perdidos é muito grande, o que pode exigir retransmissões. Isto, na prática, pode acabar implicando em maior gasto energético.

3.4 Aumentando a quantidade de usuários por *cluster* mas mantendo a quantidade de canais fixa

Com objetivo de explorar os limites do problema e avaliar o desempenho do algoritmo em várias situações, considera-se nessa seção que o número máximo de usuários por *cluster* possa ser até 6 de forma a utilizar *cluster* único. Utilizou-se 3 taxas médias de chegada de pacotes $\lambda = [3, 4, 5]$ e calculou-se a média dos valores encontrados (pacotes perdidos, tamanho do buffer, pacotes transmitidos, potência média, eficiência energética e índice de justiça) para a plotagem desses resultados em função do número de usuários. Simulou-se com $K = 1$ até 6 usuários utilizando M =2 canais e $J = 3$ modos de transmissão: BPSK, 4QAM e 8QAM. Considera-se $C_h = 2$ estados de canal e $L = 2$ para o tamanho do *buffer*. Para essa simulação utilizou-se as ações do cenário 2, que contemplam ações TDM e OFDM. As ações OFDM escolhidas foram com repetição de modo que o número de ações é dado pela equação 2.2. Nesta seção, o algoritmo em questão é a proposta 1 referente à equação 3.3 que leva em consideração tanto a ocupação do *buffer* como a quantidade de pacotes perdidos.

Com as configurações mencionadas e 6 usuários tem-se um total de $N_s = (2+1)^6 \cdot (2)^2 = 2916$ estados e $N_a = (6 \cdot 5) \cdot (3) + 1 + 6 \cdot 2 \cdot 3 = 127$ Ações. A matriz $\mathbf{P}$ terá portanto um total de 1.079.888.112 elementos. Mesmo com precisão simples, essa matriz ocupa cerca de 4 GB de memória RAM. Com 7 usuários e mantidos o número de estados de canal e número de canais já seriam necessários 48 GB ($8748 \cdot 8748 \cdot 169$ elementos).

Observa-se pela Figura (3.11) que, para a quantidade de usuários considerada, a proposta se destaca quanto à perda de pacotes (a). O tamanho médio da fila no *buffer* também é igual ou reduzido para o caso da proposta (b) em comparação aos outros algoritmos, a não ser pelo algoritmo "Ação Fixa".

Pode-se perceber que o fluxo de pacotes está limitado à quantidade de canais, mas que a proposta aproveita melhor esses canais por fazer com o que o sistema atinga uma maior taxa de transmissão de pacotes (c). Há diminuição de desempenho do algoritmo proposto a partir de 3 usuários, assim como para o algoritmo de Zhu. Quando se utiliza uma quantidade de canais menor que a quantidade de usuários uma política fixa se mostrou mais interessante em termos de vazão (Figura 3.11 (c)) mesmo com maior potência alocada e menor eficiência energética (Figuras 3.11 (d) e (e)) .

Na Figura 3.11 (d) nota-se que a potência alocada para os usuários pela proposta é bem maior, mas A partir de 3 usuários as duas propostas são praticamente equivalentes quanto a eficiência energética (Figura 3.11 (e)) mas o algoritmo proposto leva vantagem quanto à quantidade de pacotes perdidos, ocupação do *buffer* e vazão para as diferentes quantidade de usuários consideradas (Figuras 3.11 (a), (b) e (c)).

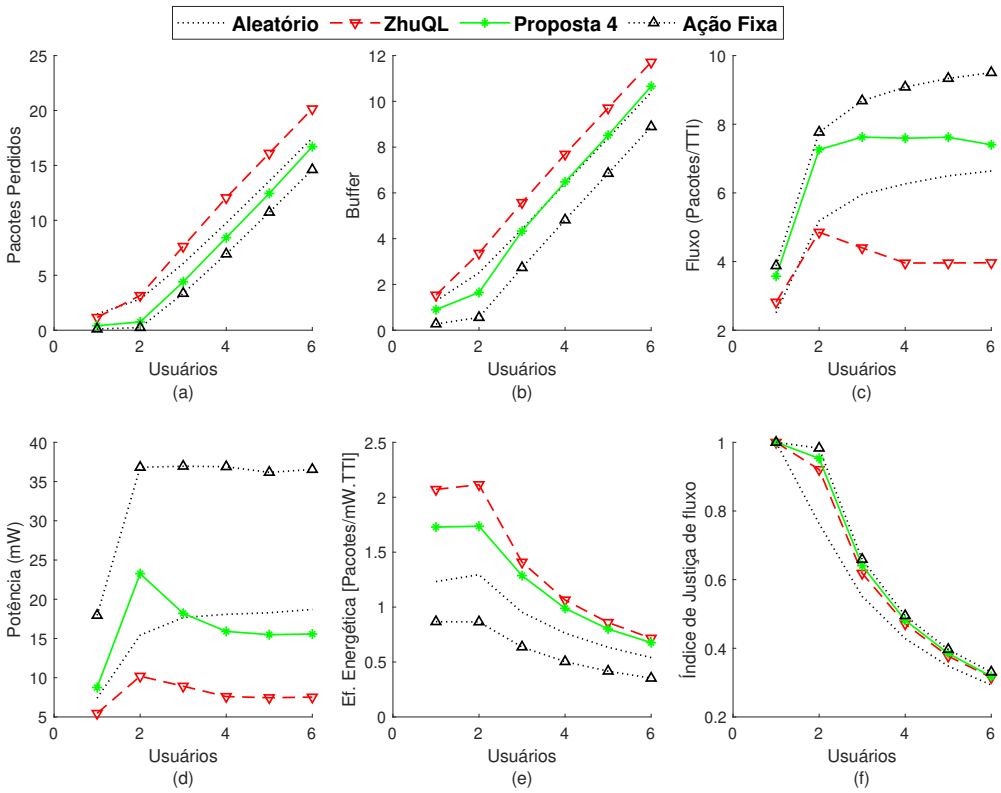

Figura 3.11 – Parâmetros de QoS versus número de usuários com aumento do tamanho do *cluster* de usuários: (a) Pacotes Perdidos, (b) ocupação do *buffer*, (c) Fluxo de pacotes, (d) Potência, (e) Eficiência energética e (f) Índice de justiça

Os valores obtidos para o índice de justiça são similares para todos os algoritmos utilizados nas simulações (Figura 3.11 (f)). Observa-se como o índice de justiça cai à medida que o número de usuários aumenta. É natural, uma vez que há a limitação de atender $M = 2$ usuários a cada TTI.

Na Figura 3.12 nota-se a quantidade média de pacotes que chegaram, que foram atendidos e que foram armazenados no *buffer* para cada uma das 3 taxas médias de chegada ($\lambda = 3$, 4 e 5) e para quantidades diferentes de usuários (1 até 6) mantendo-se 2 canais. Nota-se que as soluções utilizando aprendizado por reforço apresentam resultados competitivos com a solução de ação fixa em termos de vazão para um número de até 3 usuários, sendo a proposta ainda superior a solução de Zhu. Para quantidade de usuários maiores ou iguais a 4, a Ação Fixa leva vantagem sobre as demais soluções em termos de vazão. A proposta se mostra superior a solução aleatória para qualquer quantidade de usuários. Vale ressaltar que a Ação Fixa apresenta os piores resultados de eficiência energética (Figura 3.11 (e)).

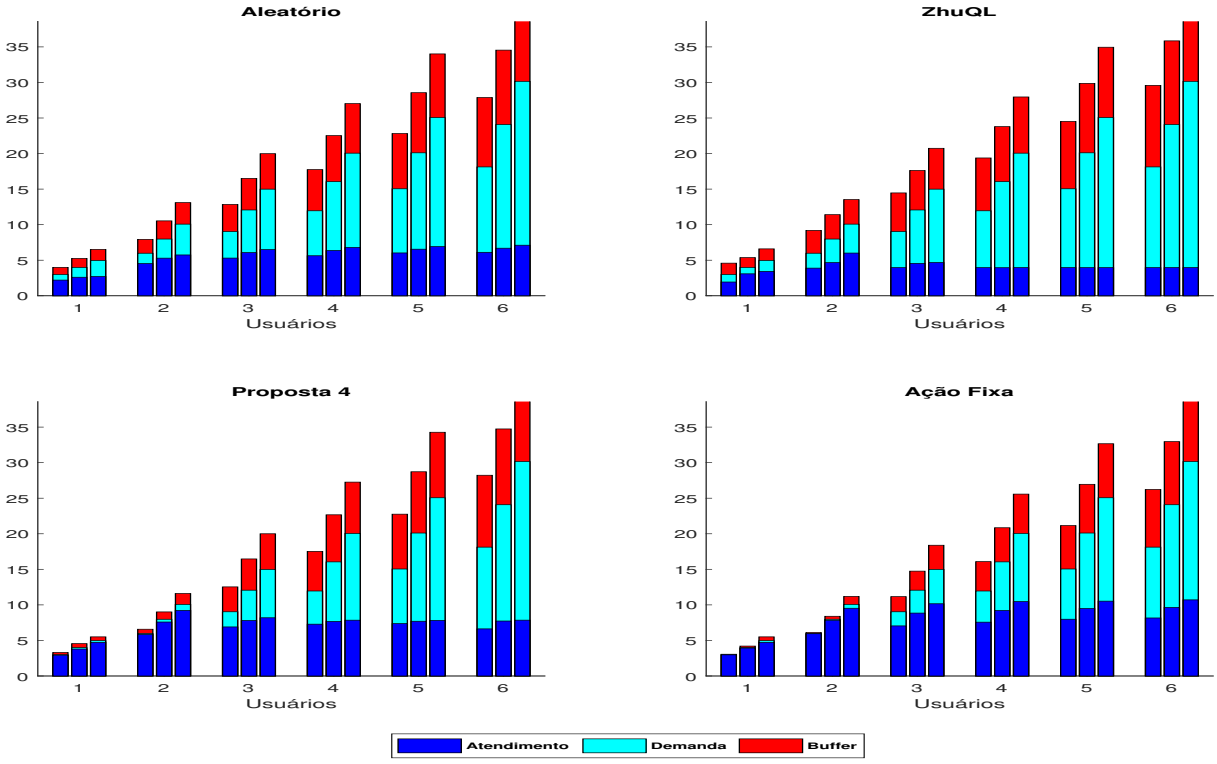

Figura 3.12 – Balanço dos pacotes para cada função de recompensa, taxa de chegada e quantidade de usuários.

3.5 Impacto no desempenho da quantidade de usuários considerando *cluster* com no máximo 3 usuários e K=M

Para essa simulação a quantidade de canais acompanha o número de usuários. A divisão dos usuários em *clusters* evita a explosão combinatória que se teria ao considerar o acoplamento, por exemplo de 10 usuários. Neste caso, resolve-se 4 problemas de 3, 3, 2 e 2 usuários e canais que são bem menores.

Os parâmetros utilizados foram: $K = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]$ usuários, $M = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]$ canais, $J = 4$ modos (4QAM, 16QAM, 64QAM e 256QAM), $C_h = 1$ estado de canal e $L = 2$ para o tamanho do *buffer*, $\lambda = [0.1, 0.7, 1.3, ..., 14.5]$.

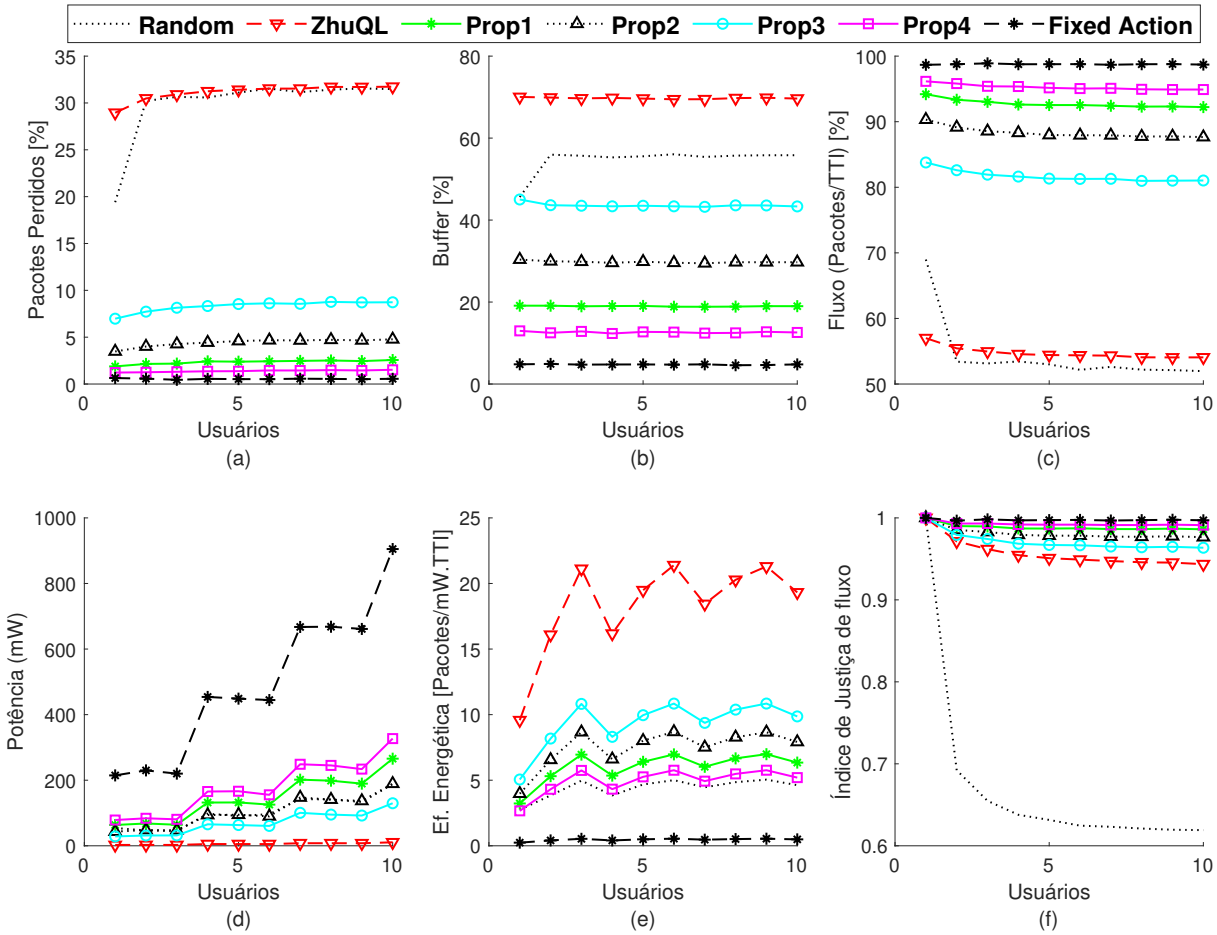

Figura 3.13 – Parâmetros de QoS versus número de usuários: (a) Pacotes Perdidos, (b) ocupação do *buffer*, (c) Fluxo de pacotes, (d) Potência, (e) Eficiência energética e (f) Índice de justiça

Conclui-se pelas figuras 3.13 (a), (b), (c), (d), (e) e (f) que o comportamento dos valores dos parâmetros de desempenho analisados é praticamente constante (em termos percentuais) ao aumentar o número de usuários se a quantidade de canais aumentar na mesma proporção. Há um aumento na potência alocada com o aumento de usuários.

Em especial, a potência alocada tem o maior aumento quando a quantidade de *clusters* aumenta (3 para 4 , 6 para 7 e 9 para 10 usuários).

3.6 Impacto do Tamanho do *Buffer* L no desempenho

Nesta seção, variou-se o tamanho máximo do *buffer* nas simulações. Os parâmetros utilizados foram: $K = 2$ usuários, $M = 2$ canais, $J = 4$ modos (4QAM, 16QAM, 64QAM e 256QAM), $C_h = 1$ estado de canal, $L = [1, 2, .., 18, 19]$ para o tamanho do *buffer* e $\lambda = [0.1, 1.1, 2.1, ..., 9.1]$. Cada gráfico é obtido, considerando-se esses 10 valores de λ para cada tamanho de *buffer*. Assim, os resultados são obtidos via normalização e cálculo da média dos valores do parâmetro referente a cada gráfico. Ou seja, os pontos dos gráficos representam valores médios dos parâmetros de QoS considerados para a faixa de lambda de 0.1 a 9.1 pacotes/(usuário.TTI).

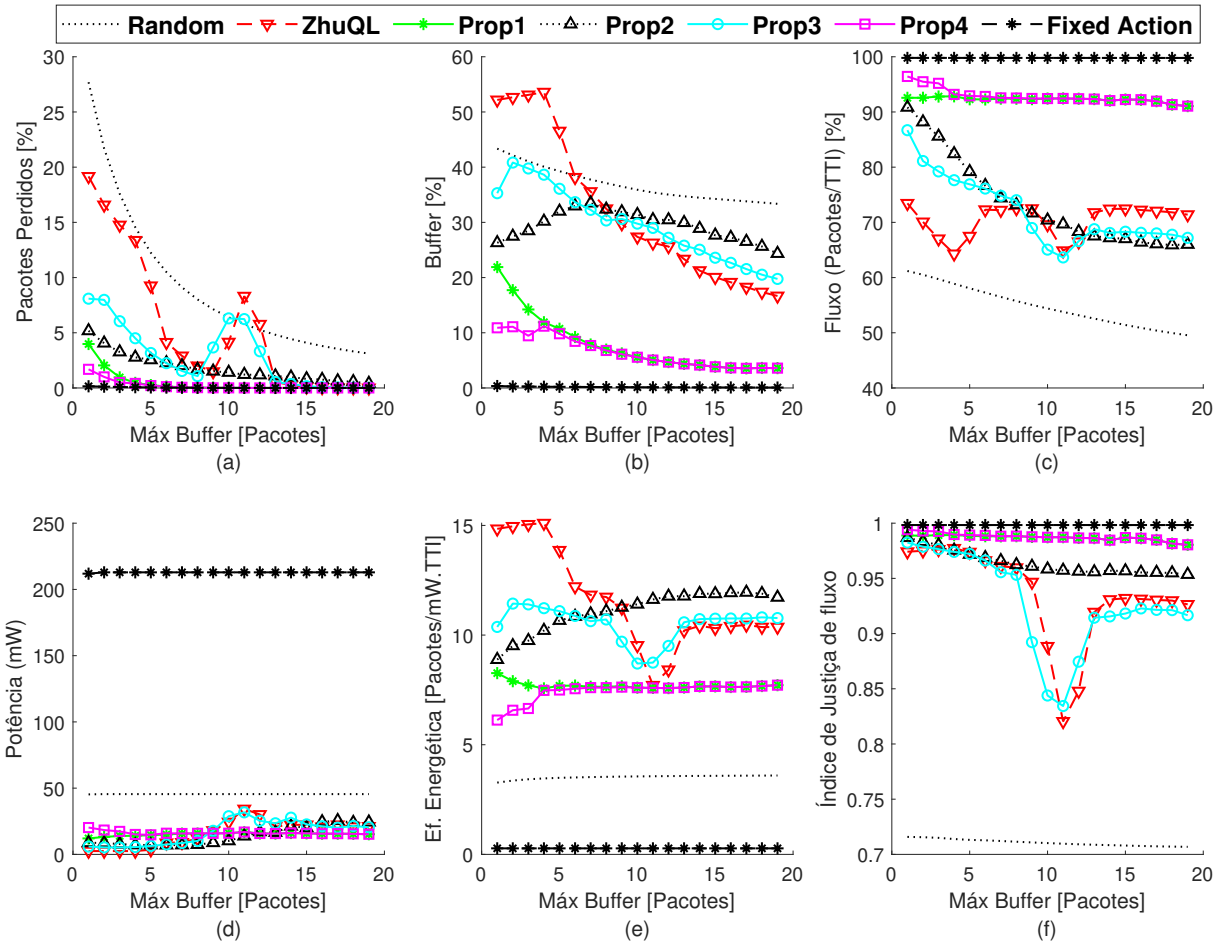

Figura 3.14 – Parâmetros de QoS versus máximo tamanho do *buffer*: (a) Pacotes Perdidos, (b) ocupação do *buffer*, (c) Fluxo de pacotes, (d) Potência, (e) Eficiência energética e (f) Índice de justiça

Ao se observar as Figuras 3.14 (a), (b), (c), (d), (e) e (f) percebe-se que o algoritmo com a função de recompensa definida em Zhu et al. (2018) se beneficia do aumento do tamanho do *buffer*, ou seja, os parâmetros de desempenho melhoram com o aumento do tamanho do *buffer*, exceto a eficiência energética e potência consumida.

Ao atingir um tamanho de *buffer* por volta de 5 nas simulações, as propostas 1 e 4 passam a perder poucos pacotes, o que corresponde ao atendimento utilizando a capacidade máxima do sistema. Porém, as soluções com aprendizado por reforço apresentam melhor eficiência energética, em especial a proposta 2 (Figura 3.14 (a) e (e)). O aumento da eficiência acontece, geralmente, com o aumento da fila no *buffer*, o que faz sentido. Ao empilhar os pacotes, pode-se atender utilizando o modo de transmissão mais eficiente. Apesar de aumentar o tempo de espera, esse tipo de atendimento tem aplicação em situações menos críticas como IoT (*Internet of Things*), comunicação de coleta de dados onde não se espera atuação imediata, entre outras. As propostas 3 e de Zhu apresentam oscilações no comportamento ao longo da variação do tamanho máximo do *buffer*. Acredita-se que a utilização do termo $E[l_k]$ na função de recompensa que representa o estado do *buffer* em média após a chegada de pacotes mas antes da ação contribua para esse comportamento. Tendo em vista os resultados apresentados em um cenário com *buffer* suficientemente grande a proposta 2 se mostra a melhor escolha em termos de eficiência energética e as propostas 1 e 4 em termos de vazão e tempo médio de espera.

3.7 Impacto do número de canais M no desempenho

Nesta seção, variou-se a quantidade de canais nas simulações. Os parâmetros utilizados foram: $K = 5$ usuários, $M = [2, 3, 4, 5, 6, 7]$ canais, $J = 4$ modos (4QAM, 16QAM, 64QAM e 256QAM), $C_h = 1$ estado de canal e $L = 2$ para o tamanho do *buffer*, $\lambda = [0.1, 0.7, 1.3, ..., 14.5]$.

Pelas Figuras 3.15 (a), (b), (c), (d), (e) e (f) pode se observar que os parâmetros de desempenho de todas as propostas melhoram com o aumento do número de canais até o limite em que $K = M$. A partir daí, há aumento da eficiência energética (Figura 3.15 (e)) pela multiplexação estatística que ocorre ao se simular mais canais que usuários. Confirma-se também, que a partir de $K = M$ ocorre redução da potência alocada aos usuários conforme mostra a Figura 3.15 (d).

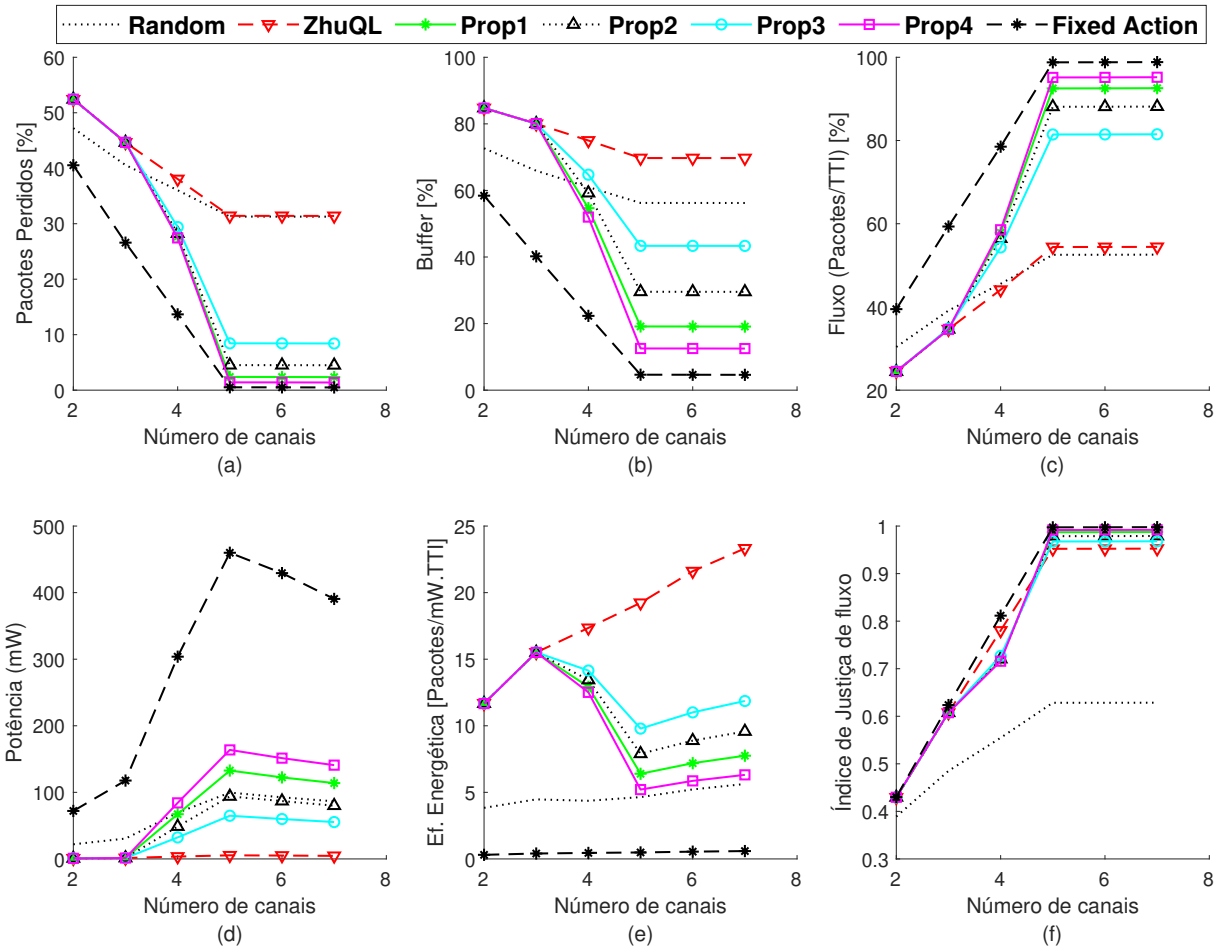

Figura 3.15 – Parâmetros de QoS versus número de canais: (a) Pacotes Perdidos, (b) ocupação do *buffer*, (c) Fluxo de pacotes, (d) Potência, (e) Eficiência energética e (f) Índice de justiça

3.8 Impacto do número de estados do canal Ch no desempenho

Nesta seção, apresenta-se os resultados de simulações variando o parâmetro C_h que representa o número de estados diferentes considerados para o canal. Na prática, o ganho do canal é uma variável real e pode assumir infinitos valores. Para fins práticos de simulação inteira, divide-se a faixa de ganho em seções de ganho médio diferentes e considera-se esses ganhos para representar melhor o estado do sistema.

Em algumas situações, uma maior quantidade de canais auxilia o agente a tomar decisões. Como por exemplo, postergar o atendimento de um usuário se este apresenta ganhos de canal ruins mas tem alta probabilidade de mudar de estado no próximo intervalo de decisão. Esse efeito pode ser mais ou menos vantajoso a depender da função de distribuição de probabilidades do ganho do canal e da capacidade do *buffer* em armazenar os dados de chegada.

A seguir apresenta-se o resultado para simulação utilizando $K = 2$ usuários, $M = 2$

canais, $L = 3$ tamanho do *buffer*, $J = 3$ modos de transmissão (BPSK, 4QAM e 8QAM) utilizando o conjunto de ações sem repetição (ver 2.2), faixa de $\lambda = [0.1, 0.2, 0.3, ..., 1]$ e número de estados de canal na faixa de $C_h = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]$. Cada ponto do gráfico corresponde a média de 10 simulações correspondentes a faixa de variação de λ. Optou-se por uma faixa restrita de $\lambda < L$ justamente para explorar a capacidade do *buffer* sem precisar aumentar demais o seu tamanho.

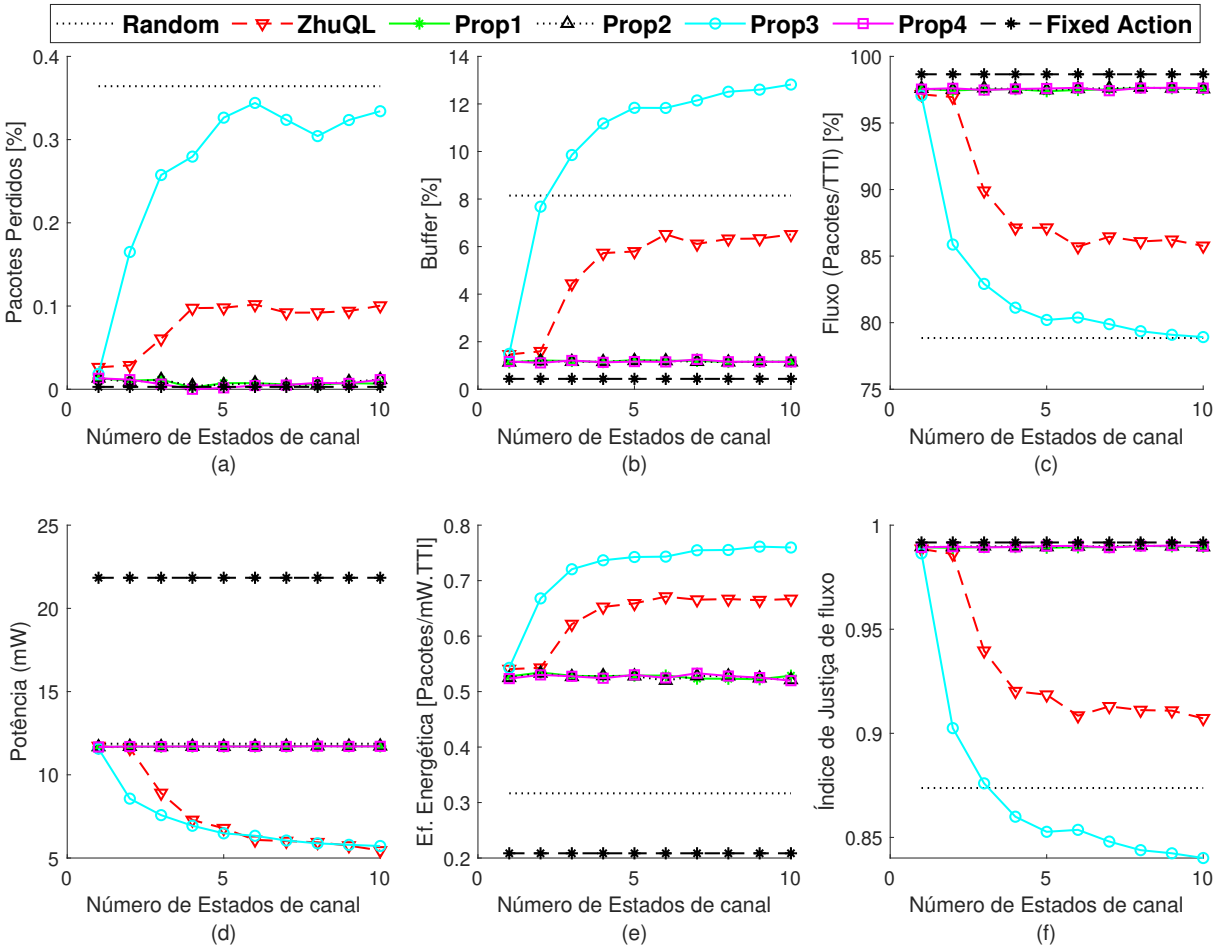

Figura 3.16 – Parâmetros de QoS versus taxa número de estados de canal para baixa taxa de chegada : (a) Pacotes Perdidos, (b) ocupação do *buffer*, (c) Fluxo de pacotes, (d) Potência, (e) Eficiência energética e (f) Índice de justiça

Observa-se pelas Figuras 3.16 (d) e (e) como a potência alocada é reduzida e ocorre um aumento na eficiência energética com o aumento da quantidade de estados de canal para a proposta 3 e de Zhu..

Para os algoritmos da proposta 3 e o de Zhu, ocorre um aumento na quantidade de pacotes perdidos (Figura 3.16 (a)) com o aumento da quantidade de estados de canal, que é compensado com um ganho em eficiência energética. A redução no fluxo de pacotes (Figura3.16 (c)) com o aumento da quantidade de estados de canal está associada a uma maior utilização do *buffer*, o que influencia na base para o cálculo percentual da

vazão. Em outras palavras, a maioria dos pacotes que deixaram de ser transmitidos foram armazenados no *buffer* e não perdidos. Pode-se notar esse comportamento na Figura 3.16 (b) que evidencia o aumento na ocupação do *buffer*. Fica clara a vantagem de se utilizar mais estados de canal e dimensionar o tamanho máximo do *buffer* de acordo com a taxa de chegada de pacotes. Nesse caso a proposta 3 se mostrou a mais vantajosa principalmente em termos de eficiência energética.

Simulações também foram feitas utilizando valores de $\lambda > L$, $\lambda = [0.1, 0.7, 1.3, ..., 14.5]$, $J = 4$ modos de transmissão (4QAM e 16QAM, 64QAM e 256QAM) e $L = 2$. Os resultados são mostrados a seguir:

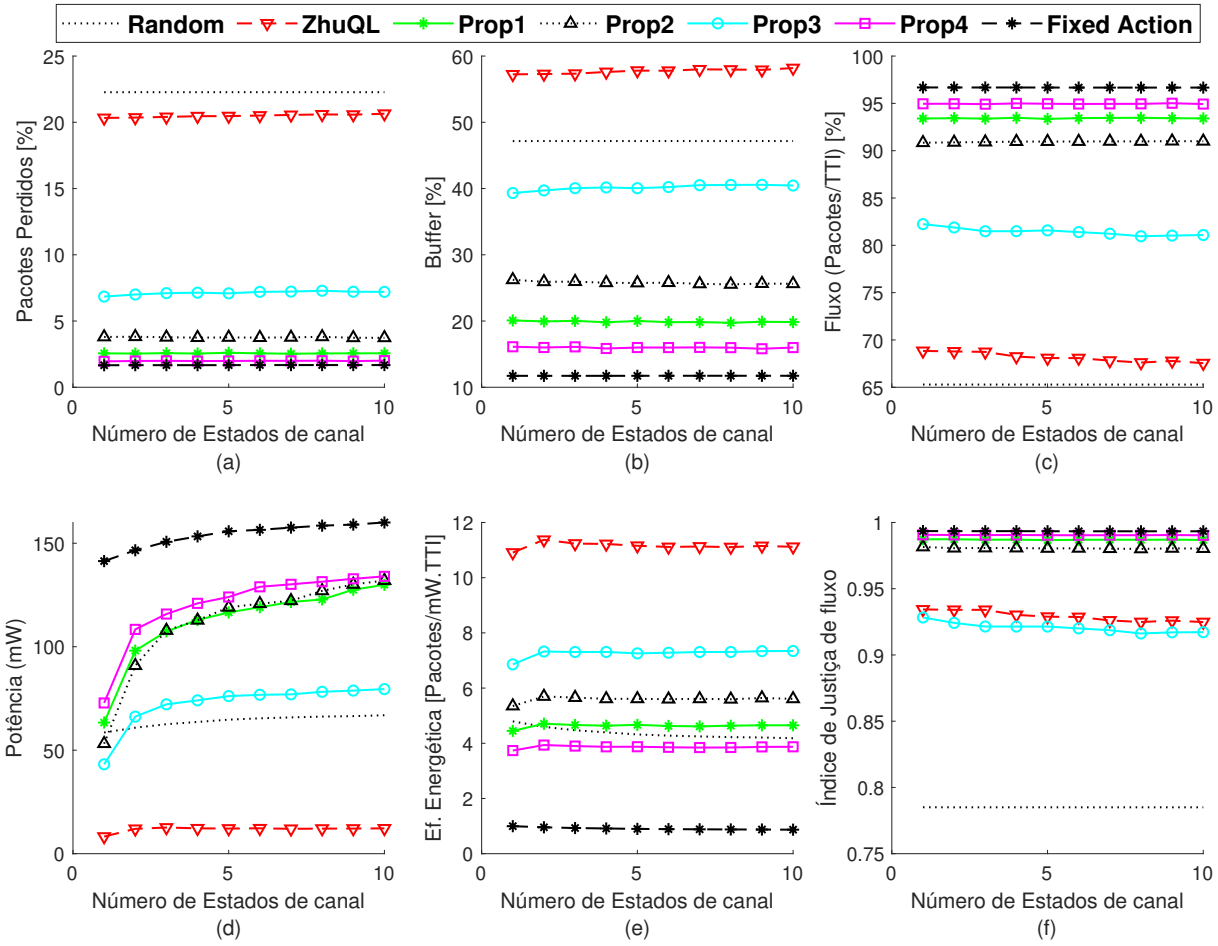

Figura 3.17 – Parâmetros de QoS versus número de estados de canal para alta taxa de chegada : (a) Pacotes Perdidos, (b) ocupação do *buffer*, (c) Fluxo de pacotes, (d) Potência, (e) Eficiência energética e (f) Índice de justiça

Pode-se notar que a partir de 3 estados, a quantidade de estados de canal apresenta maior influência nos resultados de desempenho dos algoritmos considerados e que esta influência está relacionada ao tamanho do *buffer* em relação a taxa de chegada de pacotes λ. Ao observar as Figuras 3.17, (a), (b), (c), (d), (e) e (f) confirma-se esse comportamento. Entretanto, há pouca mudança no comportamento com o aumento de C_h.

Um maior uso do *buffer* impacta no tempo de espera dos pacotes. A depender da aplicação esse parâmetro pode ser crítico. Neste caso, uma maior quantidade de estados de canal pode se tornar prejudicial para a solução em termos de vazão. Já em termos de eficiência energética os resultados são interessantes, especialmente pela pequena quantidade de pacotes perdidos.

Vale ressaltar que obteve-se baixas taxas de perda de pacotes quando utilizou-se $L \approx 3\lambda_{max}$. Um cenário com $L \approx 2\lambda_{max}$ já pode apresentar perdas de pacotes relevantes. Ao aumentar o tamanho do *buffer* aumenta-se a quantidade de estados do sistema (segundo a equação: $Estados_{buffer} = (L+1)^K$ sendo K o número de usuários). Isso pode inviabilizar a utilização da simulação apresentada em cenários com *buffers* grandes em termos de número de pacotes. Uma solução (para o sistema fluido) é aumentar o tamanho do pacote.

3.9 Selecionando as melhores subportadoras para transmissão

Percebe-se pela Figura 1.2 que há grande probabilidade de o canal de comunicação de um usuário apresentar ganhos abaixo da média. O ganho de canal tem grande influência no valor alocado de potência. Assim, para economizar energia, pode ser interessante utilizar mais canais que usuários, i.e, utilizar várias subportadoras e a informação da qualidade do canal para selecionar os "M"melhores de cada usuário-subportadora para o atendimento.

A Figura 3.18 mostra como a distribuição muda ao escolher a melhor entre 16 subportadoras para 1 usuário. Observa-se pela Figura 3.18 que o ganho médio é maior que 3 vezes o ganho médio que seria obtido ao usar uma única subportadora por usuário. Além disso, a distribuição se deslocou para a direita. Para o caso da Figura 3.18, a probabilidade de se utilizar um canal com ganho abaixo da média (igual a 1 neste caso) é pequena.

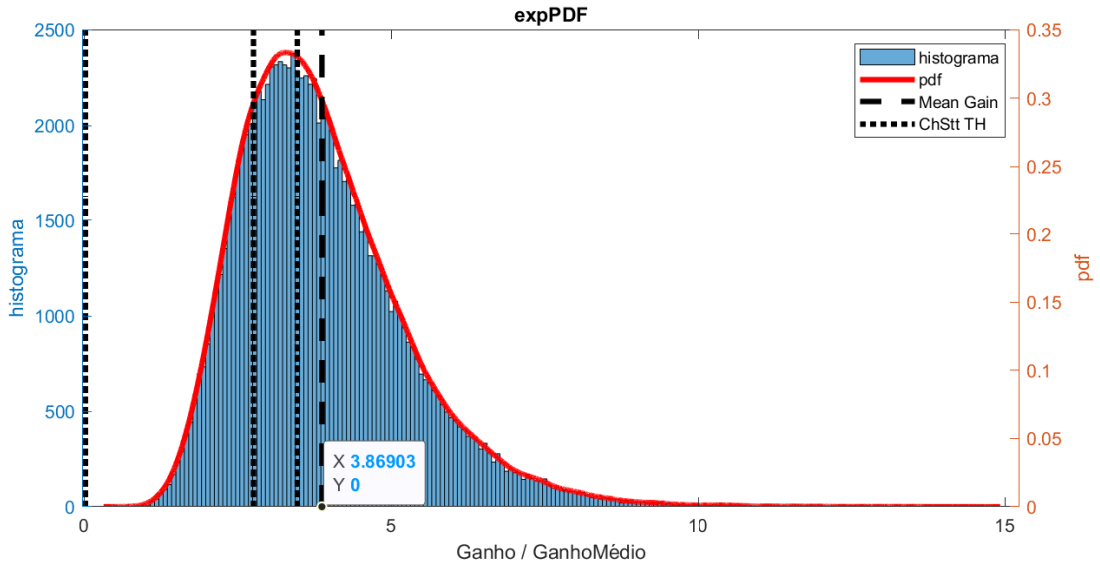

Figura 3.18 – Distribuição normalizada do ganho do canal ao utilizar as 3 melhores de cada 48 subportadoras disponíveis. Definidos 4 estados de canal.

Na Figura 3.20 (a) nota-se como a proposta 4 leva vantagem mesmo quando comparada a Ação Fixa para valores de $\lambda > 7.5$. Isso ocorre devida a falta de simetria no conjunto de ações. Como não é possível repetir o mesmo modo de transmissão para os usuários envolvidos, a decisão do agente leva vantagem sobre o escalonamento aleatório da Ação Fixa.

Na Figura 3.20 (b) observa-se a ocupação média do *buffer* para as diferentes propostas. A proposta 4 apresenta menor ocupação seguida pelas propostas 1, 2, 3 e de Zhu.

Na Figura 3.20 (c) nota-se novamente a vantagem da proposta 4 sobre a Ação Fixa para $\lambda > 7.5$. O intervalo de $\lambda < 2.5$ apresenta menores vazões, mas que são justificadas

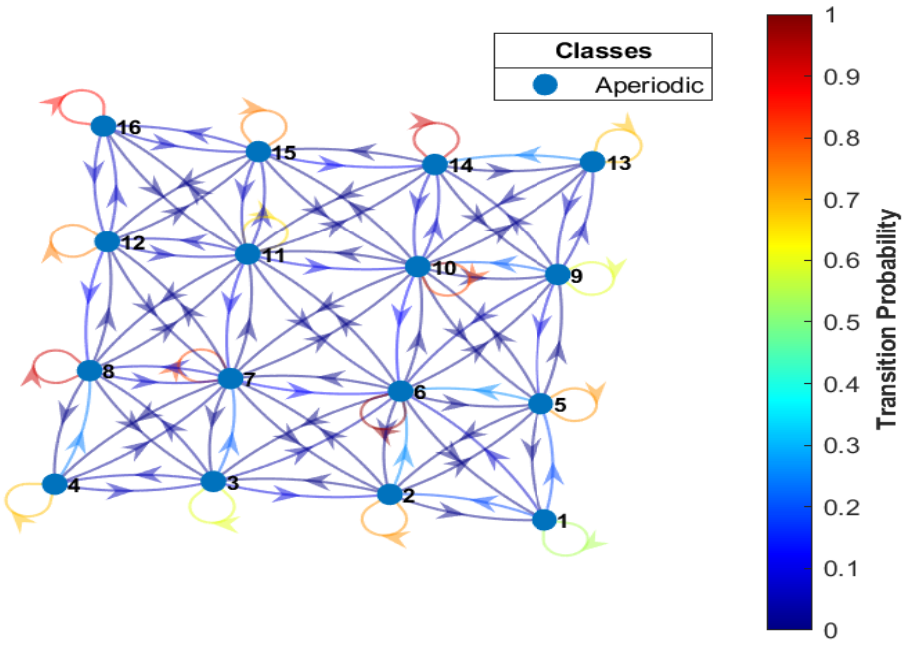

Figura 3.19 – Cadeia de Markov dos estados de canal para 2 canais ao utilizar as 2 melhores de cada 32 subportadoras disponíveis. Definidos 4 estados de canal.

pela maior ocupação do *buffer* (Figura 3.20 (b)) e leve aumento na taxa média de pacotes perdidos (Figura 3.20(a)).

De modo geral a capacidade do sistema é reduzida devido a essa limitação no conjunto de ações mas os resultados (para $2.5 < \lambda < 7.5$) são parecido aos observados na Seção 3.3.

Na Figura 3.19 pode-se notar como a cadeia de Markov para os estados do canal mudou comparada a da Figura 2.2. Há maior tendência de se manter nos estados intermediários comparado aos resultados da Figura 2.2. Ou seja, o ganho do canal tem menor probabilidade de se alterar a cada iteração.

Na Figura 3.20 (d) percebe-se como o consumo de potência foi reduzido e como houve melhora no atendimento, uma vez que a situação de $\rho <= 0.01\rho_m$ quase nunca acontece.

Ao utilizar a melhor de 16 subportadoras muita largura de banda é desperdiçada. Aplica-se esse recurso caso o consumo de energia seja muito restrito e esteja sobrando largura de banda. É comum que as subportadoras sejam melhor aproveitadas atendendo outros usuários ou subdividindo o atendimento de cada usuário. Utilizou-se os parâmetros da seção 3.2 (K,L,M,J,V,Ch =5, 2, 5, 4, 2, 2) exceto o grupo de ações que foi o mesmo da seção 3.1 (sem repetição) cujo número de ações é dado pela equação 2.1.

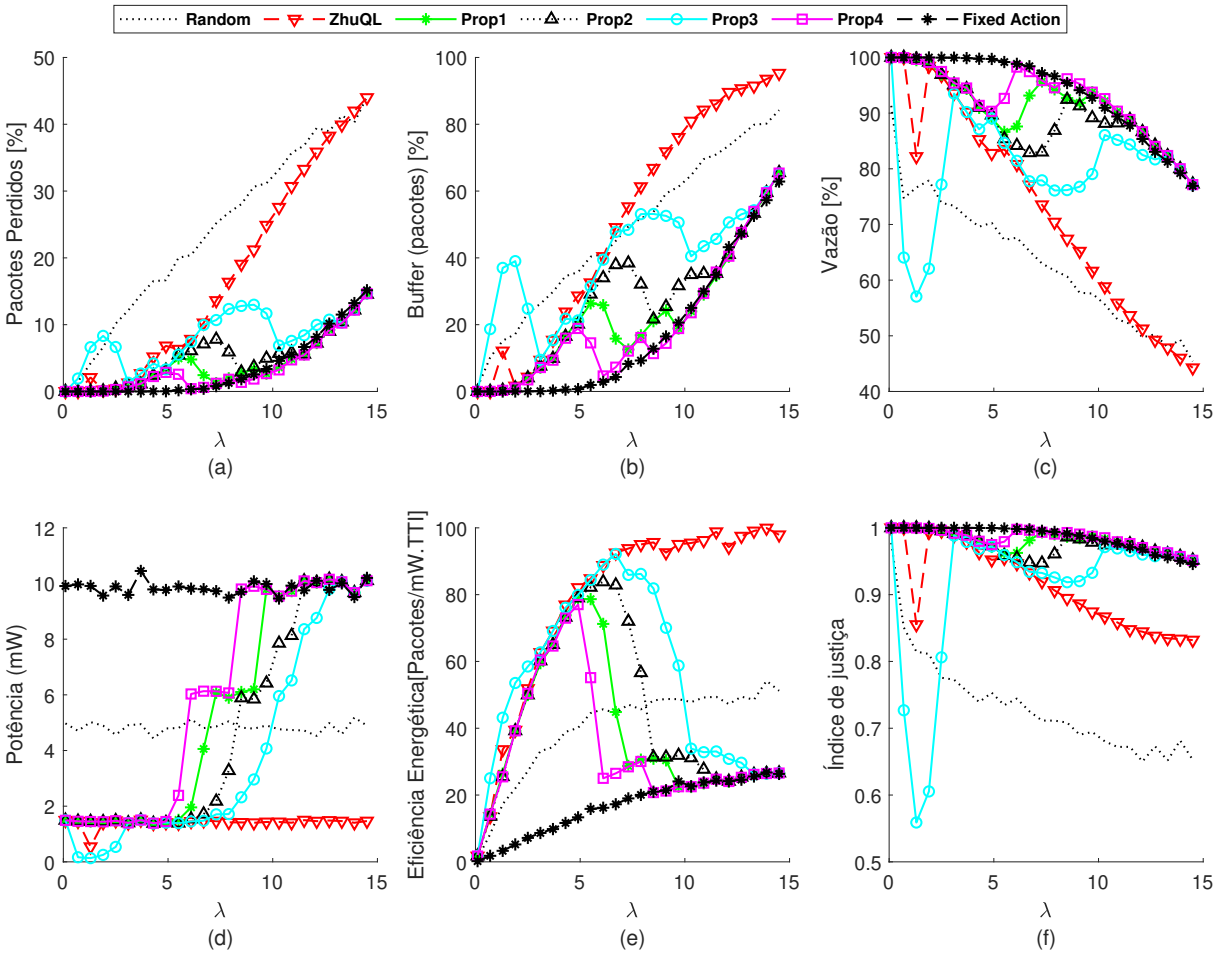

Figura 3.20 – Parâmetros de QoS versus taxa média de geração de pacotes com grande disponibilidade de subportadoras: (a) Pacotes Perdidos, (b) ocupação do *buffer*, (c) Fluxo de pacotes, (d) Potência, (e) Eficiência energética e (f) Índice de justiça

Ao comparar os resultados com os da seção 3.2 percebe-se como a solução de Zhu et al. (2018) realmente é beneficiada pelo conjunto de ações sem repetição. Também é possível observar como a capacidade do sistema é reduzida para o conjunto de ações sem repetição do modo. Quanto à grande disponibilidade de canais, percebe-se que os valores da potência são reduzidos drasticamente, como observado na (Figura 3.20 (d)). Essa abordagem mostra como o ganho do canal é o gargalo para o consumo de potência. Na prática, é possível utilizar diversidade espacial do sinal além da diversidade em frequência, para obter melhores ganhos de canal através de múltiplas antenas (MIMO) (CHO et al., 2010).

3.10 Alocação de recursos com algoritmo de aprendizagem Adaptativo considerando dados reais de tráfego

Até aqui considerou-se uma faixa fixa de λ para a chegada de dados e utilizou-se dados sintéticos nas simulações. Na prática, não se tem controle desses valores especialmente quanto ao tamanho do pacote, mas o intervalo do TTI e o tamanho da janela para o cálculo da média de chegada de pacotes podem ser ajustados. Para o atendimento adaptativo com dados reais adapta-se os algoritmos das seções anteriores deste trabalho que utilizaram dados sintéticos e propostas de função de recompensa considerando ocupação do *buffer* e pacotes perdidos com atendimento CP-OFDM (Seções 3.3, 3.5 - 3.9). Nesta alocação adaptativa de recursos também se avalia os intervalos de λ que fazem com o que o sistema de comunicação apresente melhor desempenho em termos de pacotes perdidos, ocupação média do *buffer*, vazão, potência média consumida, eficiência energética e índice de justiça. A definir o tamanho do pacote (mantido fixo durante a simulação nesse trabalho) controla-se, em partes, os valores de chegada de pacotes mesmo com dados reais. Utilizou-se média exponencial de dados e um intervalo de TTI igual a 0.5 ms o que equivale a 4 amostras dos dados reais de MAWI (2019) que foram agregados em intervalos de 0.125ms. Com isso, espera-se capturar flutuações na taxa média de chegada e obter novas políticas de atendimento que contemplem as variações no ambiente. Utiliza-se o modelo desenvolvido para dados de chegada com distribuição de poisson.

Como há flutuação nos valores da taxa média de chegada (parâmetro λ do modelo), adota-se um limiar de variação de 0.15λ para se obter uma nova solução (política de tomada de ações). Para essa simulação adotou-se tamanho do pacote igual a 1365 bytes de modo que dentro de 1 TTI as taxas médias de chegada sejam da ordem das simuladas (variando entre 3 e 15 pacotes/TTI idealmente). Com o aumento do tamanho do pacote, a largura de banda precisa ser adaptada para o modelo fluido de transmissão com alocação adaptativa de recursos. Considerou-se um fator de 13 vezes, já que o tamanho do pacote passou de 105 para 1365. Assim, para um sistema fluido equivalente próximo aos já simulados, a largura de banda considerada é da ordem de 20x13 = 260 MHz. Assim, tem-se a mesma taxa de codificação obtida no cenário com 20 MHz de banda e pacotes de 105 bytes: 2. Manteve-se a frequência da portadora (6 GHz).

O valor da intensidade da potência do ruído WN_0 aumenta com aumento da largura de banda (aproximadamente 11 dB). Isso irá impactar na potência alocada se o objetivo é manter os valores de SNR e BER. Assim, foi considerado um ganho na antena de 11dB.

Para aumentar o valor da taxa de codificação, sem mudar a estrutura do bloco de recurso (LTE), aumenta-se a largura de banda e assim a quantidade de blocos de recurso.

A quantidade de usuários simulados é de $K = 5$ usuários separados em 2 *clusters*, um de 3 usuários e outro de 2. A motivação é solucionar 2 problemas menores desacoplados

para obter uma resposta rápida o suficiente que se adapte às flutuações do sistema. Dado o intervalo reduzido do TTI de 0.5 ms o treinamento e o atendimento são feitos de maneira síncrona, i.e, ao atingir uma variação suficientemente grande em λ (adota-se 15%) uma nova política é encontrada utilizando o valor atual de λ. Nas simulações, atendimento (decisão de alocação de recursos aos dispositivos) aguarda o treinamento para ser efetivado. Na prática, é esperada uma pequena defasagem de tempo entre o treinamento e o atendimento. Espera-se que a utilização de *buffers* maiores possa contornar esse problema, entretanto, com aumento do atraso dos pacotes na fila.

Nesta simulação, considerou-se uma frequência de portadora de 6 GHz, uma distância de 60m entre transmissor e receptor e uma banda de 260 MHz. Foram simulados 18.685 segundos (equivalente a 37370 TTI's). O sistema escolhido tem K=5 usuários, tamanho de *buffer* $L = 2$, quantidade de canais $M = 5$, quantidade de modos de transmissão $J = 5$: 4QAM, 16QAM, 64QAM , 256QAM e 1024QAM e quantidade de estados de canal $C_h = 1$. O tamanho do pacote é fixado em 1365 bytes. Para taxa de desconto utilizou-se $\gamma = 0.9$. A demanda (taxa de chegada de pacotes) de entrada utiliza os dados de MAWI (2019) em bytes. Além disso utilizou-se um conjunto de ações com repetição do modo de transmissão dado pela equação 2.2. Esse conjunto reduzido permite soluções mais rápidas.

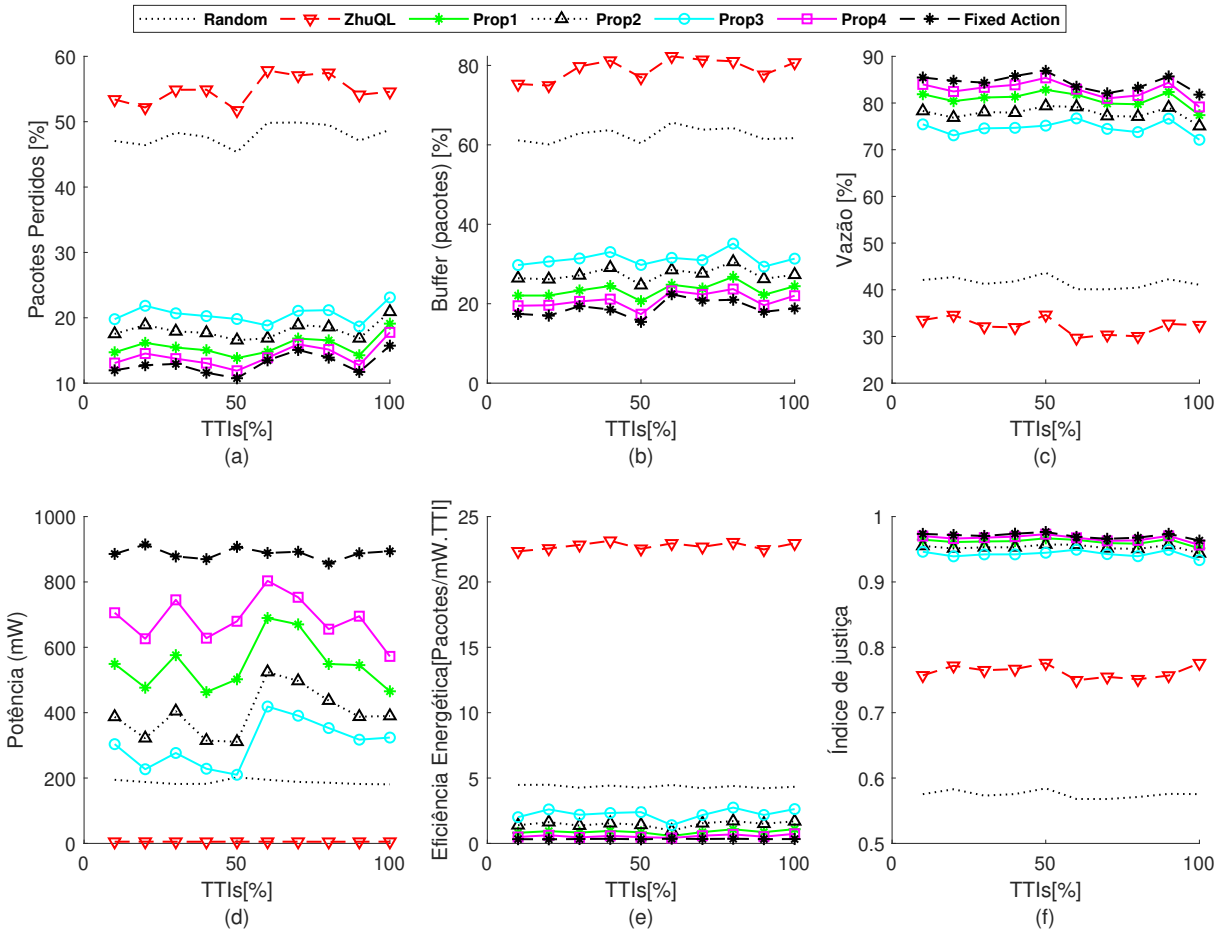

Figura 3.21 – Parâmetros de QoS versus tempo de simulação para banda de 260 MHz: (a) Pacotes Perdidos, (b) ocupação do *buffer*, (c) Fluxo de pacotes, (d) Potência, (e) Eficiência energética e (f) Índice de justiça

Comparando a Figura 3.21 (c) com a Figura 3.4 (c) da Seção 3.3 percebe-se como a vazão das propostas reduz consideravelmente quando comparado aos dados sintéticos. A taxa de perda de pacotes fica em torno de 15% para a proposta 4 (Figura 3.21 (a)). A ação fixa chega a algo em torno de 10 % com o modo constante de 1024QAM para todos os usuários. Atribui-se esse comportamento a grandes variações na chegada de pacotes que destoam da média. Os resultados das propostas se aproximam bastante da estratégia de Ação Fixa (Figuras 3.21 (b), (e) e (f) que representa um limite na capacidade do sistema com exceção dos níveis de alocação de potência (Figura 3.21 (d)) onde nota-se maior distanciamento. Assim, neste caso, observou-se que é possível obter melhores resultados aumentando a largura de banda ao invés de considerar uma modulação com mais símbolos. Mas para manter os níveis de SNR e BER sem aumentar muito a potência alocada para transmissão será necessário um aumento no ganho da antena.

Na Figura 3.22 nota-se a demanda, atendimento e ocupação do *buffer* para cada usuário e para cada uma das propostas de solução a cada 10 % do intervalo simulado

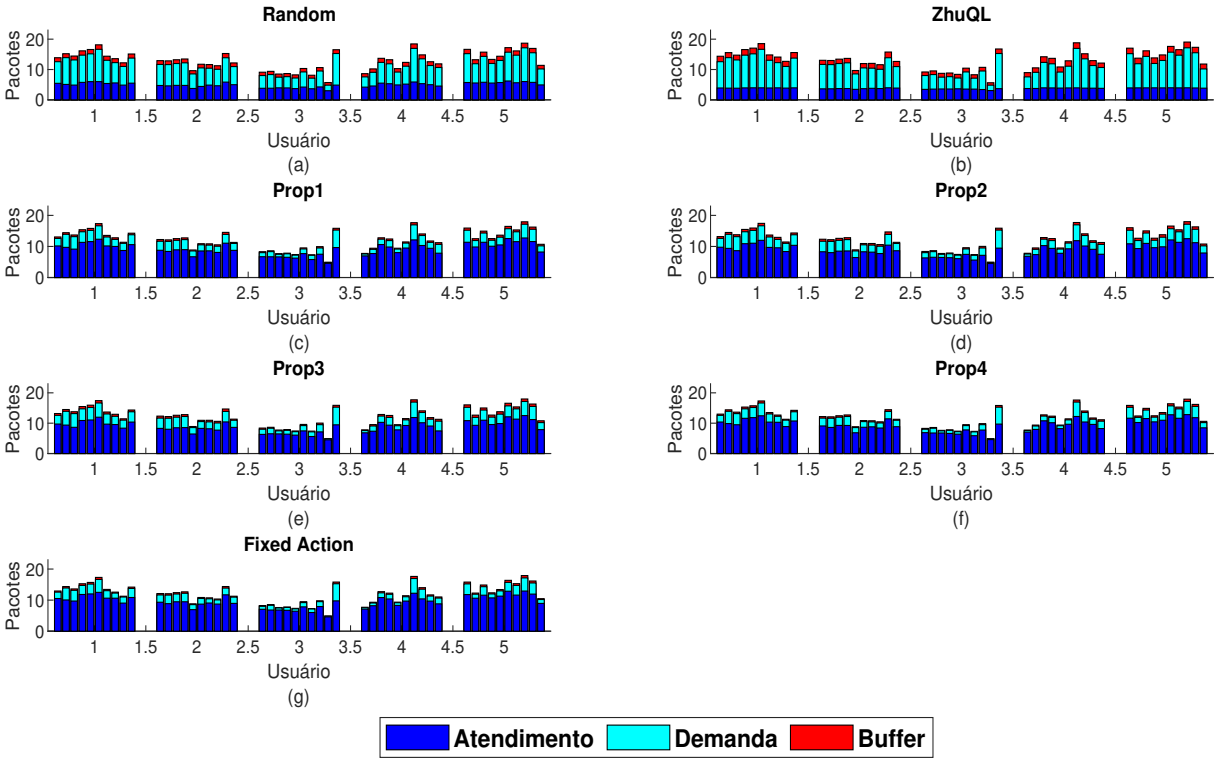

Figura 3.22 – Balanço dos pacotes para cada função de recompensa em cenário adaptativo com largura de banda igual a 260MHz.

(barras individuais). Nota-se de maneira geral que as demandas dos usuários são da mesma ordem de grandeza e as soluções são justas, ou seja, não beneficiam nenhum usuário.

Quanto à diferença entre os desempenhos das funções de recompensa, os resultados são compatíveis com os da simulação com dados sintéticos (Seção 3.3). A proposta 4 se destaca quanto à perda de pacotes, tamanho do *buffer*, taxa de transmissão, e índice de justiça (Figuras 3.21 (a), (b), (c) e (d) e a solução de Zhu et al. (2018) se destaca quanto a potência alocada e eficiência energética (Figuras 3.21 (d) e (e)). O ganho de antena considerado para essa largura de banda foi de 15+12 = 27 dB.

O tempo de processamento é uma variável importante no cenário adaptativo. O objetivo é responder às perturbações do ambiente com alterações rápidas do modelo. Observou-se que o fator limitante da resposta em tempo real é o cálculo da matriz $\mathbf{P}$, dados os novos valores de λ. O passo de obter a política (*Q-Learning*) pode ser acelerado utilizando a última política treinada como política inicial do treinamento e o valor da função de valor (V(s,a)) associado a essa política também como estimativa inicial. Com isso, parte-se de uma configuração de espaço de busca mais próximo do mostrado na Figura 2.5 do que o da Figura 2.4 porém com as matrizes do modelo atualizadas. Por outro lado, essa prática pode enviesar a solução caso o cenário mude muito em um intervalo curto. Ao se partir de uma solução prévia pode-se obter uma nova solução que seja um ótimo local.

A simulação dos 37370 *TTI's*, juntamente com o processo de cálculo adaptativo do modelo do sistema, busca da política estacionária, armazenamento e plotagem dos gráficos leva cerca de 10 minutos. O notebook utilizado possui 8 processadores lógicos Intel(R) Core(TM) I7-7700HQ CPU @ 2.80GHz e 4 núcleos. 16+4 = 20 GB de memória RAM, sistema operacional Windows 10 x64, uma placa de vídeo dedicada GeForce GTX 1050 com 4GB de memória, disco de estado sólido SSD de 120 GB e HDD de 930 GB.

Ao isolar o tempo para calcular a matriz **P** e **R** no caso do *cluster* de 3 usuários verifica-se algo em torno de 0.6 segundos e o tempo médio de convergência do algoritmo *Q-Learning* com iteração de política é de 0.4 segundos para as 5 funções de recompensa.

Na Figura 3.23 visualiza-se a sequência dos pacotes de chegada por *TTI* divididos por usuário. Os dados utilizados foram obtidos do grupo MAWI (2019).

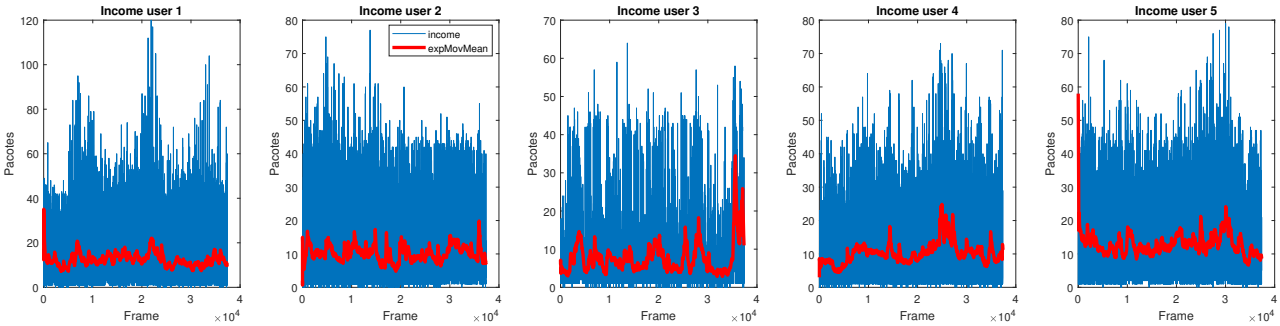

Figura 3.23 – Dados de chegada em pacotes por *TTI* gerados a partir dos dados de MAWI (2019), agrupados de 4 em 4 e com janela de média móvel igual a 500 amostras (0.25 segundos).

3.11 Aumentando a largura de banda e simulando cenário assíncrono para atendimento adaptativo utilizando dados reais de tráfego

Nesta seção, discute-se os resultados obtidos ao se aumentar a largura de banda no cenário da seção anterior (Seção 3.10) de 260 MHz para 400 MHz. Além de propor uma simulação mais realista (assíncrona) onde se estima o tempo necessário para o cálculo das matrizes $\mathbf{P}$, $\mathbf{R}$ e $\mathbf{Q}$ e limita-se o tempo de processamento ao tempo de simulação.

Com a nova largura de banda, a taxa de codificação aumenta de 2 para 3, i.e, com os modos 4QAM, 16QAM, 64QAM e 256QAM o sistema tem capacidade de transmitir até 6, 12, 18 e 24 pacotes por usuário e por TTI. O tamanho do pacote é fixado em 2100 bytes. O ganho da antena é aumentado em aproximadamente 1,8 dB = 10log(400/260) para manter os níveis de potências alocadas. Caso não seja possível aumentar o ganho da antena, é esperado um consumo de potência alocada 400/260 = 1,5 , 50% maior.

A simulação assíncrona visa simular o cenário real em que o tempo para os cálculos das matrizes $\mathbf{P}$, $\mathbf{R}$ e $\mathbf{Q}$ são maiores que o intervalo de tempo de decisão de 0.5ms (TTI). As operações para o cálculo de $\mathbf{P}$ e $\mathbf{R}$ são otimizadas utilizando vetorização da matriz $\mathbf{P}$ em suas dimensões 1 e 2, ou seja, calcula-se de forma matricial as probabilidades de transição entre estados e recompensas do estado para uma ação fixa. Um laço é iterado na dimensão das ações para obter as matrizes $\mathbf{P}$ e $\mathbf{R}$ para todas as ações. Por esse motivo, opta-se por utilizar ferramentas de computação paralela para resolver o laço em a, já que são calculados de forma independentes entre si. Na simulação deste cenário foram utilizadas 4 *threads* devido a limitação de núcleos do computador onde foram realizadas as simulações.

Os resultados a seguir consideram $K = 5$ usuários, $M = 5$ canais, $L = 2$ pacotes para o tamanho máximo do *buffer*, $J = 4$ modos de transmissão (4 QAM, 16QAM, 64QAM e 256QAM) para reduzir o número de ações, $C_h = 1$ estado de canal. A utilização de 1 estado de canal garante adaptação mais rápida e, como constatado, para taxas de chegada $\lambda > L$ tem pouca influência no desempenho. A janela para cálculo da média de chegada de pacotes de tamanho 1000 *TTI's* ou 0,5 segundos, limiar do ganho de canal para o atendimento de 2 %, e variação mínima de $\pm 0.01\lambda$ para novo treinamento. Adotou-se uma variação de 1% em λ para garantir que o agente sempre esteja treinando com um novo conjunto de λ. Assim, quando acabar o treinamento assíncrono há grande chance de já começar outro.

Observa-se que com o aumento da largura de banda tem-se a capacidade de atender até 95% dos pacotes que chegam, mas o agente continua com perdas de pacote da ordem de 15% para a melhor proposta. O desempenho das diferentes propostas mantêm suas

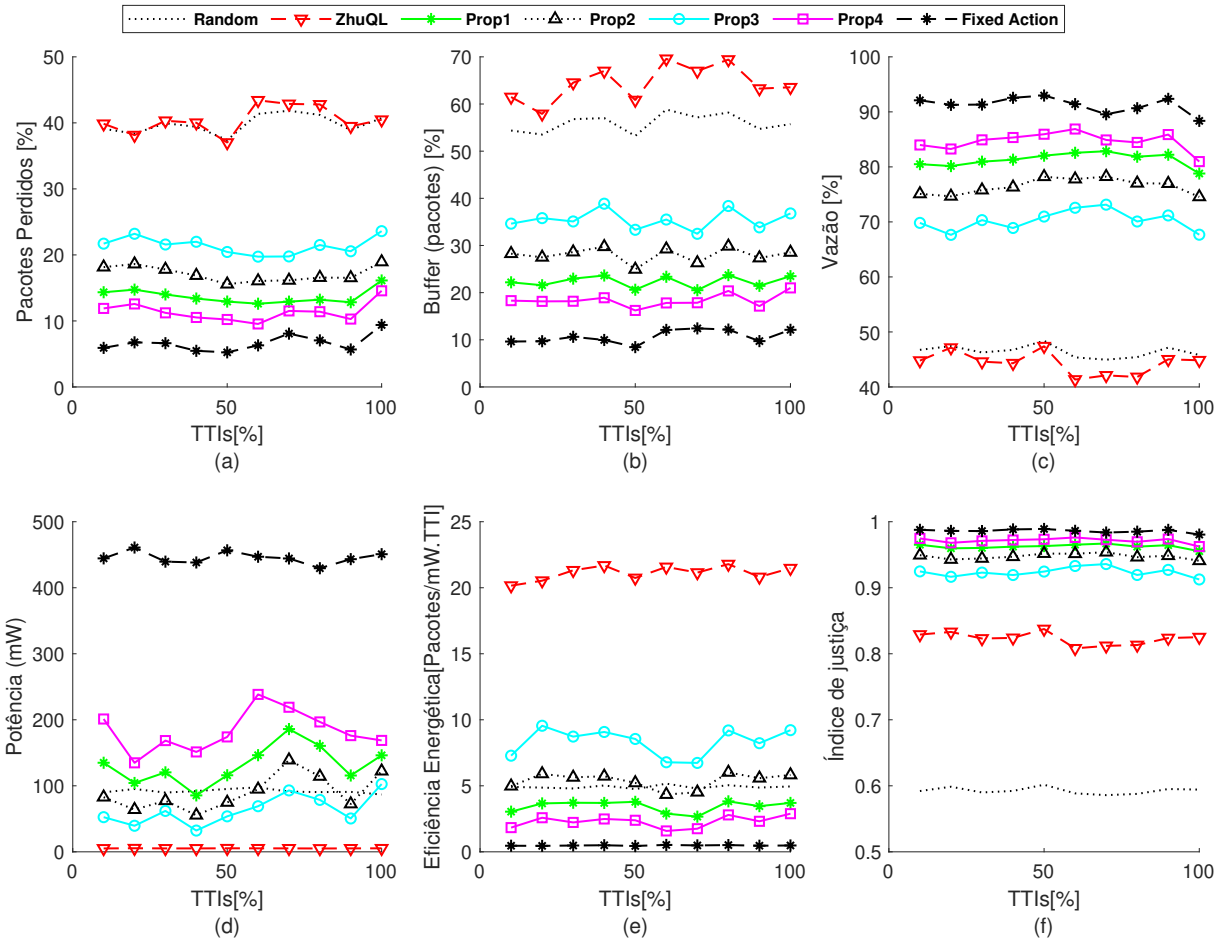

Figura 3.24 – Parâmetros de QoS versus tempo de simulação para banda de 400 MHz: (a) Pacotes Perdidos, (b) ocupação do *buffer*, (c) Fluxo de pacotes, (d) Potência, (e) Eficiência energética e (f) Índice de justiça

características, ou seja, há menor quantidade de pacotes perdidos ao utilizar a proposta 4 (Figura 3.24 (a)) e maior eficiência energética para proposta 3 e de Zhu et al. (2018) (Figura 3.24 (e)). Quanto a ocupação do *buffer* tem-se uma relação bem parecida a de pacotes perdidos (Figura 3.24 (b)). O fluxo de pacotes mostra o aumento da capacidade do sistema pela na curva de Ação Fixa (Figura 3.24 (c)). A potência alocada (Figura 3.24 (d)) é bem inferior quando comparada ao resultado da Seção 3.10 e está coerente com os níveis de vazão. O índice de justiça de fluxo acompanha as taxas de fluxo sendo a proposta 4 a mais justa (Figura 3.24 (f)).

Verificou-se um número reduzido de cálculos para o *cluster* de 3 usuários em relação ao de 2. É natural que isso ocorra, uma vez que a matriz P de probabilidade de transição de estados tem dimensão superior para o caso de 3 usuários. Em média, é preciso apenas 10% do tempo para calcular as matrizes para 2 usuários, ou seja, com *clusters* de até 2 usuários o cálculo de **P** é cerca de 10 vezes mais rápido. Por esse motivo, como simulação final da seção são apresentados os resultados da simulação assíncrona com tamanho máximo de

cluster igual a 2. Foram considerados $K = 6$ usuários e $M = 6$ canais. A largura de banda considerada é de 640 MHz e ganho de antena de 15dB. Para o tamanho máximo do *buffer* $L = 10$, $J = 4$ modos de transmissão sendo eles 4QAM, 16QAM, 64QAM e 256QAM e $C_h = 1$ estado de canal. O tamanho do pacote é fixado em 2800 bytes.

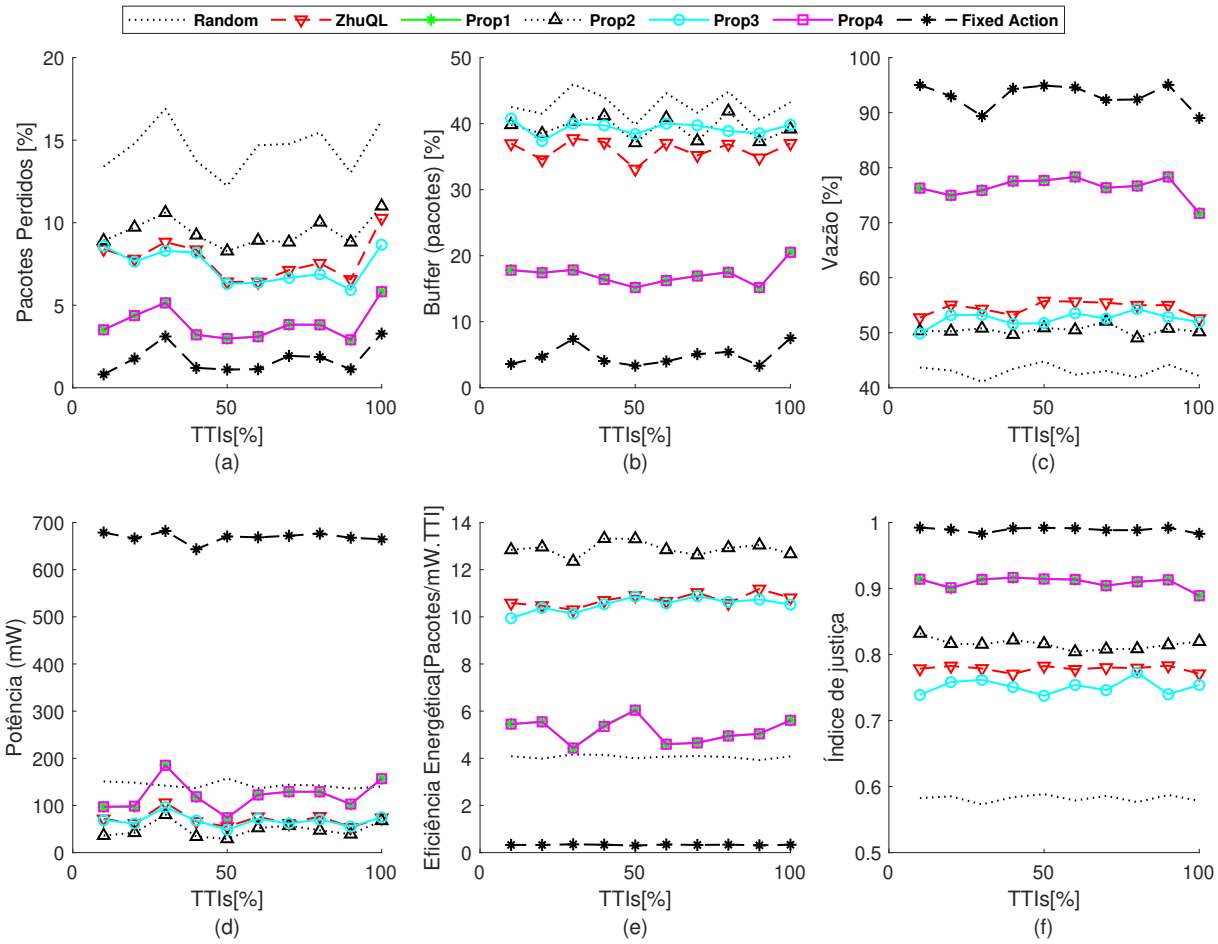

Figura 3.25 – Parâmetros de QoS versus tempo de simulação para banda de 640 MHz : (a) Pacotes Perdidos, (b) ocupação do *buffer*, (c) Fluxo de pacotes, (d) Potência, (e) Eficiência energética e (f) Índice de justiça

Os resultados para simulação assíncrona com *clusters* de até 2 usuários e *buffer* de tamanho 10 são melhores quando comparados aos resultados da Figura 3.24 que utiliza *buffer* de tamanho 2 e *cluster* máximo de 3 usuários. Em especial, para as propostas 1 e 4 que conseguem se aproximar dos níveis de pacotes perdidos e vazão do algoritmo de Ação Fixa (Figuras 3.25 (a) e 3.25 (c)) mas apresentam em média um consumo de potência inferior ao da Ação Fixa (Figura 3.25 (d)) e maior eficiência energética (Figura 3.25 (e)). Entre as propostas que utilizam aprendizado por reforço, as propostas 1 e 4 também apresentam melhor índice de justiça de vazão (Figura 3.25 (f)) e menor ocupação média do *buffer* (Figura 3.25 (b)). Já a proposta 2 apresenta o menor consumo de potência e maior eficiência energética que as demais propostas quando o tamanho do *buffer* para

cada usuários é de 10 pacotes. Espera-se comportamento similar ao se utilizar os mesmos dados de tráfego e aumentar o tamanho do *buffer*.

3.12 Utilizando rede DQN em cenário adaptativo

Conforme discutido na Seção 2.8.1, a implementação da DQN não utiliza um modelo do ambiente. Para se aplicar a rede DQN na alocação adaptativa de recursos no sistema de comunicação considerado, foi escolhido como entrada (estado **s** do sistema) um vetor coluna com os estados do *buffer* de cada usuário do *cluster* seguido pela quantidade de pacotes demandados para cada usuário, as taxas médias de chegada λ , ou seja:

$$Estado \quad \mathbf{s} = \begin{pmatrix} b_1 ... b_K & i_1 ... i_K & \lambda_1 ... \lambda_K \end{pmatrix}^T \tag{3.7}$$

Na simulação da rede DQN, utiliza-se todo o conjunto de ações possíveis, isso é, os (J+1) modos para todos os usuários, ou Na $= (J+1)^K$. A alocação de usuário no canal fica por conta do algoritmo húngaro que resolve o problema de atribuição com a informação completa dos ganhos de canal para cada usuário.

Considerou-se a seguinte configuração para o sistema de comunicação nas simulações apresentadas nesta seção: $K = 5$ usuários, tamanho do *buffer* $L = 2$, número de canais $M = 5$, número de modos $J = 4$ fixando os modos entre usuários (Eq. 2.2), e taxa de codificação $V = 2$. O número de *clusters* igual a 2, um com 3 usuários e outro com 2. Intervalo do *TTI* de 0.5 ms e chegada de dados reais obtidos de MAWI (2019) registrados em intervalos de 0.125ms e agregados em grupos de 4 para definir os pacotes de chegada do *TTI*, tamanho do pacote fixo de 3360 bytes. Ao todo são simulados 37370 *TTI's* que representam 18,685 segundos.

Quanto à configuração da rede neural adotada, o número de neurônios da camada de entrada e saída são respectivamente $3 \cdot 3 = 9$ e $5^3 = 125$ para o *cluster* de 3 usuários e $3 \cdot 2 = 6$ e $5^2 = 25$ para o *cluster* de 2 usuários . O número de neurônios das 3 camadas ocultas foram definidos como (125+9)/2, (125+9)/2 e (125+9)/2 = 67, 67, 67 para o *cluster* de 3 usuários e (25+6)/2 , (25+6)/2 e (25+6)/2 = 16, 16 e 16 para o *cluster* de 2 usuários. Logo nossa redes têm 5 camadas sendo 9, 67, 67, 67, 125 neurônios para o *cluster* de 3 usuários e 6, 16, 16, 16, 25 neurônios para o *cluster* de 2 usuários (Figura 2.6).

Foi utilizada a função de ativação ReLU (Rectified Linear Unit), e *dropout* de 9/134 = 6,7% na camada 2 e 125/134 = 93,28% na camada 4 para o *cluster* de 3 usuários. Para o *cluster* de 2 usuários o *dropout* é de 6/31 = 19,3% na camada 2 e 25/31= 80,6% na camada 4. A taxa de aprendizagem é iniciada em 0.005 e decai 10% a cada 10% dos *TTI's* (3737). A probabilidade de exploração ϵ é iniciada em 1, decai linearmente por *TTI* e chega a 0.01 a cada 9% *TTI's* e se mantém assim até completar os 10% quando retorna

novamente para 1, cai a 0.01 em 19% e assim por diante.

A condição para atualizar os pesos da rede e agregar novas memórias ao agente é uma variação de 15% na taxa de chegada de qualquer usuário do *cluster*. Quando isso ocorre, 1 episódio de 50 passos são simulados na rede DQN utilizando a taxa média de chegada de pacotes de cada usuário do *cluster* e os ganhos correntes de canal. O parâmetro ϵ e a taxa de aprendizagem dependerão da posição no tempo do *TTI* atual como discutido anteriormente.

A Figura 3.26 mostra os resultados obtidos para o sistema em um cenário adaptativo com a aplicação de uma rede DQN. Percebe-se como a proposta 4 estabiliza com comportamento de alta vazão (c) e maior consumo de potência (d). Por outro lado, a proposta de Zhu converge para soluções com maior eficiência energética (e) e maiores perdas de pacote (a). As propostas 1, 2 e 3 têm comportamento intermediário e muito parecido entre si com destaque para eficiência energética da Proposta 3 (e). Ao todo, são armazenados 600 passos para o *cluster* de 3 usuários e 700 passos para o *cluster* de 2 usuários nas memórias de *replay* dos 37370 passos simulados.

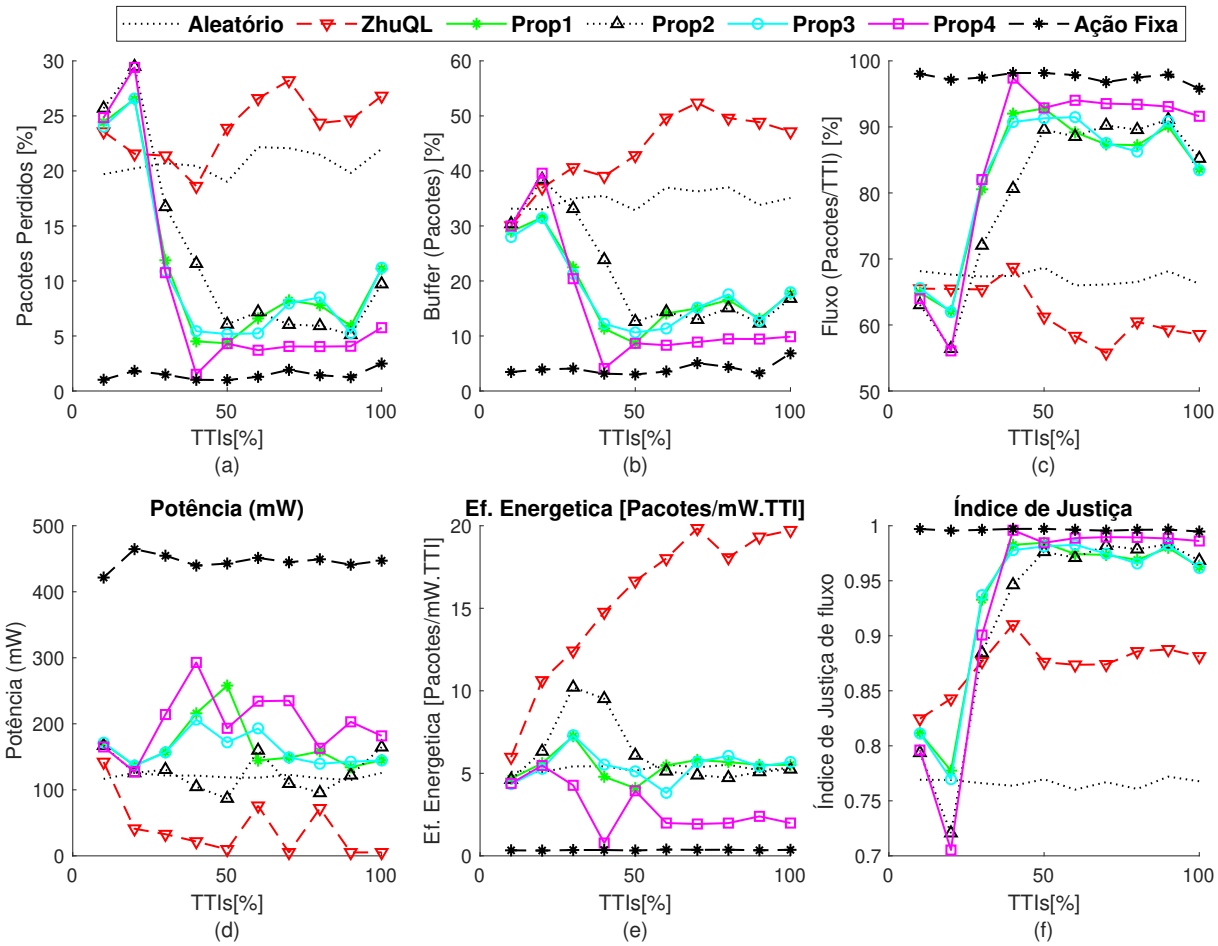

Figura 3.26 – Parâmetros de QoS versus tempo de simulação para rede DQN simulação assíncrona: (a) Pacotes Perdidos, (b) ocupação do *buffer*, (c) Fluxo de pacotes, (d) Potência, (e) Eficiência energética e (f) Índice de justiça

3.13 Utilizando QL-tabular para alocar potência por estado e ação

Nessa seção, apresenta-se um estudo de alocação de potência para obter valores de potência menos atrelados a uma BER máxima. Utiliza-se um Q-learning tabular que visa minimizar o atraso ponderado somado à potência. Dessa forma, a função de recompensa utilizada é dada pela equação:

$$R = \frac{\alpha.Buffer}{\lambda} + P \tag{3.8}$$

onde α é um peso cujo valor atribuído é 50, *buffer* o estado do *buffer* no instante e P a potência alocada para um único usuário.

O problema de alocação de potência contempla um único usuário e canal e utiliza aprendizado por reforço. Inicialmente explora valores de potência e ação (a depender de um valor de ϵ que decai a medida que a tabela é preenchida) e ao adquirir experiência vai escolhendo os valores que minimizam a função R (3.8).O parâmetro ϵ é a probabilidade de

exploração, em outras palavras, a probabilidade da potência e modo serem escolhidos de forma aleatória. Foi utilizado o banco de dados MariaDB no MATLAB para preencher a tabela (MARIADB,).

A função de valor de ação Q(s,a) é utilizada com taxa de desconto $\gamma = 0.9$ para as recompensas futuras, os modos permitidos são 0, 2, 4, 6 e 8 que representam sem transmissão, 4QAM, 16QAM, 64QAM e 256QAM respectivamente. Os parâmetros utilizados para a solução do problema de alocação de potência por QLearning tabular são: K = 1 usuário, L=2 pacotes, M=1 canal, J=4 modos, Ch=2 estados de canal, e λ = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20] pacotes/TTI. Para cada λ são simulados 20 TTI's 500 vezes, ou seja, um total de 10000 TTI's são simulados para cada valor de λ. Esse processo é realizado sem que seja monitorado o tempo de processamento com objetivo de consultar a tabela durante a execução assíncrona.

Após obter valores para potência para cada estado e ação possível do sistema e λ considerado ($P(\lambda, s, a)$) simula-se o atendimento do usuário com solução adaptativa utilizando modelo markoviano para alocação de recursos (escolha do modo de transmissão e alocação de canal para o usuário). O valor de potência alocado é obtido da tabela treinada. Verifica-se o valor atual de λ e o modo escolhido pelo agente e faz-se a consulta, ou seja, utiliza-se o λ mais próximo dos valores estipulados no treinamento (ajuste de parâmetros) do modelo.

Os resultados mostrados a seguir foram obtidos via simulações para K=5 usuários, L=2 Pacotes, M=5 canais, J=4 modos (4QAM, 16QAM, 64QAM e 256QAM), Ch=2 estados de canal, Tamanho máximo do *cluster* de 3 usuários e largura de banda de 640 MHz. Para fins de comparação, é mostrado também as soluções ao obter a potência através da restrição de BER mínima para os valores de BER = 10^{-3}, 10^{-4} e 10^{-5} e conforme a proposta 4. Além disso, também é mostrada a solução de Zhu ao respeitar a restrição de 10^{-4} para a BER. O tamanho do pacote é fixado em 3360 bytes e um ganho de 15 dB é considerado na antena.

Ao observar a Figura 3.27 nota-se como a alocação de potência feita pelo agente visando minimizar a espera e a potência consumida permite: Menores perdas de pacote e ocupação do *buffer* e maiores vazões e índice de justiça (curva em rosa $P4_{QLpower}$ das Figuras 3.27 (a), (b), (c) e (f) respectivamente). Há, por outro lado, um maior consumo de potência e diminuição da eficiência energética como observado nas Figuras 3.27 (d) e (e).

A restrição de BER mínima imprime uma variação na potência para as 3 curvas utilizando a proposta 4 ($P4_{0.1\%BER}$, $P4_{0.01\%BER}$ e $P4_{0.001\%BER}$). Quanto menor a BER admitida, maior a potência alocada. As soluções das 3 curvas (em verde , triângulo preto e ciano) são equivalentes para as Figuras 3.27 (a), (b), (c) e (f).

Na Figura 3.28 visualiza-se os valores de SNR e BER para as propostas ao longo

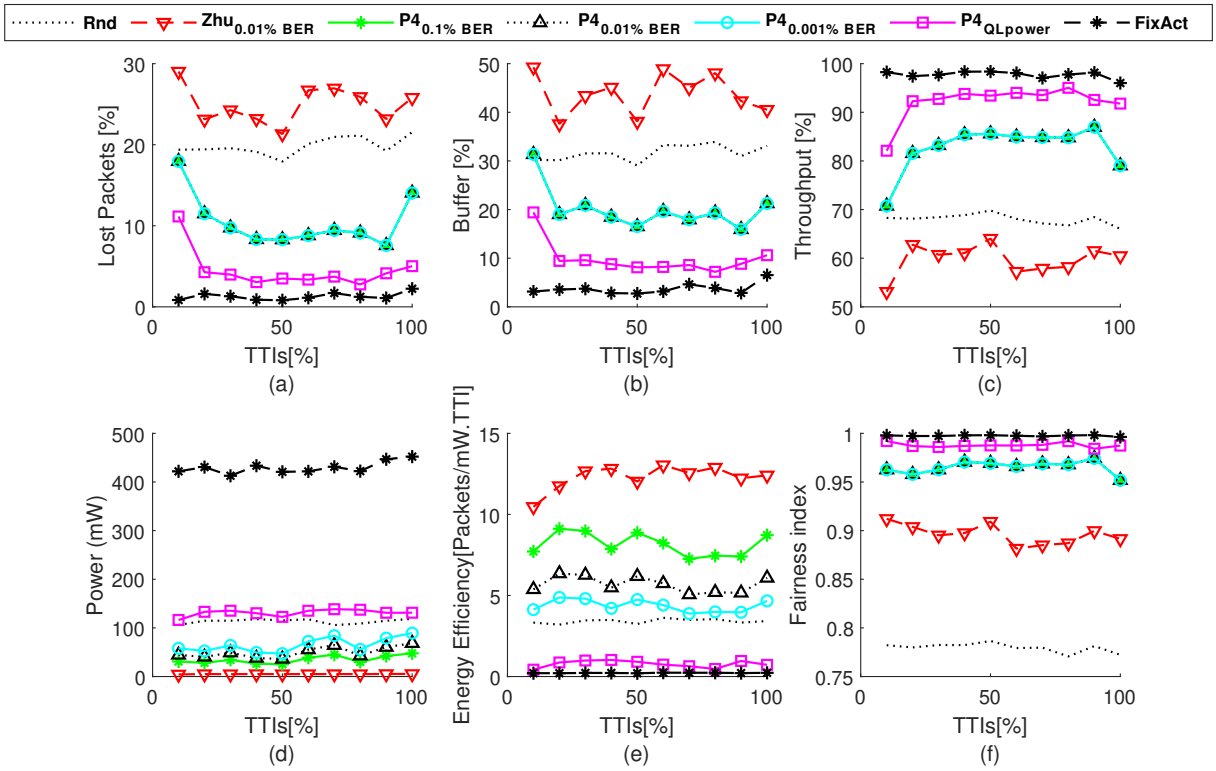

Figura 3.27 – Parâmetros de QoS versus tempo de simulação para solução com alocação de potência por reforço assíncrona: (a) Pacotes Perdidos, (b) ocupação do *buffer*, (c) Fluxo de pacotes, (d) Potência, (e) Eficiência energética e (f) Índice de justiça

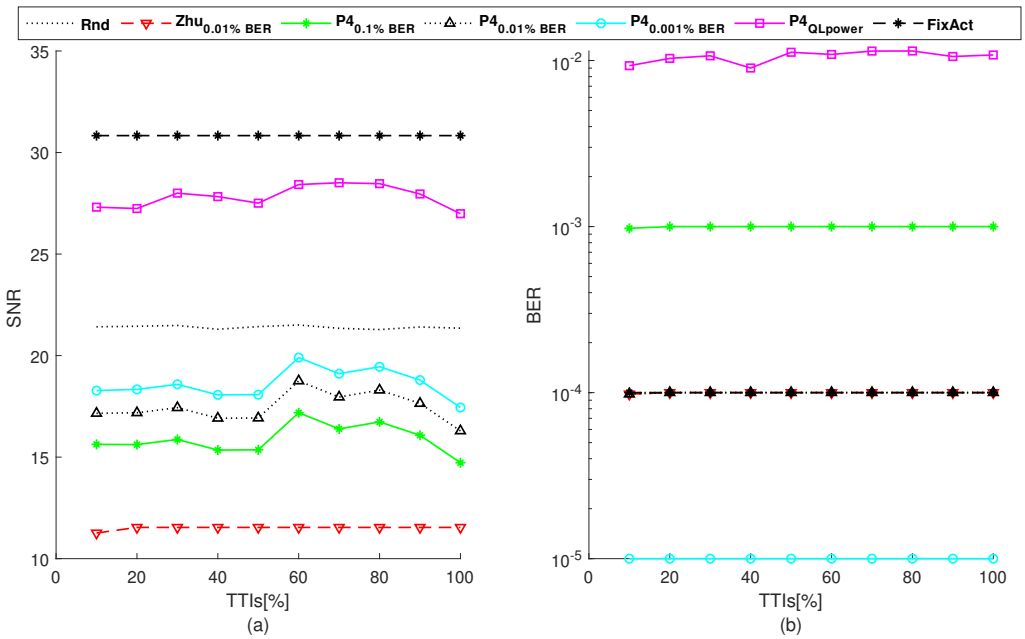

Figura 3.28 – SNR e BER versus tempo de simulação para solução com alocação de potência por reforço assíncrona: (a) SNR (b) BER

do tempo de simulação. Percebe-se como a curva em rosa ($P4_{QLpower}$) apresenta maiores valores de SNR e BER que as demais curvas. Isso evidencia que a proposta utilizando um agente treinado para alocação de potência utiliza modos com mais bits/símbolo que as demais propostas uma vez que é esperado um comportamento inverso entre SNR e BER para o mesmo modo e uma influência direta na BER a medida que aumenta-se a quantidade de bits/símbolo. Além disso, há maior liberdade na alocação de potência, levando a valores variáveis de BER que ficam em torno de 1%.

4 Conclusões

Nesta dissertação, considera-se que um sistema de comunicação CP-OFDM pode ser descrito por um modelo Markoviano e consequentemente um algoritmo de alocação de recursos baseado em aprendizado por reforço pode ser aplicado para escalonamento de recursos. Cada canal pode assumir estados diferentes (sub-portadoras) respeitando uma distribuição de *Rayleigh* para os desvanecimentos impostos ao sinal.

É adotado um modelo fluido de sistema de comunicação tomando por base a tecnologia LTE. Aplicou-se os algoritmos de aprendizagem por reforço com as funções de recompensa propostas neste sistema CP-OFDM LTE sendo analisados vários fatores como: variação da quantidade de usuários por *cluster*, variação do tamanho do buffer, variação do número de canais e de usuários, utilização de estratégia de seleção de melhores subportadoras, variações no tamanho da largura de banda, utilização de dados reais de chegada de pacotes e análise de desempenho dos algoritmos de alocação de recurso aplicados de forma adaptativa.

Na Seção 3.1, valida-se o modelo markoviano considerado utilizando dados reais de tráfego. Foi possível prever o comportamento estacionário do sistema simulado com as equações de modelo markoviano e teoria de filas para os parâmetros de QoS considerados.

Na seção 3.2 avaliou-se as vantagens do atendimento OFDM e de se utilizar uma proposta de função de recompensa levando em conta os pacotes perdidos. Diferentes taxas de chegada de pacotes foram consideradas mostrando como a proposta é mais sensível às taxas de chegada para o cenário OFDM.

Na seção 3.3 propôs-se diferentes funções de recompensa para o algoritmo baseado em aprendizado por reforço utilizando os parâmetros de QoS com vários efeitos como minimizar a quantidade de pacotes perdidos explicitando esse parâmetro na função, aumentar o uso do *buffer* (Proposta 3), aumentar a vazão do sistema, aumentar a eficiência energética, reduzir o consumo de potência. Entre as funções simuladas considera-se também a função recompensa utilizada em Zhu et al. (2018) que não explicita a quantidade de pacotes perdidos diretamente nem limita o uso da potência diretamente.

Na Seção 3.4 avaliou-se o impacto do aumento no número de usuários mantendo a quantidade de canais fixa no desempenho do sistema de comunicação com a utilização dos algoritmos de aprendizado por reforço. De maneira geral, as soluções que utilizam aprendizado por reforço proveram ao sistema melhores eficiências energéticas independente da quantidade de usuários. À medida que aumenta-se a quantidade de usuários (especificamente com mais de 3 usuários) a taxa de pacotes transmitidos cai, mas no caso da proposta (Equação 3.3) ainda fica acima da solução aleatória.

Na Seção 3.5, mostrou-se como o aumento do número de usuários e canais simultaneamente têm pouco impacto nos parâmetros de performance em termos percentuais. Observou-se um leve aumento na eficiência energética e pacotes perdidos e redução também pequena na taxa de transmissão de pacotes. A potência alocada aumenta lentamente enquanto o número de *clusters* é constante, e mais rapidamente ao aumentar o número de *clusters* (3 para 4 usuários, 6 para 7 usuários e 9 para 10 usuários).

Na Seção 3.6, verificou-se o efeito do aumento do tamanho do *buffer* nos parâmetros de desempenho do sistema. Todas as soluções são beneficiadas pelo aumento do tamanho máximo do *buffer* no que diz respeito a uma menor quantidade de pacotes perdidos. A solução de Zhu et al. (2018) tem a maior redução de pacotes perdidos com o aumento do *buffer*. As propostas 1, 3 e 4 atingem quantidades de pacotes perdidos compatíveis com a Ação Fixa (aproximadamente 0 %) e eficiência energética melhores que as soluções que não utilizam aprendizado por reforço. A proposta 2 perde em média até 5% de pacotes e possui a melhor eficiência energética entre as propostas quando o *buffer* é maior ou igual a 4.

Na Seção 3.7, mostrou-se o efeito do aumento da quantidade de canais ao manter-se a quantidade de usuários fixada em 5. Notou-se uma melhoria do desempenho de todas as soluções no que diz respeito a pacotes perdidos, ocupação do *buffer* e pacotes transmitidos até atingir-se 5 canais. A potência média aumenta e a eficiência energética reduz de 1 a 5 canais. Porém, os resultados são praticamente constantes para mais de 5 canais, com leve redução na potência alocada e leve aumento na eficiência energética devido a maior disponibilidade de canais.

Na Seção 3.8, apresentou-se uma análise dos resultados com o aumento de estados de canal. Basicamente há um aumento na eficiência energética e na taxa média de pacotes perdidos. Esse aumento é mais notável quando a taxa de chegada média se aproxima da ordem de grandeza do *buffer=2*, ou seja, quando tem-se baixas taxas de chegada de pacotes.

Na Seção 3.9, avaliou-se o efeito de utilizar 16 canais extras por canal de transmissão. O principal impacto constatado foi a redução da potência alocada e aumento da eficiência energética provocados pelo ganho estatístico (variedade de amostras) no valor médio do ganho dos canais.

Na Seção 3.10, avaliou-se um cenário onde são considerados dados reais para a chegada de pacotes e adaptação síncrona do algoritmo de alocação de recursos baseado em aprendizagem por reforço. Observou-se que as relações entre as diferentes propostas se mantiveram qualitativamente, mas a quantidade de pacotes perdidos foi bem superior à simulada com dados sintéticos.

Na Seção 3.11, mostrou-se o efeito de utilizar dados reais para a chegada de pacotes

com adaptação assíncrona do modelo, isso é, levar em conta o tempo de processamento para o cálculo dos valores dos parâmetros no modelo adaptado e obtenção da política ao se simular o atendimento. Além disso, houve aumento da largura de banda com objetivo de aumentar a capacidade do sistema simulado na seção 3.10. A principal conclusão foi que o tempo de processamento é o gargalo para a capacidade de se adaptar do agente. O uso de menos usuários por *cluster* é útil para reduzir o tempo de processamento.

Na Seção 3.12, apresentou-se os resultados em termos de desempenho do sistema ao se utilizar uma DQN - *Deep Q-Network* para aprender a política que maximiza a função Q de valor $p(s) = Arg(a), max(Q(s, a))$. Os resultados obtidos com a rede DQN de aprendizado profundo e por reforço são promissores especialmente por não utilizar o modelo do sistema e permitir abordagem adaptativa. Os resultados são condizentes com o outro método que é determinístico e se baseia no modelo estocástico do sistema. A principal limitação encontrada diz respeito às ações que devem ser discretas e não podem crescer sem limites o que limita o número máximo de usuários por *cluster*.

Na Seção 3.13 soluciona-se o problema de alocação de potência separadamente e de forma mais livre da restrição de BER considerado nos demais capítulos. É utilizado o aprendizado por reforço clássico tabular para decidir qual a melhor potência a ser alocada para cada modo e estado do sistema adotando também um intervalo vasto de λ com objetivo de minimizar o atraso e potência consumida. Os resultados mostram que a solução converge para regiões com maior vazão e menor perda de pacotes e ocupação do *buffer* quando é permitido certa liberdade para o valor de BER. Porém, apresenta menor eficiência energética que as soluções anteriores.

Entre as principais contribuições do trabalho pode-se citar soluções adaptativas tanto utilizando um modelo markoviano dinâmico como uma rede DQN produzindo diferentes valores para parâmetros de desempenho como vazão e eficiência energética.

Assim, em resumo, observou-se que as funções de recompensa podem prover resultados distintos. O uso explícito da quantidade de pacotes perdidos na função de recompensa contribui para reduzir a perda de pacotes e aumentar a taxa de transmissão. A proposta 2 se mostrou interessante em termos de eficiência energética e vazão quando o tamanho do *buffer* é suficientemente grande comparado a taxa de chegada de pacotes.

Por fim, observou-se que algoritmos baseados em aprendizado por reforço podem prover melhoria de desempenho em termos de eficiência energética e/ou vazão, quantidade de pacotes perdidos, ocupação do *buffer* e índice de justiça, para sistemas de comunicação TDM e OFDM e que a escolha da função utilidade pode influenciar nas soluções obtidas e impactar o comportamento do escalonamento de recursos.

Referências

3GPP. *Study on channel model for frequencies from 0.5 to 100 GHz (Release 15)*. [S.l.], 2018. Citado 9 vezes nas páginas 14, 19, 20, 23, 24, 27, 50, 51 e 61.

3GPP. *5G;Telecommunication management;Study on system and functional aspects of energy efficiency in 5G networks (Release 16)*. [S.l.], 2020. Citado na página 28.

CARNEIRO, D. et al. Aprendizado por reforço para escalonamento de recursos em sistema sem fio multiportadora com ondas milimétricas utilizando modelo markoviano. In: *Anais da IX Escola Regional de Informática de Goiás*. Porto Alegre, RS, Brasil: SBC, 2021. p. 12–25. ISSN 0000-0000. Disponível em: <https://sol.sbc.org.br/index.php/erigo/article/view/18430>. Citado na página 33.

CARNEIRO, D. P.; CARDOSO, Á. A.; VIEIRA, F. H. T. Aprendizado por reforço baseado em modelo markoviano para alocaçao de recursos em sistema multiportadora com ondas milimétricas. in: *Simpósio Brasileiro de Automação Inteligente-SBAI*. 2021. Citado na página 33.

CARVALHO, P. de et al. Uma ferramenta de simulação de redes multimídia baseada no modelo fluido. Citado na página 30.

CHO, Y. S. et al. *MIMO-OFDM wireless communications with MATLAB*. [S.l.]: John Wiley & Sons, 2010. Citado na página 86.

FORD, R. et al. Markov channel-based performance analysis for millimeter wave mobile networks. In: IEEE. *2017 IEEE Wireless Communications and Networking Conference (WCNC)*. [S.l.], 2017. p. 1–6. Citado 4 vezes nas páginas 19, 31, 33 e 40.

Hong Shen Wang; Moayeri, N. Finite-state markov channel-a useful model for radio communication channels. *IEEE Transactions on Vehicular Technology*, v. 44, n. 1, p. 163–171, 1995. Citado 3 vezes nas páginas 20, 31 e 33.

JAIN, R. Channel models: A tutorial. In: WASHINGTON UNIV. ST. LOUIS, DEPT. CSE. *WiMAX forum AATG*. [S.l.], 2007. v. 10. Citado na página 23.

JAIN, R.; DURRESI, A.; BABIC, G. Throughput fairness index: An explanation. In: *ATM Forum contribution*. [S.l.: s.n.], 1999. v. 99, n. 45. Citado na página 63.

KUHN, H. W. The hungarian method for the assignment problem. *Naval research logistics quarterly*, Wiley Online Library, v. 2, n. 1-2, p. 83–97, 1955. Citado na página 25.

LEE, H.; GIRNYK, M.; JEONG, J. Deep reinforcement learning approach to mimo precoding problem: Optimality and robustness. *arXiv preprint arXiv:2006.16646*, 2020. Citado na página 54.

MARIADB. *MariaDB Knowledge Base, url : https://mariadb.com/kb/en/, Acessado : 14-06-2022*. Citado na página 98.

MATZ, G.; HLAWATSCH, F. Fundamentals of time-varying communication channels. In: *Wireless Communications Over Rapidly Time-Varying Channels*. [S.l.]: Elsevier, 2011. p. 1–63. Citado na página 22.

MAWI. Mawi working group traffic archive. *https://mawi.wide.ad.jp/mawi/*, 2019. Citado 6 vezes nas páginas 13, 60, 87, 88, 91 e 95.

MNIH, V. et al. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013. Citado na página 56.

PATTETI, K.; KUMAR, T.; KALITKAR, K. M-qam ber and ser analysis of multipath fading channels in long term evolutions (lte). *International Journal of Signal Processing, Image Processing and Pattern Recognition(IJSIP)*, Vol.9, p. 361–368, 01 2016. Citado na página 19.

PROAKIS, J.; SALEHI, M. *Digital Communications*. [S.l.]: McGraw-Hill, 2008. (McGraw-Hill International Edition). ISBN 9780071263788. Citado na página 23.

PUJOLLE, G. *Mobile and Wireless Communication Networks: IFIP 19th World Computer Congress, TC-6, 8th IFIP/IEEE Conference on Mobile and Wireless Communications Networks, August 20-25, 2006, Santiago, Chile*. [S.l.]: Springer, 2006. v. 211. Citado na página 30.

RAPPAPORT, T. S. et al. *Wireless communications: principles and practice*. [S.l.]: prentice hall PTR New Jersey, 1996. v. 2. Citado 2 vezes nas páginas 38 e 61.

S, M.; MOHAMAD, A. A. A study of efficient power consumption wireless communication techniques/modules for internet of things (iot) applications. Scientific Research Publishing, 2016. Citado na página 19.

SANGAIAH, A. K. et al. Iot resource allocation and optimization based on heuristic algorithm. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 20, n. 2, p. 539, 2020. Citado na página 19.

SUTTON, R. S.; BARTO, A. G. *Reinforcement learning: An introduction*. [S.l.]: MIT press, 2018. Citado 4 vezes nas páginas 31, 41, 46 e 47.

VASCONCELOS, M. M.; CARDOSO, Á. A.; VIEIRA, F. H. T. Algoritmo baseado em aprendizado por reforço e modelo markoviano para alocação de recursos em um sistema internet das coisas cognitivo. in: *XXXVIII Simpósio Brasileiro de Telecomunicações e Processamento de Sinais - SBrT*. 2020. Citado na página 34.

ZARRINKOUB, H. *Understanding LTE with MATLAB: from mathematical modeling to simulation and prototyping*. [S.l.]: John Wiley & Sons, 2014. Citado 2 vezes nas páginas 11 e 30.

Zhu, J. et al. A new deep-q-learning-based transmission scheduling mechanism for the cognitive internet of things. *IEEE Internet of Things Journal*, v. 5, n. 4, p. 2375–2385, 2018. Citado 26 vezes nas páginas 11, 19, 20, 22, 27, 29, 31, 33, 42, 43, 44, 45, 61, 62, 63, 65, 68, 69, 70, 72, 78, 86, 90, 93, 101 e 102.