

Giovana Rodrigues Neves dos Santos

Estudo de Algoritmos de Escalonamento em Redes 5G Utilizando Simulações

Uberlândia, MG

2025

Giovana Rodrigues Neves dos Santos

Estudo de Algoritmos de Escalonamento em Redes 5G Utilizando Simulações

Trabalho de Conclusão de Curso de Engenharia Eletrônica e de Telecomunicações da Universidade Federal de Uberlândia - UFU - Campus Santa Mônica, como requisito parcial para a obtenção do título de Graduação em Engenharia Eletrônica e de Telecomunicações.

Universidade Federal de Uberlândia - UFU
Faculdade de Engenharia Elétrica - FEELT

Orientador Prof. Dr. Éderson Rosa da Silva

Uberlândia, MG

2025

Giovana Rodrigues Neves dos Santos

Estudo de Algoritmos de Escalonamento em Redes 5G Utilizando Simulações

Trabalho de Conclusão de Curso de Engenharia Eletrônica e de Telecomunicações da Universidade Federal de Uberlândia - UFU - Campus Santa Mônica, como requisito parcial para a obtenção do título de Graduação em Engenharia Eletrônica e de Telecomunicações.

Trabalho aprovado em 14 de Março de 2025.

COMISSÃO EXAMINADORA

Prof. Dr. Éderson Rosa da Silva
Orientador

Prof. Dr. Alexandre Coutinho Mateus
Membro Avaliador

Prof. Me. Gustavo Nozella Rocha
Membro Avaliador

Uberlândia, MG
2025

Agradecimentos

Primeiramente, gostaria de expressar minha profunda gratidão à minha família, especialmente aos meus pais, Fabio Santos e Simene Santos, e à minha irmã, Luana Santos, que sempre estiveram ao meu lado, oferecendo apoio incondicional e amor. Seus sacrifícios e incentivos foram fundamentais para que eu pudesse trilhar este caminho e continuar meus estudos com determinação e foco. A força e a resiliência que encontrei em vocês foram verdadeiros pilares nesta jornada, e sou eternamente grata por todo o suporte que recebi.

Agradeço de coração ao meu namorado, Guilherme Almeida, que não é apenas um companheiro incrível, mas também uma fonte constante de compreensão e motivação. Seu amor e apoio inabaláveis me inspiraram a buscar sempre o melhor de mim. Sou imensamente grata por tê-lo ao meu lado em todos os momentos, especialmente nos mais desafiadores, onde sua presença fez toda a diferença.

Meus amigos desempenharam um papel crucial nesta trajetória. Sou grata a todos que me incentivaram a escolher este curso e a cada um que fez parte da minha caminhada. Nossas histórias, risadas e o apoio mútuo tornaram essa jornada ainda mais significativa, e as memórias que construímos juntos ficarão gravadas para sempre em meu coração.

Não posso deixar de agradecer aos professores do curso pelos valiosos ensinamentos. Um agradecimento especial ao meu orientador, Éderson Rosa, cuja orientação paciente e conhecimento profundo foram essenciais na elaboração deste trabalho. Também sou grata ao professor Luiz Cláudio Theodoro, que acreditou em meu potencial e me proporcionou valiosas lições e oportunidades de aprendizado, tanto profissionais quanto pessoais, que foram fundamentais para meu desenvolvimento.

A colaboração e o incentivo de todos vocês foram essenciais para a realização deste projeto, e eu sou eternamente grata por cada contribuição. A jornada até aqui não teria sido a mesma sem o apoio e a presença de cada um de vocês.

*“Ora o último inimigo que há de ser aniquilado é a morte.”
(1 Coríntios 15:26)*

Resumo

As redes de comunicação móvel 5G exigem soluções eficientes para atender aos requisitos de Qualidade de Serviço (QoS) em cenários com alta densidade de usuários e diferentes tipos de tráfego. Este trabalho investiga o desempenho de algoritmos de escalonamento em redes 5G, utilizando simulações de nível de sistema para analisar a eficácia dos algoritmos *Round Robin* (RR), *Best Channel Quality Indicator* (Best CQI) e *Quality of Service Aware Scheduler* (QAS). A pesquisa é baseada em cenários considerando variações no número de usuários distribuídos entre serviços em Tempo Real (RT) e Não Tempo Real (NRT). Os resultados das simulações mostraram que o QAS se destacou por sua capacidade de adaptação às exigências específicas de tráfegos RT, garantindo menor latência e menores Taxas de Erro e Bloco (BLER), mesmo em condições de alta carga de rede. O Best CQI apresentou bom desempenho em situações onde a qualidade do canal favorecia serviços NRT, enquanto o RR ofereceu uma distribuição uniforme de recursos, porém com limitações em se adaptar para diferentes tipos de tráfego. As análises indicam que o QAS é a melhor opção para cenários que exigem alta QoS, com capacidade para maximizar a vazão total e proporcionar uma experiência de uso satisfatória para diferentes perfis de tráfego. Este estudo contribui para o entendimento de como diferentes algoritmos de escalonamento podem ser aplicados para otimizar a performance de redes 5G, auxiliando na escolha da solução mais adequada para atender às demandas crescentes dessas redes.

Palavras-chave: *QoS Aware Scheduler. Round Robin. Best Channel Quality Indicator. Redes 5G.*

Abstract

5G mobile communication networks require efficient solutions to meet Quality of Service (QoS) requirements in scenarios with high user density and different types of traffic. This work investigates the performance of scheduling algorithms in 5G networks, using system-level simulations to analyze the effectiveness of the Round Robin (RR), Best Channel Quality Indicator (Best CQI), and Quality of Service Aware Scheduler (QAS) algorithms. The research is based on scenarios considering variations in the number of users distributed among Real-Time (RT) and Non-Real-Time (NRT) services. The simulation results showed that the QAS stood out for its ability to adapt to the specific demands of RT traffic, ensuring lower latency and lower Block Error Rates (BLER), even under high network load conditions. The Best CQI performed well in situations where channel quality favored NRT services, while the RR provided a uniform distribution of resources but had limitations in adaptive efficiency for different types of traffic. The analyses indicate that the QAS is the best option for scenarios that require high QoS, with the capacity to maximize overall throughput and provide a satisfactory user experience for different traffic profiles. This study contributes to understanding how different scheduling algorithms can be applied to optimize the performance of 5G networks, assisting in choosing the most suitable solution to meet the growing demands of these networks.

Keywords: QoS Aware Scheduler. Round Robin. Best Channel Quality Indicator. 5G Networks.

Lista de ilustrações

Figura 1 – Categorias de serviços 5G	19
Figura 2 – Visão geral da rede 5G	20
Figura 3 – Arquitetura rede 5G	21
Figura 4 – Estrutura de Slots na Rede 5G para Diferentes Espaçamentos de Subportadora	25
Figura 5 – Estrutura de quadros suportada pelo NR	26
Figura 6 – Fluxograma da abordagem de escalonamento Round Robin.	29
Figura 7 – Fluxograma da abordagem de escalonamento Best CQI.	31
Figura 8 – Fluxograma da abordagem de escalonamento QAS.	33
Figura 9 – Modelo de tráfego <i>Constant Rate</i>	34
Figura 10 – Modelo de tráfego <i>Full buffer</i>	34
Figura 11 – Modelo de tráfego VoIP	34
Figura 12 – Modelo de tráfego <i>Gaming</i>	35
Figura 13 – Modelo de tráfego Video <i>streaming</i>	35
Figura 14 – Interface do simulador aberta no MATLAB	37
Figura 15 – Vista superior do campus Santa Mônica - UFU	41
Figura 16 – Vista da UFU - Campus Santa Mônica, com indicação da localização dos usuários e BSs	45
Figura 17 – Vazão em Mbps em relação à distribuição dos usuários no cenário 3	46
Figura 18 – Vazão média em Mbps dos três escalonadores para cada tipo de tráfego no cenário 1	48
Figura 19 – Latência por tipo de tráfego para o escalonador QAS no cenário 1	49
Figura 20 – BLER por tipo de tráfego para o escalonador RR no cenário 1	53
Figura 21 – Média geral de BLER considerando os 3 cenários	55

Lista de tabelas

Tabela 1 – Parâmetros da estrutura de slots para diferentes valores de μ .	25
Tabela 2 – Distribuição de usuários por tipo de tráfego	40
Tabela 3 – Limites geográficos do campus	40
Tabela 4 – Parâmetros das construções	41
Tabela 5 – Parâmetros da área de interferência	42
Tabela 6 – Parâmetros das BSs	42
Tabela 7 – Parâmetros de Simulação	43
Tabela 8 – Vazão Média em Mbps para diferentes tipos de usuários e escalonadores	47
Tabela 9 – Vazões médias em Mbps por escalonador em diferentes cenários	48
Tabela 10 – Estatísticas de Latência por Tipo de Tráfego (Cenário 1)	50
Tabela 11 – Estatísticas de Latência por Tipo de Tráfego (Cenário 2)	51
Tabela 12 – Estatísticas de Latência por Tipo de Tráfego (Cenário 3)	52
Tabela 13 – Média de BLER para diferentes tipos de usuários e escalonadores	53

Lista de abreviaturas e siglas

3GPP	<i>3rd Generation Partnership Project</i>
4G	4 ^a Geração
5G	5 ^a Geração
5GC	Core da Rede 5G
6G	6 ^a Geração
AF	Função de Aplicação
AMC	Modulação e Codificação Adaptativa
AMF	Função de Gerenciamento de Acesso e Mobilidade
AMR	<i>Adaptive Multi-Rate</i>
AUSF	Função de Servidor de Autenticação
BestCQI	<i>Best Channel Quality Indicator</i>
BLER	Taxa de Erro de Bloco
BS	Estação Base
BSR	Relatório de Status do <i>Buffer</i>
CQI	Indicador de Qualidade de Canal
CP-OFDM	<i>Cyclic Prefix Orthogonal Frequency Division Multiplexing</i>
CUPS	Separação de Plano de Controle e Usuário
DCIs	Indicações de Controle de Dados
DL-SCH	<i>Downlink Shared Channel</i>
DN	Rede de Dados
DFT-S-OFDM	<i>Discrete Fourier Transform Spread Orthogonal Frequency Division Multiplexing</i>
ECDF	Função de Distribuição Acumulada Empírica
eMBB	Banda Larga Móvel Aprimorada

FDD	<i>Frequency Division Duplexing</i>
gNB	<i>gNodeB</i>
IIoT	Internet das Coisas Industrial
IP	<i>Internet Protocol</i>
LTE	<i>Long-Term Evolution</i>
LOS	Linha de Visada
MAC	Controle de Acesso ao Meio
Mbps	Megabits por segundo
MCS	Esquema de Modulação e Codificação
MEC	Computação de Borda Multiacesso
mMTC	Comunicações Massivas do Tipo Máquina
MIMO	<i>Multiple Input Multiple Output</i>
mmWave	Ondas Milimétricas
ms	Milissegundos
NEF	Função de Exposição de Rede
NFs	Funções de Rede
NG-RAN	Rede de Acesso de Rádio de Próxima Geração
NLOS	Sem Linha de Visada
NRT	Não Tempo Real
NR	<i>New Radio</i>
NRF	Função de Repositório de Rede
NSSAI	Informações de Assistência para Seleção de Fatias de Rede
NSSF	Função de Seleção de Fatiamento de Rede
OFDM	<i>Orthogonal Frequency Division Multiplexing</i>
PCF	Função de Controle de Políticas
PDCCH	<i>Physical Downlink Control Channel</i>

PPP	Processo de Ponto de <i>Poisson</i>
PRBs	Blocos de Recursos Físicos
QAS	<i>Quality of Service Aware Scheduler</i>
QoS	Qualidade de Serviço
RBs	Blocos de Recursos
RR	<i>Round Robin</i>
RT	Tempo Real
SBA	Arquitetura Baseada em Serviços
SCS	Espaçamento de Subportadoras
SFI	Indicador de Formato de <i>Slot</i>
SINR	<i>Signal to Interference and Noise Ratio</i>
SLs	<i>System Level Simulator</i>
SMS	Serviço de Mensagens Curtas
SMF	Função de Gerenciamento de Sessão
SUPIs	Identificador Permanente de Assinatura
TDD	<i>Time Division Duplexing</i>
TTI	Intervalo de Transmissão
UDM	Gerenciamento Unificado de Dados
UDP	<i>User Datagram Protocol</i>
UE	Equipamento de Usuário
UFU	Universidade Federal de Uberlândia
UL-SCH	<i>Uplink Shared Channel</i>
UPF	Função de Plano do Usuário
USIM	<i>Universal Subscriber Identity Module</i>
VoIP	<i>Voice Over Internet Protocol</i>
VR	Realidade Virtual

VCCS	<i>Vienna Cellular Communications Simulators</i>
WRR	<i>Weighted Round Robin</i>

Sumário

1	INTRODUÇÃO	15
1.1	Justificativa	16
1.2	Objetivo Geral	17
1.3	Organização do trabalho	17
2	REFERENCIAL TEÓRICO	19
2.1	Rede 5G	19
2.2	Arquitetura da rede	20
2.3	Escalonamento	23
2.4	Escalonamento de Recursos na rede 5G	24
2.5	Algoritmos de escalonamento	27
2.5.1	Round Robin	27
2.5.2	Best CQI	30
2.5.3	Quality of Service Aware Scheduling	30
2.6	Modelos de Tráfego	32
2.7	<i>Vienna 5G System Level Simulator</i>	35
2.8	Trabalhos relacionados	36
2.9	Considerações Finais	38
3	METODOLOGIA DE PESQUISA	39
3.1	Ambiente de simulação	39
3.2	CrITÉrios de Avaliação	43
3.2.1	Vazão	43
3.2.2	Latência	44
3.2.3	BLER	44
3.3	Considerações Finais	44
4	RESULTADOS E DISCUSSÕES	45
4.1	Resultados das simulações	46
4.1.1	Vazão	46
4.1.2	Latência	49
4.1.3	BLER	52
4.2	Considerações Finais	55
5	CONCLUSÕES	56

REFERÊNCIAS BIBLIOGRÁFICAS	58
APÊNDICE A – CÓDIGO <i>LAUNCHER</i>	60
APÊNDICE B – CÓDIGO PARA O ESCALONADOR RR	74
APÊNDICE C – CÓDIGO PARA O ESCALONADOR BEST CQI . .	79
APÊNDICE D – CÓDIGO PARA O ESCALONADOR QAS	84

1 Introdução

O tráfego de dados móveis testemunhou um crescimento exponencial ao longo da última década, aumentando cerca de 4.000 vezes. Nos últimos 15 anos, esse crescimento é ainda mais notável, com um aumento de quase 400 milhões de vezes (CISCO, 2016). Essa explosão no consumo de dados reflete não apenas a crescente dependência da sociedade em dispositivos móveis, mas também a demanda incessante por conectividade rápida e confiável. Nesse contexto, a implementação do sistema 5G surge como uma resposta crucial para lidar com o aumento fenomenal no tráfego dos anos anteriores (OSSEIRAN et al., 2020).

O avanço das redes 5G marca uma revolução nas comunicações, proporcionando uma infraestrutura robusta e ágil para uma variedade de aplicações inovadoras. Este crescimento exponencial apresenta desafios significativos, destacando a necessidade crucial de otimizar a utilização dos recursos para suportar aplicações potenciais. A eficiência na alocação de recursos é, portanto, fundamental para atender a essa diversidade de demandas e garantir uma experiência do usuário superior. Nesse contexto, o escalonador de recursos desempenha um papel crítico ao determinar a ordem de alocação dos recursos disponíveis na rede (MAZIERO, 2019). O algoritmo escolhido para o escalonador influencia diretamente o desempenho da rede, permitindo uma alocação eficiente e rápida dos recursos para atender às necessidades variadas dos usuários, que podem abranger transmissão de dados, voz, vídeo e outros serviços. Essa seleção criteriosa do algoritmo de escalonamento é essencial para garantir uma distribuição equitativa dos recursos, maximizando a eficiência da rede e a satisfação do usuário.

A fim de investigar a alocação de recursos na rede 5G, este trabalho se propõe a analisar três algoritmos de escalonamento amplamente utilizados: o *Round Robin* (RR), *Best Channel Quality Indicator* (Best CQI) e o *Quality of Service Aware* (QAS). O RR é conhecido por sua simplicidade e popularidade ao distribuir os recursos de forma equitativa entre os usuários, atribuindo a cada um deles um intervalo de tempo de acesso à rede de maneira sequencial. No entanto, esse método não considera a qualidade do canal de cada usuário, nem busca atender a requisitos mínimos de desempenho, o que pode resultar em uma alocação de recursos menos eficiente em situações em que há demandas específicas de desempenho (SOUZA et al., 2018).

Por outro lado, o Best CQI é um algoritmo que prioriza os usuários com a melhor qualidade de canal, conforme indicado pelo Indicador de Qualidade de Canal (CQI) (GROUP, 2019). Esse algoritmo busca maximizar o desempenho da rede atribuindo recursos aos usuários que apresentam as condições mais favoráveis de comunicação, proporcionando

uma experiência de comunicação mais estável e eficiente. No entanto, a implementação eficiente do Best CQI requer um mecanismo robusto de *feedback* de qualidade do canal e uma coordenação adequada entre os nós da rede.

Em contrapartida, o QAS representa uma abordagem mais sofisticada e adaptável para o gerenciamento de recursos na rede 5G (SHIYAHIN; SCHWARZ; RUPP, 2022). Ao considerar tanto a urgência quanto a qualidade dos diferentes fluxos de tráfego, o QAS prioriza tarefas críticas ou serviços sensíveis ao atraso, garantindo uma alocação de recursos mais adaptada às necessidades específicas de cada aplicação (GUARDIEIRO; SILVA; MOREIRA, 2018). Entretanto, a implementação eficaz do QAS também apresenta desafios, incluindo a necessidade de algoritmos complexos de tomada de decisão e a coordenação entre os diferentes elementos da rede para garantir que os requisitos de Qualidade de Serviço (QoS) sejam atendidos de forma adequada.

Ao comparar e analisar o desempenho desses algoritmos, busca-se identificar suas vantagens e limitações em diferentes cenários de tráfego e carga na rede 5G.

1.1 Justificativa

A implementação da rede 5G representa um marco significativo na evolução das comunicações, revolucionando a conectividade global. No entanto, o sucesso dessa nova geração de redes depende não apenas da sua infraestrutura física, mas também da eficácia dos algoritmos de gerenciamento de recursos, como os escalonadores de rede.

Os escalonadores desempenham um papel vital na otimização da alocação de recursos nas redes, influenciando diretamente a qualidade da experiência do usuário. Em um cenário em que a demanda por largura de banda e desempenho de rede é cada vez maior, é fundamental compreender como esses algoritmos podem contribuir para a eficiência e a qualidade de serviço na rede 5G.

Ao estudar os escalonadores de rede, como o RR, Best CQI e QAS, pode-se identificar as melhores práticas para distribuir os recursos disponíveis de forma equitativa e eficiente entre os usuários e aplicações. Isso é essencial para garantir uma experiência de usuário superior, minimizar atrasos, congestionamentos e garantir a navegabilidade na internet em ambientes de rede cada vez mais complexos e exigentes.

Portanto, este trabalho justifica-se pela necessidade de explorar e compreender o papel dos escalonadores de rede na rede 5G, visando otimizar a utilização dos recursos disponíveis e melhorar a qualidade de serviço para os usuários.

1.2 Objetivo Geral

Este trabalho tem como objetivo investigar e comparar o desempenho dos algoritmos de escalonamento RR, Best CQI e QAS na rede 5G. Para alcançar esse objetivo, foram realizadas simulações utilizando o *Vienna 5G System Level Simulator* (SLS) da família *Vienna Cellular Communications Simulators* (VCCS) (MULLER et al., 2018). A escolha desse simulador se justifica pela sua robustez na modelagem de cenários complexos na rede 5G. O *Vienna 5G System Level Simulator* é reconhecido por sua capacidade de representar diversos aspectos da rede, incluindo a camada física, mobilidade do usuário e variabilidade de canal. Essa capacidade permite uma análise mais realista do ambiente operacional e dos impactos dos algoritmos de escalonamento no desempenho geral da rede 5G.

Ao realizar essas simulações, busca-se identificar as vantagens e limitações de cada algoritmo de escalonamento em diferentes cenários de tráfego, visando assim contribuir para o aprimoramento da gestão de recursos na rede 5G e para o desenvolvimento de estratégias mais eficientes para garantir uma experiência de usuário superior.

1.3 Organização do trabalho

Este trabalho está estruturado conforme descrito a seguir, cada um abordando aspectos essenciais do estudo sobre algoritmos de escalonamento em redes 5G.

O Capítulo 2, intitulado Referencial Teórico, apresenta os fundamentos teóricos da pesquisa, abordando as características das redes 5G, os algoritmos de escalonamento e os tipos de tráfego utilizados nas simulações. Além disso, inclui uma descrição do simulador utilizado e uma análise dos trabalhos relacionados que contribuem para a compreensão do tema.

Por meio do Capítulo 3, Metodologia de Pesquisa, são detalhados os procedimentos adotados para a realização das simulações. Este capítulo apresenta os passos necessários para implementar os cenários simulados, descrevendo os critérios de avaliação que são utilizados para analisar o desempenho dos algoritmos de escalonamento em diferentes contextos de tráfego.

O Capítulo 4, Resultados e Discussões, contém os resultados obtidos nas simulações, apresentando uma análise detalhada do desempenho dos escalonadores em relação aos critérios estabelecidos na metodologia. Além disso, neste capítulo realiza-se uma discussão aprofundada sobre as implicações dos resultados encontrados e sua relevância para a pesquisa em redes 5G.

Por fim, o Capítulo 5, Conclusões, sintetiza as principais descobertas e contribuições do trabalho, refletindo sobre os resultados alcançados e suas implicações para pesquisas futuras, ressaltando a importância de continuar explorando o desempenho das redes 5G e

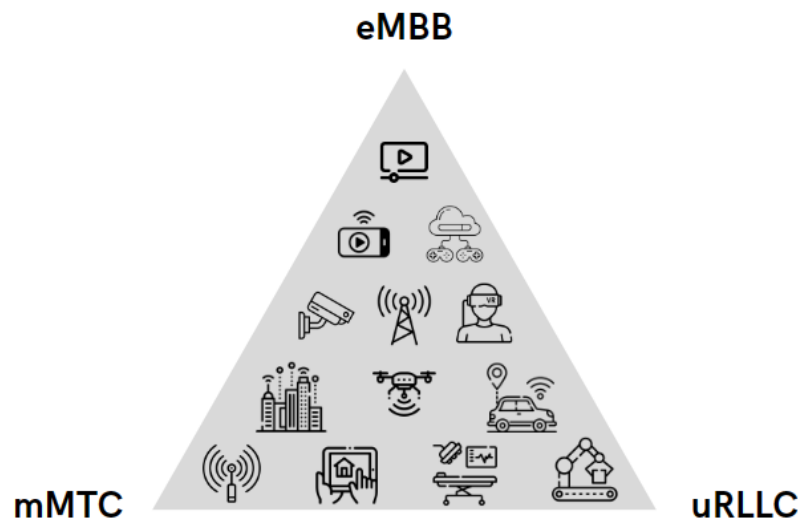
suas aplicações.

2 Referencial Teórico

2.1 Rede 5G

As redes móveis evoluíram significativamente ao longo do tempo, passando de ações simples, como chamadas telefônicas, para suportar tecnologias mais avançadas capazes de oferecer uma variedade de serviços sofisticados. Durante esse processo, houve um foco constante na expansão das faixas de espectro e na melhoria das taxas de dados e capacidades de tratamento de tráfego ([OSSEIRAN et al., 2020](#)). No entanto, a rede 5G vai além da simples busca por maiores taxas de transmissão, sendo construída sobre três pilares fundamentais, como ilustra a Figura 1:

Figura 1 – Categorias de serviços 5G



Fonte: Adaptado de ([ANACOM, 2023](#)).

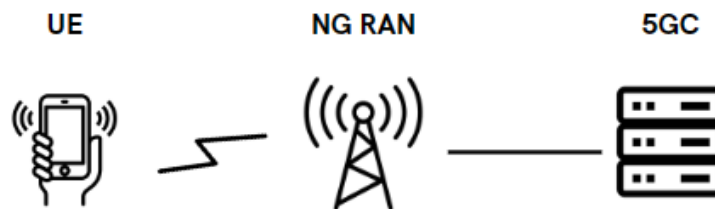
- Banda Larga Móvel Aprimorada (eMBB): Focado em fornecer altas taxas de transferência de dados para aplicativos como *streaming* de vídeo em alta definição e jogos online;
- Comunicações Ultra Confiáveis com Baixa Latência (uRLLC): Projetado para oferecer comunicações altamente confiáveis e com latência extremamente baixa para aplicações críticas, como sistemas de controle em tempo real e cirurgias remotas; e
- Comunicações Massivas do Tipo Máquina (mMTC): Destinado a conectar uma grande quantidade de dispositivos com baixo consumo de energia, como sensores em cidades inteligentes e redes de sensores sem fio para aplicações na agricultura.

A fim de atender a esses requisitos, foi essencial para a rede 5G não apenas otimizar as tecnologias já existentes, mas também adotar uma arquitetura flexível e escalável, capaz de se adequar aos diferentes cenários presentes e futuros (GUPTA; JHA, 2015).

2.2 Arquitetura da rede

Esquemáticamente, o sistema 5G utiliza os mesmos elementos das gerações anteriores: um Equipamento de Usuário (UE), composto por um dispositivo do usuário e um USIM (*Universal Subscriber Identity Module*); a Rede de Acesso de Rádio de Próxima Geração (NG-RAN), que é composta por Estações Base (BS); e o Core da Rede (5GC), como mostrado na Figura 2.

Figura 2 – Visão geral da rede 5G



Fonte: Adaptado de (SULTAN, 2022).

As Estações Base da rede 5G, designadas como *gNodeB* (gNB), desempenham diversas funções cruciais e fazem parte da NG-RAN. Além de garantir a segurança das comunicações e gerenciar recursos de rádio para otimizar a eficiência espectral, a gNB controla a mobilidade dos usuários e supervisiona o estabelecimento de conexões de rádio com os UEs. Ela também realiza a gestão da qualidade de serviço, assegurando uma experiência de usuário consistente e de alta qualidade (ARCHITECTURE, 2019).

Adicionalmente, a gNB desempenha tarefas fundamentais, como o roteamento de pacotes de dados, a coordenação entre diferentes tecnologias (como *New Radio* (NR), *Long-Term Evolution* (LTE)), proporcionando múltiplas formas de conectividade, e o compartilhamento de rede de acesso de rádio entre operadoras (SIROTKIN, 2020). Sua atuação abrange operações em diversas camadas, incluindo as camadas física, de Controle de Acesso ao Meio (MAC), de rede, bem como nas camadas de transporte e aplicação, garantindo o funcionamento eficaz e seguro da rede 5G.

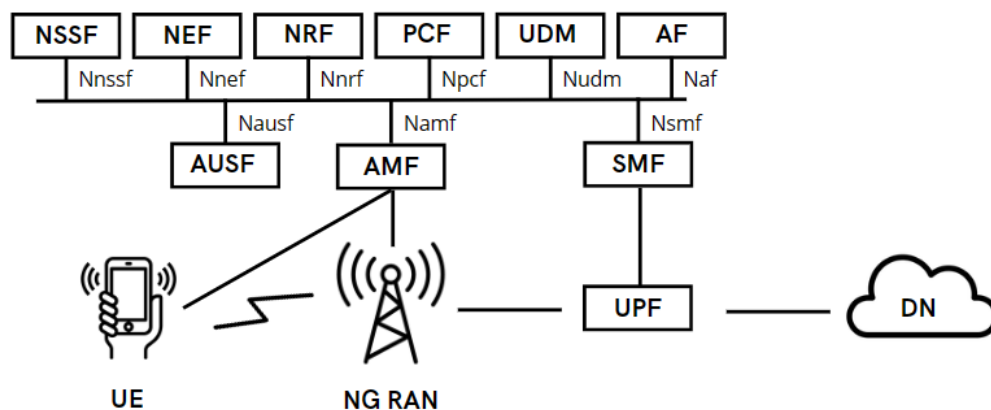
O 5GC é o elemento central que gerencia e controla todas as comunicações na rede. Ele é composto por uma Arquitetura Baseada em Serviços (SBA), onde os elementos arquiteturais são definidos como Funções de Rede (NFs), em vez das tradicionais entidades de rede. Nessa abordagem, cada NF disponibiliza seus serviços para outras NFs autorizadas

e para qualquer usuário autorizado a utilizá-los, por meio de interfaces compartilhadas (SULTAN, 2022).

Além disso, o 5GC segue novos paradigmas em várias áreas, como a Separação de Plano de Controle e Usuário (CUPS), que possibilita modelos de implantação inovadores. Ao distribuir a funcionalidade do plano de usuário mais próxima à rede de acesso, o 5GC aumenta a eficiência e viabiliza a Computação de Borda Multiacesso (MEC) (SIROTKIN, 2020). A capacidade de suportar diversas tecnologias de acesso por meio de um núcleo comum e oferecer acesso eficiente a serviços localizados e centralizados impulsiona uma ampla gama de aplicativos, desde Realidade Virtual (VR) até Internet das Coisas Industrial (IIoT). O fatiamento de rede (*network slicing*) amplia ainda mais a flexibilidade e isolamento, permitindo a personalização de grupos de usuários. Além disso, as redes privadas no 5G atendem à demanda por conectividade confiável e privacidade de dados em setores como automação industrial e logística, impulsionando novos modelos de negócios e experiências de usuário.

A Figura 3 ilustra a arquitetura de referência da rede 5G com base em interfaces orientadas a serviços. A seguir, são apresentados os principais componentes dessa arquitetura e suas respectivas funções:

Figura 3 – Arquitetura rede 5G



Fonte: Adaptado de (SULTAN, 2022).

- **Função de Gerenciamento de Acesso e Mobilidade (AMF):** Gerencia o acesso e a mobilidade dos dispositivos na rede. Suas funções principais incluem registro, estabelecimento de conexão, gerenciamento de mobilidade e controle de integridade do plano de controle. Além disso, a AMF coordena a troca de informações de contexto entre as diferentes áreas de cobertura, garantindo uma mobilidade contínua e eficiente;
- **Função de Gerenciamento de Sessão (SMF):** Responsável por estabelecer, modificar e liberar as sessões de dados para os usuários. Além do controle de sessões, a SMF

gerencia funções de ancoragem de *Internet Protocol* (IP), definição de políticas de roteamento e, em alguns casos, a funcionalidade de *roaming*, assegurando que a conectividade do usuário seja mantida em movimento;

- **Função de Servidor de Autenticação (AUSF):** Executa a autenticação de usuários e dispositivos que tentam acessar a rede. Gera vetores de autenticação, valida identidades e colabora com o Gerenciamento Unificado de Dados (UDM) para armazenar e validar as credenciais dos assinantes. A AUSF garante que apenas dispositivos autorizados acessem a rede, protegendo a integridade dos dados e a segurança das comunicações;
- **Função de Plano do Usuário (UPF):** Processa e encaminha pacotes de dados dos usuários, conectando a rede 5G aos sistemas externos, como a Internet. A UPF implementa políticas de QoS, realiza o controle de carga, gerenciamento de tráfego e suporte à ancoragem de IP, além de participar da mobilidade no plano do usuário, assegurando a entrega contínua dos dados;
- **Função de Seleção de Fatiamento de Rede (NSSF):** Seleciona as fatias de rede (*network slices*) que servirão ao dispositivo, determinando as Informações de Assistência para Seleção de Fatias de Rede (NSSAI). Além disso, a NSSF identifica as AMFs adequadas para a comunicação com o usuário, garantindo que cada dispositivo se conecte à fatia correta de rede, de acordo com os requisitos de serviço;
- **Função de Exposição de Rede (NEF):** Atua como intermediário entre a rede e serviços externos, expondo capacidades e eventos da rede de forma segura. Assim, a NEF permite que aplicativos externos acessem serviços e informações da rede 5G, facilitando a integração com provedores de serviços, enquanto mantém controle rigoroso de acesso e segurança;
- **Função de Repositório de Rede (NRF):** Mantém um repositório de funções de rede disponíveis e facilita a descoberta de serviços. Além disso, a NRF registra as instâncias de funções de rede e seus serviços suportados, permitindo que outras funções encontrem e utilizem dinamicamente as funcionalidades necessárias para atender aos usuários;
- **Função de Controle de Políticas (PCF):** Define e aplica políticas para controle de QoS, acesso e cobrança, assegurando que os recursos da rede sejam usados de forma eficiente e de acordo com as necessidades dos serviços. A PCF interage com a SMF e a UPF para garantir que as políticas definidas sejam implementadas e aplicadas corretamente durante as comunicações;
- **Gerenciamento Unificado de Dados (UDM):** Gerencia os dados dos assinantes, incluindo perfis, Identificador Permanente de Assinatura (SUPIs) e credenciais de

autenticação. O UDM também desempenha um papel importante no suporte a serviços como Serviço de Mensagens Curtas (SMS), além de gerar parâmetros para autenticação e autorização dos dispositivos na rede;

- **Função de Aplicação (AF):** Interage com a rede 5G para influenciar políticas de roteamento e priorização de tráfego, dependendo dos requisitos do serviço. A AF é utilizada, por exemplo, por aplicativos de comunicação crítica para assegurar que as exigências de baixa latência e alta prioridade sejam atendidas pela rede; e
- **Rede de Dados (DN):** Refere-se à rede de dados externa conectada ao núcleo 5G, como a Internet ou redes empresariais. O DN fornece conectividade para os dispositivos e permite a troca de informações com outros sistemas e serviços que não fazem parte do núcleo da rede 5G.

2.3 Escalonamento

Os escalonadores desempenham um papel crucial no gerenciamento eficiente dos recursos em sistemas computacionais e de comunicação. Esses algoritmos são responsáveis por determinar a ordem de execução de diferentes tarefas ou processos, otimizando o uso dos recursos disponíveis para garantir um desempenho adequado do sistema (KORHONEN JUHA, 2023).

Na área de redes de comunicação, os escalonadores são usados para gerenciar o acesso aos recursos compartilhados, como largura de banda e tempo de transmissão. Em redes sem fio, o escalonamento de recursos refere-se à atividade de alocação de recursos de rádio necessários para a transmissão de dados para os equipamentos de usuário e é realizada a cada Intervalo de Transmissão (TTI), sendo fundamental para viabilizar a troca de dados entre os usuários e a rede (CARDOSO, 2015). Os escalonadores são encarregados de determinar quais dispositivos têm permissão para transmitir dados e em que ordem, considerando fatores como prioridade de serviço, qualidade do canal e requisitos de QoS. Esses algoritmos são cruciais para garantir um desempenho eficiente do sistema de comunicação, especialmente em redes móveis, onde a vazão de dados depende significativamente do escalonamento adequado (KORHONEN JUHA, 2023).

O escalonador de dados é uma parte crucial da estação base, localizado na camada MAC. Sua função principal é coordenar as transmissões de dados nos links de *uplink* e *downlink* para todos os dispositivos dentro de sua área de cobertura (KORHONEN JUHA, 2023). O escalonador determina a distribuição de recursos com o objetivo de maximizar a eficiência da célula, garantindo uma alocação justa dos recursos disponíveis.

O 3GPP não especifica o tipo de escalonador ou política de alocação de recursos a serem utilizados, deixando essa escolha a critério dos fabricantes e desenvolvedores

de equipamentos (ALI-YAHIYA, 2011). Como resultado dessa falta de definição, uma variedade de algoritmos tem sido proposta na literatura para melhorar a qualidade de serviço em sistemas móveis, cada um com suas próprias características e abordagens de tomada de decisão. Neste trabalho, três dos escalonadores mais comuns são analisados: RR, Best CQI e QAS, cujas características e desempenhos são detalhados nos próximos capítulos.

2.4 Escalonamento de Recursos na rede 5G

A tecnologia 5G faz uso de uma variedade de técnicas de multiplexação, dentre as quais se destacam a *Cyclic Prefix Orthogonal Frequency Division Multiplexing* (CP-OFDM) e a *Discrete Fourier Transform Spread Orthogonal Frequency Division Multiplexing* (DFT-S-OFDM). Enquanto a CP-OFDM é empregada tanto no *downlink* quanto no *uplink*, o DFT-S-OFDM é uma alternativa específica para o *uplink*. Esta última se distingue pela sua capacidade de reduzir a relação entre a potência de pico pela potência média e por oferecer uma implementação menos complexa em comparação com outras estratégias de mitigação dessa relação (SIROTKIN, 2020).

O 5G suporta uma ampla gama de serviços em diversas frequências. Para atingir essa meta, são utilizadas várias numerologias definidas pelo Espaçamento de Subportadoras (SCS) no contexto do OFDM. O termo "numerologia" engloba as diferentes configurações de espaçamento de subportadoras (Δf) e duração do Prefixo Cíclico (CP) em um símbolo *Orthogonal Frequency Division Multiplexing* (OFDM). No 5G-NR, são disponibilizadas cinco numerologias distintas, com a numerologia convencional do LTE servindo como ponto de partida. A numerologia base opera com um espaçamento de subportadoras de $\Delta f = 15kHz$, enquanto as outras numerologias são múltiplos desse valor, determinados pelo fator de μ associado a cada uma delas, conforme a Equação 1:

$$SCS = 2^\mu \cdot 15kHz. \quad (1)$$

A Tabela 1 apresenta os principais parâmetros associados a cada numerologia, incluindo o espaçamento de subportadoras, a quantidade de *slots* por subquadro e o tipo de prefixo cíclico adotado. Observa-se que, para o prefixo cíclico normal, cada *slot* contém 14 símbolos OFDM independentemente do SCS, e o número de *slots* por subquadro varia de 1 a 16. No caso do prefixo cíclico estendido, há uma redução na quantidade de símbolos por *slot* para 12. A Figura 4 ilustra a relação entre o espaçamento das subportadoras, a duração dos slots e sua distribuição ao longo do tempo.

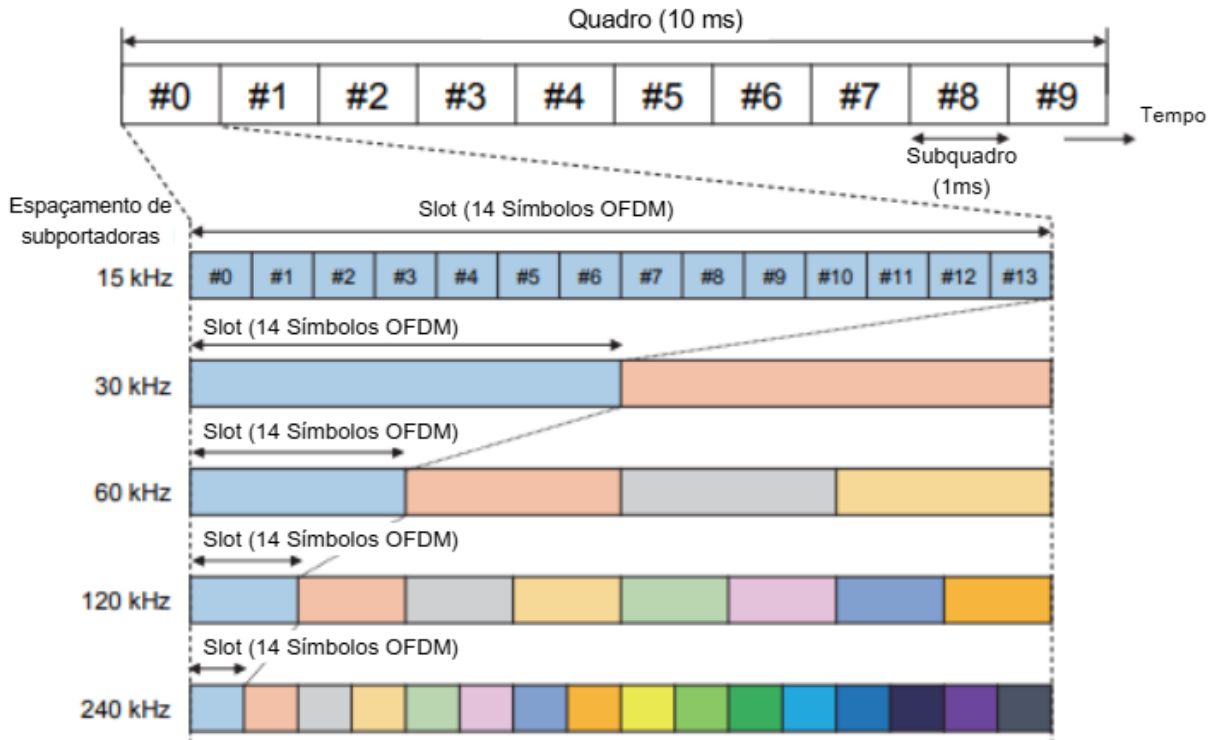
A determinação do parâmetro μ , com variação de 0 a 4, correspondendo a diferentes espaçamentos de subportadoras de 15 kHz a 240 kHz, é impactada por vários elementos, incluindo o ambiente operacional, a faixa de frequência, interferências de rádio frequência

Tabela 1 – Parâmetros da estrutura de slots para diferentes valores de μ .

μ	Num Símbolos OFDM	Num Slots	SCS (kHz)	Prefixo Cíclico
0	14	1	15	Normal
1	14	2	30	Normal
2	14	4	60	Normal
2	12	4	60	Estendido
3	14	8	120	Normal
4	14	16	240	Normal

e o tipo de serviço específico. Para cenários mMTC, as numerologias com espaçamento de subportadoras reduzido são recomendadas, pois possibilitam a conexão eficiente de um grande número de dispositivos com eficiência energética. Por outro lado, para serviços eMBB, que exigem altas taxas de dados e largura de banda substancial, as numerologias com SCS intermediário são mais adequadas. Já para aplicações que demandam baixa latência, as numerologias com SCS mais longo são preferenciais devido à sua capacidade de lidar com símbolos de curta duração, essenciais para garantir respostas rápidas (KIHERO; SOLAIJA; ARSLAN, 2019).

Figura 4 – Estrutura de Slots na Rede 5G para Diferentes Espaçamentos de Subportadora

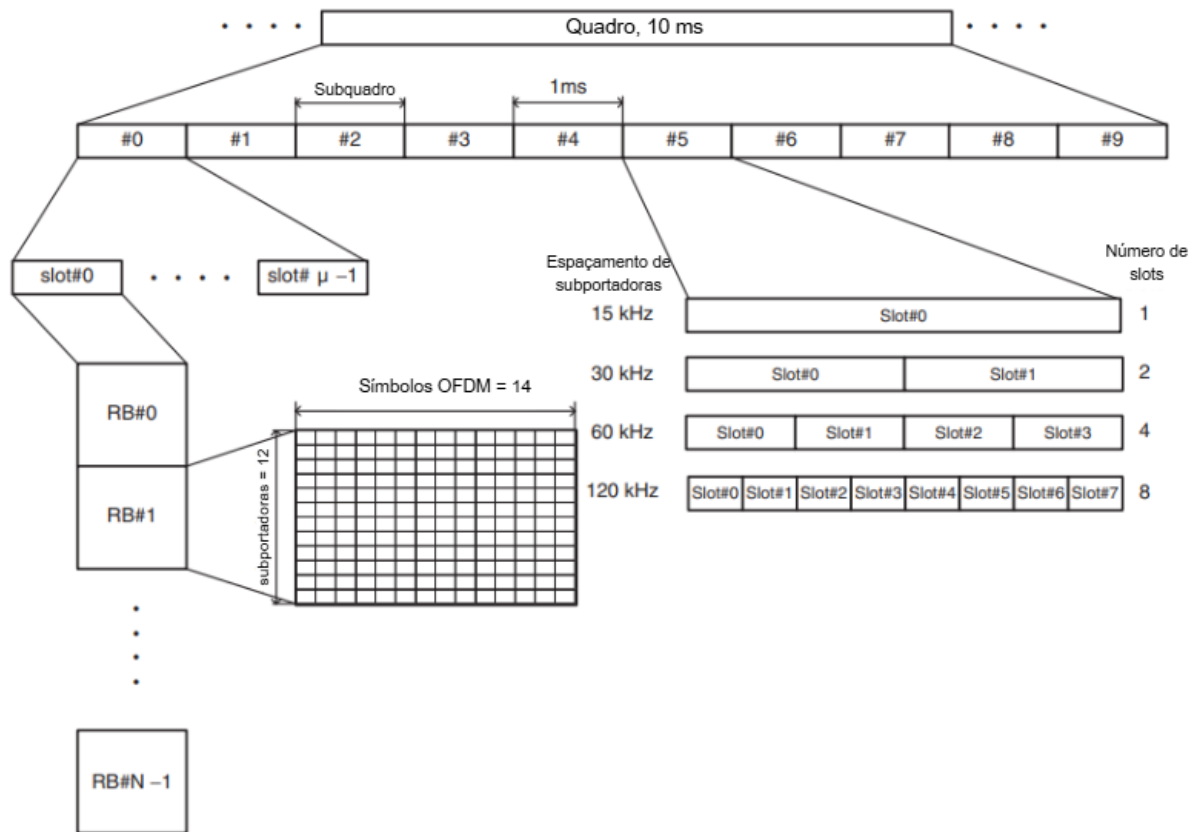


Fonte: Adaptado de (MONEM, 2024)

O NR, como o LTE, utiliza quadros no domínio do tempo com 10 ms de duração, divididos em dois semiquadros de cinco subquadros cada. Cada subquadro tem 1 ms e pode ser dividido em *slots*, ajustando a escala do parâmetro μ (3GPP-TS-38.211, 2024). A

Figura 5 ilustra essa estrutura de quadros do 5G NR, mostrando a divisão dos subframes em slots, bem como as diferentes configurações de espaçamento de subportadoras.

Figura 5 – Estrutura de quadros suportada pelo NR



Fonte: Adaptado de (SIROTKIN, 2020)

O NR emprega Blocos de Recursos Físicos (PRBs) para agrupar subportadoras e símbolos OFDM, formando unidades básicas de alocação de recursos. O número de PRBs varia de acordo com o espaçamento entre subportadoras e a faixa de frequência, com suporte a larguras de banda de 100 MHz a 400 MHz. Uma característica crucial é o alinhamento de frequência dos limites dos PRBs, conhecido como "Ponto A", permitindo a multiplexação eficiente de transmissões com diferentes configurações na mesma célula ou para o mesmo dispositivo (SIROTKIN, 2020). Essa flexibilidade assegura a interoperabilidade entre dispositivos e redes.

A estrutura de quadro do 5G NR suporta tanto operações TDD (*Time Division Duplexing*) quanto FDD (*Frequency Division Duplexing*), com atribuição de *downlink* ou *uplink* em uma granularidade de um único símbolo OFDM, em contraste com o LTE. A classificação de cada símbolo como "*downlink*", "*flexível*" ou "*uplink*" é dinamicamente determinada pelo Indicador de Formato de Slot (SFI), permitindo diversas configurações de escalonamento para o sistema TDD.

Na gNB, a camada MAC integra escalonadores dinâmicos responsáveis por alocar

recursos da camada física para os canais de transporte do *Downlink Shared Channel* (DL-SCH) e *Uplink Shared Channel* (UL-SCH). É importante notar que o escalonamento para *uplink* é feito de forma independente do *downlink* (FERREIRA, 2021).

Através de um processo contínuo e adaptável, o escalonador distribui Blocos de Recursos (RBs) em diferentes TTIs e *slots* para atender às necessidades de cada dispositivo e aplicação. Essa distribuição é feita a partir de uma série de fatores, como o Relatório de Status do *Buffer* (BSR), CQI, QoS e as características específicas de cada serviço.

Uma etapa crucial desse processo é a implementação de técnicas de Modulação e Codificação Adaptativa (AMC). Dado que as condições do canal podem variar consideravelmente entre usuários, torna-se inviável utilizar apenas uma modulação. Essas técnicas escolhem o Esquema de Modulação e Codificação (MCS) mais apropriado, considerando as condições do canal e a qualidade da transmissão, garantindo uma transmissão eficaz dos dados ao otimizar a taxa de transmissão.

Além disso, o escalonador cria as Indicações de Controle de Dados (DCIs), as quais são enviadas aos dispositivos conectados pelo canal *Physical Downlink Control Channel* (PDCCH). Estas DCIs informam aos dispositivos sobre a alocação de recursos em cada subquadro, permitindo que eles ajustem suas operações de transmissão e recepção conforme as instruções recebidas.

2.5 Algoritmos de escalonamento

Os escalonadores são construídos com base em algoritmos de escalonamento, os quais são organizados como um conjunto modular de elementos. Esses algoritmos são frequentemente divididos em políticas que desempenham diferentes funções durante o processo de escalonamento, facilitando assim o desenvolvimento e a compreensão dos algoritmos (MUNIZ, 1996).

Ao decidir como alocar recursos entre diferentes usuários, os escalonadores visam maximizar a eficiência do sistema, uma medida que abrange diversos critérios de desempenho, como a experiência de cada UE em termos de taxa de dados, a justiça na distribuição de recursos e o atraso médio de pacotes. A escolha da métrica de desempenho a ser otimizada é fundamental, pois influencia diretamente a forma como o escalonador resolve disputas de recursos e atende às necessidades variadas dos usuários (ALQAHTANI; ALHASSANY, 2013).

2.5.1 Round Robin

O algoritmo de escalonamento Round Robin é reconhecido por sua simplicidade e abordagem equitativa na distribuição de recursos entre os processos. Ao contrário de

algoritmos mais complexos, o RR não leva em consideração a qualidade do canal do usuário ou requisitos mínimos de desempenho. Sua operação é baseada na atribuição de uma fração de tempo igual para cada processo, de forma circular, tratando todos os processos sem prioridades de maneira uniforme. Essa abordagem visa alcançar uma distribuição igualitária dos recursos disponíveis, aumentando assim, o índice de justiça entre os usuários (IOSIF; BăNICĂ, 2012). No entanto, essa abordagem pode implicar em uma baixa taxa de transferência de dados dos usuários e um atraso médio maior para determinadas aplicações em comparação com outros algoritmos mais complexos, sendo difícil priorizar qualquer tipo de tráfego utilizando esta estratégia.

O algoritmo de escalonamento Round Robin é descrito pelos seguintes passos (SILBERSCHATZ, 2013):

1. Manutenção de Estruturas de Dados:

- O escalonador mantém uma fila de requisições de RBs prontas e uma lista de requisições bloqueadas.

2. Adição e Remoção de Requisições:

- Quando uma nova requisição de RB é criada, ela é adicionada ao final da fila de requisições prontas.
- Quando uma requisição é atendida e os dados são transmitidos, ela é removida das estruturas de dados de escalonamento.

3. Seleção do Próximo RB:

- O escalonador seleciona a requisição na cabeça da fila de requisições prontas e aloca os RBs disponíveis para esta requisição.

4. Reinserção após Uso dos RBs:

- Quando uma requisição termina seu *quântum* de tempo e ainda tem dados a serem transmitidos, ela é movida para o final da fila de requisições prontas.

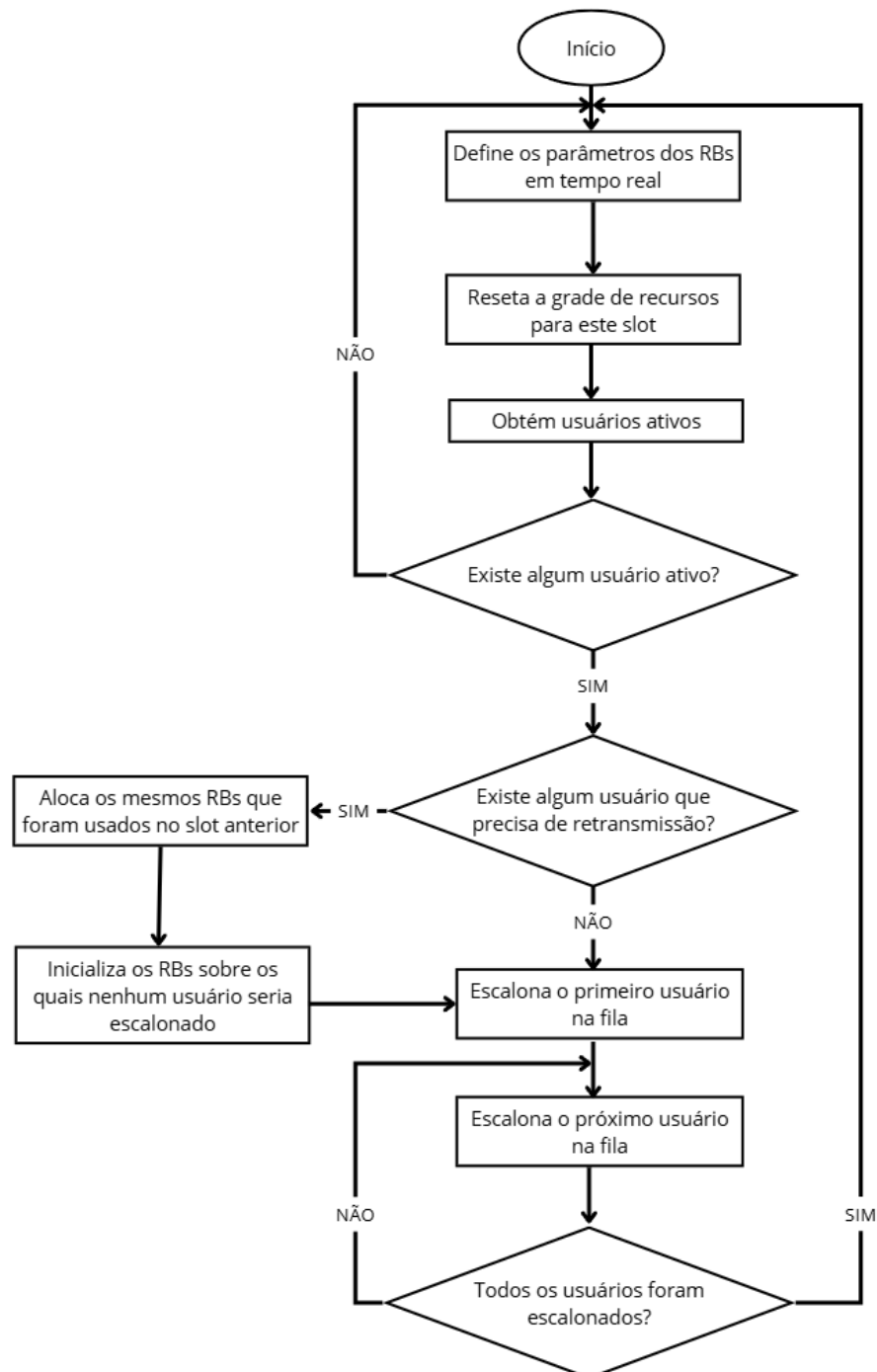
5. Manipulação de Eventos:

- O manipulador de eventos realiza as seguintes ações:
 - a) **Solicitação de Entrada/Saída ou Bloqueio:** Quando uma requisição faz uma solicitação de entrada/saída ou é bloqueada por qualquer motivo (como espera de RBs), sua entrada é removida da fila de requisições prontas e adicionada à lista de bloqueados.

- b) **Conclusão de Entrada/Saída ou Desbloqueio:** Quando a operação de entrada/saída termina ou o bloqueio é resolvido, a requisição é movida de volta para o final da fila de requisições prontas.

A Figura 6 ilustra o processo descrito.

Figura 6 – Fluxograma da abordagem de escalonamento Round Robin.



Fonte: Adaptado de (QUEIROZ, 2022)

2.5.2 Best CQI

O Best CQI é um algoritmo de escalonamento desenvolvido para otimizar a alocação de recursos em redes móveis, projetado para atribuir blocos de recursos aos usuários com as melhores condições de enlace de rádio ou qualidade de canal em cada TTI. Para realizar esse escalonamento, a estação base transmite sinais de referência, conhecidos como pilotos de *downlink*, para os UEs. Esses sinais são então utilizados pelos dispositivos para realizar medições e calcular seus respectivos CQIs, com valores mais altos indicando melhores condições de canal (HABAEBI et al., 2013).

Apesar de seu potencial para aumentar a vazão individual na célula, o Best CQI apresenta desvantagens significativas, especialmente no que diz respeito à equidade entre os usuários. Essa abordagem tende a favorecer usuários com valores de CQI mais elevados, frequentemente aqueles que estão próximos à estação base. Como consequência, os usuários localizados nas bordas da célula, que normalmente apresentam CQIs mais baixos, enfrentam uma menor probabilidade de serem escalonados. Isso pode resultar em uma distribuição desigual da capacidade da rede, afetando a qualidade do serviço para uma parte significativa dos usuários que dependem da cobertura da rede em áreas de sinal mais fraco.

O processo de tomada de decisão desse escalonador é mostrado na Figura 7.

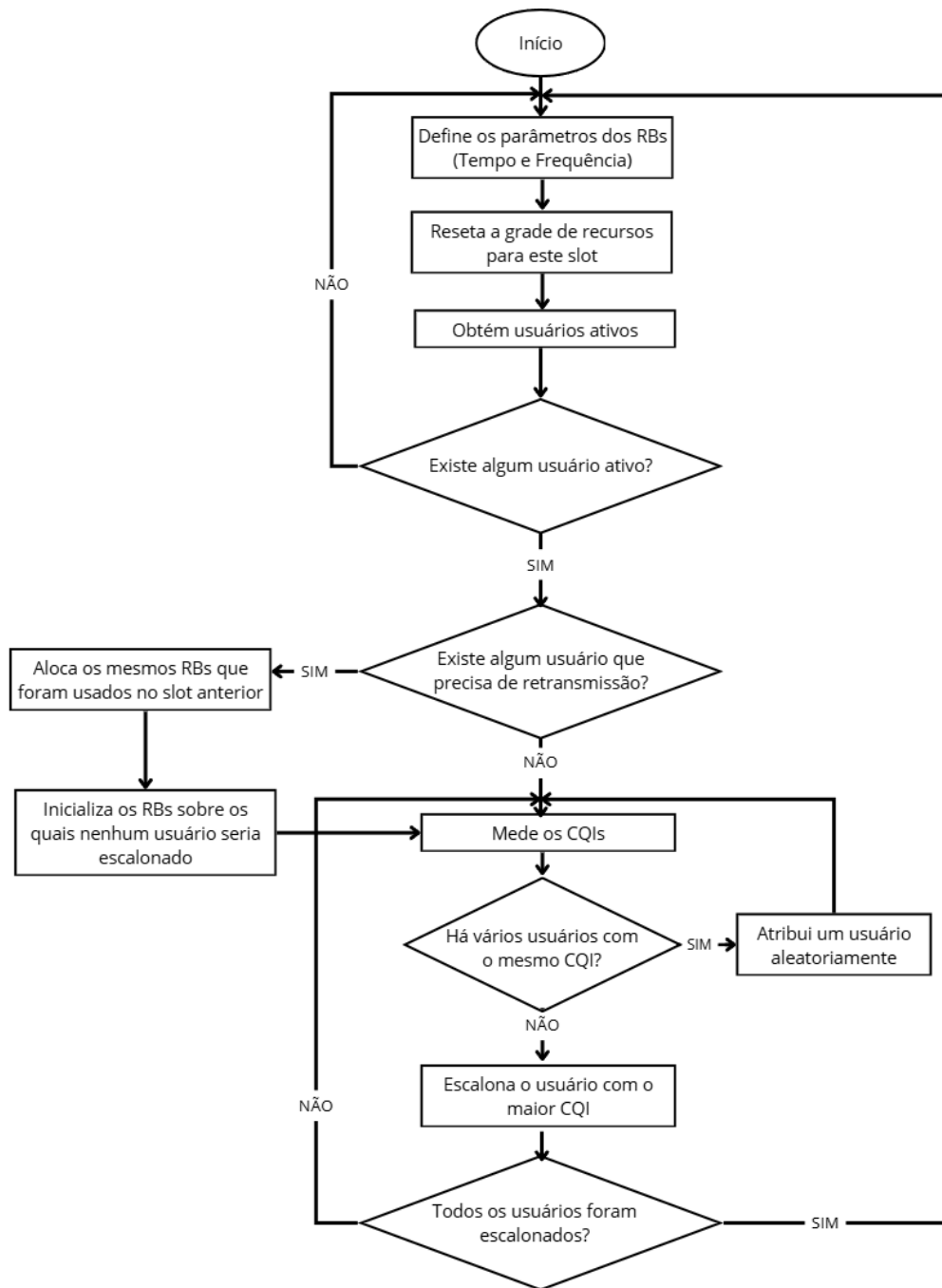
2.5.3 Quality of Service Aware Scheduling

O *QoS Aware Scheduling* é um algoritmo de escalonamento avançado projetado para otimizar a alocação de RBs, em redes 5G. Sua principal função é equilibrar as necessidades de tráfego em tempo real e não tempo real, permitindo que a rede atenda de maneira eficiente a uma ampla gama de aplicações e serviços com diferentes requisitos de desempenho. O QAS realiza um monitoramento contínuo dos requisitos de QoS dos diversos fluxos de tráfego, permitindo ajustes na alocação de recursos e considerando a ocupação dos *buffers* dos usuários, contrastando-se com os algoritmos tradicionais. Um *buffer* é uma espécie de "área de espera" onde os dados ficam armazenados temporariamente antes de serem processados ou transmitidos.

Uma das características especiais do QAS é sua capacidade de priorizar fluxos de tráfego que exigem uma resposta rápida, o que resulta na redução da latência para aplicações em tempo real, como chamadas de voz e jogos online. Para o tráfego NRT, o algoritmo assegura que as taxas de dados sejam adequadas, contribuindo para a qualidade das transferências de dados. O QAS resolve problemas de otimização não convexa para a alocação de RBs, empregando transformações matemáticas específicas que permitem encontrar soluções eficientes, maximizando a utilização do espectro disponível (SHIYAHIN; SCHWARZ; RUPP, 2022).

Além disso, a alocação de recursos é ajustada com base em fatores de prioridade

Figura 7 – Fluxograma da abordagem de escalonamento Best CQI.



Fonte: Adaptado de (QUEIROZ, 2022)

relacionados à latência e confiabilidade, melhorando o desempenho geral da vazão da rede sem comprometer significativamente a BLER ou a latência dos usuários em tempo real (SHIYAHIN; SCHWARZ; RUPP, 2022). Para isso, a alocação é formulada como um problema de otimização inteira binária, no qual cada RB é atribuído a um único usuário, respeitando restrições de latência e confiabilidade. O modelo considera o atraso atual de cada fluxo em relação ao seu limite máximo, além da BLER, garantindo que fluxos mais críticos recebam prioridade. Além de buscar uma alocação equitativa entre os

usuários, especialmente aqueles com *buffers* completos (FASTENBAUER et al., 2023), o algoritmo utiliza o índice de Jain para avaliar a equidade na distribuição dos RBs. Essa métrica, que varia de 0 a 1, assegura que poucos usuários não monopolizem os recursos, evitando desigualdades na vazão. A solução desse problema é implementada por meio de programação matemática, utilizando ferramentas como o *Gurobi Optimizer* e a biblioteca de programação convexa do MATLAB.

O fluxograma da Figura 8 demonstra o processo de tomada de decisão desse escalonador.

2.6 Modelos de Tráfego

Com a crescente complexidade das redes 5G, é essencial simular os diferentes tipos de tráfego para entender o desempenho da rede de maneira mais realista. Cada modelo de tráfego representa um tipo específico de usuário ou aplicação, sendo implementado de forma a reproduzir as condições encontradas no mundo real. De modo geral, esses modelos podem ser divididos em duas categorias principais: RT e NRT.

O tráfego em tempo real se destaca pela sua demanda por baixa latência e alta confiabilidade. Aplicações como chamadas de voz ou jogos online, por exemplo, requerem que os dados sejam transmitidos quase instantaneamente para garantir uma boa experiência ao usuário. Nesses casos, os pacotes de dados são enviados em intervalos regulares e o tamanho desses pacotes pode variar conforme a aplicação. Qualquer atraso significativo na entrega dos pacotes pode prejudicar a qualidade do serviço, tornando a comunicação menos eficiente.

Já o tráfego em não tempo real é mais flexível em relação a atrasos. Ele inclui atividades como transferência de arquivos ou o envio de e-mails, onde a entrega imediata dos dados não é tão crítica. Nesse tipo de tráfego, os usuários geralmente têm uma quantidade ilimitada de dados para transmitir, e os pacotes podem ser gerados de maneira contínua ou em intervalos específicos, dependendo das necessidades da aplicação.

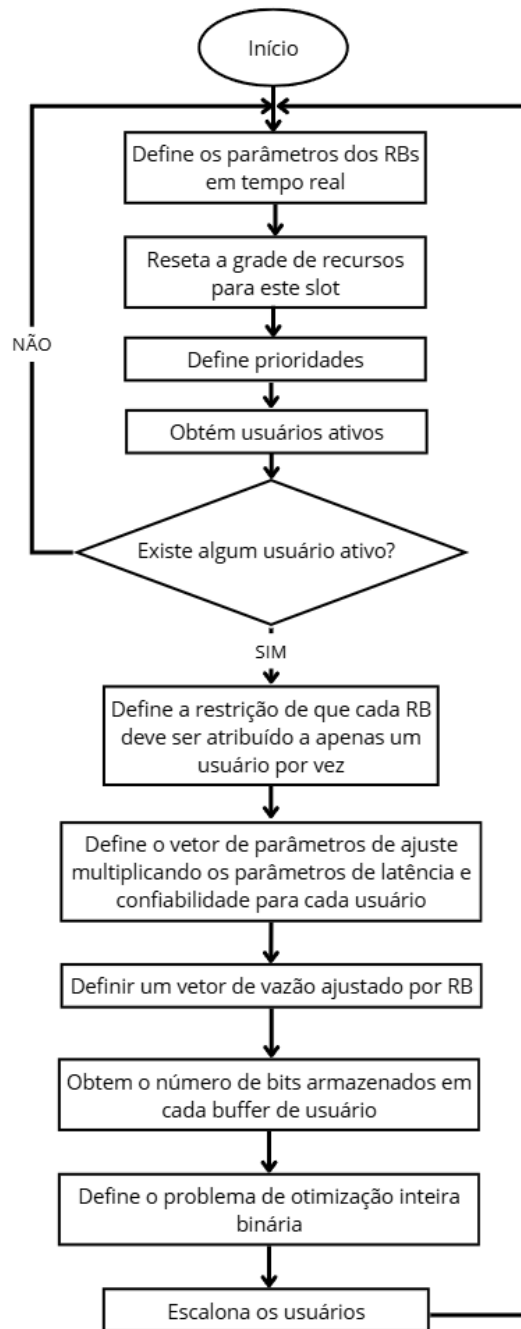
Outro aspecto importante é a forma como o sistema lida com os *buffers*. Para tráfego NRT, os *buffers* ajudam a adaptar a carga de dados de maneira eficiente, otimizando a transmissão. Já para o tráfego RT, o gerenciamento dos *buffers* precisa ser feito com cautela, evitando que os atrasos se acumulem e comprometam a qualidade da comunicação.

Os modelos de tráfego utilizados neste trabalho são descritos a seguir:

Constant Rate: Neste modelo, pacotes de tamanho fixo são gerados com uma taxa de chegada determinada, conforme apresentado na Figura 9. O tempo inicial para a geração dos pacotes é aleatório, o que proporciona simulações mais realistas.

Full buffer: Este modelo assume que os usuários possuem uma quantidade infinita

Figura 8 – Fluxograma da abordagem de escalonamento QAS.



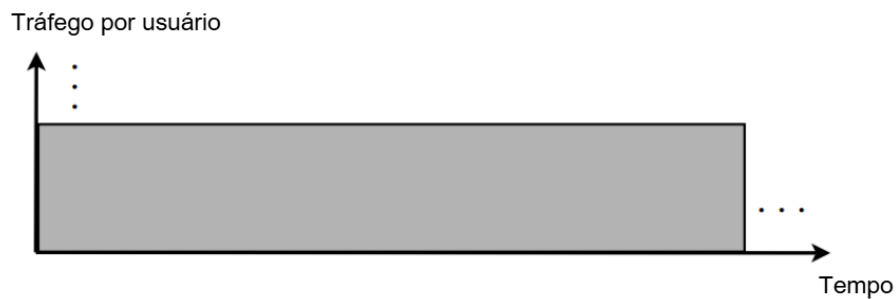
Fonte: Adaptado de (QUEIROZ, 2022)

de dados a serem transmitidos. Cada usuário tem um único pacote de tamanho infinito que pode ser observado na Figura 10. Como resultado, para esse modelo de tráfego, não é calculada a latência, já que os pacotes não são completamente transmitidos ao final da simulação.

Voice Over Internet Protocol (VoIP): O modelo de atividade de voz é representado por um simples modelo de Markov em dois estados. O *codec Adaptive Multi-rate* (AMR) é utilizado, com taxas de fonte que variam de 4,75 kbps a 12,2 kbps.

Figura 9 – Modelo de tráfego *Constant Rate*

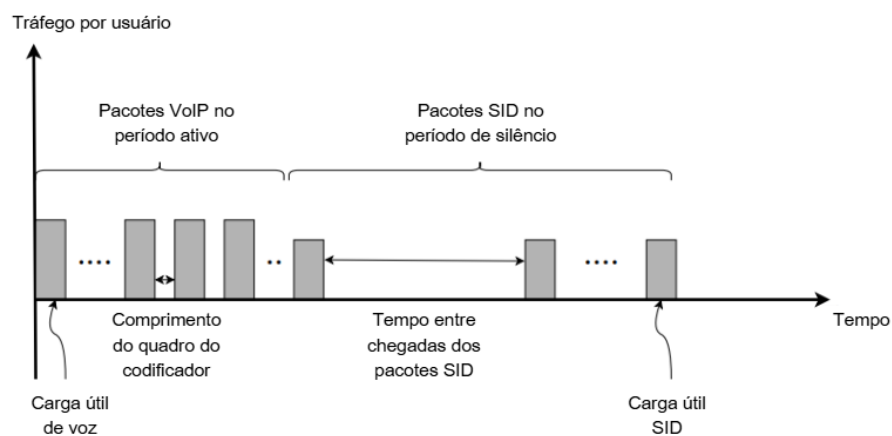
Fonte: Adaptado de (FASTENBAUER et al., 2023)

Figura 10 – Modelo de tráfego *Full buffer*

Fonte: Adaptado de (FASTENBAUER et al., 2023)

Pacotes de voz são gerados a cada 20 ms durante períodos de atividade e a cada 160 ms durante períodos de silêncio. A latência da interface de rádio não deve exceder 40 ms para garantir uma comunicação eficaz. Esta configuração pode ser observada na Figura 11.

Figura 11 – Modelo de tráfego VoIP

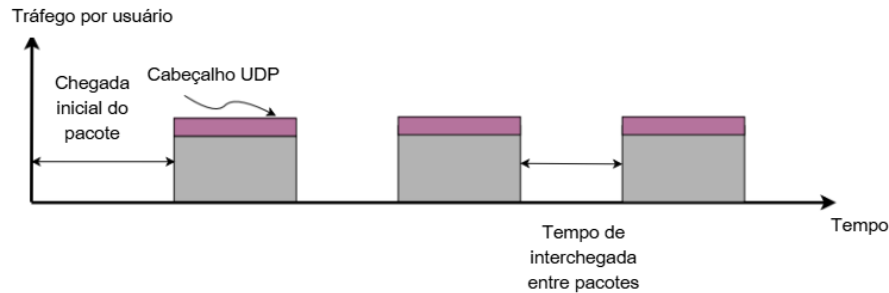


Fonte: Adaptado de (FASTENBAUER et al., 2023)

Gaming: O tráfego gerado por usuários em jogos interativos é caracterizado por um tempo de chegada de pacotes uniformemente distribuído, simulando a aleatoriedade do tráfego, como pode ser observado na Figura 12. Os parâmetros incluem o tempo de interchegada dos pacotes, o tamanho do pacote e um tamanho determinístico adicional

para o cabeçalho do protocolo UDP (*User Datagram Protocol*). A latência dos pacotes deve ser curta, com uma média de atraso não excedendo 60 ms para garantir uma experiência de jogo fluida.

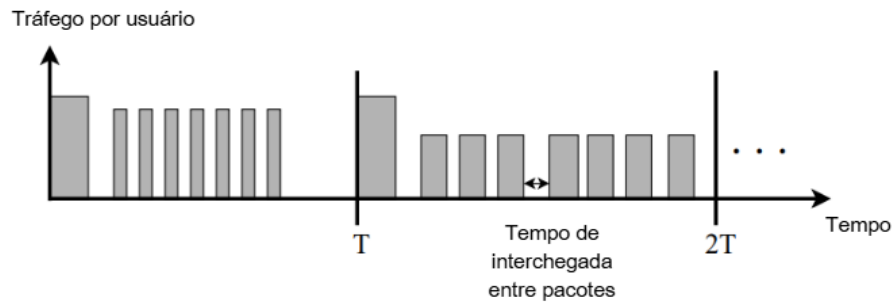
Figura 12 – Modelo de tráfego *Gaming*



Fonte: Adaptado de (FASTENBAUER et al., 2023)

Video *streaming*: Os dados de vídeo chegam em intervalos regulares, e cada quadro de vídeo consiste em múltiplos pacotes de tamanhos aleatórios, ver Figura 13. O modelo considera tempos de interchegada entre os pacotes, que são afetados por intervalos de atraso introduzidos pelo codificador. A taxa de origem de vídeo geralmente é em torno de 64 kbps e a latência não deve exceder 100 ms para garantir uma boa experiência para o usuário.

Figura 13 – Modelo de tráfego Video *streaming*



Fonte: Adaptado de (FASTENBAUER et al., 2023)

2.7 Vienna 5G System Level Simulator

O *Vienna 5G System Level Simulator* é um software desenvolvido para a simulação de redes de comunicação móvel, incluindo as tecnologias 4G, 5G e em desenvolvimento para 6G, funcionando com a plataforma Matlab. Este simulador permite a criação de layouts variados que abrangem múltiplas camadas de estações base, além de diferentes tipos de usuários, proporcionando um ambiente flexível para a análise de desempenho de redes. Ele realiza simulações Monte-Carlo para obter um desempenho médio da rede, calculando métricas como:

- **Vazão Média:** Medido por usuário e estação base;
- **SINR (*Signal to Interference and Noise Ratio*):** Avaliado a partir de diversas condições de canal;
- **BLER:** Determina a taxa de sucesso das transmissões;
- **Latência:** Calculada com base no tempo de transmissão e no processamento de pacotes, fornecendo informações sobre a qualidade do serviço para aplicações sensíveis ao tempo.

O simulador também é capaz de modelar ambientes complexos, incluindo a colocação de obstruções tridimensionais, como paredes e edifícios, utilizando dados reais de cidades para melhorar a precisão das simulações. Além disso, o 5G SLS representa as novas características de transmissão do 5G, como o uso de Ondas Milimétricas (*mmWave*) e *Massive MIMO* (*Multiple Input Multiple Output*), empregando modelos de canal apropriados para diferentes faixas de frequência.

Outro aspecto importante do 5G SLS é sua capacidade de avaliar a perda de pacotes, considerando fatores como perda de sinal e interferência. Isso é crucial para a análise da qualidade da transmissão em condições reais, permitindo a identificação de áreas onde a cobertura pode ser melhorada.

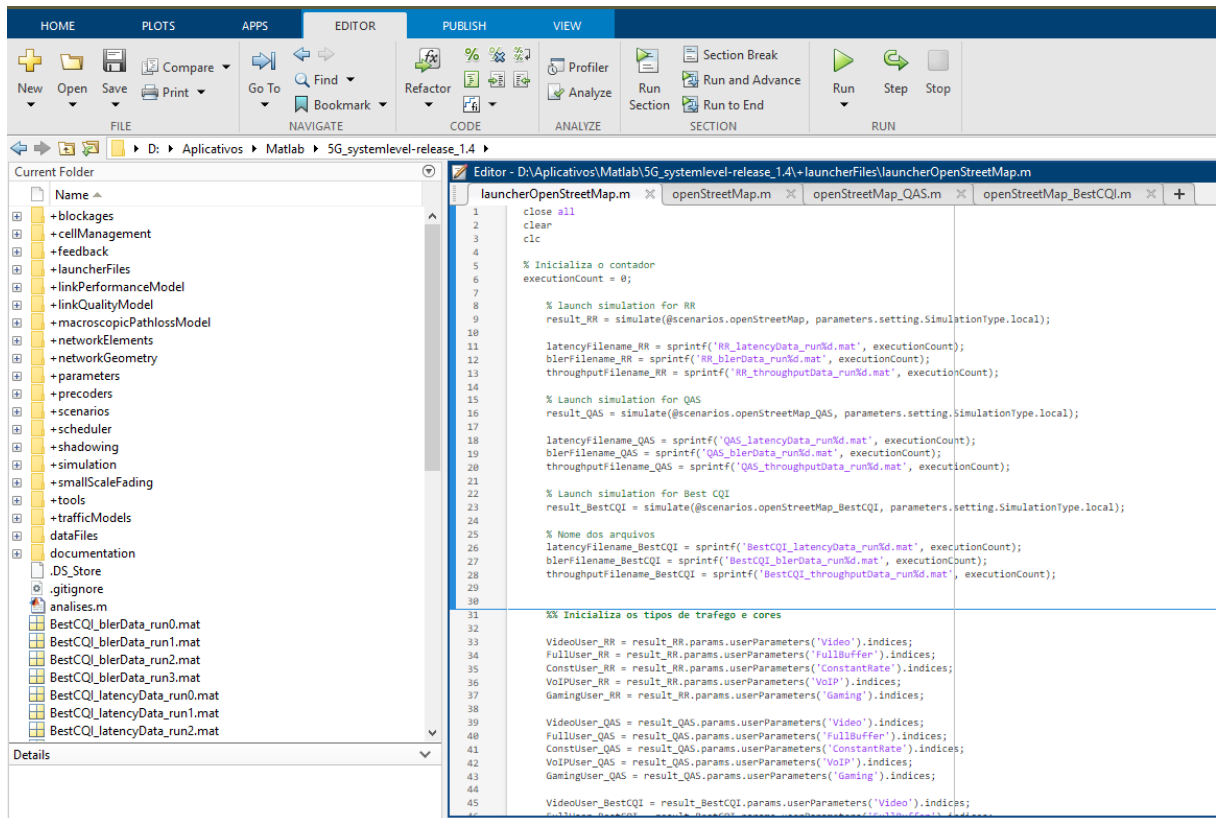
Com sua estrutura modular e suporte para programação orientada a objetos em Matlab, o 5G SLS permite que os usuários personalizem e expandam suas funcionalidades conforme necessário, facilitando a incorporação de novos algoritmos e técnicas de simulação. Informações adicionais sobre o simulador estão disponíveis no site oficial [Vienna 5G Simulators](#).

A Figura 14 mostra uma captura de tela do MATLAB com a interface do simulador aberta, ilustrando sua estrutura e principais elementos.

2.8 Trabalhos relacionados

A importância dos escalonadores no desempenho das redes móveis é destacada por diversos trabalhos publicados sobre o tema. No trabalho de (CARDOSO, 2015), foram avaliados sete algoritmos de escalonamento com foco na eficiência espectral e na qualidade de serviço. Esses algoritmos foram divididos em dois grupos: independentes do canal e dependentes do canal, que por sua vez, podem ser classificados em: baseados em QoS e não baseados em QoS. A avaliação foi realizada por meio de simulações utilizando o software NS-3, comparando os algoritmos em termos de vazão, atraso, *jitter*, justiça e complexidade computacional na rede LTE.

Figura 14 – Interface do simulador aberta no MATLAB



Fonte: Autoria própria (2025).

Os resultados destacaram que os algoritmos independentes do canal, embora mais simples de implementar, atingem menor vazão de dados por não considerarem as variações nas condições de canal. Por outro lado, os algoritmos dependentes do canal demonstraram melhor desempenho, alcançando melhor eficiência espectral e maiores taxas de transmissão, mas apresentam maior complexidade computacional, proporcional ao número de usuários e às métricas necessárias.

O estudo de (HABAEBI et al., 2013) discute a influência de diferentes algoritmos de escalonamento de tráfego na tecnologia LTE, com foco nas técnicas de *Round Robin*, *Best CQI* e *Proportional Fair*. Foi feita uma análise de desempenho dos algoritmos em termos de vazão e taxa de erro de bloco através de simulações de nível de sistema baseadas em Matlab. Os resultados indicam que o algoritmo *Best CQI* supera os outros em termos de vazão, porém às custas da justiça para usuários com más condições de canal. Enquanto isso, o algoritmo *Round Robin* oferece justiça, mas reduz significativamente a vazão do sistema. Por fim, é destacada a necessidade de encontrar um equilíbrio entre vazão e justiça ao projetar algoritmos de escalonamento.

O trabalho de (IOSIF; BăNICĂ, 2012) analisa o desempenho da rede LTE em *downlink* e *uplink*, focando em estratégias de escalonamento de pacotes como *RR* e *Weighted Round Robin* (WRR). Ele destaca que limitações como o número de PDCCHs e a

largura de banda influenciam o desempenho do sistema. Através das simulações, concluiu-se que alocar mais de 1 RB por usuário é mais eficaz em cenários com restrição de energia e serviços com baixa latência.

No trabalho de (SHIYAHIN; SCHWARZ; RUPP, 2022) é destacada a importância de um escalonamento consciente de QoS em ambientes de tráfego misto, propondo o QAS como uma solução que equilibra as necessidades de latência, taxa de transferência e confiabilidade dos usuários. As simulações apresentadas no artigo mostram que o QAS supera significativamente os escalonadores de referência, como RR e Best CQI, em termos de vazão média, confiabilidade média e latência, especialmente em cenários de tráfego variados, como comunicações veiculares, VoIP, *streaming* de vídeo e HTTP.

Além disso, o artigo ressalta a necessidade de esforços futuros para lidar com a complexidade computacional e expandir o QAS para incluir técnicas de transmissão avançadas, como MIMO, relevantes para redes 5G.

2.9 Considerações Finais

Neste capítulo, foram discutidos os fundamentos teóricos que sustentam a pesquisa, incluindo as características da rede 5G e dos escalonadores. Além disso, foram detalhados os algoritmos de escalonamento a serem utilizados, juntamente com os modelos de tráfego empregados nas simulações. De maneira semelhante foi apresentado o simulador utilizado e foram citados trabalhos relacionados que contextualizam a relevância deste estudo.

No próximo capítulo, Metodologia de Pesquisa, será detalhado o ambiente de simulação e os parâmetros de avaliação adotados para mensurar o desempenho dos algoritmos nos cenários modelados.

3 Metodologia de Pesquisa

Neste capítulo, são descritos os métodos e ferramentas aplicados no desenvolvimento deste trabalho, abrangendo os ambientes de simulação, os parâmetros configurados juntamente com as métricas avaliadas.

3.1 Ambiente de simulação

Neste trabalho, foram desenvolvidos três cenários de simulação para avaliar o desempenho dos escalonadores abordados no Capítulo 2: RR, Best CQI e QAS. As simulações foram realizadas no software *Vienna 5G System Level Simulator* baseado em Matlab e projetadas para quantidades de usuários variando entre 150, 350 e 550, permitindo uma análise abrangente das capacidades dos algoritmos sob diferentes condições de carga. Os códigos utilizados para a implementação desses cenários podem ser consultados nos Apêndices A, B, C e D.

As porcentagens de usuários foram definidas com base no trabalho (SHIYAHIN; SCHWARZ; RUPP, 2022), buscando criar uma simulação que refletisse as condições típicas de uso das redes. Os usuários foram distribuídos entre dois tipos principais de tráfego: RT e NRT. O tráfego RT, que inclui *Video streaming*, VoIP e *Gaming*, é típico de aplicações que exigem baixa latência e alta qualidade de serviço. Os cenários foram estruturados de forma que 25% dos usuários são designados para Vídeo, 25% para VoIP e 25% para *Gaming*. Por outro lado, o tráfego NRT, que abrange aplicações que podem tolerar atrasos maiores, como *Full Buffer* e *Constant Rate*, representa 15% do total para *Constant Rate* e 10% para *Full Buffer*.

Como a distribuição percentual nem sempre resultava em valores inteiros, foi aplicado o arredondamento para baixo, assegurando a coerência na alocação dos usuários em cada cenário. A Tabela 2 apresenta a quantidade final de usuários alocados para cada tipo de tráfego após a aplicação do arredondamento.

Para garantir uma variabilidade controlada e permitir uma comparação justa, cada cenário foi simulado três vezes para cada escalonador, utilizando uma semente aleatória (*random seed*) diferente em cada execução. A utilização de *random seeds* permite inicializar o gerador de números aleatórios de maneira que as condições de transmissão e os parâmetros sejam consistentes entre as execuções. Isso assegura que os resultados não sejam influenciados por flutuações aleatórias, permitindo uma avaliação mais precisa e repetível do desempenho de cada escalonador em diferentes cenários.

Os cenários foram posicionados no campus Santa Mônica da Universidade Federal

Tabela 2 – Distribuição de usuários por tipo de tráfego

Cenário 1: 150 usuários			
Tipo de Tráfego	Porcentagem	Valor Exato	Valor Arredondado
Vídeo	25	37,5	37
VoIP	25	37,5	37
<i>Gaming</i>	25	37,5	37
<i>Constant Rate</i>	15	22,5	22
<i>Full Buffer</i>	10	15	15
Total	100	150	148
Cenário 2: 350 usuários			
Tipo de Tráfego	Porcentagem	Valor Exato	Valor Arredondado
Vídeo	25	87,5	87
VoIP	25	87,5	87
<i>Gaming</i>	25	87,5	87
<i>Constant Rate</i>	15	52,5	52
<i>Full Buffer</i>	10	35	35
Total	100	350	348
Cenário 3: 550 usuários			
Tipo de Tráfego	Porcentagem	Valor Exato	Valor Arredondado
Vídeo	25	137,5	137
VoIP	25	137,5	137
<i>Gaming</i>	25	137,5	137
<i>Constant Rate</i>	15	82,5	82
<i>Full Buffer</i>	10	55	55
Total	100	550	548

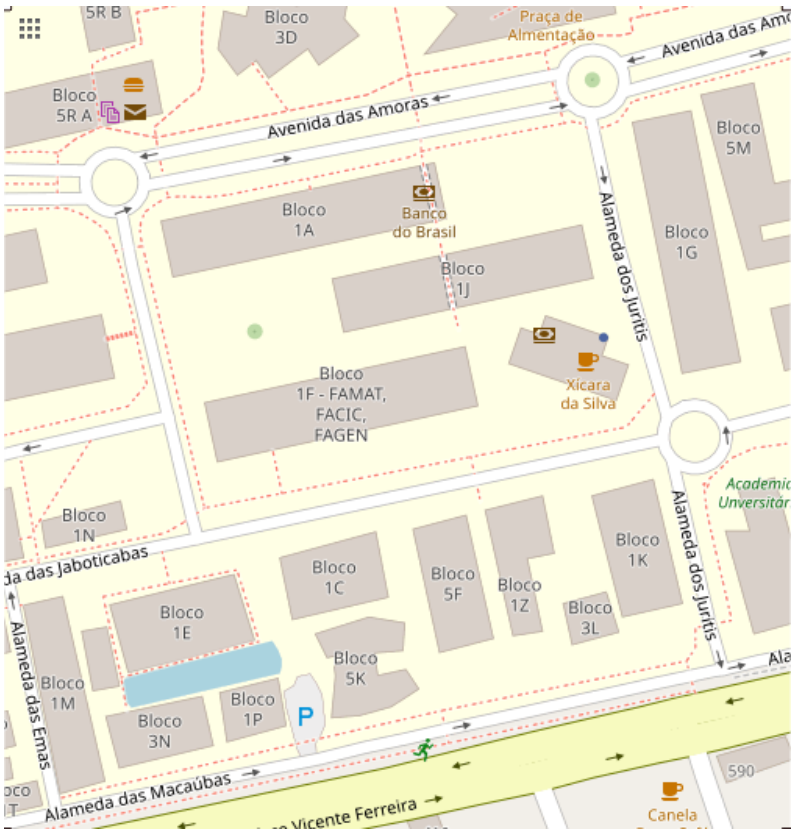
de Uberlândia (UFU), com o auxílio do site [OpenStreetMap](#), que identifica e exibe prédios e construções no mapa. Isso ajuda a calcular as interferências e atenuações, proporcionando uma simulação mais fiel às condições urbanas reais. As coordenadas exatas do campus, assim como uma imagem ilustrativa do local, podem ser consultadas na Tabela 3 e na Figura 15. As coordenadas com valores negativos correspondem às direções sul (latitude) e oeste (longitude).

Tabela 3 – Limites geográficos do campus

	Inferior	Superior
Latitude	-18.920252	-18.917542
Longitude	-48.259442	-48.256738

No entanto, o *OpenStreetMap* não oferece informações sobre a altura dos edifícios. Para resolver essa questão, o simulador gera alturas de forma aleatória, utilizando uma

Figura 15 – Vista superior do campus Santa Mônica - UFU



Fonte: Autoria própria (2024).

distribuição uniforme entre os limites máximos e mínimos definidos. Esses valores de altura foram mantidos conforme o padrão original, apresentados na Tabela 4.

Tabela 4 – Parâmetros das construções

Parâmetros	Valores
Altura máxima	25 m
Altura mínima	10 m

Por se tratar de uma área real, é fundamental definir a área de interesse e interferência no simulador para delimitar o espaço onde as construções e outros elementos urbanos afetam a propagação do sinal. Essa delimitação permite que o simulador estime com precisão os efeitos de interferências e atenuações nas redes 5G. Neste trabalho, mantiveram-se as configurações padrão do simulador, que podem ser consultadas na Tabela 5.

As coordenadas usadas no simulador são cartesianas (x, y, z), sendo que as coordenadas x e y correspondem, respectivamente, à longitude e latitude, enquanto a coordenada z indica a elevação em relação ao solo.

Além disso, foram posicionadas duas estações base em posições predefinidas, essas

Tabela 5 – Parâmetros da área de interferência

Valores	Parâmetros
Eixo x	400 m
Eixo y	300 m
Eixo z	50 m

posições foram mantidas conforme as configurações originais do programa, como pode ser observado na Tabela 6. Cada estação foi equipada com uma antena 5G setorizada, composta por três setores e um elemento de transmissão. As coordenadas também correspondem ao sistema cartesiano, sendo que os valores positivos correspondem ao sentido norte no eixo y e leste no eixo x.

Tabela 6 – Parâmetros das BSs

	Eixo x	Eixo y
BS 1	-191.968	70.4545
BS 2	147.677	-30.2005

A distribuição dos usuários foi realizada de forma aleatória, utilizando o Processo de Ponto de Poisson (PPP), que é amplamente empregado em simulações de redes para representar de maneira estatisticamente realista a distribuição espacial dos usuários. Esse método assegura uma dispersão aleatória e independente dos pontos (usuários) no espaço, o que reflete cenários práticos de redes celulares. Na simulação, os usuários foram considerados estáticos, ou seja, suas posições permanecem fixas ao longo da análise, facilitando a avaliação do desempenho dos algoritmos de escalonamento sob condições controladas.

Para a modelagem da propagação do sinal, foi utilizado o modelo de perda de percurso *UrbanMacro5G*, que é adequado para cenários urbanos. Este modelo considera tanto as condições de Linha de Visada (LOS) quanto as de Sem Linha de Visada (NLOS), além das condições internas (*indoor*) e externas (*outdoor*). Essa abordagem permite uma análise mais abrangente das interações entre a rede e os usuários.

A definição dos parâmetros principais para a simulação foi realizada com base em uma análise detalhada dos trabalhos relacionados citados na seção 2.8, especialmente o trabalho de (SHIYAHIN; SCHWARZ; RUPP, 2022), que serviu como principal referência. Foram mantidos alguns dos parâmetros originais do simulador, como a frequência e largura de banda, além daqueles já mencionados. Um resumo das principais características e parâmetros da simulação pode ser encontrado na Tabela 7.

Tabela 7 – Parâmetros de Simulação

Parâmetros	Valores
Tempo de simulação	2000 time slots
Duração de time slot	1 ms
Frequência central	2 GHz
Largura de banda	10 MHz
Número de ERBs	2 BSs
Antenas	Antena setorizada de três setores com um elemento transmissor
Tipo de escalonador	RR, Best CQI e QAS
Número de usuários	150, 350, 550 usuários
Tipo de tráfego	NRT: 10% <i>Full Buffer</i> , 15% <i>Constant Rate</i> RT: 25% Vídeo, 25% VoIP, 25% <i>Gaming</i>
Atraso aceitável	Vídeo: 100 ms VoIP: 40 ms <i>Gaming</i> : 60 ms
Modo de canal	Pedestre
Path loss	<i>UrbanMacro5G</i>

3.2 Critérios de Avaliação

A fim de analisar a eficácia dos algoritmos utilizados, uma série de critérios de avaliação são detalhados a seguir. Esses critérios possibilitam uma avaliação do desempenho, qualidade de serviço, capacidade e confiabilidade de cada escalonador nos cenários propostos.

3.2.1 Vazão

A vazão ou taxa de transferência é um dos principais indicadores de desempenho em telecomunicações, representando a quantidade de dados que um usuário pode transmitir ou receber por segundo. Geralmente medida em bits por segundo (bps), a vazão determina a eficiência de comunicação em uma rede e está relacionada a vários fatores, incluindo a largura de banda disponível, a eficiência dos protocolos de comunicação, a qualidade do sinal, o congestionamento da rede e a tecnologia de acesso utilizada. A vazão é calculada conforme a Equação 2:

$$Vazao = \frac{\sum R_x \cdot TamanhoPacote}{TempoEntrega}, \quad (2)$$

onde R_x é a taxa de transmissão de cada pacote.

3.2.2 Latência

A latência refere-se ao tempo que um pacote de dados leva para viajar de um ponto a outro na rede, sendo medida em segundos e muitas vezes expressa em Milissegundos (ms). A latência é dada pela Equação 3:

$$Latencia = T_{rx} - T_{tx}, \quad (3)$$

onde T_{rx} é o instante em que os dados são recebidos e T_{tx} é o instante de transmissão. A latência pode ser afetada por vários fatores, incluindo o tempo de processamento nos nós da rede, a distância física entre o remetente e o destinatário, e o congestionamento da rede. Redes com alta latência podem resultar em experiências frustrantes para os usuários.

3.2.3 BLER

A BLER é uma métrica fundamental que mede a confiabilidade do canal de comunicação, indicando a proporção de blocos de dados recebidos incorretamente em relação ao total de blocos transmitidos. Essa métrica é especialmente importante em redes onde a precisão na entrega de dados é crítica. A BLER pode ser calculada pela Equação 4:

$$BLER = \frac{B_e}{B_t}, \quad (4)$$

onde B_e é o número de blocos com erro, e B_t é o número total de blocos transmitidos. Um BLER alto sugere que a rede está enfrentando problemas significativos na transmissão, como interferência, ruído excessivo ou congestionamento, o que pode levar a retransmissões e diminuição da eficiência geral do sistema.

3.3 Considerações Finais

Neste capítulo, foi detalhado o ambiente de simulação e os parâmetros configurados, assim como as métricas de desempenho escolhidas, incluindo vazão, latência e BLER. Essa abordagem metodológica permitirá uma compreensão aprofundada do desempenho de cada algoritmo em diferentes cenários de tráfego, identificando suas principais vantagens e desvantagens, além de garantir a reprodutibilidade e comparabilidade dos resultados obtidos. No próximo capítulo, Resultados e Discussões, é analisada a eficácia dos algoritmos de escalonamento com base nos resultados das simulações.

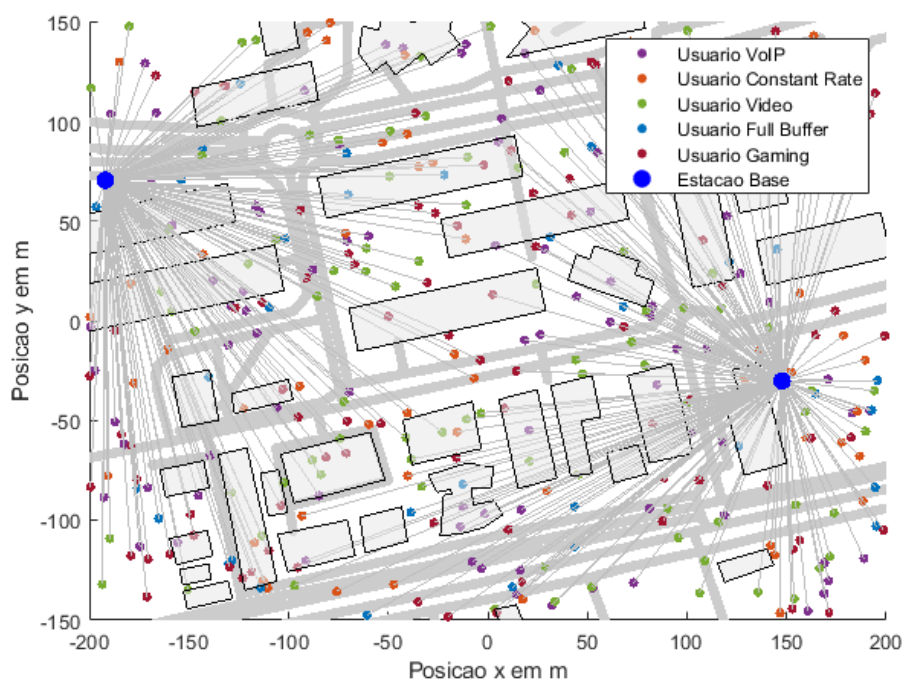
4 Resultados e Discussões

Neste capítulo, são apresentados e analisados os resultados obtidos por meio das simulações realizadas no *Vienna 5G System Level Simulator* com os diferentes algoritmos de escalonamento em cenários 5G. O objetivo principal é avaliar o desempenho dos algoritmos com base em parâmetros cruciais para as redes de comunicação, os quais foram especificados na Seção 3.2.

Para a apresentação dos resultados, esta seção conta com tabelas e figuras que ilustram os dados obtidos a partir das simulações. Os resultados foram calculados com base na média de três execuções realizadas para cada escalonador em cada cenário e estão organizados de forma comparativa entre os escalonadores dentro de cada cenário. A análise enfatiza o comportamento de cada métrica de desempenho, considerando os diferentes tipos de tráfego e as distribuições de usuários.

Como o estudo envolve três escalonadores, gráficos semelhantes são apresentados apenas uma vez, como pode ser observado na Figura 16, que mostra o mapa do cenário, onde apenas as posições e quantidades de usuários variam. Essa escolha visa otimizar a apresentação, já que o objetivo desta seção é apenas ilustrar os resultados das simulações individuais dos cenários estudados.

Figura 16 – Vista da UFU - Campus Santa Mônica, com indicação da localização dos usuários e BSs



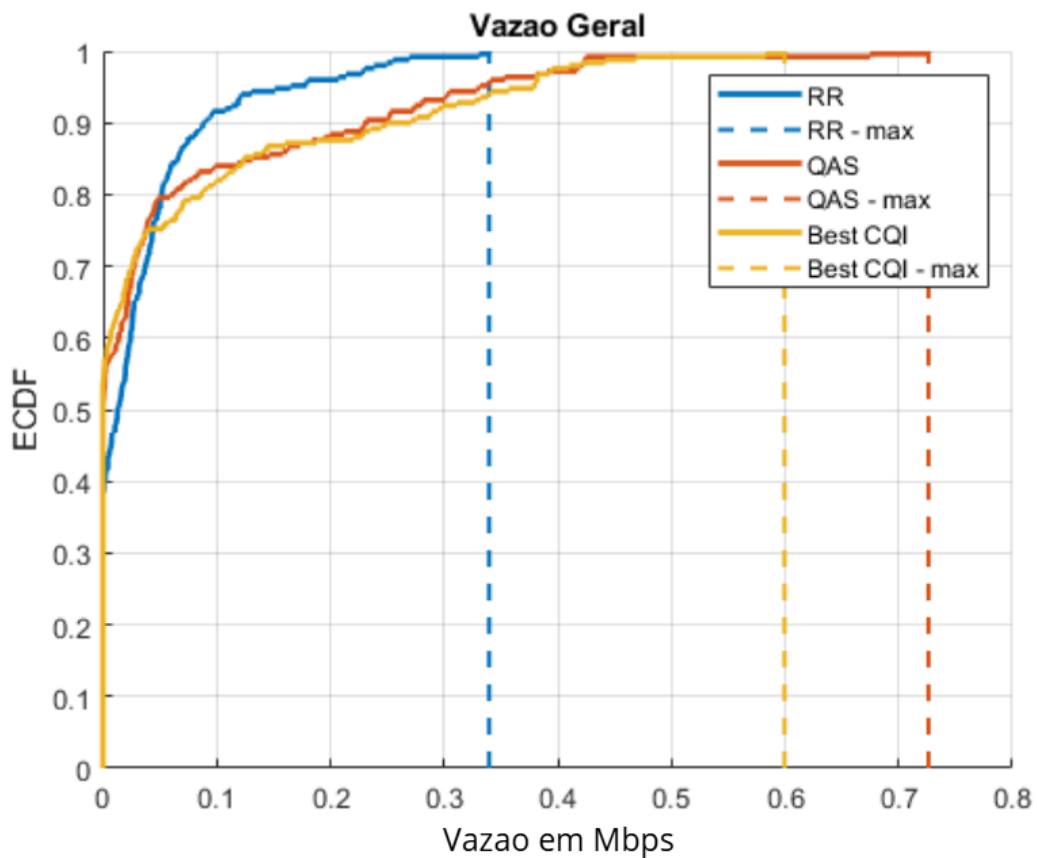
4.1 Resultados das simulações

A Figura 16 mostra a vista superior do *campus* Santa Mônica da UFU, onde foram simulados todos os cenários, gerada pelo simulador com base no site *OpenStreetMap*, que exhibe as construções e ruas da área. No mapa, observa-se uma distribuição de pontos menores e coloridos, que representam os 350 usuários do cenário 2, divididos em diferentes tipos de tráfego. Já os pontos maiores, em azul, indicam as estações base, responsáveis pela transmissão dos sinais 5G aos usuários na região de interesse da simulação.

4.1.1 Vazão

A Figura 17 apresenta a vazão geral da simulação com 550 usuários, sem diferenciar os tipos de tráfego. A curva exibida corresponde à Função de Distribuição Acumulada Empírica (ECDF), gerada com base nos valores coletados durante a simulação. A figura mostra a relação entre a vazão em Megabits por segundo (Mbps) e a distribuição dos usuários para os três diferentes algoritmos de escalonamento.

Figura 17 – Vazão em Mbps em relação à distribuição dos usuários no cenário 3



Observa-se que o algoritmo QAS apresenta o melhor desempenho, destacando-se pela maior vazão máxima, indicada pela linha tracejada "QAS - max". Além disso, sua curva geral é ligeiramente superior à do Best CQI, evidenciando sua vantagem tanto na vazão média quanto na máxima. O Best CQI, embora próximo do QAS em desempenho,

apresenta uma vazão máxima um pouco inferior, ficando em segundo lugar. Já o algoritmo RR possui o pior desempenho, com uma vazão máxima significativamente menor e uma curva geral mais baixa, refletindo uma vazão média também inferior.

A Tabela 8 apresenta os resultados de vazão média para diferentes tipos de tráfego e escalonadores nos três cenários estudados, que consideram 150, 250 e 550 usuários, respectivamente. A Tabela 9 resume a vazão média geral por escalonador em cada um desses cenários. Os resultados indicam variações de desempenho significativas, refletindo as características e a eficiência de cada algoritmo de escalonamento.

Tabela 8 – Vazão Média em Mbps para diferentes tipos de usuários e escalonadores

Cenário 1: 150 usuários					
Escalonador	VoIP	<i>Constant Rate</i>	Video	<i>Gaming</i>	<i>Full Buffer</i>
RR	1,35	2,11	1,51	1,22	1,50
QAS	1,58	2,16	1,55	1,37	1,59
Best CQI	1,39	2,28	1,49	1,27	1,76
Cenário 2: 350 usuários					
Escalonador	VoIP	<i>Constant Rate</i>	Video	<i>Gaming</i>	<i>Full Buffer</i>
RR	1,02	1,23	1,01	0,98	1,13
QAS	1,18	1,37	1,15	1,16	1,05
Best CQI	1,11	1,44	1,14	1,06	1,19
Cenário 3: 550 usuários					
Escalonador	VoIP	<i>Constant Rate</i>	Video	<i>Gaming</i>	<i>Full Buffer</i>
RR	0,41	0,45	0,39	0,34	0,49
QAS	0,59	0,58	0,57	0,55	0,61
Best CQI	0,51	0,59	0,52	0,41	0,69

No Cenário 1, com 150 usuários, o escalonador QAS apresentou o melhor desempenho em quase todos os tipos de tráfego, destacando-se nos serviços RT, como VoIP (1,58 Mbps) e *Gaming* (1,37 Mbps). Essa superioridade reflete sua capacidade de priorizar tráfegos sensíveis à latência e requisitos de qualidade de serviço. O Best CQI demonstrou uma performance intermediária, superando os demais no tráfego Constant Rate, com uma vazão de 2,28 Mbps, e também se mostrou competitivo para *Full Buffer*, com 1,76 Mbps. Por outro lado, o RR apresentou os menores valores médios, com desempenho mais equilibrado entre os diferentes tipos de tráfego, mas sem se destacar em nenhuma categoria.

No Cenário 2, com 350 usuários, o aumento na densidade da rede evidenciou diferenças mais significativas entre os algoritmos. O QAS continuou liderando para VoIP (1,18 Mbps), *Gaming* (1,16 Mbps) e Vídeo (1,15 Mbps), demonstrando sua eficiência em gerenciar tráfego RT mesmo sob condições mais congestionadas. O Best CQI apresentou a

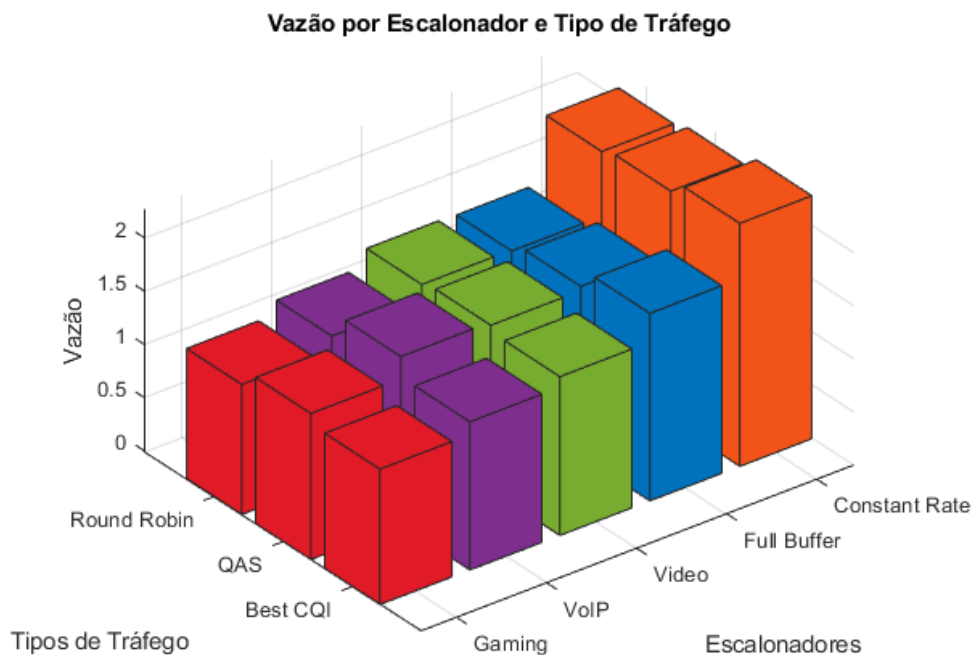
maior vazão média para Constant Rate, com 1,44 Mbps, reafirmando sua aptidão para tráfegos NRT. O RR, por sua vez, manteve o pior desempenho, registrando os menores valores em todas as categorias, como 1,02 Mbps para VoIP e 0,98 Mbps para *Gaming*, destacando suas limitações em cenários mais densos.

No Cenário 3, com 550 usuários, a pressão na rede ampliou ainda mais as diferenças entre os escalonadores. O QAS novamente se destacou, apresentando os melhores resultados em quase todos os tipos de tráfego, com 0,59 Mbps para VoIP, 0,57 Mbps para Video e 0,55 Mbps para *Gaming*, evidenciando sua robustez em cenários de alta densidade. O Best CQI se mostrou competitivo, alcançando a maior vazão para *Full Buffer* (0,69 Mbps) e resultados próximos ao QAS em outras categorias, como 0,59 Mbps para *Constant Rate*. O RR, entretanto, ficou significativamente atrás, com os menores valores médios, como 0,41 Mbps para VoIP e apenas 0,34 Mbps para *Gaming*, indicando sua incapacidade de gerenciar eficientemente os recursos em situações de alta demanda.

Tabela 9 – Vazões médias em Mbps por escalonador em diferentes cenários

Escalonador	Cenário 1	Cenário 2	Cenário 3
RR	1,49	1,05	0,40
QAS	1,61	1,18	0,58
Best CQI	1,55	1,16	0,52

Figura 18 – Vazão média em Mbps dos três escalonadores para cada tipo de tráfego no cenário 1



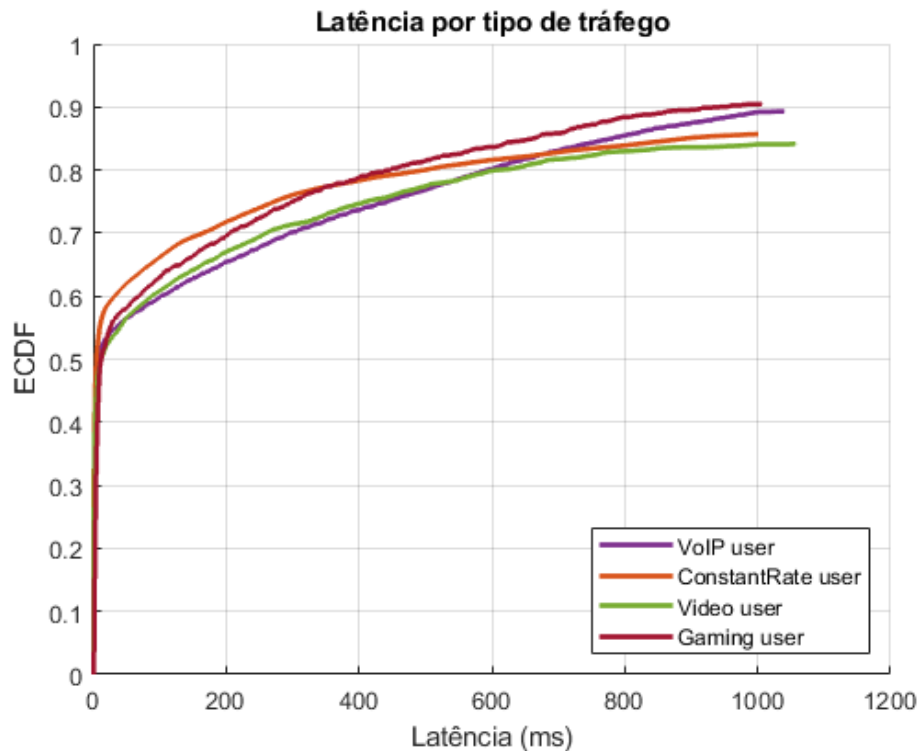
A Figura 18 mostra um gráfico 3D que ilustra a vazão média dos três escalonadores para cada tipo de tráfego no primeiro cenário, onde o total de usuários é 150. De forma geral,

os resultados indicam que o QAS é o algoritmo mais eficaz em termos de maximização da vazão média, especialmente em cenários mais congestionados. Sua abordagem adaptativa permite um melhor gerenciamento da alocação de recursos, o que é essencial para garantir uma experiência de qualidade para todos os tipos de tráfego, especialmente aqueles que exigem uma maior quantidade de dados. Em comparação, o RR mostrou-se o menos eficiente, refletindo sua limitação em termos de flexibilidade e adaptação às necessidades variáveis de tráfego. O Best CQI conseguiu bons resultados para tipos de tráfego que se beneficiam de boas condições de canal, mas mostrou-se inferior ao QAS em cenários com alta densidade de usuários, onde a alocação adaptativa de recursos é crucial para maximizar a vazão total.

4.1.2 Latência

Os resultados de latência para os três cenários de simulação, incluindo a latência média, o percentual de violação das metas e os limites estabelecidos para cada tipo de tráfego, estão apresentados nas Tabelas 10, 11 e 12. Para os tráfegos em tempo real (RT), as metas de latência foram definidas conforme discutido na Seção 2.6, levando em consideração as exigências específicas de qualidade de serviço (QoS) associadas a esses tipos de tráfego.

Figura 19 – Latência por tipo de tráfego para o escalonador QAS no cenário 1



No caso dos tráfegos não em tempo real (NRT), apenas a latência média foi avaliada, uma vez que não há metas rígidas relacionadas a esse tipo de tráfego. Além disso, a latência

do tráfego *Full Buffer* não foi medida devido às características particulares do modelo adotado. Conforme descrito na Seção 2.6, os usuários nesse modelo transmitem uma quantidade infinita de dados, configurados como um único pacote de tamanho infinito. Essa configuração inviabiliza a medição da latência total, já que os pacotes nunca são completamente transmitidos durante a simulação. Por essa razão, não há ECDF associada à latência para o tráfego *Full Buffer*.

A Figura 19 ilustra a latência por tipo de tráfego para o escalonador QAS no primeiro cenário, apresentada em função da ECDF. Essa representação evidencia a capacidade do QAS em priorizar tráfegos com maiores restrições de latência, reafirmando sua adequação para aplicações sensíveis à latência em ambientes com demandas diversas de tráfego.

Tabela 10 – Estatísticas de Latência por Tipo de Tráfego (Cenário 1)

Cenário 1: 150 usuários				
Escalonador	Tipo de Tráfego	Latência Média (ms)	Percentual de Violação (%)	Latência Máxima Permitida (ms)
RR	VoIP	156,36	34,61	40
	<i>Constant Rate</i>	123,39	-	-
	Video	160,83	29,08	100
	<i>Gaming</i>	127,81	31,88	60
QAS	VoIP	99,56	21,21	40
	<i>Constant Rate</i>	83,49	-	-
	Video	112,03	14,48	100
	<i>Gaming</i>	115,07	22,54	60
Best CQI	VoIP	119,52	29,79	40
	<i>Constant Rate</i>	81,78	-	-
	Video	118,72	27,03	100
	<i>Gaming</i>	122,07	26,28	60

No Cenário 1, conforme apresentado na Tabela 10, o escalonador QAS demonstrou um desempenho superior em relação à latência média para tráfegos em tempo real (RT), como VoIP e *Gaming*, alcançando 99,56 ms e 115,07 ms, respectivamente. Esses resultados também refletiram nas menores taxas de violação para esses tipos de tráfego: 21,21% para VoIP e 22,54% para *Gaming*, evidenciando a capacidade do QAS em priorizar tráfegos sensíveis à latência. Em comparação, o Best CQI obteve uma latência média ligeiramente maior, mas ainda competitiva, destacando-se na gestão de tráfegos menos sensíveis a latências extremas. Já o RR, por sua vez, apresentou um desempenho mais equilibrado entre os tipos de tráfego, sem priorização de nenhum deles, o que resultou em latências médias mais altas e taxas de violação superiores, refletindo suas limitações em lidar com os requisitos de tráfegos RT.

Para tráfegos não em tempo real (NRT), como *Constant Rate*, todos os escalonadores apresentaram latências médias próximas, com o Best CQI liderando ligeiramente (81,78

ms). Contudo, devido à natureza menos exigente dos tráfegos NRT em relação a latências rígidas, as diferenças de desempenho não impactaram de maneira crítica a rede.

Tabela 11 – Estatísticas de Latência por Tipo de Tráfego (Cenário 2)

Cenário 2: 350 usuários				
Escalonador	Tipo de Tráfego	Latência Média (ms)	Percentual de Violação (%)	Latência Máxima Permitida (ms)
RR	VoIP	186,27	50,52	40
	<i>Constant Rate</i>	218,77	-	-
	Video	231,42	46,19	100
	<i>Gaming</i>	224,34	46,98	60
QAS	VoIP	146,67	35,12	40
	<i>Constant Rate</i>	211,19	-	-
	Video	216,78	41,93	100
	<i>Gaming</i>	164,03	37,91	60
Best CQI	VoIP	171,38	45,78	40
	<i>Constant Rate</i>	212,71	-	-
	Video	241,19	51,02	100
	<i>Gaming</i>	194,61	41,36	60

No Cenário 2, apresentado na Tabela 11, houve um aumento nas latências médias e nas taxas de violação devido ao maior congestionamento da rede com 350 usuários. O escalonador QAS continuou a apresentar o melhor desempenho para tráfegos em tempo real (RT), com as menores latências médias, como 146,67 ms para VoIP e 164,03 ms para *Gaming*. As taxas de violação também foram mais baixas em comparação aos outros escalonadores, com 35,12% para VoIP e 37,91% para *Gaming*, reforçando a capacidade do QAS de priorizar eficazmente tráfegos sensíveis à latência, mesmo sob maior carga de usuários.

Por outro lado, o RR manteve seu desempenho mais equilibrado entre os tipos de tráfego, mas ainda registrou as maiores latências médias para a maioria dos tráfegos RT, como 186,27 ms para VoIP e 224,34 ms para *Gaming*, além de taxas de violação elevadas, como 50,52% para VoIP. O Best CQI, embora tenha apresentado uma latência média próxima à do RR para VoIP (171,38 ms), teve um desempenho inferior para vídeo, com a maior latência média (241,19 ms) e a maior taxa de violação (51,02%). No tráfego *Constant Rate*, os três escalonadores apresentaram resultados semelhantes em termos de latência média, dada a natureza mais flexível dos tráfegos NRT.

No Cenário 3, apresentado na Tabela 12, com 550 usuários, as latências médias e as taxas de violação aumentaram ainda mais, refletindo o congestionamento da rede. O QAS continuou a se destacar para tráfegos RT, alcançando 250,33 ms para VoIP e 231,52 ms para *Gaming*. Embora essas latências médias tenham aumentado em relação aos cenários anteriores, elas ainda foram as menores entre os escalonadores, com taxas de violação de

Tabela 12 – Estatísticas de Latência por Tipo de Tráfego (Cenário 3)

Cenário 3: 550 usuários				
Escalonador	Tipo de Tráfego	Latência Média (ms)	Percentual de Violação (%)	Latência Máxima Permitida (ms)
RR	VoIP	310,10	69,43	40
	<i>Constant Rate</i>	252,66	-	-
	Video	324,39	75,40	100
	<i>Gaming</i>	313,74	63,70	60
QAS	VoIP	250,33	52,25	40
	<i>Constant Rate</i>	259,15	-	-
	Video	230,49	54,93	100
	<i>Gaming</i>	231,52	55,37	60
Best CQI	VoIP	273,84	64,25	40
	<i>Constant Rate</i>	271,25	-	-
	Video	262,31	58,43	100
	<i>Gaming</i>	310,88	73,87	60

52,25% para VoIP e 55,37% para *Gaming*. Isso demonstra a eficácia do QAS em priorizar tráfegos sensíveis à latência, mesmo sob alta carga de usuários.

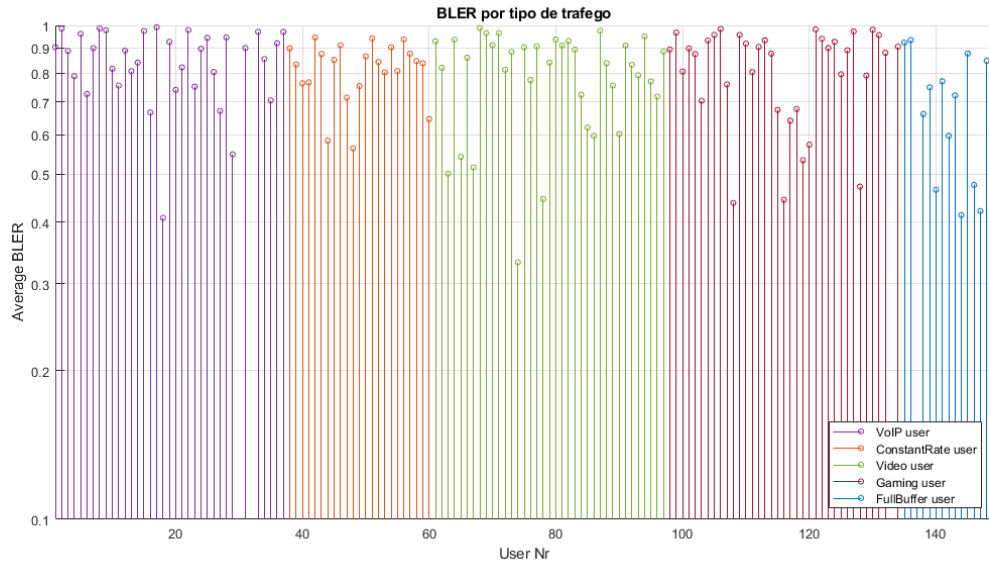
O Best CQI registrou latências médias mais altas, como 273,84 ms para VoIP, com uma taxa de violação de 64,25%, e 310,88 ms para *Gaming*, com 73,87% de violação. Embora o Best CQI tenha superado o RR para VoIP (que registrou 310,10 ms), ele ainda ficou atrás do QAS em termos de latência e apresentou taxas de violação mais altas, especialmente para *Gaming*. O RR, por sua vez, com sua abordagem de escalonamento sem priorização específica de tráfego, apresentou as maiores latências para todos os tipos de tráfego, como 310,10 ms para VoIP e 324,39 ms para vídeo, além das taxas de violação mais altas, com 69,43% para VoIP e 75,40% para vídeo, evidenciando suas claras limitações em cenários de alta demanda de tráfego.

Em cenários NRT, como o tráfego de vídeo, o Best CQI se aproximou mais do QAS, com latências médias semelhantes. No entanto, para tráfegos RT, o QAS se destacou por sua capacidade de atender aos requisitos de latência mais rigorosos, algo que o Best CQI não consegue replicar de maneira tão eficiente. O Round Robin, com sua abordagem simplificada e sem priorização de tráfego, continuou apresentando as maiores latências e as piores taxas de violação.

4.1.3 BLER

Os resultados da Tabela 13 mostram a média de BLER para diferentes tipos de tráfego e escalonadores nos três cenários avaliados, com 150, 350 e 550 usuários. Apesar das variações observadas, nota-se que os valores de BLER permanecem relativamente próximos entre os diferentes escalonadores, o que pode indicar uma capacidade geral semelhante de

Figura 20 – BLER por tipo de tráfego para o escalonador RR no cenário 1



lidar com erros de transmissão, mesmo com a variação na carga da rede. Na Figura 20 é possível identificar um gráfico de BLER por tipo de tráfego para o escalonador RR no primeiro cenário.

Tabela 13 – Média de BLER para diferentes tipos de usuários e escalonadores

Cenário 1: 150 usuários					
Escalonador	VoIP	<i>Constant Rate</i>	Video	<i>Gaming</i>	<i>Full Buffer</i>
RR	0,74	0,68	0,75	0,79	0,67
QAS	0,68	0,68	0,70	0,74	0,66
Best CQI	0,71	0,69	0,68	0,76	0,67
Cenário 2: 350 usuários					
Escalonador	VoIP	<i>Constant Rate</i>	Video	<i>Gaming</i>	<i>Full Buffer</i>
RR	0,75	0,74	0,76	0,79	0,69
QAS	0,71	0,70	0,72	0,75	0,67
Best CQI	0,77	0,72	0,69	0,76	0,68
Cenário 3: 550 usuários					
Escalonador	VoIP	<i>Constant Rate</i>	Video	<i>Gaming</i>	<i>Full Buffer</i>
RR	0,78	0,78	0,79	0,79	0,72
QAS	0,76	0,71	0,70	0,70	0,69
Best CQI	0,79	0,69	0,77	0,74	0,68

No Cenário 1 (150 usuários), o escalonador QAS apresenta um desempenho levemente superior em relação aos demais, com menores valores de BLER para tráfegos como VoIP (0,68) e *Gaming* (0,74). O Best CQI também se destaca em Video (0,68),

enquanto o escalonador RR apresenta valores ligeiramente mais altos, como 0,79 para *Gaming*. No entanto, as diferenças não são expressivamente amplas, sugerindo que, em cenários de baixa carga, todos os escalonadores conseguem lidar de forma razoável com os requisitos de qualidade.

Com o aumento da carga no Cenário 2 (350 usuários), observa-se um aumento nos valores de BLER para todos os escalonadores. O QAS continua apresentando os menores valores para tráfegos críticos, como VoIP (0,71) e *Gaming* (0,75), enquanto o Best CQI e o RR apresentam valores próximos, como 0,77 e 0,75 para VoIP, respectivamente. Essa proximidade reforça que, embora o QAS mostre alguma vantagem em priorizar tráfegos sensíveis, as diferenças de BLER entre os escalonadores ainda não são significativas a ponto de comprometer drasticamente a qualidade de serviço.

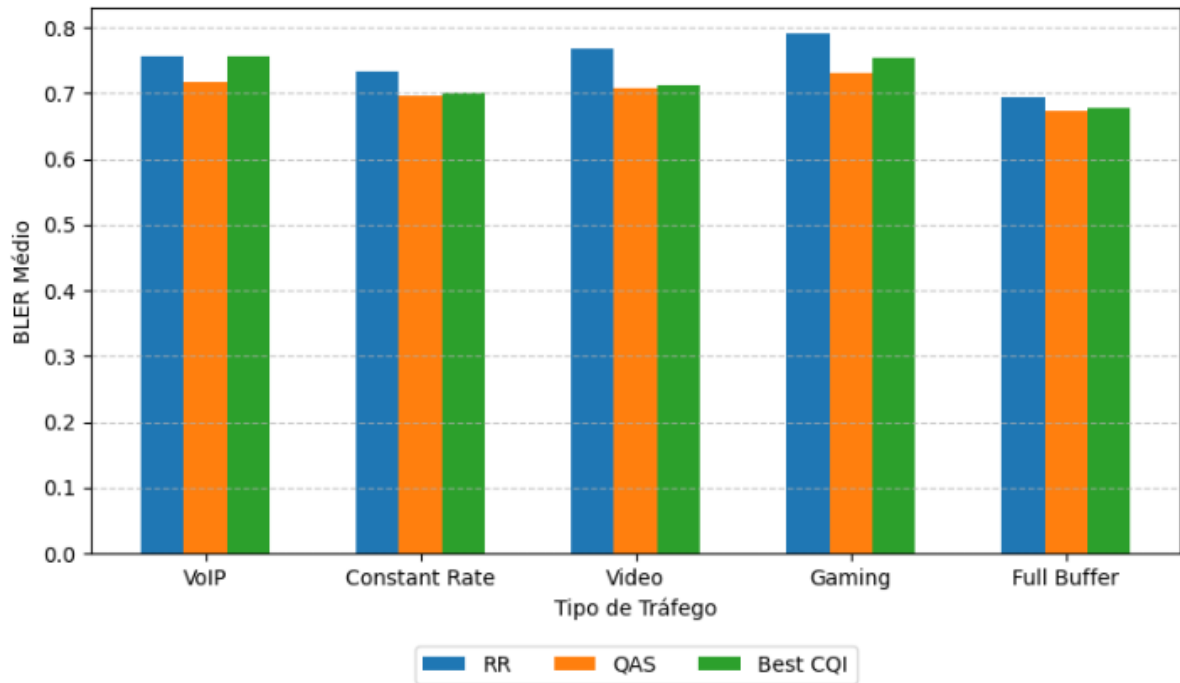
No Cenário 3 (550 usuários), com maior congestionamento, os valores de BLER aumentam para todos os tipos de tráfego e escalonadores. Apesar disso, os escalonadores continuam apresentando valores relativamente similares, como VoIP (0,76 para QAS, 0,78 para RR e 0,79 para Best CQI) e *Gaming* (0,70 para QAS e 0,74 para Best CQI). Embora o QAS demonstre uma ligeira superioridade em cenários críticos, a diferença de BLER entre ele e os demais escalonadores não é acentuada.

De maneira geral, os resultados apresentados na Tabela 13 indicam que, embora as diferenças de BLER entre os escalonadores não sejam muito grandes, elas revelam uma hierarquia de desempenho clara. O QAS apresenta os menores valores de BLER para tráfego RT, especialmente em cenários mais carregados, como o Cenário 3. Essa superioridade reflete sua capacidade de gerenciar recursos de forma eficiente, atendendo melhor às demandas de qualidade de serviço, particularmente em tráfegos sensíveis.

A média geral de BLER nos três cenários avaliados pode ser observada na Figura 21. Essa métrica consolida os resultados obtidos para cada tipo de tráfego, permitindo uma análise comparativa mais ampla entre os escalonadores. Conforme apresentado, o Best CQI apresenta um desempenho intermediário, com valores de BLER próximos aos do QAS, mas com pior eficiência em tráfegos mais exigentes e em cenários com maior congestionamento. Por outro lado, o escalonador RR, por não priorizar tipos específicos de tráfego, registra consistentemente os maiores valores de BLER, sendo o menos eficiente entre os três.

Apesar de as diferenças de BLER parecerem sutis, elas se tornam importantes em aplicações que exigem alta qualidade de serviço e eficiência, como cenários com alta densidade de usuários. Nesse contexto, o QAS se mostra a opção mais confiável, seguido pelo Best CQI, enquanto o RR apresenta limitações evidentes, sendo mais adequado para situações de menor criticidade.

Figura 21 – Média geral de BLER considerando os 3 cenários



4.2 Considerações Finais

Neste capítulo, foram apresentados e analisados os resultados de vazão, latência e BLER dos escalonadores RR, Best CQI e QAS, em diferentes densidades de usuários e tipos de tráfego. O QAS se destacou para o tráfego RT, enquanto o Best CQI beneficiou usuários em boas condições de canal, especialmente em situações menos congestionadas. O escalonador RR, sem priorização específica, apresentou desempenho estável, porém abaixo dos demais, especialmente em cenários com tráfegos em tempo real. No próximo capítulo, Conclusões, são discutidas as principais contribuições e limitações de cada escalonador.

5 Conclusões

O desempenho das redes 5G está diretamente ligado à eficiência na alocação de recursos, especialmente em cenários de alta demanda e diversidade de tráfego. Nesse contexto, os algoritmos de escalonamento desempenham um papel crucial na distribuição dos recursos da rede. Diante disso, este estudo comparou três escalonadores: RR, Best CQI e QAS, com o objetivo de avaliar seus impactos e limitações em diferentes condições de rede. Os resultados obtidos a partir das simulações permitiram uma análise detalhada do comportamento desses algoritmos em cenários variados de densidade de usuários e tipos de tráfego.

Em termos de BLER, observou-se que o escalonador RR teve os piores resultados, especialmente para tráfegos sensíveis ao atraso, como VoIP e *Gaming*, devido à sua abordagem de distribuição equitativa de recursos, o que resulta em uma alocação menos eficiente e maiores taxas de erro de transmissão. Em contraste, o QAS, embora com valores semelhantes aos dos outros escalonadores, demonstrou maior eficiência na redução do BLER para tráfegos em tempo real (RT), evidenciando uma melhor adaptação às necessidades específicas de cada tipo de usuário.

Quanto à latência, o QAS também se destacou, apresentando menores médias e taxas de violação em comparação aos outros algoritmos, especialmente em cenários com maior carga de rede. Sua capacidade de priorizar tráfegos em tempo real foi crucial para aplicações com requisitos de qualidade de serviço, mesmo sob condições de maior congestionamento. Já o Best CQI, embora eficiente ao favorecer usuários em boas condições de canal, teve dificuldades em garantir latências aceitáveis para tráfegos em tempo real, revelando suas limitações em situações onde a diversidade na qualidade de canal e o número elevado de usuários impactam significativamente o desempenho.

Em relação à vazão, o QAS demonstrou novamente sua superioridade, maximizando o uso da banda e garantindo um desempenho consistente para serviços RT, como VoIP, Video e *Gaming*. Mesmo em cenários mais congestionados, o QAS conseguiu alocar recursos de forma adaptativa, melhorando a experiência dos usuários que necessitavam de maior largura de banda e menor latência. O Best CQI foi competitivo para tráfegos NRT, como *Constant Rate*, onde as boas condições de canal favorecem um desempenho mais estável. No entanto, mostrou-se menos eficaz que o QAS em cenários de alta densidade. O escalonador RR, embora tenha distribuído os recursos de forma mais uniforme, não conseguiu competir com a eficiência dos outros dois escalonadores em termos de maximização de vazão, o que reflete sua limitação em adaptar-se às necessidades específicas de diferentes tipos de tráfego.

De maneira geral, o estudo destacou que o escalonador QAS é a melhor opção para cenários que exigem alta qualidade de serviço e baixos índices de latência e erro, especialmente em redes congestionadas com tráfegos RT. Sua abordagem adaptativa permite um melhor gerenciamento dos recursos disponíveis, assegurando uma experiência de uso mais satisfatória para os usuários de diferentes tipos de tráfego. O Best CQI mostrou-se uma alternativa viável para situações em que a qualidade do canal é o fator predominante, enquanto o RR pode ser útil em contextos que priorizam a distribuição equitativa de recursos, apesar de suas limitações em termos de flexibilidade e eficiência em ambientes dinâmicos e de alta carga.

Esses resultados sugerem que, na prática, a escolha do escalonador deve considerar não apenas a densidade de usuários, mas também a diversidade e a criticidade dos tipos de tráfego. O QAS provou ser uma solução robusta para redes modernas que necessitam lidar com múltiplos serviços simultâneos, oferecendo um desempenho superior em termos de latência e vazão, essenciais para uma experiência de qualidade em aplicações exigentes. No entanto, essa eficácia vem acompanhada de uma complexidade significativa em comparação com os demais algoritmos, principalmente em relação ao RR, que é mais simples e direto na alocação de recursos.

Além disso, a simulação revelou que, conforme a densidade de usuários aumentou, os parâmetros de desempenho, especialmente a latência e a vazão, sofreram degradação significativa. Isso indica que, além da seleção do algoritmo de escalonamento, é essencial considerar o dimensionamento da capacidade da rede. Um dimensionamento adequado assegura que a infraestrutura da rede possa atender a flutuações na demanda, minimizando a degradação da qualidade do serviço e garantindo a continuidade em condições de alta carga.

Para trabalhos futuros, sugere-se investigar novas estratégias de escalonamento que combinem as vantagens do QAS e do Best CQI, de forma a desenvolver algoritmos que possam adaptar a alocação de recursos com maior flexibilidade, considerando tanto a qualidade de canal quanto os requisitos específicos de cada tipo de tráfego. Outra possibilidade seria considerar a integração de técnicas de aprendizado de máquina para otimizar a alocação de recursos, aprendendo padrões de uso e ajustando a estratégia de escalonamento de forma autônoma e em tempo real.

Referências Bibliográficas

3GPP-TS-38.211. *Physical channels and modulation (Release 18) 3GPP TS 38.211*. 2024. Disponível em: <<https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=32>>. Citado na página 25.

ALI-YAHIYA, T. *Understanding LTE and its Performance*. New York, NY: Springer New York, 2011. ISBN 978-1-4419-6456-4. Disponível em: <<https://link.springer.com/10.1007/978-1-4419-6457-1>>. Citado na página 24.

ALQAHTANI, S.; ALHASSANY, M. Performance modeling and evaluation of novel scheduling algorithm for lte networks. In: . [S.l.: s.n.], 2013. p. 101–105. ISBN 978-0-7695-5043-5. Citado na página 27.

ANACOM. *Que serviços estão a ser implementados com o 5G?* 2023. Resposta atualizada em 21.08.2023. O Portal 5G foi criado pela ANACOM para partilhar, de forma alargada e participada, informação sobre o 5G. Disponível em: <<https://portal5g.pt/perguntas-frequentes/quais-os-servicos-que-deverao-ser-implementados-com-o-5g-e-a-partir-de-quando/>>. Citado na página 19.

ARCHITECTURE. In: 5G Explained. John Wiley Sons, Ltd, 2019. cap. 4, p. 71–103. ISBN 9781119275695. Disponível em: <<https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119275695.ch4>>. Citado na página 20.

CARDOSO, E. H. S. Análise de desempenho de algoritmos de escalonamento de recursos em redes LTE heterogêneas. 2015. Citado 2 vezes nas páginas 23 e 36.

CISCO. Cisco visual networking index: Global mobile data traffic forecast update, 2015–2020. 2016. Citado na página 15.

FASTENBAUER, A. et al. *User Manual: The Vienna 5G System Level Simulator*. [S.l.]: Institute of Telecommunications, TU Wien, 2023. Citado 3 vezes nas páginas 32, 34 e 35.

FERREIRA, M. V. G. Escalonamento de recursos em redes sem fio 5g baseado em otimização de retardo e de alocação de potência considerando comunicação dispositivo a dispositivo. 2021. Citado na página 27.

GROUP, T. . F. Engineering innovation and design. 2019. Citado na página 15.

GUARDIEIRO, P.; SILVA, E.; MOREIRA, J. Algoritmos qos-aware para o escalonamento downlink lte-a em cenários de tráfego misto: Uma comparação de desempenho. In: . [S.l.: s.n.], 2018. Citado na página 16.

GUPTA, A.; JHA, R. A survey of 5g network: Architecture and emerging technologies. *Access, IEEE*, v. 3, p. 1206–1232, ago. 2015. Citado na página 20.

HABAEBI, M. H. et al. Comparison between scheduling techniques in long term evolution. *IJUM Engineering Journal*, v. 14, n. 11, mar. 2013. ISSN 2289-7860. Disponível em:

- <<https://journals.iium.edu.my/ejournal/index.php/iiumej/article/view/354>>. Citado 2 vezes nas páginas 30 e 37.
- IOSIF, O.; BĂNICĂ, I. Lte performance evaluation based on two scheduling models. 2012. Citado 2 vezes nas páginas 28 e 37.
- KIHERO, A. B.; SOLAIJA, M. S. J.; ARSLAN, H. Inter-numerology interference for beyond 5g. *IEEE Access*, v. 7, p. 146512–146523, 2019. ISSN 2169-3536. Disponível em: <<https://ieeexplore.ieee.org/document/8861343/>>. Citado na página 25.
- KORHONEN JUHA, G. M. *Scheduling of data transmissions*. 2023. Disponível em: <<https://www.3gpp.org/technologies/scheduling>>. Citado na página 23.
- MAZIERO, C. Sistemas operacionais: Conceitos e mecanismos. 2019. Citado na página 15.
- MONEM, M. A. *Frame Structure for 5G New Radio Release 15*. 2024. Disponível em: <<https://moniem-tech.com/2024/09/15/frame-structure-for-5g-new-radio-release-15/>>. Citado na página 25.
- MULLER, M. K. et al. Flexible multi-node simulation of cellular mobile communications: the Vienna 5G System Level Simulator. *Eurasip Journal on Wireless Communications and Networking*, 2018. Citado na página 17.
- MUNIZ, F. J. Parallel load-balancing on message passing architectures. 1996. Accepted: 2016-08-29T19:13:27Z. Disponível em: <<http://www.repositorio.cdn.br:8080/jspui/handle/123456789/662>>. Citado na página 27.
- OSSEIRAN, A. et al. *Whitepaper on 5G wireless access: an overview*. 2020. Disponível em: <<https://www.ericsson.com/en/reports-and-papers/white-papers/5g-wireless-access-an-overview>>. Citado 2 vezes nas páginas 15 e 19.
- QUEIROZ, G. A. *A new channel and QoS-aware scheduling algorithm for real-time and non-real-time traffic in 5G heterogeneous networks*. Dissertação (Mestrado) — Universidade Federal de Pernambuco, 2022. Disponível em: <<http://orcid.org/0000-0003-4073-9435>>. Citado 3 vezes nas páginas 29, 31 e 33.
- SHIYAHIN, A.; SCHWARZ, S.; RUPP, M. Quality of service aware scheduling in mixed traffic wireless networks. In: *2022 IEEE 27th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*. [S.l.: s.n.], 2022. p. 159–165. Citado 6 vezes nas páginas 16, 30, 31, 38, 39 e 42.
- SILBERSCHATZ, A. *Operating System Concepts (9th, 2012.12)*. [S.l.: s.n.], 2013. ISBN 978-1-118-06333-0. Citado na página 28.
- SIROTKIN, S. *5G Radio Access Network Architecture: The Dark Side of 5G*. [S.l.]: Wiley-IEEE Press; 1st edition, 2020. Citado 4 vezes nas páginas 20, 21, 24 e 26.
- SOUZA, D. et al. O impacto de escalonadores de recursos e modulações no desempenho do downlink de redes 5g baseadas em lte. *Revista de Sistemas de Informação da FSMA*, 2018. Citado na página 15.
- SULTAN, A. *5G System Overview*. 2022. Disponível em: <<https://www.3gpp.org/technologies/5g-system-overview>>. Citado 2 vezes nas páginas 20 e 21.

APÊNDICE A – Código *Launcher*

O código a seguir foi desenvolvido em *MATLAB* utilizando o *Vienna 5G System Level Simulator*, baseado em um código pré-existente que foi modificado e aprimorado com novas funções para executar a simulação, exibir os resultados na forma de gráfico e texto, e armazenar os dados em arquivos do tipo *.mat*.

```
close all
clear
clc

% Inicializa o contador
executionCount = 0;

% launch simulation for RR
result_RR = simulate(@scenarios.openStreetMap,
    parameters.setting.SimulationType.local);

latencyFilename_RR = sprintf('RR_latencyData_run%d.mat',
    executionCount);
blerFilename_RR = sprintf('RR_blerData_run%d.mat', executionCount);
throughputFilename_RR = sprintf('RR_throughputData_run%d.mat',
    executionCount);

% Launch simulation for QAS
result_QAS = simulate(@scenarios.openStreetMap_QAS,
    parameters.setting.SimulationType.local);

latencyFilename_QAS = sprintf('QAS_latencyData_run%d.mat',
    executionCount);
blerFilename_QAS = sprintf('QAS_blerData_run%d.mat',
    executionCount);
throughputFilename_QAS = sprintf('QAS_throughputData_run%d.mat',
    executionCount);

% Launch simulation for Best CQI
result_BestCQI = simulate(@scenarios.openStreetMap_BestCQI,
    parameters.setting.SimulationType.local);

% Nome dos arquivos
latencyFilename_BestCQI = sprintf('BestCQI_latencyData_run%d.mat',
    executionCount);
blerFilename_BestCQI = sprintf('BestCQI_blerData_run%d.mat',
    executionCount);
```

```

throughputFilename_BestCQI =
    sprintf('BestCQI_throughputData_run%d.mat', executionCount);

%% Inicializa os tipos de trafego e cores

VideoUser_RR = result_RR.params.userParameters('Video').indices;
FullUser_RR = result_RR.params.userParameters('FullBuffer').indices;
ConstUser_RR =
    result_RR.params.userParameters('ConstantRate').indices;
VoIPUser_RR = result_RR.params.userParameters('VoIP').indices;
GamingUser_RR = result_RR.params.userParameters('Gaming').indices;

VideoUser_QAS = result_QAS.params.userParameters('Video').indices;
FullUser_QAS =
    result_QAS.params.userParameters('FullBuffer').indices;
ConstUser_QAS =
    result_QAS.params.userParameters('ConstantRate').indices;
VoIPUser_QAS = result_QAS.params.userParameters('VoIP').indices;
GamingUser_QAS = result_QAS.params.userParameters('Gaming').indices;

VideoUser_BestCQI =
    result_BestCQI.params.userParameters('Video').indices;
FullUser_BestCQI =
    result_BestCQI.params.userParameters('FullBuffer').indices;
ConstUser_BestCQI =
    result_BestCQI.params.userParameters('ConstantRate').indices;
VoIPUser_BestCQI =
    result_BestCQI.params.userParameters('VoIP').indices;
GamingUser_BestCQI =
    result_BestCQI.params.userParameters('Gaming').indices;

trafficUsers_RR = {VoIPUser_RR, ConstUser_RR, VideoUser_RR,
    GamingUser_RR, FullUser_RR};
trafficUsers_QAS = {VoIPUser_QAS, ConstUser_QAS, VideoUser_QAS,
    GamingUser_QAS, FullUser_QAS};
trafficUsers_BestCQI = {VoIPUser_BestCQI, ConstUser_BestCQI,
    VideoUser_BestCQI, GamingUser_BestCQI, FullUser_BestCQI};

trafficLabels = {'VoIP user', 'ConstantRate user', 'Video user',
    'Gaming user', 'FullBuffer user'};
trafficTypes = {'VoIP', 'ConstantRate', 'Video', 'Gaming',
    'FullBuffer'};
colors = {tools.myColors.matlabPurple, tools.myColors.matlabOrange,
    tools.myColors.matlabGreen, tools.myColors.matlabRed,
    tools.myColors.matlabBlue};

```

```

%% display results
%RR
plotLatency(result_RR, trafficUsers_RR, trafficLabels, colors,
    latencyFilename_RR);
plotBler(result_RR, trafficUsers_RR, trafficLabels, colors,
    blerFilename_RR, 'RR');
plotThroughput(result_RR, trafficUsers_RR, trafficLabels, colors,
    throughputFilename_RR, 'RR');
%QAS
plotLatency(result_QAS, trafficUsers_QAS, trafficLabels, colors,
    latencyFilename_QAS);
plotBler(result_QAS, trafficUsers_QAS, trafficLabels, colors,
    blerFilename_QAS, 'QAS');
plotThroughput(result_QAS, trafficUsers_QAS, trafficLabels, colors,
    throughputFilename_QAS, 'QAS');
%BestCQI
plotLatency(result_BestCQI, trafficUsers_BestCQI, trafficLabels,
    colors, latencyFilename_BestCQI);
plotBler(result_BestCQI, trafficUsers_BestCQI, trafficLabels,
    colors, blerFilename_BestCQI, 'Best CQI');
plotThroughput(result_BestCQI, trafficUsers_BestCQI, trafficLabels,
    colors, throughputFilename_BestCQI, 'Best CQI');
%% Vazao Geral
figure();
subplot(3, 2, 6);
plotNumerologyThroughputCombined(result_RR.networkResults.userList,
    result_RR.userThroughputMBitPerSec, ...
    result_QAS.networkResults.userList,
    result_QAS.userThroughputMBitPerSec, ...
    result_BestCQI.networkResults.userList,
    result_BestCQI.userThroughputMBitPerSec);
title('Vazao Geral');

%% ESTATISTICAS

maxAllowedLatencies = containers.Map(...
{'VoIP user', 'Video user', 'Gaming user'}, ...
[40, 100, 60]);

calculateLatencyStatistics(result_RR, trafficTypes, trafficLabels,
    maxAllowedLatencies, 'RR');
calculateLatencyStatistics(result_QAS, trafficTypes,
    trafficLabels, maxAllowedLatencies, 'QAS');
calculateLatencyStatistics(result_BestCQI, trafficTypes,
    trafficLabels, maxAllowedLatencies, 'Best CQI');

```

```

%% plot scene per user type

result_RR.plotScene(1);

% Plota as linhas em cinza claro conectando os usuarios as antenas
for iBaseStation = result_RR.networkResults.baseStationList
    for iAntenna = iBaseStation.antennaList
        posAnt = iAntenna.positionList(:,1);
        for iUser = iBaseStation.attachedUsers
            posUe = iUser.positionList(:,1);
            % Plota a linha cinza conectando usuario e antena
            plot3([posUe(1),posAnt(1)], [posUe(2),posAnt(2)],
                [posUe(3),posAnt(3)], 'Color', tools.myColors.lightGray);
        end
        pAntenna = plot3(posAnt(1), posAnt(2), posAnt(3), '0',
            'MarkerSize', 8, 'MarkerEdgeColor','b', 'MarkerFaceColor',
            'b');
    end
end

% Plota os pontos coloridos de acordo com o tipo de trafego

for iUser = VoIPUser_RR
    posUe = result_RR.networkResults.userList(iUser).positionList(:,1);
    pVoIP = plot3(posUe(1), posUe(2), posUe(3), 'o', 'MarkerSize', 4,
        'MarkerEdgeColor', tools.myColors.matlabPurple,
        'MarkerFaceColor', tools.myColors.matlabPurple);
end

for iUser = ConstUser_RR
    posUe = result_RR.networkResults.userList(iUser).positionList(:,1);
    pConst = plot3(posUe(1), posUe(2), posUe(3), 'o', 'MarkerSize', 4,
        'MarkerEdgeColor', tools.myColors.matlabOrange,
        'MarkerFaceColor', tools.myColors.matlabOrange);
end

for iUser = FullUser_RR
    posUe = result_RR.networkResults.userList(iUser).positionList(:,1);
    pFull = plot3(posUe(1), posUe(2), posUe(3), 'o', 'MarkerSize', 4,
        'MarkerEdgeColor', tools.myColors.matlabBlue, 'MarkerFaceColor',
        tools.myColors.matlabBlue);
end

for iUser = VideoUser_RR
    posUe = result_RR.networkResults.userList(iUser).positionList(:,1);
    pVideo = plot3(posUe(1), posUe(2), posUe(3), 'o', 'MarkerSize', 4,

```

```

        'MarkerEdgeColor', tools.myColors.matlabGreen,
        'MarkerFaceColor', tools.myColors.matlabGreen);
end

for iUser = GamingUser_RR
    posUe = result_RR.networkResults.userList(iUser).positionList(:,1);
    pGame = plot3(posUe(1), posUe(2), posUe(3), 'o', 'MarkerSize', 4,
        'MarkerEdgeColor', tools.myColors.matlabRed, 'MarkerFaceColor',
        tools.myColors.matlabRed);
end

% Legenda para os usuarios
legend([pVoIP, pConst, pVideo, pFull, pGame, pAntenna],{'Usuario VoIP',
    'Usuario Constant Rate', 'Usuario Video', 'Usuario Full Buffer',
    'Usuario Gaming', 'Estacao Base'});

% Definindo limites do grafico
roi = result_RR.params.regionOfInterest;
xlim([roi.xMin, roi.xMax]);
ylim([roi.yMin, roi.yMax]);
xlabel('Posicao x em m');
ylabel('Posicao y em m');
zlabel('Altura em m');

%% plot latency per user type

function plotLatency(result, trafficUsers, trafficLabels, colors,
    filename)
    figure();
    hold on;

    % Inicializa as celulas de dados de latencia
    latencyData = {};
    latencyLabels = {};

    % Indice para armazenamento valido
    validIndex = 1;

    for i = 1:length(trafficUsers)
        % Get user indices and calculate latency
        trafficUser = trafficUsers{i};
        LatencyCell = result.transmissionLatency(trafficUser, :);

        cellSize = size(result.transmissionLatency(trafficUser, :));
        n_User = cellSize(1);

        % Reshape and extract latency

```

```

    packetsLatency = reshape(LatencyCell,
        [n_User*result.params.time.numberOfChunks, 1]);
    Latency = [packetsLatency{:}];

    % Check if latency data is valid
    if all(isinf(Latency)) || isempty(Latency)
        warning(['Transmission latency plot for ' trafficLabels{i}
            ' will not be shown since insufficient number of packets
            has been transmitted.']);
        continue;
    end

    % Store latency data and labels
    latencyData{validIndex} = Latency;
    latencyLabels{validIndex} = trafficLabels{i};

    % Atualiza o indice de armazenamento valido
    validIndex = validIndex + 1;

    % Plot latency
    tools.myEcdf(Latency, 'Color', colors{i}, 'LineWidth', 2);
end

% Salvar dados
save(filename, 'latencyData', 'latencyLabels');

% Configura es do gr fico
grid on;
xlabel('Lat ncia (ms)');
ylabel('ECDF');
legend(trafficLabels, 'Location', 'southeast');
title('Lat ncia por tipo de tr fego');
hold off;

end

%%

function calculateLatencyStatistics(result, trafficUsers,
    trafficLabels, maxAllowedLatencies, schedulerType)

    % Inicializa as celulas de dados de latencia e rotulos
    latencyData = {};
    latencyLabels = {};

    % ndice para armazenamento valido
    validIndex = 1;

```

```

fprintf('--- Estatísticas de Latência por Tipo de Tráfego (%s)
---\n', schedulerType);

% Loop através de cada grupo de tráfego
for i = 1:length(trafficUsers)
    % Obter índices de usuários e calcular latência
    trafficUser = trafficUsers{i};
    LatencyCell = result.transmissionLatency(trafficUser, :);

    % Calcular o número de usuários e redimensionar os dados de
    % latência
    cellSize = size(result.transmissionLatency(trafficUser, :));
    n_User = cellSize(1);

    % Redimensionar e extrair latência dos pacotes transmitidos
    packetsLatency = reshape(LatencyCell, [n_User *
        result.params.time.numberOfChunks, 1]);
    Latency = [packetsLatency{:}];

    % Remover valores 'Inf' e 'NaN' antes de calcular as
    % estatísticas
    Latency = Latency(~isinf(Latency) & ~isnan(Latency));

    % Verificar se os dados de latência são válidos após a
    % filtragem
    if isempty(Latency)
        warning(['Transmission latency plot for ', trafficLabels{i}
            ' will not be shown since insufficient number of packets
            has been transmitted.']);
        continue;
    end

    % Calcular as estatísticas de latência
    minLatency = min(Latency); % Latência mínima
    maxLatency = max(Latency); % Latência máxima
    meanLatency = mean(Latency); % Latência média

    % Definir a latência máxima permitida para o tipo atual de
    % tráfego
    if isKey(maxAllowedLatencies, trafficLabels{i})
        maxAllowedLatency = maxAllowedLatencies(trafficLabels{i});
    else
        maxAllowedLatency = meanLatency; % Valor padrão se o tipo
        % não for RT
    end

    % Calcular o percentual de usuários que violam a latência

```

```

        maxima permitida
        violatingUsers = sum(Latency > maxAllowedLatency);
        totalUsers = length(Latency);
        violationPercent = (violatingUsers / totalUsers) * 100;

        % Exibir as estatísticas no console para cada tipo de trafego e
        escalonador
        fprintf('Escalonador: %s\n', schedulerType);
        fprintf('Tipo de Tr fego: %s\n', trafficLabels{i});
        fprintf(' - Lat ncia m nima: %.2f ms\n', minLatency);
        fprintf(' - Lat ncia m xima: %.2f ms\n', maxLatency);
        fprintf(' - Lat ncia m dia: %.2f ms\n', meanLatency);
        fprintf(' - Percentual de usuarios que violam a latencia
            m xima permitida (%.2f ms): %.2f%%\n\n', maxAllowedLatency,
            violationPercent);

        % Store latency data and labels
        latencyData{validIndex} = Latency;
        latencyLabels{validIndex} = strcat(trafficLabels{i}, ' - ',
            schedulerType);

        % Atualizar ndice de armazenamento
        validIndex = validIndex + 1;
    end

end

%% plot Bler per user type

function plotBler(result, trafficUsers, trafficLabels, colors,
    filename, schedulerType)
    % Inicializa a figura

    fprintf('Escalonador: %s\n', schedulerType);
    figure('Position', [100, 100, 1200, 600]);
    hold on;

    % Inicializar a posi o inicial para cada grupo de usuarios
    startPos = 1;

    % Inicializa o valor maximo do eixo x
    xMax = 0;

    % Inicializa as celulas de dados de BLER
    blerData = cell(1, length(trafficUsers));
    blerLabels = cell(1, length(trafficUsers));

```

```
for i = 1:length(trafficUsers)
    % Obter índices de usuarios e calcular o BLER
    trafficUser = trafficUsers{i};
    BlerData = result.BLER.DL(trafficUser, :);

    % Filtrar BLER dos usuarios que n o sao agendados em todos os
    slots
    rowsOfUnscheduled = all(BlerData == 0, 2);

    % Substituir zeros por NaNs para usuarios nao agendados e n o
    ROI
    BlerData(BlerData == 0) = nan;

    % Calcular a media de BLER para cada tipo de usuario (omitindo
    NaNs)
    meanBler = mean(BlerData, 2, 'omitnan');

    % Atribuir BLER zero para usu rios n o agendados
    meanBler(rowsOfUnscheduled) = 0;

    % Remover NaNs
    meanBler = meanBler(~isnan(meanBler));

    % Atualizar o valor maximo do eixo x
    xMax = max(xMax, startPos + length(meanBler) - 1);

    % Definir o fim da posicao para o grupo atual
    endPos = startPos + length(meanBler) - 1;

    % Plotar BLER para o tipo de usuario atual
    stem(startPos:endPos, meanBler, 'DisplayName',
        trafficLabels{i}, ...
        'LineWidth', 0.5, 'MarkerSize', 4, ...
        'MarkerEdgeColor', colors{i}, 'Color', colors{i});

    % Atualizar a posicao inicial para o proximo grupo de usuarios
    startPos = endPos + 1;

    % Armazenar dados de BLER
    blerData{i} = meanBler;
    blerLabels{i} = trafficLabels{i};

    BlerMean = mean(meanBler);

    fprintf('Media de Bler para %s: %.2f\n', trafficLabels{i},
        BlerMean);
```

```

end

% Salvar dados
save(filename, 'blerData', 'blerLabels');

% Ajustar o limite do eixo x dinamicamente
xlim([1, xMax]);
ylim([1e-1, 1]);

% Definindo as marcacoes do eixo Y de 0 a 1 com incrementos
regulares
yticks(0:0.1:1);

% Configuracoes do grafico
set(gca, 'YScale', 'log');
grid on;
xlabel('User Nr');
ylabel('Average BLER');
title('BLER por tipo de trafego');
legend('show', 'Location', 'southeast');
hold off;
end

%% plot throughput per user type

function plotThroughput(result, trafficUsers, trafficLabels, colors,
filename, schedulerType)
% Inicializa a figura
figure();
hold on;

throughputData = cell(1, length(trafficUsers));
throughputLabels = cell(1, length(trafficUsers));

% Plotar throughput para cada tipo de usu rio
for i = 1:length(trafficUsers)
    % Obter throughput para o tipo de usu rio atual
    trafficUser = trafficUsers{i};
    ThroughputData =
        result.userThroughputMBitPerSec.DL(trafficUser, :);

    % Armazenar dados de throughput
    throughputData{i} = ThroughputData(:);
    throughputLabels{i} = trafficLabels{i};

    % Plotar a ECDF do throughput

```

```

tools.myEcdf(ThroughputData(:), 'Color', colors{i},
            'LineWidth', 2);

% Calcular a m dia de throughput
Vazao = ThroughputData(~isinf(ThroughputData) &
    ~isnan(ThroughputData) & ThroughputData ~= 0);
meanThroughput = mean(Vazao);

% Exibir a m dia de throughput
fprintf('Escalonador: %s\n', schedulerType);
fprintf('Tipo de Tr fego: %s\n', trafficLabels{i});
fprintf(' - Vaz o m dia: %.2f Mbit/s\n\n', meanThroughput);
end

% Salvar dados de throughput em um arquivo .mat
save(filename, 'throughputData', 'throughputLabels');

% Configura es do gr fico
grid on;
xlabel('Vaz o em Mbit/s');
ylabel('ECDF');
legend(trafficLabels, 'Location', 'southeast');
title('Vaz o por por tipo de tr fego');
hold off;
end

% Helper functions for plotting
function plotNumerologyThroughputCombined(userListRR, throughputRR,
    userListQAS, throughputQAS, userListBestCQI, throughputBestCQI)
% Plot throughput by numerology/technology for all schedulers
numerologies = unique([userListRR.numerology, userListQAS.numerology,
    userListBestCQI.numerology]);

colorRR = [0 0.4470 0.7410]; % Azul
colorQAS = [0.8500 0.3250 0.0980]; % Laranja
colorBestCQI = [0.9290 0.6940 0.1250]; % Amarelo

subplot(1,1,1);
hold on;
legends = {};
% Para acumular as m dias
meanThroughputRR = [];
meanThroughputQAS = [];
meanThroughputBestCQI = [];

% Para calcular o percentual de usu rios acima da m dia

```

```

totalUsersRR = 0;
aboveAvgUsersRR = 0;
totalUsersQAS = 0;
aboveAvgUsersQAS = 0;
totalUsersBestCQI = 0;
aboveAvgUsersBestCQI = 0;

for num = numerologies
    userMaskRR = [userListRR.numerology] == num;
    userMaskQAS = [userListQAS.numerology] == num;
    userMaskBestCQI = [userListBestCQI.numerology] == num;

    % Plot para RR
    if any(userMaskRR)
        rrThroughput = mean(throughputRR.DL(userMaskRR, :), 2,
            'omitnan');
        tools.myEcdf(rrThroughput, 'Color', colorRR, 'LineWidth',
            2);
        legends{end+1} = 'RR';

        % Linha pontilhada para RR
        maxRR = max(rrThroughput);
        plot([maxRR, maxRR], [0, 1], 'LineStyle', '--',
            'LineWidth', 1.5, 'Color', colorRR);
        legends{end+1} = 'RR - max';

        % Acumula a media e o numero total de usuarios
        meanThroughputRR = [meanThroughputRR; mean(rrThroughput)];
        totalUsersRR = totalUsersRR + sum(userMaskRR);
        aboveAvgUsersRR = aboveAvgUsersRR + sum(rrThroughput >
            mean(rrThroughput)); % Contagem de usuarios acima da
            media
    end

    % Plot para QAS
    if any(userMaskQAS)
        gasThroughput = mean(throughputQAS.DL(userMaskQAS, :), 2,
            'omitnan');
        tools.myEcdf(gasThroughput, 'Color', colorQAS, 'LineWidth',
            2);
        legends{end+1} = 'QAS';

        % Linha pontilhada para QAS
        maxQAS = max(gasThroughput);
        plot([maxQAS, maxQAS], [0, 1], 'LineStyle', '--',
            'LineWidth', 1.5, 'Color', colorQAS);
        legends{end+1} = 'QAS - max';
    end
end

```

```

        % Acumula a media e o numero total de usuarios
        meanThroughputQAS = [meanThroughputQAS;
            mean(qasThroughput)];
        totalUsersQAS = totalUsersQAS + sum(userMaskQAS);
        aboveAvgUsersQAS = aboveAvgUsersQAS + sum(qasThroughput >
            mean(qasThroughput)); % Contagem de usuarios acima da
            media
    end

    % Plot para Best CQI
    if any(userMaskBestCQI)
        bestCQIThroughput =
            mean(throughputBestCQI.DL(userMaskBestCQI, :), 2,
                'omitnan');
        tools.myEcdf(bestCQIThroughput, 'Color', colorBestCQI,
            'LineWidth', 2);
        legends{end+1} = 'Best CQI';

        % Linha pontilhada para Best CQI
        maxBestCQI = max(bestCQIThroughput);
        plot([maxBestCQI, maxBestCQI], [0, 1], 'LineStyle', '--',
            'LineWidth', 1.5, 'Color', colorBestCQI);
        legends{end+1} = 'Best CQI - max';

        % Acumula a media e o numero total de usuarios
        meanThroughputBestCQI = [meanThroughputBestCQI;
            mean(bestCQIThroughput)];
        totalUsersBestCQI = totalUsersBestCQI +
            sum(userMaskBestCQI);
        aboveAvgUsersBestCQI = aboveAvgUsersBestCQI +
            sum(bestCQIThroughput > mean(bestCQIThroughput)); %
            Contagem de usuarios acima da media
    end
end

% Exibe a media de throughput por escalonador
fprintf('Escalonador: %s\n', 'RR');
fprintf(' - Vazao media: %.4f Mbit/s\n', mean(meanThroughputRR));
fprintf(' - Percentual de usuarios acima da media: %.2f%%\n\n',
    (aboveAvgUsersRR / totalUsersRR) * 100);

fprintf('Escalonador: %s\n', 'QAS');
fprintf(' - Vazao media: %.4f Mbit/s\n', mean(meanThroughputQAS));
fprintf(' - Percentual de usuarios acima da media: %.2f%%\n\n',
    (aboveAvgUsersQAS / totalUsersQAS) * 100);

```

```
fprintf('Escalonador: %s\n', 'Best CQI');
fprintf(' - Vazao edia: %.4f Mbit/s\n',
    mean(meanThroughputBestCQI));
fprintf(' - Percentual de usuarios acima da media: %.2f%%\n\n',
    (aboveAvgUsersBestCQI / totalUsersBestCQI) * 100);

title('Vazao');
xlabel('Vazao em Mbit/s');
ylabel('ECDF');
grid on;
legend(legends);

end
```

Listing A.1 – Launcher para simulação com os escalonadores

APÊNDICE B – Código para o escalonador RR

O código a seguir foi desenvolvido em *MATLAB* utilizando o *Vienna 5G System Level Simulator*. Ele se baseia em um código de cenário pré-existente, que foi modificado e aprimorado com novas funções para avaliar o desempenho do escalonador RR. O número de usuários foi ajustado para 150, 350 e 550, permitindo a análise do comportamento do algoritmo sob diferentes condições de carga.

```
function params = openStreetMap(params)

%% General Configuration
rng(241222)

% time config
params.time.slotDuration      = 1e-3;
params.time.slotsPerChunk     = 2000;

% Scheduler type
params.schedulerParameters.spectrumScheduler =
    parameters.setting.SchedulerType.roundRobin;

% enable/disable ini
params.calculateIni = true;
params.iniOversampling = 2;

% define the region of interest and interference region
params.regionOfInterest.xSpan = 400;
params.regionOfInterest.ySpan = 300;
params.regionOfInterest.zSpan = 50;

% transmission parameters
params.carrierDL.centerFrequencyGHz = 2;
params.transmissionParameters.DL.bandwidthHz = 10e6;

%% Blockages
% The following parameters describe the specification of an
    OpenStreetMap
% city. The desired real-world area is specified via latitudes and
% longitudes, which can easily be obtained from openstreetmap.com via
    the
```

```

% "Export" feature.
openStreetMapCity = parameters.city.OpenStreetMap;
openStreetMapCity.makeStreets      = true;
openStreetMapCity.latitude         = [-18.920252, -18.917542];
openStreetMapCity.longitude        = [-48.259442, -48.256738];
openStreetMapCity.streetWidth      = 5;
% All buildings have random heights, which are sampled from a uniform
% distribution bounded by the minimum and maximum building height. For
% reproducibility it is possible to specify an integer seed for the
% height
% random number generator.
openStreetMapCity.minBuildingHeight = 10;
openStreetMapCity.maxBuildingHeight = 25;
openStreetMapCity.heightRandomSeed  = 'shuffle';
openStreetMapCity.wallLossdB        = 0;
% The configuration of a city can be stored to and loaded from JSON
% files.
% This also allows for manually setting building heights and wall loss.
openStreetMapCity.saveFile          = [];
openStreetMapCity.loadFile          = [];
params.cityParameters('OSMCity') = openStreetMapCity;

%% pathloss model container
outdoor = parameters.setting.Indoor.outdoor;
indoor  = parameters.setting.Indoor.indoor;
LOS     = parameters.setting.Los.LOS;
NLOS    = parameters.setting.Los.NLOS;
macro   = parameters.setting.BaseStationType.macro;

if tools.checkToolbox('Communications')

    pathloss = parameters.pathlossParameters.UrbanMacro5G;

    params.pathlossModelContainer.modelMap{macro, outdoor, LOS}
        = pathloss;
    params.pathlossModelContainer.modelMap{macro, outdoor, NLOS}
        = pathloss;
    params.pathlossModelContainer.modelMap{macro, outdoor, LOS}
        = pathloss;
    params.pathlossModelContainer.modelMap{macro, outdoor, NLOS}
        = pathloss;
    params.pathlossModelContainer.modelMap{macro, indoor, LOS}
        = pathloss;
    params.pathlossModelContainer.modelMap{macro, indoor, NLOS}
        = pathloss;
    params.pathlossModelContainer.modelMap{macro, indoor, LOS}
        = pathloss;

```

```

        params.pathlossModelContainer.modelMap{macro,    indoor,    NLOS}
            = pathloss;
end

%% Base Stations
antenna5G = parameters.basestation.antennas.ThreeSector;
antenna5G.nTX          = 1;
antenna5G.numerology    = 1; % assign numerology 1
antenna5G.technology    =
    parameters.setting.NetworkElementTechnology.NRMN_5G;
antenna5G.alwaysOn      = true;

% precoders
precoderTechnology = parameters.precoders.Technology();
precoderTechnology.setTechPrecoder(
    parameters.setting.NetworkElementTechnology.NRMN_5G,
    parameters.precoders.LteDL);

% First Base Station
BS1 = parameters.basestation.PredefinedPositions;
BS1.positions = [-191.968; 70.4545]; % [x;y] coordinate in the OSM
    city
BS1.nSectors    = 3;
BS1.antenna      = antenna5G;
BS1.precoder.DL = precoderTechnology;
params.baseStationParameters('BS1') = BS1;

% Second Base Station
BS2 = parameters.basestation.PredefinedPositions;
BS2.positions = [147.677; -30.2005]; % [x;y] coordinate in the OSM
    city
BS2.nSectors    = 3;
BS2.antenna      = antenna5G;
BS2.precoder.DL = precoderTechnology;
params.baseStationParameters('BS2') = BS2;

%% Users
% Video users
Users_Video = parameters.user.Poisson2D;
Users_Video.nElements          = 37;
Users_Video.nRX                 = 1;
Users_Video.channelModel        =
    parameters.setting.ChannelModel.PedB;
Users_Video.trafficModel.type   =
    parameters.setting.TrafficModelType.Video;
Users_Video.trafficModel.delayConstraint = 100;
Users_Video.numerology          = 1;
Users_Video.technology          =

```

```

    parameters.setting.NetworkElementTechnology.NRMN_5G;
params.userParameters('Video') = Users_Video;

% VoIP users
Users_voip = parameters.user.Poisson2D;
Users_voip.nElements          = 37;
Users_voip.nRX                = 1;
Users_voip.channelModel       =
    parameters.setting.ChannelModel.PedB;
Users_voip.trafficModel.type   =
    parameters.setting.TrafficModelType.VoIP;
Users_voip.trafficModel.delayConstraint = 40;
Users_voip.numerology          = 1;
Users_voip.technology          =
    parameters.setting.NetworkElementTechnology.NRMN_5G;
params.userParameters('VoIP') = Users_voip;

% Gaming users
Users_game = parameters.user.Poisson2D;
Users_game.nElements          = 37;
Users_game.nRX                = 1;
Users_game.channelModel       =
    parameters.setting.ChannelModel.PedB;
Users_game.trafficModel.type   =
    parameters.setting.TrafficModelType.Gaming;
Users_game.trafficModel.delayConstraint = 60;
Users_game.numerology          = 1;
Users_game.technology          =
    parameters.setting.NetworkElementTechnology.NRMN_5G;
params.userParameters('Gaming') = Users_game;

% Constant rate users
Users_CR = parameters.user.Poisson2D;
Users_CR.nElements          = 22;
Users_CR.nRX                = 1;
Users_CR.channelModel       = parameters.setting.ChannelModel.PedB;
Users_CR.trafficModel.type   =
    parameters.setting.TrafficModelType.ConstantRate;
Users_CR.numerology          = 1;
Users_CR.technology          =
    parameters.setting.NetworkElementTechnology.NRMN_5G;
params.userParameters('ConstantRate') = Users_CR;

% Full buffer users
Users_FB = parameters.user.Poisson2D;
Users_FB.nElements          = 15;
Users_FB.nRX                = 1;

```

```
Users_FB.channelModel          = parameters.setting.ChannelModel.PedB;
Users_FB.trafficModel.type     =
    parameters.setting.TrafficModelType.FullBuffer;
Users_FB.trafficModel.numSlots = 0;
Users_FB.numerology            = 1;
Users_FB.technology            =
    parameters.setting.NetworkElementTechnology.NRMN_5G;
params.userParameters('FullBuffer') = Users_FB;
end
```

Listing B.1 – Código para o escalonador RR

APÊNDICE C – Código para o escalonador Best CQI

O código a seguir foi desenvolvido em *MATLAB* utilizando o *Vienna 5G System Level Simulator*. Ele se baseia em um código de cenário pré-existente, que foi modificado e aprimorado com novas funções para avaliar o desempenho do escalonador Best CQI. O número de usuários foi ajustado para 150, 350 e 550, permitindo a análise do comportamento do algoritmo sob diferentes condições de carga.

```
function params = openStreetMap_BestCQI(params)

%% General Configuration

rng(241222)

% time config
params.time.slotDuration      = 1e-3;
params.time.slotsPerChunk     = 2000;

% Scheduler type
params.schedulerParameters.spectrumScheduler =
    parameters.setting.SchedulerType.bestCqi;

% enable/disable ini
params.calculateIni = true;
params.iniOversampling = 2;

% define the region of interest and interference region
params.regionOfInterest.xSpan = 400;
params.regionOfInterest.ySpan = 300;
params.regionOfInterest.zSpan = 50;

% transmission parameters
params.carrierDL.centerFrequencyGHz = 2;
params.transmissionParameters.DL.bandwidthHz = 10e6;

%% Blockages
% The following parameters describe the specification of an
    OpenStreetMap
% city. The desired real-world area is specified via latitudes and
% longitudes, which can easily be obtained from openstreetmap.com via
```

```

    the
% "Export" feature.
openStreetMapCity = parameters.city.OpenStreetMap;
openStreetMapCity.makeStreets      = true;
openStreetMapCity.latitude         = [-18.920252, -18.917542];
openStreetMapCity.longitude        = [-48.259442, -48.256738];
openStreetMapCity.streetWidth      = 5;
% All buildings have random heights, which are sampled from a uniform
% distribution bounded by the minimum and maximum building height. For
% reproducibility it is possible to specify an integer seed for the
    height
% random number generator.
openStreetMapCity.minBuildingHeight = 10;
openStreetMapCity.maxBuildingHeight = 25;
openStreetMapCity.heightRandomSeed  = 'shuffle';
openStreetMapCity.wallLossdB        = 0;
% The configuration of a city can be stored to and loaded from JSON
    files.
% This also allows for manually setting building heights and wall loss.
openStreetMapCity.saveFile          = [];
openStreetMapCity.loadFile          = [];
params.cityParameters('OSMCity') = openStreetMapCity;

%% pathloss model container
outdoor = parameters.setting.Indoor.outdoor;
indoor  = parameters.setting.Indoor.indoor;
LOS     = parameters.setting.Los.LOS;
NLOS    = parameters.setting.Los.NLOS;
macro   = parameters.setting.BaseStationType.macro;

if tools.checkToolbox('Communications')

    pathloss = parameters.pathlossParameters.UrbanMacro5G;

    params.pathlossModelContainer.modelMap{macro, outdoor, LOS}
        = pathloss;
    params.pathlossModelContainer.modelMap{macro, outdoor, NLOS}
        = pathloss;
    params.pathlossModelContainer.modelMap{macro, outdoor, LOS}
        = pathloss;
    params.pathlossModelContainer.modelMap{macro, outdoor, NLOS}
        = pathloss;
    params.pathlossModelContainer.modelMap{macro, indoor, LOS}
        = pathloss;
    params.pathlossModelContainer.modelMap{macro, indoor, NLOS}
        = pathloss;
    params.pathlossModelContainer.modelMap{macro, indoor, LOS}

```

```

        = pathloss;
    params.pathlossModelContainer.modelMap{macro,    indoor,    NLOS}
        = pathloss;
end

%% Base Stations
antenna5G = parameters.basestation.antennas.ThreeSector;
antenna5G.nTX          = 1;
antenna5G.numerology    = 1; % assign numerology 1
antenna5G.technology    =
    parameters.setting.NetworkElementTechnology.NRMN_5G;
antenna5G.alwaysOn      = true;

% precoders
precoderTechnology = parameters.precoders.Technology();
precoderTechnology.setTechPrecoder(
    parameters.setting.NetworkElementTechnology.NRMN_5G,
    parameters.precoders.LteDL);

% First Base Station
BS1 = parameters.basestation.PredefinedPositions;
BS1.positions = [-191.968; 70.4545]; % [x;y] coordinate in the OSM
    city
BS1.nSectors    = 3;
BS1.antenna      = antenna5G;
BS1.precoder.DL = precoderTechnology;
params.baseStationParameters('BS1') = BS1;
% Second Base Station
BS2 = parameters.basestation.PredefinedPositions;
BS2.positions = [147.677; -30.2005]; % [x;y] coordinate in the OSM
    city
BS2.nSectors    = 3;
BS2.antenna      = antenna5G;
BS2.precoder.DL = precoderTechnology;
params.baseStationParameters('BS2') = BS2;

%% Users
% Video users
Users_Video = parameters.user.Poisson2D;
Users_Video.nElements = 37;
Users_Video.nRX        = 1;
Users_Video.channelModel =
    parameters.setting.ChannelModel.PedB;
Users_Video.trafficModel.type =
    parameters.setting.TrafficModelType.Video;
Users_Video.trafficModel.delayConstraint = 100;
Users_Video.numerology = 1;

```

```

Users_Video.technology          =
    parameters.setting.NetworkElementTechnology.NRMN_5G;
params.userParameters('Video') = Users_Video;

% VoIP users
Users_voip = parameters.user.Poisson2D;
Users_voip.nElements          = 37;
Users_voip.nRX                 = 1;
Users_voip.channelModel        =
    parameters.setting.ChannelModel.PedB;
Users_voip.trafficModel.type    =
    parameters.setting.TrafficModelType.VoIP;
Users_voip.trafficModel.delayConstraint = 40;
Users_voip.numerology           = 1;
Users_voip.technology           =
    parameters.setting.NetworkElementTechnology.NRMN_5G;
params.userParameters('VoIP') = Users_voip;

% Gaming users
Users_game = parameters.user.Poisson2D;
Users_game.nElements          = 37;
Users_game.nRX                 = 1;
Users_game.channelModel        =
    parameters.setting.ChannelModel.PedB;
Users_game.trafficModel.type    =
    parameters.setting.TrafficModelType.Gaming;
Users_game.trafficModel.delayConstraint = 60;
Users_game.numerology           = 1;
Users_game.technology           =
    parameters.setting.NetworkElementTechnology.NRMN_5G;
params.userParameters('Gaming') = Users_game;

% Constant rate users
Users_CR = parameters.user.Poisson2D;
Users_CR.nElements          = 22;
Users_CR.nRX                 = 1;
Users_CR.channelModel        = parameters.setting.ChannelModel.PedB;
Users_CR.trafficModel.type    =
    parameters.setting.TrafficModelType.ConstantRate;
Users_CR.numerology           = 1;
Users_CR.technology           =
    parameters.setting.NetworkElementTechnology.NRMN_5G;
params.userParameters('ConstantRate') = Users_CR;

% Full buffer users
Users_FB = parameters.user.Poisson2D;
Users_FB.nElements          = 15;

```

```
Users_FB.nRX                      = 1;
Users_FB.channelModel              = parameters.setting.ChannelModel.PedB;
Users_FB.trafficModel.type        =
    parameters.setting.TrafficModelType.FullBuffer;
Users_FB.trafficModel.numSlots    = 0;
Users_FB.numerology                = 1;
Users_FB.technology                =
    parameters.setting.NetworkElementTechnology.NRMN_5G;
params.userParameters('FullBuffer') = Users_FB;
end
```

Listing C.1 – Código para o escalonador Best CQI

APÊNDICE D – Código para o escalonador QAS

O código a seguir foi desenvolvido em *MATLAB* utilizando o *Vienna 5G System Level Simulator*. Ele se baseia em um código de cenário pré-existente, que foi modificado e aprimorado com novas funções para avaliar o desempenho do escalonador QAS. O número de usuários foi ajustado para 150, 350 e 550, permitindo a análise do comportamento do algoritmo sob diferentes condições de carga.

```
function params = openStreetMap_QAS(params)

%% General Configuration

rng(241222)

% time config
params.time.slotDuration      = 1e-3;
params.time.slotsPerChunk     = 2000;

% Scheduler type
params.schedulerParameters.spectrumScheduler =
    parameters.setting.SchedulerType.qosAware;

% enable/disable ini
params.calculateIni = true;
params.iniOversampling = 2;

% define the region of interest and interference region
params.regionOfInterest.xSpan = 400;
params.regionOfInterest.ySpan = 300;
params.regionOfInterest.zSpan = 50;

% transmission parameters
params.carrierDL.centerFrequencyGHz = 2;
params.transmissionParameters.DL.bandwidthHz = 10e6;

%% Blockages
% The following parameters describe the specification of an
    OpenStreetMap
% city. The desired real-world area is specified via latitudes and
% longitudes, which can easily be obtained from openstreetmap.com via
```

```

    the
% "Export" feature.
openStreetMapCity = parameters.city.OpenStreetMap;
openStreetMapCity.makeStreets      = true;
openStreetMapCity.latitude         = [-18.920252, -18.917542];
openStreetMapCity.longitude        = [-48.259442, -48.256738];
openStreetMapCity.streetWidth      = 5;
% All buildings have random heights, which are sampled from a uniform
% distribution bounded by the minimum and maximum building height. For
% reproducibility it is possible to specify an integer seed for the
    height
% random number generator.
openStreetMapCity.minBuildingHeight = 10;
openStreetMapCity.maxBuildingHeight = 25;
openStreetMapCity.heightRandomSeed  = 'shuffle';
openStreetMapCity.wallLossdB        = 0;
% The configuration of a city can be stored to and loaded from JSON
    files.
% This also allows for manually setting building heights and wall loss.
openStreetMapCity.saveFile          = [];
openStreetMapCity.loadFile          = [];
params.cityParameters('OSMCity') = openStreetMapCity;

%% pathloss model container
outdoor = parameters.setting.Indoor.outdoor;
indoor  = parameters.setting.Indoor.indoor;
LOS     = parameters.setting.Los.LOS;
NLOS    = parameters.setting.Los.NLOS;
macro   = parameters.setting.BaseStationType.macro;

if tools.checkToolbox('Communications')

    pathloss = parameters.pathlossParameters.UrbanMacro5G;

    params.pathlossModelContainer.modelMap{macro, outdoor, LOS}
        = pathloss;
    params.pathlossModelContainer.modelMap{macro, outdoor, NLOS}
        = pathloss;
    params.pathlossModelContainer.modelMap{macro, outdoor, LOS}
        = pathloss;
    params.pathlossModelContainer.modelMap{macro, outdoor, NLOS}
        = pathloss;
    params.pathlossModelContainer.modelMap{macro, indoor, LOS}
        = pathloss;
    params.pathlossModelContainer.modelMap{macro, indoor, NLOS}
        = pathloss;
    params.pathlossModelContainer.modelMap{macro, indoor, LOS}

```

```

        = pathloss;
    params.pathlossModelContainer.modelMap{macro,    indoor,    NLOS}
        = pathloss;
end

%% Base Stations
antenna5G = parameters.basestation.antennas.ThreeSector;
antenna5G.nTX          = 1;
antenna5G.numerology    = 1; % assign numerology 1
antenna5G.technology    =
    parameters.setting.NetworkElementTechnology.NRMN_5G;
antenna5G.alwaysOn      = true;

% precoders
precoderTechnology = parameters.precoders.Technology();
precoderTechnology.setTechPrecoder(
    parameters.setting.NetworkElementTechnology.NRMN_5G,
    parameters.precoders.LteDL);

% First Base Station
BS1 = parameters.basestation.PredefinedPositions;
BS1.positions = [-191.968; 70.4545]; % [x;y] coordinate in the OSM
    city
BS1.nSectors    = 3;
BS1.antenna      = antenna5G;
BS1.precoder.DL  = precoderTechnology;
params.baseStationParameters('BS1') = BS1;
% Second Base Station
BS2 = parameters.basestation.PredefinedPositions;
BS2.positions = [147.677; -30.2005]; % [x;y] coordinate in the OSM
    city
BS2.nSectors    = 3;
BS2.antenna      = antenna5G;
BS2.precoder.DL  = precoderTechnology;
params.baseStationParameters('BS2') = BS2;

%% Users
% Video users
Users_Video = parameters.user.Poisson2D;
Users_Video.nElements          = 37;
Users_Video.nRX                = 1;
Users_Video.channelModel        =
    parameters.setting.ChannelModel.PedB;
Users_Video.trafficModel.type   =
    parameters.setting.TrafficModelType.Video;
Users_Video.trafficModel.delayConstraint = 100;
Users_Video.numerology          = 1;

```

```

Users_Video.technology          =
    parameters.setting.NetworkElementTechnology.NRMN_5G;
params.userParameters('Video') = Users_Video;

% VoIP users
Users_voip = parameters.user.Poisson2D;
Users_voip.nElements          = 37;
Users_voip.nRX                 = 1;
Users_voip.channelModel        =
    parameters.setting.ChannelModel.PedB;
Users_voip.trafficModel.type    =
    parameters.setting.TrafficModelType.VoIP;
Users_voip.trafficModel.delayConstraint = 40;
Users_voip.numerology           = 1;
Users_voip.technology           =
    parameters.setting.NetworkElementTechnology.NRMN_5G;
params.userParameters('VoIP') = Users_voip;

% Gaming users
Users_game = parameters.user.Poisson2D;
Users_game.nElements          = 37;
Users_game.nRX                 = 1;
Users_game.channelModel        =
    parameters.setting.ChannelModel.PedB;
Users_game.trafficModel.type    =
    parameters.setting.TrafficModelType.Gaming;
Users_game.trafficModel.delayConstraint = 60;
Users_game.numerology           = 1;
Users_game.technology           =
    parameters.setting.NetworkElementTechnology.NRMN_5G;
params.userParameters('Gaming') = Users_game;

% Constant rate users
Users_CR = parameters.user.Poisson2D;
Users_CR.nElements          = 22;
Users_CR.nRX                 = 1;
Users_CR.channelModel        = parameters.setting.ChannelModel.PedB;
Users_CR.trafficModel.type    =
    parameters.setting.TrafficModelType.ConstantRate;
Users_CR.numerology           = 1;
Users_CR.technology           =
    parameters.setting.NetworkElementTechnology.NRMN_5G;
params.userParameters('ConstantRate') = Users_CR;

% Full buffer users
Users_FB = parameters.user.Poisson2D;
Users_FB.nElements          = 15;

```

```
Users_FB.nRX                      = 1;
Users_FB.channelModel              = parameters.setting.ChannelModel.PedB;
Users_FB.trafficModel.type        =
    parameters.setting.TrafficModelType.FullBuffer;
Users_FB.trafficModel.numSlots    = 0;
Users_FB.numerology                = 1;
Users_FB.technology                =
    parameters.setting.NetworkElementTechnology.NRMN_5G;
params.userParameters('FullBuffer') = Users_FB;
end
```

Listing D.1 – Código para o escalonador QAS