**UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ**

**RAMON DAMIAN AREVALOS VILLALBA**

# MODELO PARA GARANTIR O ATRASO MÁXIMO PARA CONEXÕES VOIP EM UMA REDE SDN ODL

**PONTA GROSSA**

**2023**

**RAMON DAMIAN AREVALOS VILLALBA**

# MODELO PARA GARANTIR O ATRASO MÁXIMO PARA CONEXÕES VOIP EM UMA REDE SDN ODL

## Model to Ensure the Maximum Delay for VoIP Connections in an ODL SDN Network

Dissertação apresentada como requisito para obtenção do título de Mestre em Ciência da Computação do Departamento Acadêmico de Informática da Universidade Tecnológica Federal do Paraná (UTFPR).

Orientador: Prof. Dr. Augusto Foronda

**PONTA GROSSA**

**2023**

**Ministério da Educação**
**Universidade Tecnológica Federal do Paraná**
**Campus Ponta Grossa**

RAMON DAMIAN AREVALOS VILLALBA

# MODELO PARA GARANTIR O ATRASO MÁXIMO PARA CONEXÕES VOIP EM UMA REDE SDN ODL

Trabalho de pesquisa de mestrado apresentado como requisito para obtenção do título de Mestre Em Ciência Da Computação da Universidade Tecnológica Federal do Paraná (UTFPR). Área de concentração: Sistemas E Métodos De Computação.

Data de aprovação: 21 de Junho de 2023

Dr. Augusto Foronda, Doutorado - Universidade Tecnológica Federal do Paraná

Dr. Lourival Aparecido De Gois, Doutorado - Universidade Tecnológica Federal do Paraná

Dr. Mauricio Zadra Pacheco, Doutorado - Universidade Estadual de Ponta Grossa (Uepg)

Dedico este trabalho à minha família.

## AGRADECIMENTOS

Este trabalho de mestrado foi uma longa caminhada, acompanhado de inúmeros desafios, incertezas, alegrias e percalços encontrados pelo caminho, mas nada seria possível sem o apoio e contribuição de várias pessoas a quem deixo meu imenso agradecimento.

Agradeço, antes de tudo, a Deus por nunca me abandonar.

Ao meu orientador Prof. Dr. Augusto Foronda, pela atenção, paciência, sabedoria e segurança para me guiar neste mundo de descobertas.

A minha esposa Cibeli May pelo amor, apoio, compreensão e incentivo de sempre.

Aos meus filhos, Victor Rafael e Henry Lionel por serem o motor da minha vida e pelo amor incondicional.

Ao meu querido pai Victor e minha mãe Marisa pelo apoio e pelas orações.

Aos meus irmãos Victor, Ernesto e Ana pelo carinho.

Aos meus colegas do PPGCC, em especial a Tathi e Raimundo.

Ao meu amigo Eng. Eduardo Sallum por ser um dos incentivadores desta caminhada.

À UTFPR e ao PPGCC.

À Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, processo 88887.630451/2021-00, pelo apoio financeiro provido.

## RESUMO

As redes tradicionais trabalham de forma distribuída, onde cada elemento tem um plano de controle. Possuem desvantagens como: a tomada de decisão é distribuída por vários componentes de rede e equipamentos de diferentes fabricantes tem cada um a sua interface de configuração aumentando a complexidade e o tempo de configuração dos equipamentos. Para solucionar estes problemas, surgiram as redes SDN (*Software Defined Networking*), onde o plano de dados, responsável pelo encaminhamento de dados é separado do plano de controle, responsável pela configuração e gerenciamento dos elementos da rede, sendo representado por um controlador. Para a comunicação entre o controlador e os componentes da rede, o protocolo *OpenFlow* é usado. Assim como nas redes tradicionais, as redes SDN precisam prover QoS (*Quality of Service)* para os usuários, que é um mecanismo para melhorar o desempenho em uma rede de comunicação e administrar os recursos da rede com mais eficiência. Uma rede SDN pode prover QoS através de duas ferramentas que foram implementadas até o momento do desenvolvimento deste trabalho: medidor e fila. Medidor é um mecanismo usado na entrada da rede para limitar a taxa de dados e fila é usado na saída da rede onde podem ser configuradas várias filas para separar o tráfego. Como ainda não se tem um modelo definido usando estes dois mecanismos para prover QoS em uma rede com diferentes tráfegos, neste trabalho foi desenvolvido um modelo para determinar o tamanho da fila em função do número de usuários e do atraso máximo para conexões VoIP (*Voice Over Internet Protocol*) em uma rede SDN. A proposta foi desenvolvida através de regressão linear simples e múltiplas e os resultados com a equação desenvolvida foram comparados com simulações no ambiente de simulação mininet com controlador ODL (*OpenDaylight*). Os resultados mostram que o tamanho da fila é diretamente proporcional ao número de conexões VoIP e os valores teóricos se aproximam dos valores obtidos através da simulação.



**Palavras-chave:** SDN; *OpenFlow*; QoS; atraso; filas; VoIP.

## ABSTRACT

Traditional networks work in a distributed manner, where each element has a control plane. They have disadvantages such as: decision-making is distributed across several network components and equipment from different manufacturers each has its own configuration interface, increasing the complexity and time of equipment configuration. To solve these problems, SDN networks emerged, where the data plane, responsible for forwarding data, is separated from the control plane, responsible for configuring and managing the network elements, being represented by a controller. For communication between the controller and network components, the OpenFlow protocol is used. As with traditional networks, SDN networks need to provide QoS to users, which is a mechanism to improve performance in a communication network and manage network resources more efficiently. An SDN network can provide QoS through two tools that have been implemented until the moment of the development of this work: meter and queue. Meter is a mechanism used at the network entrance to limit the data rate and queue is used at the network exit where multiple queues can be configured to separate the traffic. As there is still no defined model using these two mechanisms to provide QoS in a network with different traffics, in this work a model was developed to determine the queue size as a function of the number of users and the maximum delay for VoIP connections over an SDN network. The proposal was developed through simple and multiple linear regression and the results with the developed equation were compared with simulations in the mininet simulation environment with ODL controller. The results show that the queue size is directly proportional to the number of VoIP connections and the theoretical values are close to the values obtained through the simulation.



**Keywords:** SDN; OpenFlow; QoS; delay; queue; VoIP.

## LISTA DE EQUAÇÕES

**Equação 1 – Regressão linear simples.....41**
**Equação 2 – Notação matricial para regressão linear simples .....42**
**Equação 3 – Regressão linear múltipla .....43**
**Equação 4 – Notação matricial para regressão linear múltipla.....44**
**Equação 5 – R-quadrado .....45**
**Equação 6 – Notação matricial para regressão linear simples .....51**
**Equação 7 – Método dos mínimos quadrados como dados do modelo .....51**
**Equação 8 – Reta melhor ajuste entre atraso máximo e tamanho da fila .....51**
**Equação 9 – Método dos mínimos quadrados como dados do modelo .....52**
**Equação 10 – Reta melhor ajuste entre atraso máximo e número de usuários 52**
**Equação 11 – Notação matricial para regressão linear múltipla.....54**
**Equação 12 – Método dos mínimos quadrados como dados do modelo .....54**
**Equação 13 – Reta melhor ajuste do atraso máximo em função do tamanho da fila e número de usuários.....54**
**Equação 14 – Reta melhor ajuste do tamanho da fila em função do número de usuários e atraso máximo .....56**

## LISTA DE FIGURAS

**Figura 1 – Rede Tradicional....18**
**Figura 2 – Rede SDN....19**
**Figura 3 – Arquitetura SDN....20**
**Figura 4 – Arquitetura *OpenFlow*....22**
**Figura 5 – Tipo de atraso em uma rede....28**
**Figura 6 – Ferramenta Medidor e Fila....28**
**Figura 7 – Esquema do banco de dados OVSDB....29**
**Figura 8 – Exemplo de classes HTB....30**
**Figura 9 – (a) Um balde furado com água. (b) Um balde furado com pacotes...33**
**Figura 10 – Funcionamento de uma aplicação VoIP....33**
**Figura 11 – Interpretação geométrica dos parâmetros $\beta_0$ e $\beta_1$....42**
**Figura 12 – Ilustração da regressão linear simples....43**
**Figura 13 – Hiperplano no espaço p-dimensional das variáveis $x_p$....44**
**Figura 14 – Interpretação da fila em um switch....46**
**Figura 15 – Comportamento das filas....47**
**Figura 16 – Criação da fila e definir o tamanho....48**
**Figura 17 – Fluxograma do modelo de tráfego....49**
**Figura 18 – Equação de regressão linear simples....50**
**Figura 19 – Equação de regressão linear múltipla....53**
**Figura 20 – Topologia 1....59**
**Figura 21 – Topologia 2....59**

## LISTA DE GRÁFICOS

**Gráfico 1 – Atraso máximo pelo tamanho da fila ..... 52**
**Gráfico 2 – Atraso máximo pelo número de usuários ..... 53**
**Gráfico 3 – Atraso máximo em função do número de usuários e do tamanho da fila ..... 55**
**Gráfico 4 – Simulação VoIP com uma única fila FIFO ..... 60**
**Gráfico 5 – Simulação VoIP com até 10 switches ..... 61**
**Gráfico 6 – Simulação VoIP com fila de 9Mbps ..... 63**
**Gráfico 7 – Simulação VoIP com fila de 3Mbps até 10Mbps ..... 64**
**Gráfico 8 – Simulação VoIP com 20 usuários ..... 65**
**Gráfico 9 – Tamanho da fila em função do atraso e número de usuários ..... 66**

## LISTA DE QUADROS

**Quadro 1 – Componentes de um fluxo .....23**
**Quadro 2 – Características dos *codecs*.....35**
**Quadro 3 – Comparação entre trabalhos relacionados e a proposta.....40**
**Quadro 4 – Características do ambiente de simulação .....57**

## LISTA DE TABELAS

**Tabela 1 – Classificação MOS .....37**
**Tabela 2 – Dados das 3 variáveis .....50**
**Tabela 3 – Dados para verificar as equações.....62**
**Tabela 4 – Erro percentual entre atraso máximo e número de usuários .....63**
**Tabela 5 – Erro percentual entre atraso máximo e tamanho da fila .....65**
**Tabela 6 – Erro percentual entre tamanho da fila, atraso máximo e número de usuários .....67**
**Tabela 7 – Previsão de tamanho do enlace em uma rede .....68**

## LISTA DE ABREVIATURAS E SIGLAS

| | |
|---|---|
| ACELP | *Algebraic Code Excited Linear Prediction* |
| ADPCM | *Adaptive Differential Pulse Code Modulation* |
| AEV | *Environment Virtual Experimentation* |
| API | *Application programming interface* |
| AQM | *Active Queue Management* |
| CBBPM | *Collaborative Borrowing Based Packet-Marking* |
| DSCP | *Differentiated Services Codepoint* |
| FIFO | *First in First out* |
| FQ | *Fair Queueing* |
| HFSC | *Hierarchical Fair-Service Curve* |
| HTB | *Hierarchical Token Bucket* |
| ICMP | *Internet Control Message Protocol* |
| IP | *Internet Protocol* |
| ITU-T | *International Telecommunication Union-Telecommunication* |
| LLDP | *Link Layer Discovery Protocol* |
| MAC | *Media Access Control* |
| MOS | *Mean Opinion Score* |
| MPC-MLQ | *Multi Pulse-Maximum Likelihood Quantization* |
| MPLS | *Multi-Protocol Lavel Switching* |
| ODL | *OpenDaylight* |
| OF | *OpenFlow* |
| ONF | *Open Networking Foundation* |
| ONOS | *Open Network Operating System* |
| OVS | *Open vSwitch* |
| OVSDB | *Open vSwitch Database Management Protocol* |
| PCM | *Pulse Code Modulation* |
| PPS | *Packet per second* |
| PSNR | *Peak Signal-to-Noise Ratio* |
| PHB | *Per-Hop Behavior* |
| QoE | *Quality of Experience* |
| QoS | *Quality of Service* |
| RED | *Random Early Detection* |

| | |
|---|---|
| RTP | *Real-time Transport Protocol* |
| RTCP | *Real-time Transport Control Protocol* |
| RTT | *Round Trip Time* |
| RSVP | *Resource ReSerVation Protocol* |
| SDN | *Software Defined Networking* |
| SFQ | *Stochastic Fairness Queuing* |
| SSIM | *Structural Similarity* |
| TCP | *Transmission Control Protocol* |
| UDP | *User Datagram Protocol* |
| UFPE | Universidade Federal de Pernambuco |
| VLAN | *Virtual Local Area Network* |
| VMN | *Virtual Machine Network* |
| VoIP | *Voice Over Internet Protocol* |
| VQM | *Video Quality Metric* |

**SUMÁRIO**


**1 INTRODUÇÃO. . . . . 15**
**1.1 Objetivos. . . . . 16**
1.1.1 Objetivo Geral . . . . . 16
1.1.2 Objetivos Específicos. . . . . 16
**1.2 Estrutura da dissertação . . . . . 17**
**2 REFERENCIAL TEÓRICO . . . . . 18**
**2.1 Rede Definida por *Software* . . . . . 18**
**2.2 Arquitetura SDN . . . . . 20**
**2.3 Controlador ODL . . . . . 21**
**2.4 *OpenFlow* . . . . . 22**
2.4.1 Versões *OpenFlow*. . . . . 23
**2.5 OVSDB . . . . . 24**
**2.6 QoS – IntServ – DiffServ . . . . . 25**
**2.7 Métricas para Medição de QoS. . . . . 26**
**2.8 QoS em uma Rede SDN. . . . . 28**
2.8.1 Filas . . . . . 29
2.8.2 Medidor . . . . . 31
**2.9 VoIP . . . . . 33**
2.9.1 RTP e RTCP . . . . . 34
2.9.2 *Codecs* de voz . . . . . 34
2.9.3 QoS para VoIP . . . . . 36
**2.10 Trabalhos Relacionados . . . . . 37**
**3 REGRESSÃO LINEAR . . . . . 39**
**3.1 Regressão Linear Simples . . . . . 41**
**3.2 Regressão Linear Múltipla . . . . . 43**
**3.3 R-quadrado. . . . . 45**
**4 DESENVOLVIMENTO DO MODELO . . . . . 46**
**4.1 Tráfegos . . . . . 46**
**4.2 Criação das Filas . . . . . 47**
**4.3 Fluxograma. . . . . 48**
**4.4 Cálculo do Tamanho da Fila . . . . . 49**
**5 RESULTADOS EXPERIMENTAIS . . . . . 57**
**5.1 Descrição do Ambiente de Simulação . . . . . 57**
**5.2 Topologia . . . . . 58**
**5.3 Simulações e Resultados . . . . . 60**
5.3.1 Simulação para mostrar o problema. . . . . 60
5.3.2 Simulação com número diferente de switches . . . . . 61
5.3.3 Simulação que relaciona atraso máximo com número de usuários . . . . 62
5.3.4 Previsão da fila em função do atraso e número de usuários . . . . . 66
**6 CONCLUSÃO . . . . . 69**
**6.1 Trabalhos futuros. . . . . 70**
**REFERÊNCIAS . . . . . 71**

# 1 INTRODUÇÃO

As redes tradicionais trabalham de forma distribuída, onde cada elemento da rede tem um plano de controle e usam dispositivos de *hardware* como roteadores e *switches* para controlar o tráfego de rede. Elas possuem desvantagens como: a tomada de decisão é distribuída por vários componentes de rede, o que faz com que seja muito trabalhoso adicionar quaisquer novos dispositivos de rede, e isso torna o gerenciamento e a configuração da rede extremamente trabalhosa e sujeita a erros o que não atende mais a exigência das redes atuais que possuem sistemas cada vez mais complexos (LI, 2016). Equipamentos de diferentes fabricantes tem cada um a sua interface de configuração, o que aumenta a complexidade e o tempo de configuração dos equipamentos.

Para solucionar estes problemas surgiram as redes *Software Defined Networking* (SDN) onde o plano de dados é separado do plano de controle. O plano de dados é formado pelos equipamentos de rede e é responsável pelo encaminhamento dos dados. O plano de controle é retirado dos equipamentos de rede e fica em um elemento externo chamado controlador, que é responsável pela configuração e gerenciamento dos elementos da rede e possui vantagens como: infraestrutura de rede personalizável, maior controle porque ao se programar apenas um controlador baseado em *software* de padrão aberto é possível controlar o fluxo de tráfego, a segurança é reforçada, pois é possível ter uma visão de toda a rede, sendo assim, é possível ter uma visão mais global relacionada as ameaças da rede; é mais flexível, os administradores podem controlar a rede, alterar as definições de configuração, aprovisionar recursos e aumentar a capacidade da rede (HU, 2014).

A arquitetura SDN é formada por três camadas: camada de aplicação, camada de controle e camada de infraestrutura. Na camada de aplicação se encontra os aplicativos utilizados numa rede e que interagem com o controlador. Já na camada de controle está o software que controla a rede SDN e finalmente a camada de infraestrutura representa todos os dispositivos físicos da rede como roteadores, pontos de acesso e *switches*.

Para a comunicação entre o controlador e os componentes da rede usa-se o protocolo *OpenFlow,* que permite definir fluxos de dados por meio de *software* e projetar regras para orientar os *switches* sobre como direcionar o tráfego de rede (LI, 2016). O protocolo *OpenFlow* não faz o gerenciamento do *switch*, que deve ser feito

por outro protocolo como, por exemplo, *Open vSwitch Database Management Protocol* (OVSDB) ou *OpenFlow Configuration* (OF-Config). As operações de *OpenvSwitch* (OVS)*,* como criação de interfaces, definição de políticas de *Quality of Service* (QoS) ou desligamento de uma porta física, são gerenciados pelo OVSDB (HKRISHNA, 2016).

Assim como nas redes tradicionais, as redes SDN precisam prover QoS para os usuários. QoS é um mecanismo para melhorar o desempenho em uma rede de comunicação e administrar os recursos da rede com mais eficiência. Uma rede SDN pode prover QoS através de duas ferramentas: medidor e fila. Medidor é um mecanismo usado na entrada da rede para limitar a taxa de dados e fila é usado na saída da rede onde podem ser configuradas várias filas para separar o tráfego (BOLEY, 2016). O medidor é implementado através do algoritmo do balde furado, que tem a principal função de limitar o tráfego que entra na rede. Existem vários algoritmos de enfileiramento, neste trabalho será utilizado o algoritmo *Hierarchical Token Bucket (HTB),* pois é um dos mais utilizado na literatura para controle de fluxos em redes *SDN* (B. HUBERT, 2022).

A arquitetura SDN não define como usar o medidor e a fila para prover QoS em uma rede com diferentes tráfegos. Então, o objetivo desde trabalho é propor um modelo para determinar o tamanho da fila em função do número de usuários e do atraso máximo para conexões VoIP em rede SDN. A proposta será desenvolvida através de regressão linear simples e múltiplas para encontrar uma equação que relaciona estas 3 variáveis e comparar com os dados obtidos no ambiente de simulação mininet com controlador *OpenDaylight* (ODL).

## 1.1 Objetivos

### 1.1.1 Objetivo Geral

Desenvolver um modelo para garantir atraso máximo para conexões VoIP em uma rede SDN ODL.

### 1.1.2 Objetivos Específicos

- Analisar a relação entre número de usuários, número de switches, tamanho da fila e atraso com o uso de filas;
- Desenvolver uma equação para o cálculo do tamanho da fila;
- Simular tráfegos VoIP para diferentes quantidades de usuários e diferentes tamanhos de fila;
- Comparar os dados obtidos através do modelo teórico com os dados obtidos através das simulações.

## 1.2 Estrutura da Dissertação

O presente trabalho está organizado em seis capítulos. O capítulo 1 é referente a contextualização do tema, assim como os objetivos e organização do trabalho. O capítulo 2 apresenta a fundamentação teórica do trabalho de modo a explicar conceitos da rede SDN, QoS, VoIP e os trabalhos relacionados. O capítulo 3 é referente aos conceitos e equações da regressão linear simples e múltipla. No capítulo 4 é apresentado o modelo com a criação da fila, fluxograma e as equações obtidas para o cálculo da fila. O capítulo 5 descreve os resultados experimentais da simulação para avaliar o modelo. E o capítulo 6 apresenta as conclusões e trabalhos futuros.

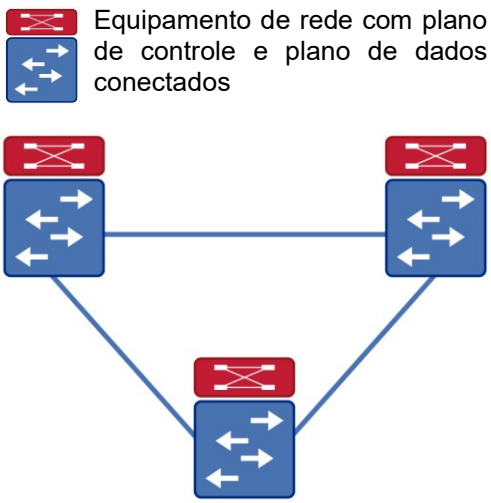
# 2 REFERENCIAL TEÓRICO

Este capítulo aborda os conceitos fundamentais para a compreensão do tema do presente trabalho. A seção 2.1 descreve a diferença de uma rede tradicional para uma rede SDN. A seção 2.2 apresenta a arquitetura de uma rede SDN. A seção 2.3 descreve o *OpenFlow* e suas versões. A seção 2.4 explica o OVSDB. A seção 2.5 define o controlador ODL. A seção 2.6 descreve QoS. A seção 2.7 explica as métricas para medição de QoS. A seção 2.8 trata sobre QoS em uma rede SDN. E a seção 2.10 aborda os trabalhos relacionados ao tema.

## 2.1 Rede Definida por *Software*

As redes tradicionais trabalham de forma distribuída, onde cada elemento da rede tem um plano de controle e um plano de dados. O plano de controle é responsável por algumas tarefas como, por exemplo, configuração do equipamento e definição do caminho para o tráfego dos dados. O plano de dados é responsável pelo encaminhamento dos dados. Este modelo de rede tradicional pode ser visto na Figura 1.

**Figura 1 – Rede Tradicional**



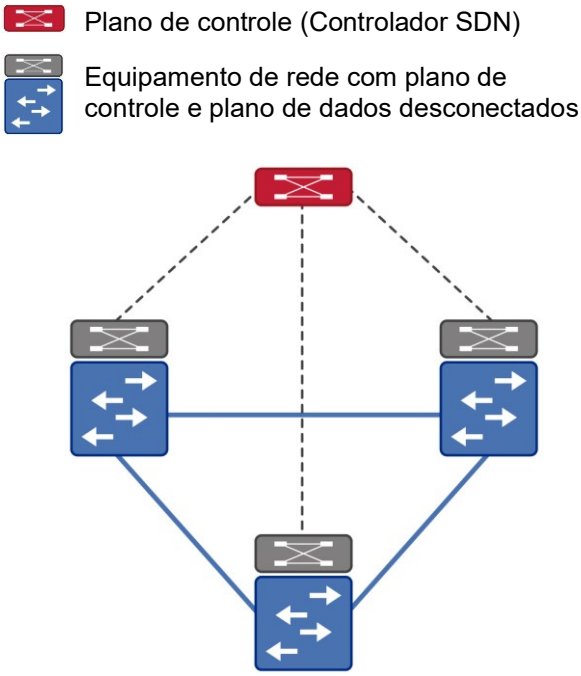

**Fonte: Adaptado de KRISHNA (2016)**

As redes tradicionais usam dispositivos de *hardware* como roteadores e *switches* para controlar o tráfego de rede. Pode-se citar como desvantagem da rede tradicional:

- A tomada de decisão é distribuída por vários componentes de rede;
- Adicionar quaisquer novos dispositivos de rede, isso torna o gerenciamento e a configuração da rede extremamente trabalhosa e sujeita a erros (LI, 2016);
- Os equipamentos de diferentes fabricantes têm cada um à sua interface de configuração, o que aumenta a complexidade e o tempo de configuração dos equipamentos.

A rede SDN usa controladores baseados em *software* ou interfaces de programação de aplicativos (API) e se comunica com a infraestrutura de *hardware*. Desta forma, equipamentos como roteadores e *switches* podem ser controlados por *software*. Nas redes SDN, o plano de dados é separado do plano de controle, conforme pode ser visto na Figura 2. O plano de dados é formado pelos equipamentos de rede e é responsável pelo encaminhamento dos dados. O plano de controle é retirado dos equipamentos de rede e fica em um elemento externo chamado controlador, que é responsável pela configuração e gerenciamento dos elementos da rede (HU, 2014).

**Figura 2 – Rede SDN**




**Fonte: Adaptado de KRISHNA (2016)**

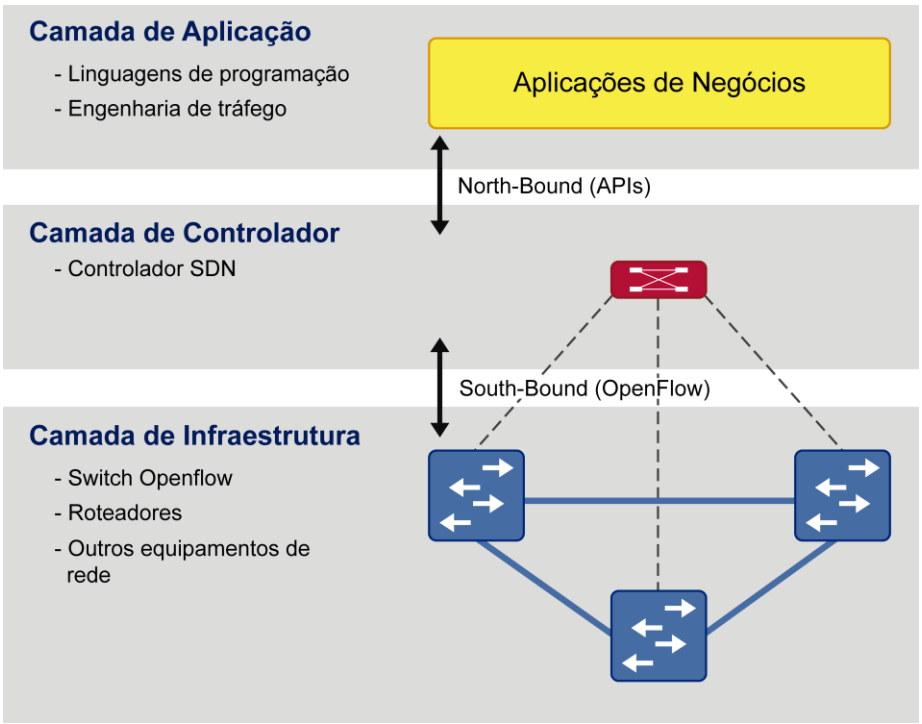
Em relação a rede tradicional, a rede SDN possui vários benefícios:

- Infraestrutura de rede personalizável e os administradores podem otimizar o fluxo de dados na rede;
- Possui maior controle porque ao se programar apenas um controlador baseado em *software* de padrão aberto é possível controlar o fluxo de tráfego;
- A segurança é reforçada, pois é possível ter uma visão mais global relacionada as ameaças da rede;
- A rede SDN é mais flexível, os administradores podem controlar a rede, alterar as definições de configuração, provisionar recursos e aumentar a capacidade da rede.

## 2.2 Arquitetura SDN

A arquitetura SDN representada na Figura 3 é formada por três camadas: camada de aplicação, camada de controle e camada de infraestrutura.

**Figura 3 – Arquitetura SDN**




**Fonte: Adaptado de ONF (2022)**

- Camada de aplicação: esta consiste em um ou mais aplicativos de rede, que interagem com o controlador para utilizar a visão global de toda infraestrutura da rede para seus processos internos de tomada de decisão e assim fazer o encaminhamento de dados nos *switches* e roteadores. A comunicação com os controladores ocorre por meio da API *north-bound* (KARAKUS, 2017);
- Camada de controle: tem a função de controlar e supervisionar o comportamento de encaminhamento de rede por meio da API *south-bound* e é composto por um controlador SDN baseado em *software*, que é centralizado, o que permite que as funções da rede sejam programáveis de forma rápida e fácil (KARAKUS, 2017);
- Camada de Infraestrutura: é um plano inferior que consiste em uma rede de dispositivos como roteadores, pontos de acesso, *switches*, entre outros. Estes elementos de rede representam o plano de dados e fazem o encaminhamento de pacotes de acordo com as instruções do plano de controle (KARAKUS, 2017).

## 2.3 Controlador ODL

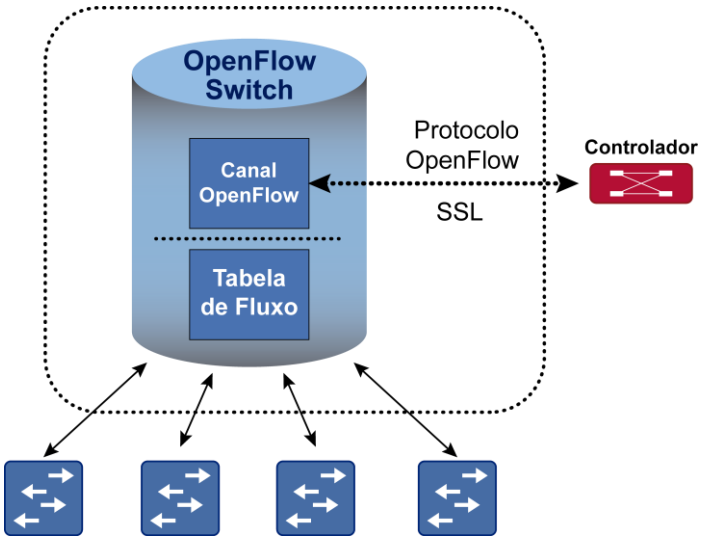
O ODL é um *software* implementado para ser utilizado em redes SDN, desenvolvida pelo consórcio Linux *Foundation*. Tem o objetivo de atender desde ambientes de testes até redes de produção. Possui uma ampla variedade de protocolos de controle suportados, entre eles *OpenFlow*, fornecendo diversidade de serviços de rede e permitindo a fácil inclusão de novas funcionalidades, na forma de serviços de rede, *plug-ins* e aplicações. Ele pode ser executado em qualquer Sistema Operacional, é uma ferramenta de código aberto que é financiada por diversas empresas de tecnologia, tais como: Cisco e Intel (OPENDAYLIGHT, 2021).

Para a comunicação entre o controlador e os componentes da rede usa-se o protocolo *OpenFlow* (ANDRIOLI, 2017), no entanto, há outros protocolos de comunicação, como por exemplo o OpFlex. O protocolo *OpenFlow* permite definir fluxos de dados por meio de *software* e projetar regras para orientar os *switches* sobre como direcionar o tráfego de rede (LI, 2016).

## 2.4 *OpenFlow*

O protocolo *OpenFlow* demonstrado na Figura 4 surgiu na Universidade de Stanford em 2008 e é atualmente gerenciado pela *Open Networking Foundation* (ONF) (KREUTZ, 2014).

**Figura 4 – Arquitetura *OpenFlow***




**Fonte: Adaptado de MCKEOWN (2008)**

O *OpenFlow* foi proposto para padronizar a forma como um controlador se comunica com dispositivos de rede em uma arquitetura SDN. Ele fornece especificações para migrar a lógica de controle de um *switch* para o controlador e também define um protocolo para comunicação entre o controlador e os *switches* (LARA, 2014). Os *switches OpenFlow* dependem da configuração dos controladores, pois um controlador SDN instala entradas de tabela de fluxo em *switches*, para que estes possam encaminhar o tráfego de acordo com essas entradas (BRAUN, 2014).

Por meio do protocolo *OpenFlow*, o controlador pode manipular (adicionar, atualizar e excluir) entradas nas tabelas de fluxo, reativamente ou proativamente (ONF, 2015). Ao chegar um novo fluxo, o *switch* verifica a tabela com esses dados, caso nos registros não encontre uma entrada correspondente a este fluxo, ele envia o pacote para o controlador por meio de uma mensagem "*packet_in*". Por meio de sua aplicação, o controlador tem que decidir o que fazer com o fluxo (encaminhar, descartar, enviar de volta) e o controlador passa essa informação ao *switch* instalando um novo fluxo em sua tabela (CUNHA, 2017).

Os campos de uma entrada de fluxo são mostrados no Quadro 1. Uma tabela de fluxo é composta de entradas de fluxos.

**Quadro 1 – Componentes de um fluxo**

| *Match Fields* | *Priority* | *Counters* | *Instructions* | *Timeouts* | *Cookie* | *Flags* |
|---|---|---|---|---|---|---|

**Fonte: Adaptado de ONF (2015)**

- *Match fields*: campos para corresponder com os pacotes;
- *Priority*: prioridade de correspondência de fluxo;
- Contadores: serão atualizados quando os pacotes são correspondidos;
- Instruções: é usado para modificar um conjunto de ações ou processamento de *pipeline;*
- *Timeouts*: quantidade de precedência máxima de tempo ou tempo ocioso antes que o fluxo seja expirado pelo *switch*;
- *Cookie*: valor escolhido pelo controlador. Pode servir para filtrar entradas de fluxo afetadas por requisições de estatísticas, modificação e de exclusão de fluxos. No processamento dos pacotes não é usado;
- *Flags*: modificam a forma como as entradas de fluxo são gerenciadas.

Os campos de correspondência e prioridade identificam uma entrada na tabela de fluxo, que identificam uma entrada única em uma tabela específica (ONF, 2015).

Conforme o *OpenFlow* foi evoluindo, diversas versões de sua especificação surgiram (BRAUN, 2014).

### 2.4.1 Versões *OpenFlow*

Versão *OpenFlow* 1.0: existe o enfileiramento que encaminha pacotes através de uma fila conectada a uma porta. Um *Switch OpenFlow* pode ter uma ou mais filas, dependendo de suas portas. O controlador SDN pode consultar informações sobre as filas de um *switch*. O comportamento da fila pode ser configurado através do OF-CONFIG, fora do escopo do *OpenFlow* (KARAKUS, 2017).

Versão *OpenFlow* 1.1: realiza correspondência e marcação de rótulos VLAN (*Virtual Local Area Network*) e MPLS (*Multi Protocol Lavel Switching*) e classes de tráfego. O novo suporte possibilita que se adicione, modifique e remova tags VLAN (KARAKUS, 2017).

Versão *OpenFlow* 1.2: possui a capacidade do controlador SDN consultar todas as filas em um *switch* e especificar uma taxa máxima para a fila. É uma versão que específica que as filas podem ser anexadas às portas e ser usado para mapear fluxos nelas (KARAKUS, 2017).

Versão *OpenFlow* 1.3: foi introduzida a funcionalidade de limitação de taxa de entrada por meio de um medidor, que tem alguns componentes: identificador, bandas e contadores. Os contadores ajudam o controlador na coleta de estatísticas da rede. Os medidores podem associar um pacote a uma porta para implementar estruturas complexas de QoS. Ou seja, é possível monitorar a taxa de entrada de tráfego, conforme definido por uma regra de fluxo (KARAKUS, 2017).

Versão *OpenFlow* 1.4: permite um controlador monitorar em tempo real as alterações feitas por outros controladores em qualquer subconjunto das tabelas de fluxo. Para que isso seja possível, um controlador define um número de monitores que seleciona um subconjunto das tabelas de fluxo. Assim, ao se adicionar, modificar ou remover em um dos subconjuntos definidos por um monitor de fluxo, é enviado para o controlador um evento informando-o do ocorrido (KARAKUS, 2017).

Versão *OpenFlow* 1.5: substitui a instrução do medidor, das versões anteriores, assim vários medidores podem ser conectados a uma entrada de fluxo e pode-se usar os medidores em grupos (KARAKUS, 2017).

Versão *OpenFlow* 1.6: foi disponibilizada desde setembro de 2016, porém somente membros da ONF (*Open Networking Foundation*) podem acessá-la (ONF, 2015).

A versão *OpenFlow* utilizada neste trabalho é o *OpenFlow* 1.3 pois atende o objetivo do modelo em criar filas para mapear os fluxos e também possui contadores que ajudam na coleta de estatísticas da rede.

O protocolo *OpenFlow* é usado para a comunicação entre o controlador e o *switch*, mas ele não faz o gerenciamento do *switch*, que deve ser feito por outro protocolo complementar como, por exemplo, OF-Config (KREUTZ, 2014) ou OVSDB que é o protocolo utilizado neste modelo.

## 2.5 OVSDB

OVSDB é projetado para fornecer recursos de gerenciamento avançados para *Open vSwitches*, como por exemplo:

- Definir políticas de QoS em interfaces;
- Gerenciar filas e coletar estatísticas.

Além dos recursos do *OpenFlow* para configurar o comportamento dos fluxos em um dispositivo de encaminhamento, um *Open vSwitch* oferece outras funções de rede, como por exemplo, permite que os elementos de controle criem várias instâncias de *switch* virtual, define políticas de QoS em interfaces, conecta interfaces aos *Open vSwitch*, configura interfaces de túnel em caminhos de dados *OpenFlow*, gerencia filas e coleta estatísticas (KREUTZ, D., 2014).

As operações de *Open vSwitch,* como criação de interfaces, definição de políticas de QoS ou desligamento de uma porta física, são gerenciados pelo OVSDB. O OVSDB é formado por um servidor de banco de dados e por um *daemon vswitch*, que monitora o banco de dados para adições, exclusões e modificações nessas informações. O servidor OVSDB se comunica por meio do protocolo de gerenciamento OVSDB e armazena informações sobre *Open vSwitch* na forma de banco de dados (HKRISHNA, 2016).

O *OpenFlow* e o OVSDB fornecem mecanismos para prover políticas QoS que vão ser explicados a seguir.

## 2.6 QoS – IntServ – DiffServ

QoS melhora o desempenho em uma rede de comunicação e administra os recursos da rede com mais eficiência. QoS é implementado em duas abordagens: rígida e flexível. O *Integrated Services* (IntServ) é o método rígido que garante os requisitos de QoS de conexões individuais**.** Já o *Differentiated Services* (DiffServ) é o método flexível que garante os requisitos de QoS de conexões de vários fluxos juntos (KARAKUS, 2017).

O IntServ foi a primeira tentativa de estabelecer o controle de QoS na rede IP. IntServ emula o conceito de alocação de recursos de comutação de circuito. Os elementos da rede são forçados a alocar recursos para um fluxo de tráfego específico, de forma análoga a uma sessão de chamada de comutação de circuito (HKRISHNA, 2016).

IntServ usa o protocolo de reserva de recursos *Resource ReSerVation Protocol* (RSVP) para notificar/solicitar a cada dispositivo de rede ao longo do caminho de um fluxo para reservar a quantidade específica de recursos para o fluxo. A transmissão

só começa se cada dispositivo ao longo do caminho puder reservar a largura de banda necessária. Apesar do IntServ ser uma forma eficaz de fornecer uma garantia de QoS, ele possui algumas desvantagens: todos os roteadores ao longo do tráfego devem oferecer suporte a IntServ, cada roteador em um determinado caminho de rede precisará armazenar todos os possíveis estados e informações das diferentes conexões. Por não ser escalonável, IntServ nunca foi implantado na Internet, mas tem sido bastante usado em redes locais e pequenas empresas. Como vantagem, o IntServ fornece uma garantia rígida e pode ser usado para fornecer diferenciação de classe de serviço (BRADEN, 1994).

O modelo DiffServ foi introduzido para resolver o problema de escalabilidade do IntServ, pois enquanto o IntServ trata o tráfego de ponta a ponta, o DiffServ aplica políticas em cada salto, o que é chamada como comportamento *Per-Hop Behavior* (PHB). O DiffServ não trabalha por fluxo individual, ele funciona em fluxos agregados. Os pacotes podem ser classificados usando bits de acordo com o ponto de código de serviços diferenciados *Differential Services Codepoint* (DSCP) contidos no cabeçalho do pacote. Independendo do fluxo que fazem parte, todos os pacotes com mesmo cabeçalho recebem o mesmo tratamento. Quando os pacotes são classificados, os PHB correspondentes são aplicados àqueles pacotes. Por ser mais escalonável que o IntServ, o DiffServ é usado na Internet, para que seja dado prioridade a uma classe de fluxos em relação a outra (BLACK, 2015).

Neste trabalho é usado o método flexível DiffServ pois garante os requisitos de QoS para vários fluxos juntos que neste caso são as conexões VoIP.

## 2.7 Métricas para Medição de QoS

Existem algumas métricas para medir o desempenho de redes de comunicação. Entre elas, estão: vazão, atraso, *jitter* e perda de pacotes (NACIMENTO, 2004).

- Vazão é a média ou a quantidade de entrega de pacotes que é possível fazer ao final de um período de tempo;
- Atraso é o tempo entre o envio de uma mensagem de uma determinada origem e a recepção desta mensagem ao destino;
- *Jitter* é a variação no atraso de pacotes em um determinado (ALALAWI, 2015);

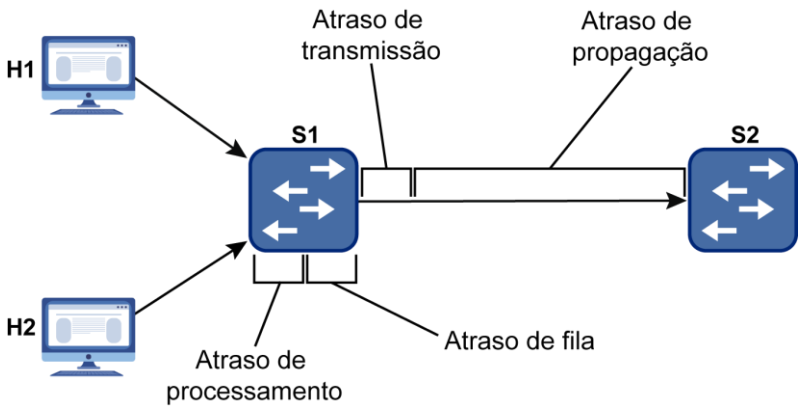
- Perda de pacotes é a porcentagem de pacotes enviados que não chegaram ao destino, devido ao congestionamento, problemas nos equipamentos de transmissão e atraso excessivo.

Atraso também chamado de atraso fim a fim é o tempo que o pacote leva para percorrer o caminho desde a sua origem até o destino. Ao longo da transmissão os pacotes podem sofrer vários tipos de atraso como processamento, fila, transmissão e propagação.

- Atraso de processamento: se refere ao tempo necessário para o equipamento examinar o cabeçalho do pacote e decidir para onde encaminhar o pacote. Outros fatores também podem influenciar como o tempo requerido para verificar se há erros que ocorreram eventualmente ao transmitir o pacote do equipamento a outro na rede. Estes atrasos estão na ordem de microssegundos ou menos.
- Atraso de fila: quando o pacote está na fila ele sofre um atraso de enfileiramento ao esperar ser enviado ao próximo nó da rede. Esse atraso depende do congestionamento na rede. Se a fila está vazia, o atraso é zero, porém, se muitos outros pacotes estiverem esperando para serem transmitidos, o atraso aumenta.
- Atraso de transmissão: depende da largura de banda (velocidade) do enlace. É o tempo necessário para transportar todos os bits de dentro do equipamento para o meio de transmissão e é da ordem de alguns micros a milisegundos.
- Atraso de propagação: quando um bit é enviado no enlace, ele irá se deslocar para o roteador seguinte. Este tempo gasto para se propagar do começo do link até o outro equipamento é o atraso de propagação. Este tempo depende do meio físico do link (fibra ótica, cabo coaxial, cabo par-trançado e outros) (KUROSE, 2013).

A Figura 5 apresenta os tipos de atraso em uma rede.

**Figura 5 – Tipos de atraso em uma rede**



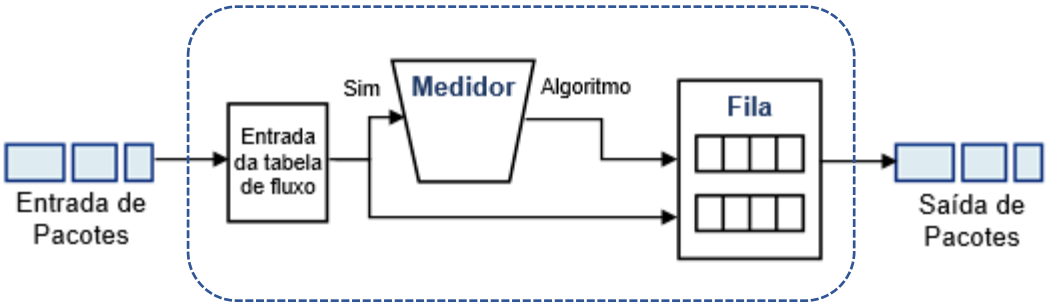

**Fonte: Adaptado de KUROSE (2013)**

## 2.8 QoS em uma Rede SDN

Uma rede SDN pode prover QoS através de duas ferramentas: Medidores (*Meters)* e Filas (*Queues)* como pode ser visto na Figura 6.

Medidor é um mecanismo usado na entrada da rede para limitar a taxa de dados e a fila é usado na saída da rede onde podem ser configuradas várias filas para separar o tráfego (BOLEY, 2016).

**Figura 6 – Ferramenta Medidor e Fila**



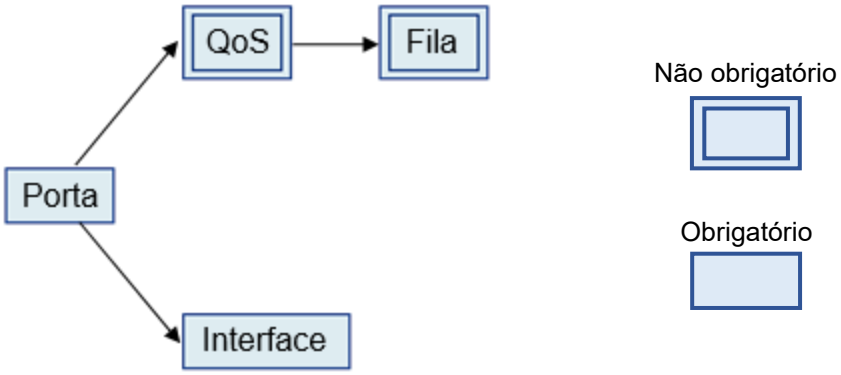

**Fonte: Adaptado de BOLEY (2016)**

### 2.8.1 Filas

A fila consiste em um mecanismo de enfileiramento de pacotes de saída na porta do *switch OpenFlow*. A fila foi primeiro implementada no *OpenFlow* 1.0 com uma taxa mínima garantida em cada fila e depois foi estendida para *OpenFlow* 1.2 com também uma taxa máxima em cada fila. O gerenciamento da fila é realizado pelo

OVSDB, sendo que o *OpenFlow* apenas pode consultar as estatísticas da fila do *switch* (PRADEEP, 2017).

**Figura 7 – Esquema do banco de dados OVSDB**




**Fonte: Adaptado de KRISHNA (2016)**

O esquema da tabela de portas no OVSDB é mostrado na Figura 7. A tabela de portas está relacionada a tabela de QoS e a tabela de interface. A relação com a tabela de interface é indispensável e isso significa que cada porta deve ter uma interface. Já a relação com a tabela de QoS não é obrigatória. Uma porta de *switch* pode ter apenas um QoS configurado, já um QoS pode ter várias filas.

Quando uma fila é criada na entrada de fluxo, os pacotes que correspondem à entrada do fluxo por meio desta fila serão encaminhados e a taxa de transferência real desse fluxo será limitada pelos parâmetros de configuração *max_rate* e *min_rate*.

- O parâmetro *min_rate* define a taxa de dados mínima garantida para uma fila. Se a propriedade *min_rate* for definida, o *switch* irá priorizar esta fila para atingir a taxa mínima mencionada, ao custo das taxas de outros fluxos. Sendo que a capacidade é compartilhada proporcionalmente com base na *min_rate* de cada fila;
- O parâmetro *max_rate d*efine a taxa máxima de dados permitida para esta fila. Caso a taxa real de fluxos que for usada por esta fila acabar sendo maior do que a *max_rate* especificada, o *switch* irá atrasar os pacotes ou descartá-los para satisfazer a *max_rate* (PRADEEP, 2017).

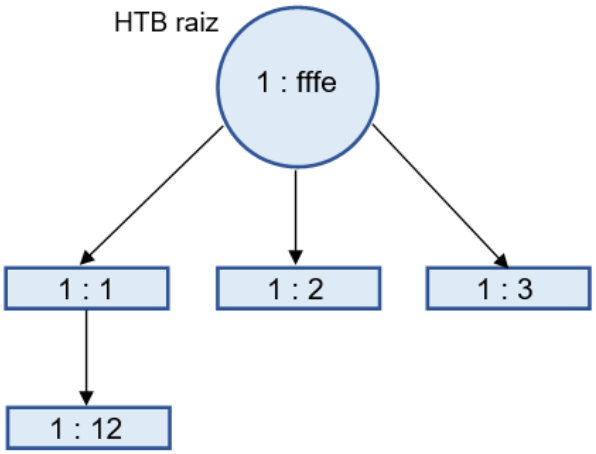
A ferramenta fila do *OpenFlow* usa algoritmos de enfileiramento, que afetam o atraso experimentado por um pacote e determina quanto tempo ele espera para ser transmitido. Existem alguns algoritmos de enfileiramento, incluindo algoritmos simples como *First In First Out* (FIFO) ou aqueles mais elaborados como os

algoritmos HTB, o *Stochastic Fairness Queuing* (SFQ) e o *Hierarchical Fair-Service Curve* (HFSC). Destacas-se também os algoritmos conhecidos como Gerenciamento Ativo de Filas ou *Active Queue Management* (AQM), os algoritmos *Random Early Detection* (RED), *Controlled Delay* (Codel) e o *Fair Queuing Controlled Delay* (FQ Codel) (B. HUBERT, 2022).

Neste trabalho será utilizado o algoritmo HTB, pois é um dos mais utilizados na literatura para controle de fluxos em redes SDN. No algoritmo HTB, os tokens são gerados a uma taxa fixa e logo são armazenados em um balde de capacidade fixa. Os pacotes só podem ser retirados da fila ou enviados para uma porta de saída se houver token disponível no balde.

Dentro de uma instância HTB, podem existir várias classes. A Figura 8 mostra um exemplo de arranjo de classes HTB. Classe 1: fffe é a classe raiz HTB, ela tem três classes filho: classe 1: 1, 1: 2 e 1: 3. A classe 1: 1 é outra classe HTB que tem seu próprio filho, classe 1:12. Uma classe que não é pai de outra classe é conhecida como classe folha. Aqui, para cada uma das classes folha, uma *qdisc* FIFO é anexada (KRISHNA, 2016).

**Figura 8 – Exemplo de classes HTB**




**Fonte: Adaptado de KRISHNA (2016)**

As classes HTB têm várias propriedades importantes:

- *Default*: os pacotes que não são classificados serão enviados a um nó configurado no campo padrão (*default*). O nó padrão é definida utilizando o parâmetro *default*;

- *Priority*: Define a prioridade da classe. A classe com o campo (prio 0) tem a maior prioridade;

- *Rate*: usado para definir a transmissão mínima que a classe poderá atingir;

- *Ceil*: usado para definir a transmissão máxima que a classe poderá atingir;

- *Burst*: este é o tamanho do balde de *rate*. Define quantos *bytes* poderão ser transmitidos em rajadas;

- *Cburst*: este é o tamanho do balde de *ceil*. Determina quantos bytes poderão ser enviados na capacidade máxima do dispositivo durante uma rajada (DEVERA, 2022).

HTB estende o sistema de modelagem de tráfego de *token bucket* com um modelo de empréstimo de token. Usando este modelo de empréstimo, quando uma classe usa menos largura de banda do que a quantidade atribuída, a largura de banda ociosa fica disponível para qualquer outra classe usar. É importante notar que o termo “emprestar” não é totalmente preciso, uma vez que a classe que toma emprestado não tem nenhuma obrigação de devolver o recurso que foi emprestado.

Quando a taxa de uma classe filha é excedida, é permitido tomar emprestados tokens de sua classe pai até atingir o teto (taxa máxima permitida). Quando atingir o teto, o sistema começará a enfileirar os pacotes. Na fila, os pacotes serão retirados da fila se houver *tokens* suficientes ou descartados quando a fila estiver cheia. Na Figura 7, as classes 1: 1, 1: 2 e 1: 3 podem “emprestar” *tokens* da classe raiz, enquanto a classe 1:12 pode emprestar de 1: 1 (KRISHNA, 2016).

### 2.8.2 Medidor

A tabela medidor foi introduzida no *OpenFlow* 1.3 como um novo recurso para prover QoS. Ela permite monitorar a taxa de ingresso de um fluxo e, em seguida, executa as operações com base na taxa do fluxo. O medidor é conectado às entradas de fluxo e é uma propriedade de uma porta de *switch*. Uma tabela de medidores é composta por um identificador medidor, bandas de medidor e contador. Quando a taxa do fluxo é maior do que a taxa da banda especificada, a operação especificada no tipo de banda será realizada para o fluxo. Existem dois tipos de operação de banda que definem como os pacotes são processados: “*drop*” que descarta pacotes que excedem o especificado na taxa da banda e é parecido com o *min_rate* de uma fila, e também se tem o “*dscp remark*”, que aumenta a precedência do campo DSCP no

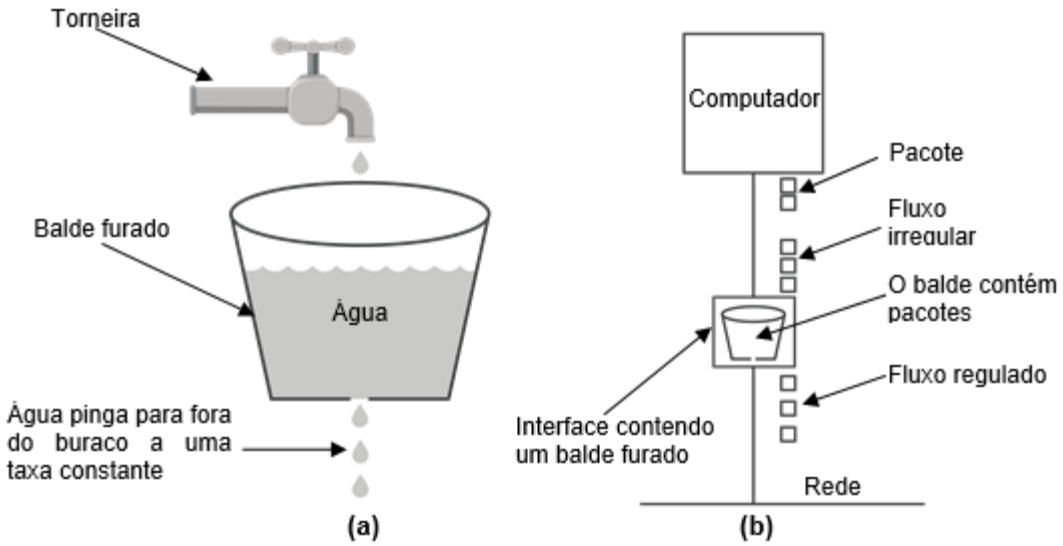
cabeçalho IP do pacote. O DSCP se refere a um conjunto de valores de QoS e estes valores podem ser utilizados para priorizar o tráfego, onde terá três níveis de prioridade e quatro classes de serviço (PRADEEP, 2017).

A tabela medidor é um complemento da fila, sendo que a fila *OpenFlow* pode garantir uma taxa mínima para um fluxo usando modelagem de tráfego e policiamento em *switches*, enquanto que com a tabela medidor não é possível fazer isso. Já o medidor possibilita realizar a alteração de *bits* DSCP em vez de apenas descartar pacotes como a *min_rate* da fila. Fila não pode ser configurada com o canal *OpenFlow*, sendo que o medidor pode ser instalado, modificado e removido em tempo de execução usando o protocolo *OpenFlow* (KRISHNA, 2016).

Medidor é implementado através do algoritmo chamado *leaky bucket* (balde furado). A mesma ideia de um balde furado pode ser aplicada aos pacotes como ilustrado na Figura 9, ou seja, em um balde furado, não importa a velocidade que a água entra no balde, o fluxo de saída sempre terá uma taxa constante havendo água no balde e será zero se ele estiver vazio e se ele estiver cheio a água que escorre pelas bordas será perdida (TANEMBAUM, 2011).

O algoritmo de balde furado é uma fila finita quando chega um novo pacote, ele vai para a fila se tem espaço e se não tem, ele é descartado. A cada pulso de relógio, um pacote é transmitido. Da mesma forma funciona o balde furado de contagem de bytes, a cada pulso, um contador é inicializado em n. Quando o primeiro pacote da fila tem menos bytes que o valor do contador, ele é transmitido, e o contador será decrementado por esse número de bytes. Se o contador tiver um valor suficientemente alto é possível enviar pacotes adicionais. Caso o contador fica abaixo do comprimento do próximo pacote na fila, a transmissão para até a contagem de bytes residuais ser reinicializada e o fluxo pode continuar (TANEMBAUM, 2011).

**Figura 9 – (a) Um balde furado com água. (b) Um balde furado com pacotes**



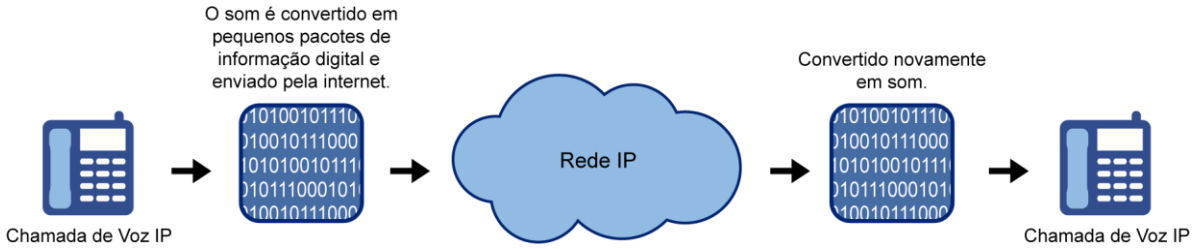

**Fonte: Adaptado de TANEMBAUN (2011)**

Estes mecanismos de QoS podem ser usados em diferentes tipos de tráfego como dados, voz ou vídeo. Na seguinte seção será apresentado o protocolo VoIP que é o foco neste trabalho.

## 2.9 VoIP

VoIP é a abreviação de *Voice Over Internet Protocol*, em português Voz sobre Protocolo de Internet (VOIP DO BRASIL, 2019). A tecnologia VoIP converte ondas sonoras em dados digitais como ilustrado na Figura 10. Desta forma, a chamada telefônica convertida agora em dados, trafega pela Internet igual os e-mails, páginas da *web* ou *downloads* de arquivos.

**Figura 10 – Funcionamento de uma aplicação VoIP**




**Fonte: Adaptado de ORADE (2022)**

A transmissão de dados é feita pelo protocolo *Realtime Transport Protocol* (RTP) e este utiliza como transporte o protocolo *User Datagram Protocol* (UDP). Os dados são codificados por *codecs*, eles são algoritmos que permite o transporte de um sinal analógico através de linhas digitais e variam em complexidade. (KARAPANTAZIS; PAVLIDOU, 2009). O protocolo RTP e os *codecs* serão descritos detalhadamente nas subseções 2.9.1 e 2.9.2.

### 2.9.1 RTP e RTCP

O RTP é um protocolo de transporte em tempo real que provê o transporte de dados fim a fim de áudio, vídeo ou ambos simultaneamente. O protocolo não garante qualidade de serviço e não fornece um mecanismo para assegurar a entrega das informações e nem a entrega em ordem. Deste modo, um número de sequência é incluído no pacote RTP, permitindo ao destino reconstruir a sequência de pacotes (SCHULZRINNE *et al.*, 2003). O transporte de dados é complementado pelo protocolo de controle *Real-time Transfer Control Protocol* (RTCP).

O protocolo de controle de transporte em tempo real pode ser usado juntamente com o RTP, mas o RTP e RTCP utilizam portas diferentes. O protocolo RTCP tem como finalidade transmitir periodicamente pacotes de controle para todos os participantes da sessão contendo informações sobre a qualidade da distribuição dos dados.

### 2.9.2 *Codecs* de voz

Os *codecs* são algoritmos que convertem um sinal analógico de voz em sinal digital, desta forma disponibilizando a transmissão através de linhas de comunicação digital. Os *codecs* variam em complexidade, na qualidade do áudio e largura de banda necessária (MEHTA; UDANI, 2001). Cada chamada utiliza um tamanho de banda para trafegar os dados, então quanto maior a qualidade das chamadas, maior a largura de banda necessária para a transmissão.

Todos os *codecs* são controlados e aprovados pela entidade *International Telecommunication Union-Telecommunication* (ITU-T). A ITU avalia e atribui pontuação de 1 (ruim) até 5 (excelente) para o *codec* após o processo de testes (ITU-

T, 2022). O que os diferencia são os algoritmos utilizados, a média de atraso e a qualidade de voz.

- O *Codec* G.711 é um *Pulse Code Modulation* (PCM) de 13bits e 14bits para amostras logarítmicas de *byte* e opera em uma taxa de transmissão de 64Kbps. Existe dois tipos de G.711, o padrão que é o G.711.1 com uma amostra por pacote e o G.711.2 com duas amostras por pacote.
- O *Codec* G.723.1 pode operar em duas taxas de transmissão: 6.3Kbps usando o algoritmo *Multi Pulse-Maximum Likelihood Quantization* (MPC-MLQ) e 5.3Kbps usando o algoritmo *Algebraic Code Excited Linear Prediction* (ACELP).
- O *Codec* G.726 pode operar em quatro taxas de transmissão, 16, 24, 32 e 40Kbps e utiliza o algoritmo *Adaptive Differential Pulse Code Modulation* (ADPCM).
- O *Codec* G.729 opera em uma taxa de transmissão de 8Kbps. Este *codec* consome menos largura de banda e a qualidade de áudio é considerado bom, mas o consumo de CPU é alto, além de ser um *codec* pago (ITU-T, 2022).

As principais características dos *codecs* são apresentadas no Quadro 2:

**Quadro 2 – Características dos *codecs***

| Codec & Bit Rate (Kbps) | Codec Sample Interval (ms) | Codec Sample Size (Bytes) | Mean Opinion Score (MOS) | Voice Payload Size (Bytes) | Voice Payload Size (ms) | Packets Per Second (PPS) | Bandwidth Ethernet (Kbps) |
|---|---|---|---|---|---|---|---|
| **G.711 (64 Kbps)** | 10 | 80 | 4.1 | 160 | 20 | 50 | 87.2 |
| **G.729 (8 Kbps)** | 10 | 10 | 3.92 | 20 | 20 | 50 | 31.2 |
| **G.723.1 (6.3 Kbps)** | 30 | 24 | 3.9 | 24 | 30 | 33.3 | 21.9 |
| **G.723.1 (5.3 Kbps)** | 30 | 20 | 3.8 | 20 | 30 | 33.3 | 20.8 |
| **G.726 (32 Kbps)** | 5 | 20 | 3.85 | 80 | 20 | 50 | 55.2 |
| **G.726 (24 Kbps)** | 5 | 15 | 3.5 | 60 | 20 | 50 | 47.2 |

**Fonte: Adaptado de VOICE OVER IP (2016)**

- *Bit Rate* (Kbps) é o número de *bits* por segundo necessários para serem transmitidos;

- *Codec Sample Size* é o número de *bytes* gerados a cada intervalo de amostragem;
- *Codec sample interval* é o intervalo de amostragem com o qual o Codec opera.
- MOS é a pontuação do Codec de acordo com a ITU-T;
- *Voice Payload Size* é o número de *bytes* preenchido no pacote com voz;
- *Packets per second* (PPS) é o número de pacotes que precisam ser transmitidos para entregar o *bit rate* do Codec.

As características no tráfego de voz dependem principalmente do Codec adotado. Neste trabalho, adotou-se o G.711 pois é um dos Codecs mais utilizados no mercado.

### 2.9.3 QoS para VoIP

Nos roteadores ou switches a capacidade de processamento e tráfego de pacotes nos canais de comunicação são compartilhados por diversas conexões simultâneas, normalmente o tratamento de congestionamento é o descarte dos pacotes em excesso. Sendo assim, o principal objetivo da QoS em aplicações VoIP que são consideradas aplicações sensíveis a atrasos de propagação é priorizar o tráfego de pacotes (BERNAL, 2007).

Em transmissões de dados, os principais fatores que determinam a QoS são: o atraso que é a principal métrica neste trabalho, *jitter*, perda de pacotes, vazão e *Mean Opinion Score* (MOS).

- O atraso em aplicações VoIP é caracterizado como sendo o tempo que leva para a fala sair da boca de um interlocutor e chegar ao ouvido do receptor (MONKS, 2006). Caso o valor do tempo de atraso seja alto, ele influenciará na qualidade da chamada, sendo recomendado não exceder 150ms (SINGH, 2014);
- O *jitter* é uma variação do atraso dos pacotes propagados pela rede e isso conduz a uma dificuldade do restabelecimento do sincronismo dos sinais de voz transportados pelos pacotes (BERNAL, 2007);
- As perdas de pacotes de voz em uma rede sem recursos de QoS são tratados da mesma forma que outros pacotes de dados, ou seja, em casos de

congestionamento e tráfego intenso os pacotes de voz são descartados sem distinção (MONKS, 2006);

- A vazão é o volume de dados por segundo necessário para o tráfego dos pacotes de voz (ALMEIDA, 2008);
- MOS é um método subjetivo de teste para medir a qualidade de voz. Nesse método os usuários VoIP são selecionados para realizarem uma chamada ou ouvirem uma amostra de voz e avaliar a qualidade atribuindo uma nota. A nota vai de 1 a 5 e corresponde ao nível de qualidade da chamada (SINGH, 2014).
- A Tabela 1 apresenta a classificação MOS.

**Tabela 1 – Classificação MOS**

| Qualidade da chamada | MOS |
|---|---|
| Excelente | 5 |
| Boa | 4 |
| Regular | 3 |
| Insatisfatória | 2 |
| Ruim | 1 |

**Fonte: Adaptado de SINGH (2014)**

## 2.10 Trabalhos Relacionados

É possível encontrar na literatura trabalhos relacionados que abordaram problemas de QoS em tecnologia VoIP em redes SDN. Seguem alguns trabalhos relacionados com esta pesquisa.

No trabalho de Thorpe *et al.* (2016) a proposta foi uma solução baseada em rede SDN para fornecer um serviço avançado de monitoramento intermediário de chamadas VoIP, onde a solução é detectar e localizar problemas de qualidade em tráfegos com tecnologia VoIP. O nome da solução é *Intermediate Mean Opinion Score* (iMOS), ela foi desenvolvida para utilizar como um módulo do controlador *Floodlight*, além do módulo para o controlador a solução utiliza uma versão modificada do OVS para coletar a métrica proposta também chamada de métrica iMOS. O módulo iMOS coleta as métricas de QoS que podem ser utilizadas para calcular o MOS de nós intermediários no caminho de um fluxo VoIP, encontrando a origem da degradação do QoS e permitindo ações corretivas. Para calcular a métrica iMOS e enviá-las para o

módulo no *Floodlight*, os nós intermediários coletam informações de pacotes RTCP. O módulo no controlador *Floodlight* possui uma interface gráfica que permite ligar e desligar a coleta das informações realizada pelos nós intermediários, além de mostrar todos os valores coletados.

Na pesquisa Cardoso (2015), foi proposto a criação de uma rede lógica que possibilitasse assegurar qualidade de serviço a certos fluxos. Para isso foram desenvolvidas duas lógicas de processamento para controladores SDN utilizando o *Floodlight*, sendo uma destinada à zona de ingresso da rede e outra à zona central. As métricas consideradas foram: número de pacotes perdidos pelas diferentes classes, o *jitter* e atraso dos pacotes. Para a criação de uma rede simulada foi utilizado o Mininet com *switches* OVS e o protocolo de comunicação *OpenFlow* 1.3. O programa utilizado para gerar tráfego foi o iperf3 que permite a limitação de largura de banda em tráfegos. Como resultado no caso do tráfego P2P, que tem como características ser um tráfego que cria uma grande quantidade de fluxos, estes entraram em competição consigo mesmo, pois um fluxo criado por um host, tenta obter garantias tais como outros fluxos oriundos do mesmo host. Já o tráfego de voz teve suas garantias atendidas, independentemente de qualquer que seja o estado da rede. Foi observado que a taxa de ocupação das ligações protegidas foi elevada, acima dos 93%, o que permite que um controlador assegure um maior volume de tráfego com qualidade de serviço do que aquele que supostamente é suportado pelo *switch* e existiu um aumento de 20% de largura de banda com garantidas, em comparação com uma rede onde esta técnica não seria utilizada, portanto obteve-se uma rede mais eficiente.

Maribondo e Fernandes (2016) propuseram uma nova forma de Adaptação de *codecs* em SDN (CAoS). Para isso, o estudo foi baseado em uma arquitetura SDN e de acordo com os autores, implementa uma abordagem eficiente e inovadora para evitar a degradação de voz em redes corporativas e reduz a qualidade em chamadas de voz quando ocorre congestionamentos para chamadas que utilizam Codec com menor largura de banda. O CAoS é composto de três módulos, um módulo para monitoramento do tráfego, um módulo para as decisões e um módulo para as mensagens SIP. Conforme os resultados obtidos, o CAoS permite que uma quantidade maior de chamadas possa ser efetuada em períodos de tráfego intenso sem degradar a qualidade de voz.

Na pesquisa de Hasrouty *et al.* (2016) para que chamadas de conferência de voz possa atingir alta qualidade pela Internet é uma tarefa difícil. Para que as redes dinâmicas e dispositivos móveis heterogêneos possam garantir uma boa qualidade de experiência, devem ser adequadamente gerenciados por sistemas de chamadas VoIP multipartes (*Called Multiparty VoIP - MVoIP*). Para os autores, o MVoIP é uma sessão ou uma chamada entre três ou mais clientes chamados participantes ou partes. O sistema VoIP multipartes é baseado na tecnologia SDN e para otimizar a qualidade de uma chamada em conferência para cada participante, utiliza distribuição *multicast* e adaptação dinâmica de fluxos. Com a utilização de SDN, as redes podem ser observadas globalmente a partir de uma unidade central. Uma vez que a topologia de rede e as condições dos canais são conhecidas, o MVoIP atua monitorando caminhos com menor latência criando fluxos nesses caminhos e entregando dinamicamente as taxas de *bit rates*.

A proposta de Costa *et al.* (2015) foi investigar a adequação do servidor PABX Asterisk para fornecer capacidades de comunicação aceitável entre VoIP com MOS para um grande número de chamadas. Para medir a capacidade do servidor PBX foi usado a métrica de probabilidade de bloqueio enquanto o MOS é utilizado para avaliar a qualidade das chamadas VoIP. A proposta mostra nos resultados que o servidor PBX Asterisk pode lidar simultaneamente com 165 chamadas de voz com probabilidade de bloqueio de menos 5%, com isso, proporcionando chamadas VoIP com MOS acima da média 4. Ainda na investigação, o MOS é medido por uma ferramenta chamada VoIPMonitor que observa o tráfego VoIP.

Já o Kwon *et al.* (2014) propuseram em redes SDN uma arquitetura de serviço chamada de *Adaptive Mobile Voice over Internet Protocol* (mVoIP) com o objetivo de melhorar a qualidade de serviços VoIP para usuários móveis. Melhorar a *Quality of Experience* (QoE) é um dos pontos chave. Na arquitetura que foi proposta, possui agentes chamados mVoIP *QoE measurement agentes* que são implantados em redes *wireless* e tem como objetivo coletar informações de QoS. O *mVoIP QoS Manager* é outro elemento da arquitetura que é uma aplicação SDN que coleta dados dos mVoIP *QoE measurement agents* e também dos OVS em redes SDN. O mVoIP *QoS Manager* analisa os dados coletados e decide qual o Codec mais adequado e qual o melhor caminho para os fluxos mVoIP de acordo com as condições da rede. Contudo, é apresentado apenas uma proposta de arquitetura, e como trabalhos futuros serão realizados testes experimentais.

O Quadro 3 apresenta a comparação entre as métricas analisadas dos trabalhos relacionados e o modelo proposto. Nota-se que os outros trabalhos não garantem um atraso máximo na transmissão de dados VoIP que é proposto neste trabalho e também é proposto determinar o tamanho da fila em função do número de usuários para garantir o atraso e para isso será usado uma ferramenta disponível no protocolo *OpenFlow*, que são as filas.

**Quadro 3 – Comparação entre trabalhos relacionados e a proposta**

| | **Métrica analisadas** | | | | |
|---|---|---|---|---|---|
| **Referência** | **Atraso máximo** | **Vazão** | **Número de usuários** | **Filas** | **MOS** |
| Cardoso (2015) | | **X** | | | |
| Costa *et al.* (2015) | | | **X** | | **X** |
| Hasrouty et al (2016) | | | **X** | | |
| Kwon (2014) | | | **X** | | |
| Thorpe *et al.* (2016) | | | | | **X** |
| Maribondo e Fernandes (2016) | | **X** | **X** | | |
| **Proposta** | **X** | **X** | **X** | **X** | |

**Fonte: Autoria propria (2023)**

Observa-se na tabela de comparação que as métricas analisadas nos trabalhos são variadas, mas para a tecnologia VoIP, que é a proposta deste trabalho a métrica mais importante da proposta é garantir o atraso.

# 3 REGRESSÃO LINEAR

A análise de regressão estuda a relação entre uma variável chamada variável dependente $\gamma$ e outras variáveis chamadas variáveis independentes $x1, \dots, x\rho$.

Este relacionamento é representado por um modelo matemático, isto é, por uma equação que associa a variável dependente com as variáveis independentes.

Os dois tipos de análise de regressão que foram utilizados neste trabalho são:

- Regressão Linear Simples: define-se como uma relação linear entre a variável dependente e uma variável independente.
- Regressão Linear Múltipla: define-se como uma relação linear entre a variável dependente e várias variáveis independentes (ANTUNES, 2012).

## 3.1 Regressão Linear Simples

A Equação 1 que representa o modelo de regressão linear simples é:

$$\gamma_i = \beta_0 + \beta_1 x_i + \varepsilon_i, \qquad i = 1, \dots, n \qquad (1)$$

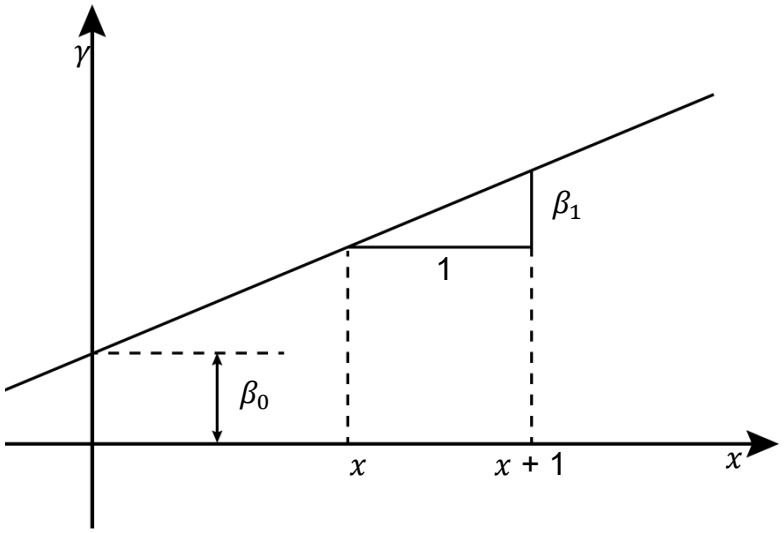
onde:

- $\gamma_i$ representa o valor da variável chamada variável dependente $\gamma$ e a observação $i$, $i = 1, \dots, n$;
- $x_i$ representa o valor da variável chamada variável independente $x$ e a observação $i$, $i = 1, \dots, n$;
- $\varepsilon_i$, $i = 1, \dots, n$ observação $i$ que correspondem ao erro $\varepsilon$;
- $\beta_0$ e $\beta_1$ correspondem aos parâmetros ou coeficientes do modelo.

$\beta_0$ é o ponto em que a reta da regressão corta o eixo *Y* quando *X = 0* e é chamado de coeficiente linear.

$\beta_1$ é a inclinação da reta, mostrando a taxa de mudança em *Y*. Na Figura 11 observa-se a interpretação geométrica dos parâmetros $\beta_0$ e $\beta_1$ (ANTUNES, 2012).

**Figura 11 – Interpretação geométrica dos parâmetros $\boldsymbol{\beta_0}$ e $\boldsymbol{\beta_1}$**




**Fonte: Adaptado de ANTUNES (2022)**

Para encontrar os coeficientes $\beta_0$, $\beta_1$, tem-se a seguinte notação matricial:

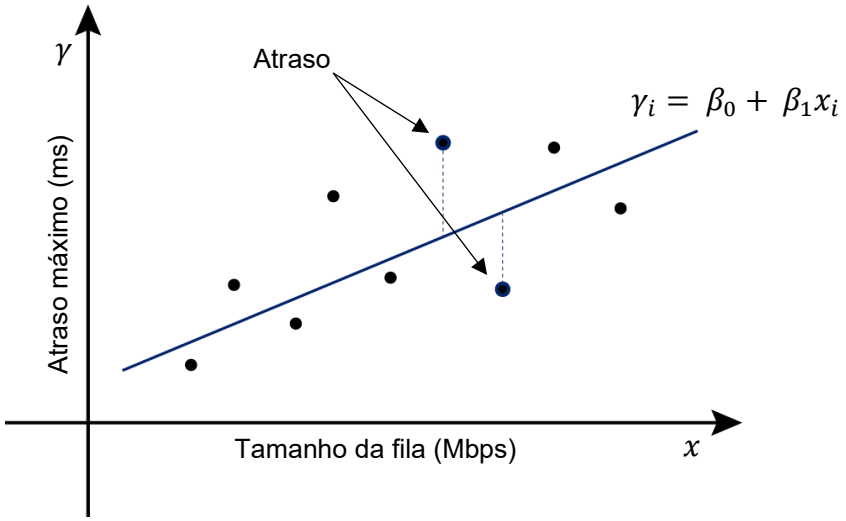
$$\begin{cases} \beta_0 + \beta_1 x_1 = \gamma_1 \\ \beta_0 + \beta_2 x_2 = \gamma_2 \\ \quad \vdots \\ \beta_0 + \beta_k x_k = \gamma_k \end{cases} \leftrightsquigarrow \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_k \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} = \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_k \end{bmatrix} \quad (2)$$

As variáveis devem ser substituídas pelos valores obtidos e procurar por uma solução de mínimos quadrados. Em seguida, a matriz pode ser reduzida por escalonamento para obter a estimativa dos coeficientes de regressão:

$$\begin{cases} \simeq \beta_0 \\ \simeq \beta_1 \end{cases}$$

Na Figura 12 pode-se observar que a partir das variáveis obtém-se uma equação da reta que neste exemplo nos ajuda a prever o atraso máximo dado o tamanho da fila. Na ilustração vê-se que a previsão não é muito boa, em alguns pontos o atraso está acima do esperado e em outros casos abaixo.

**Figura 12 – Ilustração da regressão linear simples**




**Fonte: Autoria própria (2023)**

## 3.2 Regressão Linear Múltipla

A diferença entre a regressão linear simples e a regressão linear múltipla é que na múltipla são consideradas duas ou mais variáveis independentes. Assume-se que na regressão linear múltipla existe uma relação linear entre uma variável dependente $y$ e $p$ variáveis independentes, $x_1, x_2, \dots, x_p$ (ANTUNES, 2012).

A Equação 3 que representa o modelo de regressão linear múltipla é:

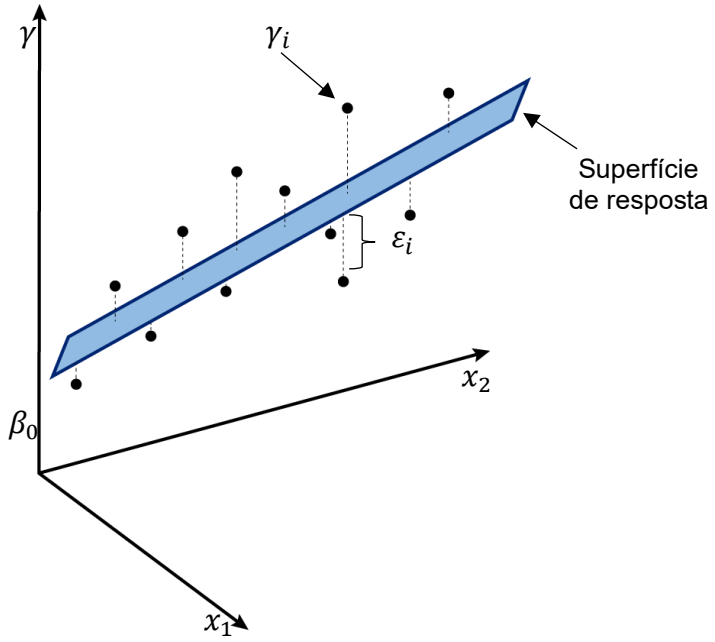
$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} + \varepsilon_i, \qquad i = 1, \dots, n \qquad (3)$$

onde:

- $y_i$ representa o valor da variável resposta $i$; $i = 1, \dots, n$;
- $x_{i1}, x_{i2}, \dots, x_{ip}$, $i = 1, \dots, n$ são os valores da $i$-ésima observação das $p$ variáveis independentes;
- $\beta_0, \beta_1, \beta_2, \dots, \beta_p$ são os coeficientes de regressão;
- $\varepsilon_i$, $i = 1, \dots, n$ observação $i$ que correspondem ao erro.

Na regressão linear múltipla a função de regressão é chamada de superfície de resposta e descreve-se um hiperplano no espaço p-dimensional das variáveis de entrada $x_p$ conforme mostrado na Figura 13.

**Figura 13 – Hiperplano no espaço p-dimensional das variáveis $x_p$**




**Fonte: Adaptado de ANTUNES (2022)**

Para encontrar os coeficientes $\beta_0, \beta_1, \beta_2, \ldots, \beta_p$, tem-se a seguinte notação matricial:

$$\begin{cases} \beta_0 + \beta_1 x_1 + \beta_2 x_2 = \gamma_1 \\ \beta_0 + \beta_2 x_2 + \beta_3 x_3 = \gamma_2 \\ \vdots \\ \beta_0 + \beta_k x_k + \beta_k x_k = \gamma_k \end{cases} \leftrightsquigarrow \begin{bmatrix} 1 & x_1 & x_2 \\ 1 & x_2 & x_3 \\ \vdots & \vdots & \vdots \\ 1 & x_k & x_k \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix} = \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_k \end{bmatrix} \quad (4)$$

Após substituídas as variáveis pelos valores obtidos e preciso encontrar uma solução de mínimos quadrados e reduzir a matriz por escalonamento para obter a estimativa dos coeficientes de regressão:

$$\begin{cases} \simeq \beta_0 \\ \simeq \beta_1 \\ \simeq \beta_2 \end{cases}$$

### 3.3 R-quadrado

Depois de ajustar o modelo linear, é preciso determinar quão próximos os dados estão da reta de regressão ajustada. Para isso é utilizada uma medida estatística chamada R-quadrado ($\mathcal{R}^2$), também conhecido como coeficiente de determinação ou coeficiente de determinação múltipla para regressão linear múltipla e ele representa a proporção da variância dos valores estimados pela variância dos valores observados.

A Equação 5 que representa o $\mathcal{R}^2$ é:

$$\mathcal{R}^2 = \frac{\sum_{i=1}^{n}(\hat{\gamma}_i - \bar{\gamma})^2}{\sum_{i=1}^{n}(\gamma_i - \bar{\gamma})^2} \qquad (5)$$

onde:

- $\hat{\gamma}_i$ valores estimados;
- $\gamma_i$ valores observados;
- $\bar{\gamma}_i$ média dos valores $\gamma_i$.

O $\mathcal{R}^2$ está sempre entre 0% a 100%, geralmente quanto maior o $\mathcal{R}^2$, melhor a reta se ajusta aos dados (QUININO, 2011).

# 4 DESENVOLVIMENTO DO MODELO

O objetivo deste trabalho é propor um modelo para determinar o tamanho da fila em função do número de usuários e do atraso máximo utilizando rede SDN. Para garantir o atraso máximo foi levado em consideração a largura de banda nas filas que é um mecanismo que fornece ao tráfego QoS dentro do OVS. A admissão e reserva de largura de banda são realizadas no controlador para limitar quantos fluxos de QoS podem ser admitidos.

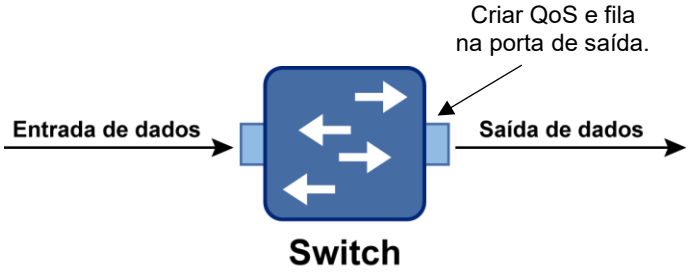
O modelo de garantia usado neste trabalho é o DiffServ, pois é uma proposta que agrupa todos os usuários de um tráfego específico em um único fluxo. Neste capítulo serão explicados em detalhes este modelo.

## 4.1 Tráfegos

Neste modelo, existem dois tipos de tráfego: tráfego de melhor esforço e tráfego de QoS. Um tráfego de melhor esforço é um tráfego sem reservas e garantias de largura de banda comprometendo o atraso nas conexões, enquanto que um tráfego QoS é um tráfego com largura de banda garantida, já que é possível criar filas exclusivas para as conexões.

Por padrão os switches contêm uma fila única usada para todo tipo de dados, onde os dados têm as mesmas prioridades e competem a largura de banda que não são reservadas, desta forma a QoS não é garantido nas conexões VoIP. Como objetivo é determinar o tamanho da fila em função do número de usuário e do atraso máximo nas conexões, é necessário implementar filas com QoS para cada tipo de dados. As filas são implementadas na saída de um switch conforme mostrado na Figura 14.

**Figura 14 – Implementação da fila em um switch**



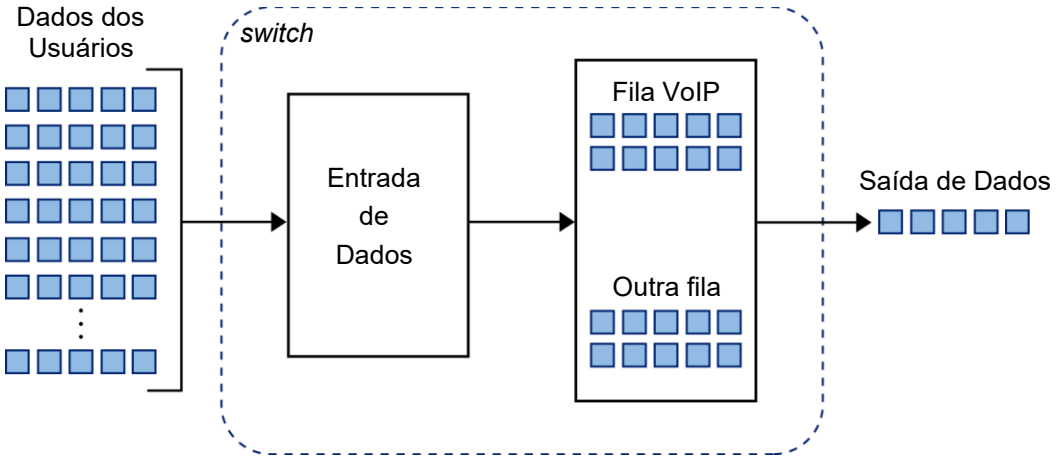

**Fonte: Autoria própria (2023)**

O objetivo do modelo é fazer que seja possível determinar o tamanho da fila em função do número de usuários e do atraso máximo na rede, através da regressão linear como será mostrado no capítulo 5.

## 4.2 Criação das Filas

O objetivo de criar filas é para encaminhar em cada uma delas um determinado tipo de tráfego. Neste modelo foram criadas duas filas, uma para as conexões VoIP e a outra para outros tipos de dados, seguindo a ideia do modelo de QoS DiffServ para que os dados sejam separados de acordo com classes de serviços pré-determinados. A Figura 15 mostra os usuários enviando vários tipos de dados, dentro do switch são separados de acordo com o tipo de QoS implementado em cada fila.

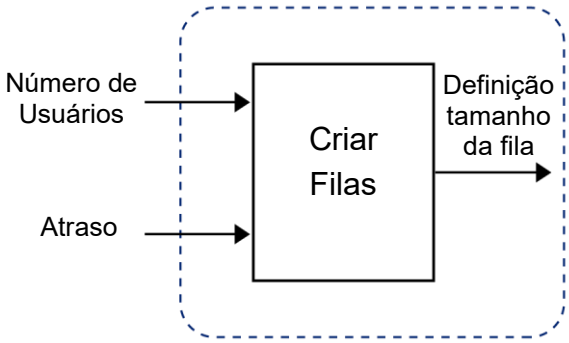
**Figura 15 – Comportamento das filas**




**Fonte: Autoria própria (2023)**

Para criar QoS e tabela de filas no OVSDB do switch, o controlador envia um comando ovs-vsctl. É nesse comando que é definido o tamanho da fila como mostrado na Figura 16, essa definição é um fator determinante para a quantidade de usuários dentro de uma rede para poder garantir o atraso máximo em conexões VoIP. Então o modelo proposto tem 2 variáveis de entrada e uma saída que é o tamanho da fila.

**Figura 16 – Criação da fila e definir o tamanho**




**Fonte: Autoria própria (2023)**

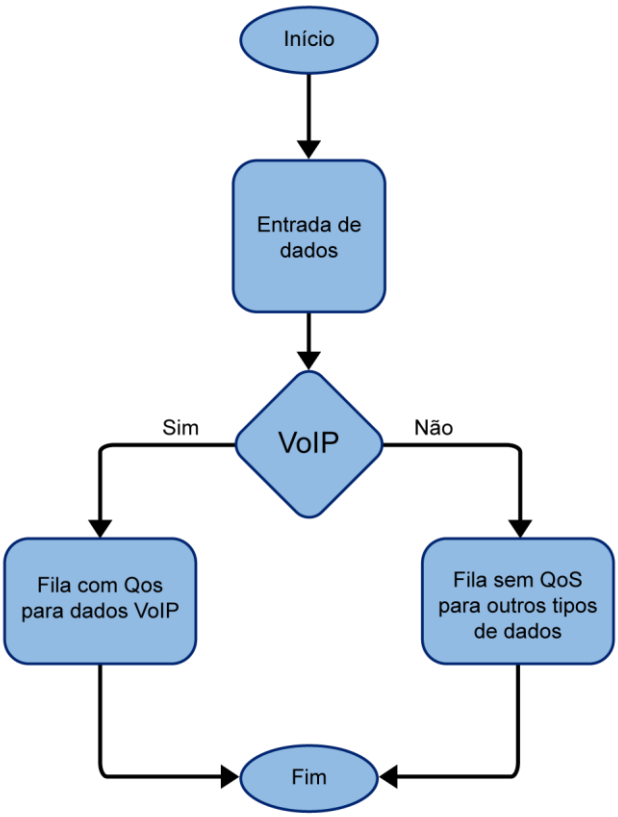
Após a criação da fila foram feitas as simulações para levantar os dados e verificar se as amostrar tem evidências de relação através do valor de $\mathcal{R}^2$.

## 4.3 Fluxograma

A Figura 17 mostra o fluxograma com as etapas usadas no modelo. Quando os dados enviados pelos usuários entram no switch é acionado o processo de admissão no controlador, ele verifica o pacote para determinar o tipo de dado.

Se os fluxos forem conexões VoIP, os dados são enviados para a fila com QoS e o controlador verifica o OVSDB para determinar o tamanho da fila que foi configurada. Essa largura de banda depende da quantidade de usuários na rede para ver se atende o atraso máximo determinado para conexões VoIP. Se o fluxo de dados não for VoIP eles serão direcionados para fila chamada outros.

**Figura 17 – Fluxograma do modelo de tráfego**




**Fonte: Autoria própria (2023)**

## 4.4 Cálculo do Tamanho da Fila

O modelo deste trabalho determina o tamanho da fila para atender 150ms de atraso máximo para conexões VoIP. Então tem-se 3 variáveis que são o número de usuários na rede, o tamanho da fila em Mbps e o atraso.

A seguir será mostrado como foram usados os dados das simulações para obter o tamanho da fila em função do número de usuários e do atraso máximo. Na Tabela 2 tem-se 3 colunas que contém as 3 variáveis do modelo que foram obtidos das simulações que vão ser detalhadas no Capítulo 5.

**Tabela 2 – Dados das 3 variáveis**

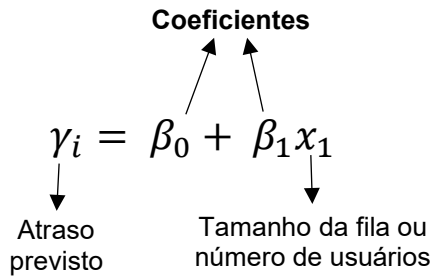
| Atraso Máximo (ms) | Tamanho da fila (Mbps) | Número de usuários |
|---|---|---|
| 18,040 | 8 | 5 |
| 49,453 | 8 | 10 |
| 81,828 | 8 | 15 |
| 114,461 | 8 | 20 |
| 139,903 | 8 | 25 |
| 164,862 | 8 | 30 |
| 17,624 | 9 | 5 |
| 43,994 | 9 | 10 |
| 69,408 | 9 | 15 |
| 95,793 | 9 | 20 |
| 125,456 | 9 | 25 |
| 155,483 | 9 | 30 |
| 16,483 | 10 | 5 |
| 39,215 | 10 | 10 |
| 58,201 | 10 | 15 |
| 83,147 | 10 | 20 |
| 108,587 | 10 | 25 |
| 137,342 | 10 | 30 |

**Fonte: Autoria própria (2023)**

Pode-se observar que na coluna de atraso máximo (ms) tem-se valores distintos em cada linha, o tamanho das filas está dimensionado em 8, 9 e 10 Mbps e o número de usuários de 5 até 30 com intervalos de 5 em 5. Desse modo, nota-se que cada linha é referente ao atraso máximo de uma quantidade de usuários VoIP, com um tamanho de fila específico. Na Figura 18 ilustra-se a regressão linear simples que serve para prever um valor de uma variável dependente $\gamma$ que é o atraso máximo previsto por uma variável independente $x_1$ que pode ser o tamanho da fila ou número de usuários e $\beta_0$, $\beta_1$ que são os coeficientes que serão encontrados.

.

**Figura 18 – Equação de regressão linear simples**




**Fonte: Autoria própria (2023)**

Para encontrar os coeficientes $\beta_0$, $\beta_1$, tem-se o seguinte sistema escrito em notação matricial:

$$\begin{cases} \beta_0 + \beta_1 x_1 = \gamma_1 \\ \beta_0 + \beta_2 x_2 = \gamma_2 \\ \quad \vdots \\ \beta_0 + \beta_k x_k = \gamma_k \end{cases} \leftrightsquigarrow \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_k \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} = \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_k \end{bmatrix} \quad (6)$$

Para realizar uma apresentação de gráfico com dados mais específicos, será substituindo os dados da Tabela 2 para o atraso máximo previsto $\gamma$ e tamanho da fila $x_1$ com número de usuários fixado em 20, tem-se:

$$\begin{bmatrix} 1 & 8 \\ 1 & 9 \\ 1 & 10 \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} = \begin{bmatrix} 114{,}461 \\ 95{,}793 \\ 83{,}147 \end{bmatrix} \leftrightsquigarrow \mathrm{A}\vec{x} = \vec{\beta} \quad (7)$$

Após realizar o cálculo, obtém-se a estimativa dos coeficientes de regressão:

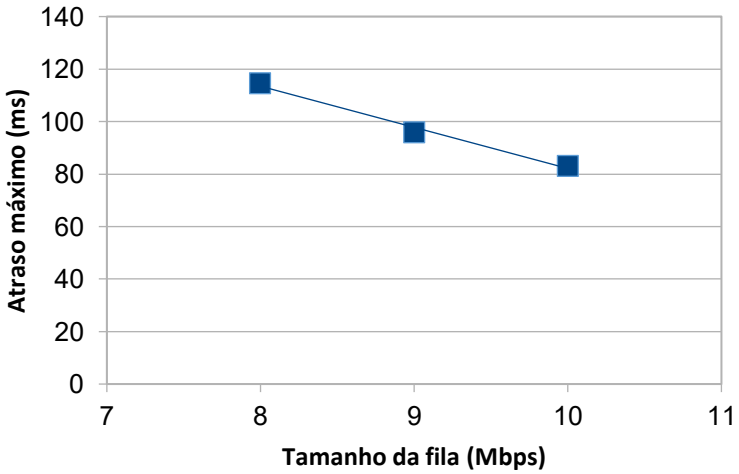
$$\begin{cases} \beta_0 \simeq 238{,}71 \\ \beta_1 \simeq -15{,}66 \end{cases}$$

A Equação 8 de melhor ajuste procurado é aproximadamente,

$$\gamma \backsimeq 238{,}71 - 15{,}66.\, x_1 \quad (8)$$

No Gráfico 1 obteve-se a reta de melhor ajuste apresentado na Equação 6 e o valor de $\mathcal{R}^2$ igual a 98%, a relação destas duas variáveis é decrescente.

**Gráfico 1 – Atraso máximo pelo tamanho da fila**




**Fonte: Autoria própria (2023)**

A seguir será substituindo os dados da Tabela 2 para o atraso máximo previsto $\gamma$ e número de usuários $x_1$ com tamanho da fila fixado em 9Mbps, tem-se:

$$\begin{bmatrix} 1 & 5 \\ 1 & 10 \\ 1 & 15 \\ 1 & 20 \\ 1 & 25 \\ 1 & 30 \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} = \begin{bmatrix} 17{,}624 \\ 43{,}994 \\ 69{,}408 \\ 95{,}793 \\ 125{,}456 \\ 155{,}483 \end{bmatrix} \leftrightsquigarrow \quad A\vec{x} = \vec{\beta} \tag{9}$$

Após realizar o cálculo, obtém-se a estimativa dos coeficientes de regressão:

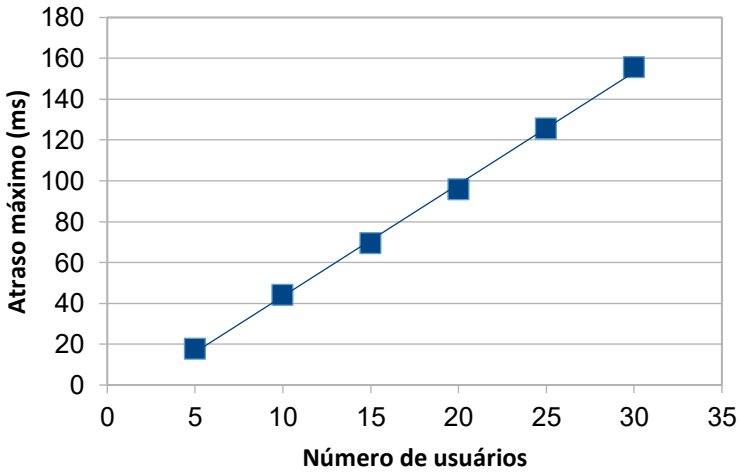
$$\begin{cases} \beta_0 \simeq -11{,}38 \\ \beta_1 \simeq 5{,}46 \end{cases}$$

A Equação 10 de melhor ajuste procurado é aproximadamente,

$$\gamma \simeq -11{,}38 + 5{,}49.\, x_1 \tag{10}$$

O Gráfico 2 ilustra a reta de melhor ajuste representada pela Equação 10 e o $\mathcal{R}^2$ obtido é igual a 99%.

**Gráfico 2 – Atraso máximo pelo número de usuários**

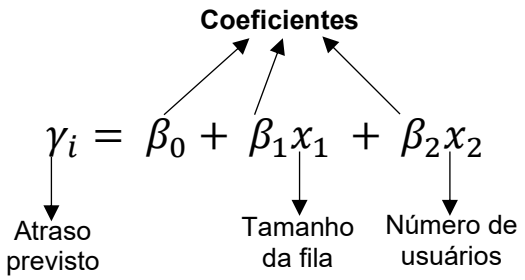



**Fonte: Autoria própria (2023)**

Nota-se que em ambos os casos se tem uma regressão linear e pode-se concluir que o tamanho da fila e o número de usuários tem relação linear com o atraso máximo e percebe-se também que para ver a relação entre o número de usuários, tamanho da fila e atraso máximo pode ser usado regressão linear múltipla.

A Figura 19 ilustra a equação de regressão linear múltipla. Vê-se que a variável dependente $\gamma$ é o atraso máximo previsto, a variável independente $x_1$ é o tamanho da fila, a variável independente $x_2$ é a quantidade de usuários e $\beta_0$, $\beta_1$, $\beta_2$ são os coeficientes que serão encontrados.

**Figura 19 – Equação de regressão linear múltipla**




**Fonte: Autoria própria (2023)**

Para encontrar os coeficientes, tem-se o seguinte sistema escrito em notação matricial:

$$\begin{cases} \beta_0 + \beta_1 x_1 + \beta_2 x_2 = \gamma_1 \\ \beta_0 + \beta_2 x_2 + \beta_3 x_3 = \gamma_2 \\ \vdots \\ \beta_0 + \beta_k x_k + \beta_k x_k = \gamma_k \end{cases} \leftrightsquigarrow \begin{bmatrix} 1 & x_1 & x_2 \\ 1 & x_2 & x_3 \\ \vdots & \vdots & \vdots \\ 1 & x_k & x_k \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix} = \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_k \end{bmatrix} \quad (11)$$

Substituindo os dados da Tabela 2, tem-se:

$$\begin{bmatrix} 1 & 8 & 5 \\ 1 & 8 & 10 \\ 1 & 8 & 15 \\ 1 & 8 & 20 \\ 1 & 8 & 25 \\ 1 & 8 & 30 \\ 1 & 9 & 5 \\ 1 & 9 & 10 \\ 1 & 9 & 15 \\ 1 & 9 & 20 \\ 1 & 9 & 25 \\ 1 & 9 & 30 \\ 1 & 10 & 5 \\ 1 & 10 & 10 \\ 1 & 10 & 15 \\ 1 & 10 & 20 \\ 1 & 10 & 25 \\ 1 & 10 & 30 \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix} = \begin{bmatrix} 18{,}040 \\ 49{,}453 \\ 81{,}828 \\ 114{,}461 \\ 139{,}903 \\ 164{,}862 \\ 17{,}624 \\ 43{,}994 \\ 69{,}408 \\ 95{,}793 \\ 125{,}456 \\ 155{,}483 \\ 16{,}483 \\ 39{,}215 \\ 58{,}201 \\ 83{,}147 \\ 108{,}587 \\ 137{,}342 \end{bmatrix} \leftrightsquigarrow \quad \mathrm{A}\vec{x} = \vec{\beta} \quad (12)$$

Após realizar o cálculo, obtém-se a estimativa dos coeficientes de regressão:

$$\begin{cases} \beta_0 \simeq 84{,}066 \\ \beta_1 \simeq -10{,}464 \\ \beta_2 \simeq 5{,}401 \end{cases}$$

A Equação 13 de melhor ajuste procurado é aproximadamente,

$$\gamma \simeq 84{,}066 - 10{,}464.x_1 + 5{,}401.x_2 \quad (13)$$

Com a Equação 13 obtida, pode-se prever a variável $\gamma$ que é o atraso máximo. Ao se observar a última linha da Tabela 2, é mostrado que em uma rede com 30 usuários e com tamanho da fila de 10Mbps o atraso máximo fica em 137,342ms, lembrando que o limite para conexões VoIP é de 150ms. Substituindo a variável $x_2$,

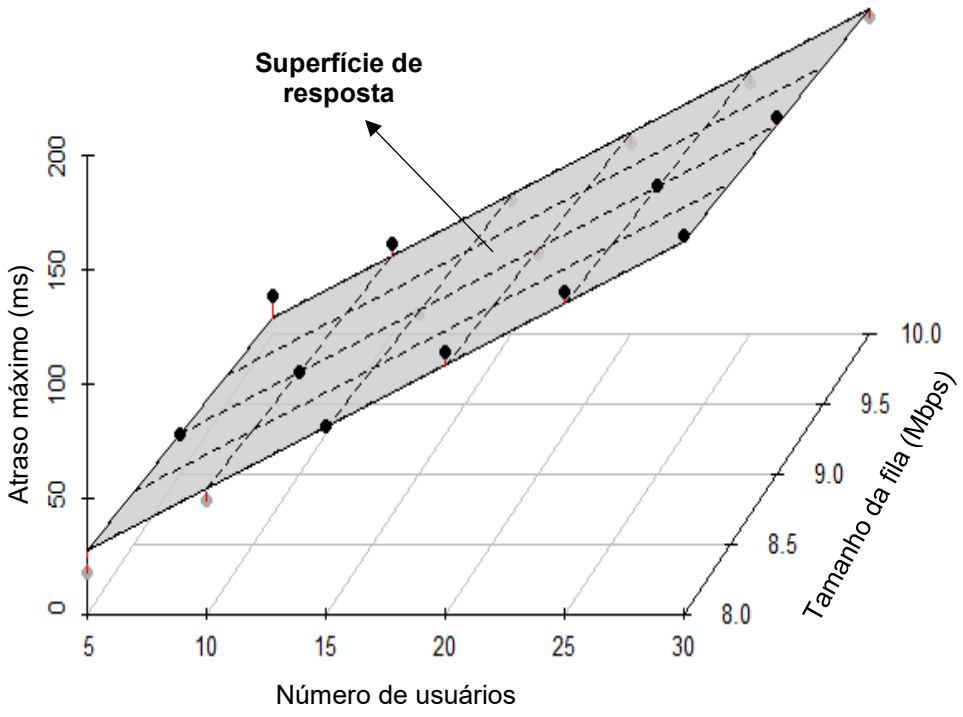
para 32 usuários e deixando a variável $x_1$ com o mesmo tamanho da fila que é 10Mbps, tem-se

$$\gamma \backsimeq 84{,}066 - 10{,}464.10 + 5{,}401.32$$
$$\gamma \backsimeq 152{,}254\ ms$$

Observa-se que o atraso previsto ficou em 152,254ms, isso significa que foi inserido mais dois usuários na rede que tem o tamanho de fila de 10Mbps, o atraso máximo passa do limite máximo de 150ms. Para avaliar se os dados tem correlação calculou-se o $\mathcal{R}^2$ e obteve-se a porcentagem de 98% que representa a proporção da variabilidade na variável dependente pela variável independente.

O Gráfico 3 mostra a regressão linear múltipla para os dados das 3 variáveis apresentadas na Tabela 2. A superfície de resposta é um hiperplano no espaço p-dimensional para as variáveis número de usuários e tamanho da fila em função do atraso máximo. Os pontos na cor preto são valores acima da superfície de resposta, enquanto os pontos na cor cinza estão abaixo.

**Gráfico 3 – Atraso máximo em função do número de usuários e do tamanho da fila**




**Fonte: Autoria própria (2023)**

Conclui-se que o atraso máximo aumenta com o aumento do número de usuários e diminui com o aumento do tamanho da fila. Por exemplo, para 5 usuários, o atraso é de 16,483ms e para 30 usuários, o atraso é de 137,342ms. A visualização do gráfico 3D está posicionada para destacar a superfície de resposta e por esse motivo não fica muito claro os valores do tamanho da fila em função do atraso máximo, mas foi verificado e o tamanho da fila igual a 8Mbps tem atraso de 164,862ms e para 10Mbps de fila o atraso é de 137,342ms.

Ao se isolar a variável independente que corresponde ao tamanho da fila, na Equação 14 é possível achar o tamanho da fila em função da quantidade de usuários e do atraso máximo.

$$x_1 \backsimeq \frac{5{,}401.x_2 + 84{,}066 - \gamma}{10{,}464} \qquad (14)$$

Por exemplo, utilizando a Equação 14 e substituindo os valores para número de usuários $x_2$ igual a 31 e o atraso $\gamma$ em 150ms, o tamanho da fila a ser utilizado na rede é em torno de 9,699Mbps. Como o objetivo do modelo é determinar o tamanho da fila em função do número de usuários e do atraso máximo, a Equação 14 é a principal contribuição deste trabalho.

Os dados que foram apresentados neste capítulo serão comprovados no Capítulo 5 através das simulações realizadas.

# 5 RESULTADOS EXPERIMENTAIS

Este capítulo descreve o ambiente de simulação e suas características, as topologias utilizadas e os resultados obtidos para comprovar a eficácia do modelo proposto.

## 5.1 Descrição do Ambiente de Simulação

Para a realização das simulações, utilizou-se um computador Acer Aspire A515-51 com processador Intel i7-8550U, pertencente a oitava geração de processadores Intel i7, com 4 núcleos físicos e base de processamento em 4GHz, além de 8 megabytes de memória do tipo DDR-4. O computador possui 1 *terabyte* de disco rígido. Sendo assim, esta máquina suporta o ambiente de teste.

O ambiente de simulação é composto por uma máquina virtual, o gerenciador de máquinas virtuais Oracle VM Virtualbox (versão 6.1), contendo:

- Sistema operacional Ubuntu 14.04 LTS;
- Emulador de rede Mininet (versão 2.2.1), juntamente com o software OVS;
- Controlador ODL.

O Quadro 4 apresenta as características do ambiente de simulação.

**Quadro 4 – Características do ambiente de simulação**

| | |
|---|---|
| Computador | Acer Aspire A515-51 |
| Processador | Intel® Core™ i7-8550U CPU @ 1.80GHz 1.99GHz |
| Memória RAM | 8 GB |
| Disco Rígido | 1 TB |
| Máquina Vitual | Oracle VM Virtualbox v.6.1 |
| Sistema Operacional | Ubuntu 14.04 LTS |
| Emulador | Mininet v.2.2.1 |
| *Switches* | Open vSwitch v.2.9 |
| Controlador | OpenDaylight |

**Fonte: Autoria própria (2023)**

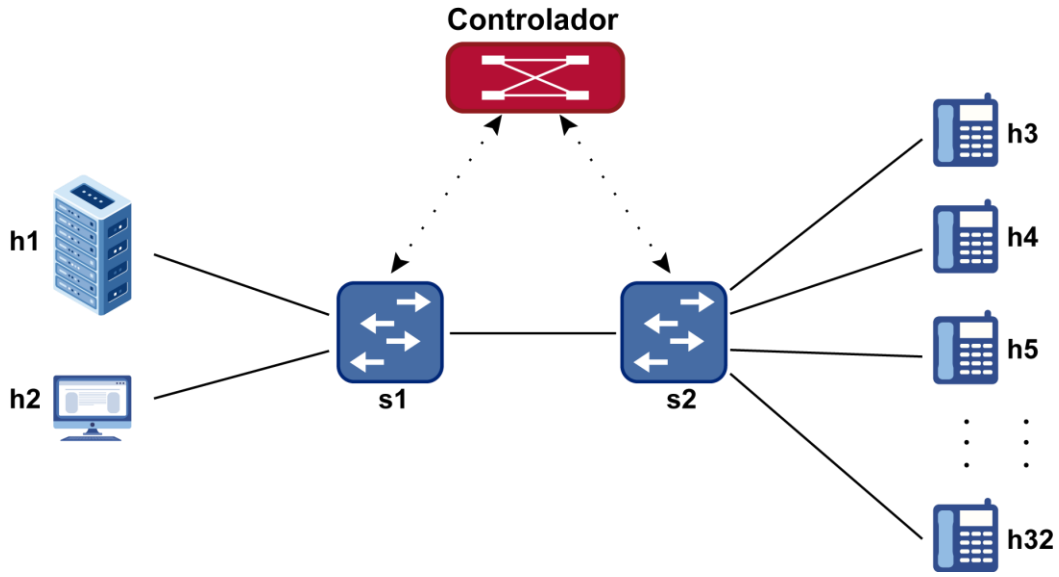
Para a obtenção dos resultados foram emulados vários experimentos gerando tráfegos VoIP na rede. Os programas utilizados foram os seguintes:

- Mininet - é capaz de emular redes e cria uma rede virtual composta por *hosts*, *links*, *switchs* e controladores (MININET, 2022).
- D-ITG - o software *Distributed Internet Traffic Generator* (D-ITG) é uma ferramenta que gera tráfegos IPv4 e IPv6, também consegue mensurar métricas de desempenho em nível de pacote através de relatórios, como por exemplo taxa de transferência, atraso, *jitter*. Alem disso, a ferramenta suporta protocolo TCP quanto o UDP. Essa ferramenta consegue simular tráfegos do tipo ICMP e VoIP. Na arquitetura do D-ITG os recursos são gerados pelos comandos, ITGSend e ITGRecv. O ITGSend se comunica com o ITGRecv reproduzindo tráfegos realizando uma comunicação do tipo cliente/servidor. O ITGSend pode encaminhar diversos tráfego em paralelo para diversas instâncias do ITGRecv, isso depende da configuração implementada pelo usuário. O comando ITGLog é utilizado para coletar as informações dos testes de desempenho e o ITGDec serve para analisar esses dados que foram coletados (D-ITG, 2022).

## 5.2 Topologia

Foram criadas duas topologias para as simulações. A topologia 1 é apresentada na Figura 20. Possui um total de 32 *hosts*, ligados a 2 *switches*, onde o *switch* s1 está ligado ao h1 que é utilizado para enviar os pacotes, o h2 responsável por receber e armazenar informações de log e o *switch* s2 ligado aos demais host h3, h4, h5,...,h32 e cada *switch* controlado pelo controlador ODL.

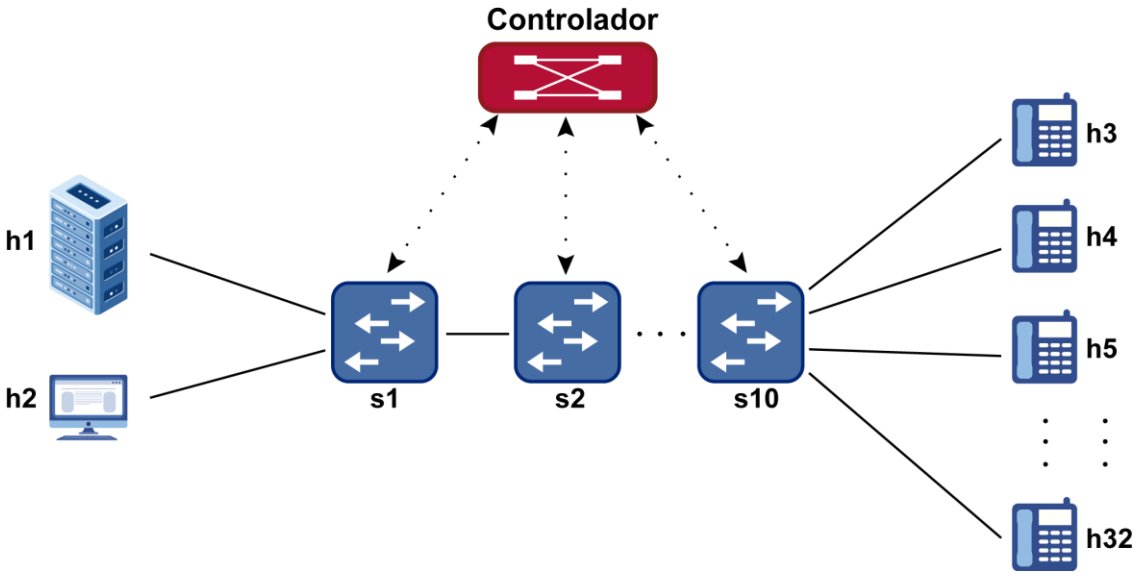
**Figura 20 – Topologia 1**




**Fonte: Autoria própria (2023)**

A topologia 2 é apresentada na Figura 21. Possui um total de 32 *hosts*, ligados em até 10 *switches* em série, onde o *switch* s1 está ligado ao h1 que é utilizado para enviar os pacotes, o h2 responsável por receber e armazenar informações de log e s2, o *switch* s2 ligado simultaneamente até o s10, o *switch* s10 aos demais host H3, H4, H5,...,H32 e cada *switch* controlado pelo controlador ODL.

**Figura 21 – Topologia 2**




**Fonte: Autoria própria (2023)**

Os *hosts* e *switches* estão configurados com IPs fixo.

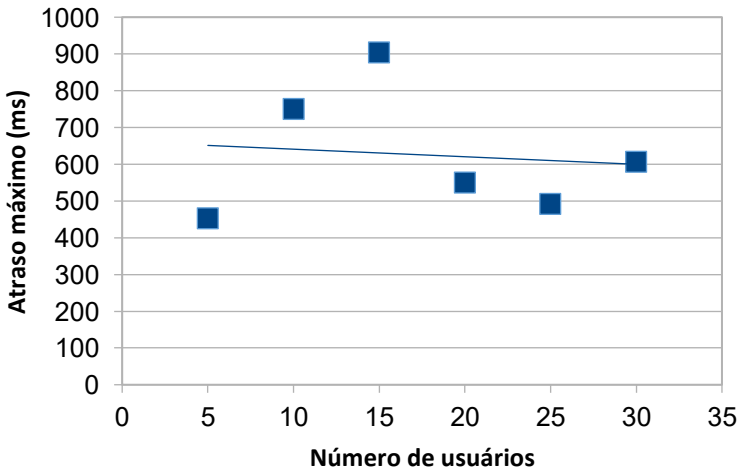
## 5.3 Simulações e Resultados

Nesta seção é apresentada a simulação que mostra o problema da fila padrão do *switch,* simulação que verifica se a quantidade de *switch* na rede altera o valor do atraso, simulação que relaciona o atraso máximo com número de usuários e a previsão do tamanho da fila em função do atraso e número de usuários. No desenvolvimento do modelo no Capítulo 4 foram utilizados dados mostrados na Tabela 2 e aqui são utilizadas outras simulações que são visualizadas no Tabela 3 para verificar se as equações do modelo correspondem com as simulações.

### 5.3.1 Simulação para mostrar o problema

Primeiramente realizou-se a simulação que mostra o problema, para isso utilizou-se a topologia 1 e foram criados tráfegos simultâneos entre TCP e VoIP com codec G711 e com a fila FIFO padrão do *switch*. As conexões foram simuladas em grupo de 5 usuários até 30, os tráfegos foram gerados pelo D-ITG e o enlace entre os *switches* é de 10 Mbps. Nesta simulação foram repetidas 20 vezes para cada grupo de usuários, o resultado é apresentado no Gráfico 4. Foi utilizado regressão linear para verificar se existe relação entre as variáveis.

**Gráfico 4 – Simulação VoIP com uma única fila FIFO**




**Fonte: Autoria própria (2023)**

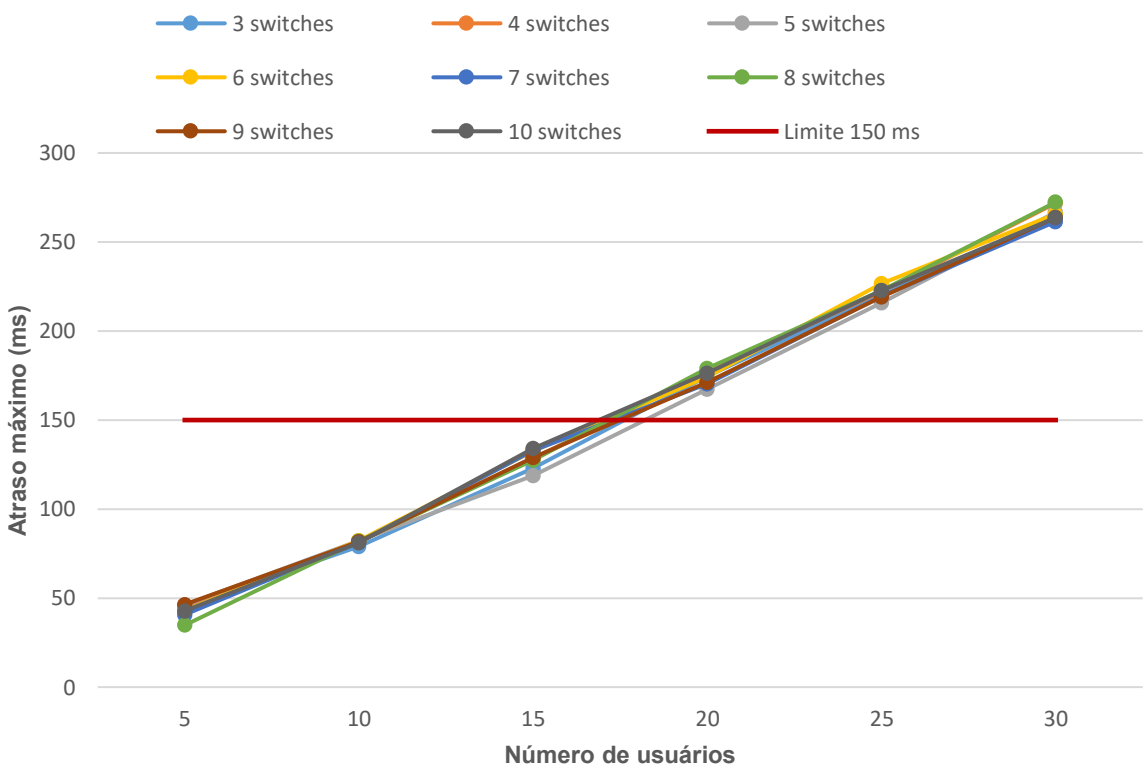
Nesta simulação obteve-se $\mathcal{R}^2 = 1\%$ para o tráfego VoIP e o atraso máximo ficou em 900ms. Como o *switch* contêm uma fila única, observou-se que os dados sofreram um atraso de enfileiramento, ou seja, atraso de fila conforme explicado na seção 2.7 e demostrado na Figura 5. Com isso, as conexões VoIP ficaram com o atraso máximo muito alto.

Com o resultado desta simulação apresentou-se o problema que será tratado neste trabalho. O modelo pretende usar uma fila exclusiva para o tráfego VoIP com o objetivo de obter uma relação linear entre tamanho de fila, atraso máximo e número de usuários.

### 5.3.2 Simulação com número diferente de switches

Para simular uma rede mais real, foi realizada simulação com a topologia 2 aumentando a quantidade de *switches* em até 10, posicionados em série e com a fila VoIP em 5Mbps. Os resultados são mostrados no Gráfico 5.

**Gráfico 5 – Simulação VoIP com até 10 switches**




**Fonte: Autoria própria (2023)**

Observa-se que o limite do atraso continua aproximadamente com 18 usuários, isso mostra que aumentando a quantidade de *switches* na rede não altera o atraso. Por este motivo a topologia 2 não vai ser usada, pois não tem correlação com o atraso, mas foi importante verificar se a quantidade de switches altera o valor do atraso.

### 5.3.3 Simulação que relaciona atraso máximo com número de usuários

Esta simulação foi realizada com o objetivo de verificar a Equação 10 que é referente a regressão linear simples que relaciona atraso máximo e número de usuários. Os dados desta seção são apresentados na Tabela 3 também foram repetidos 20 vezes para cada grupo de usuários.

**Tabela 3 – Dados para verificar as equações**

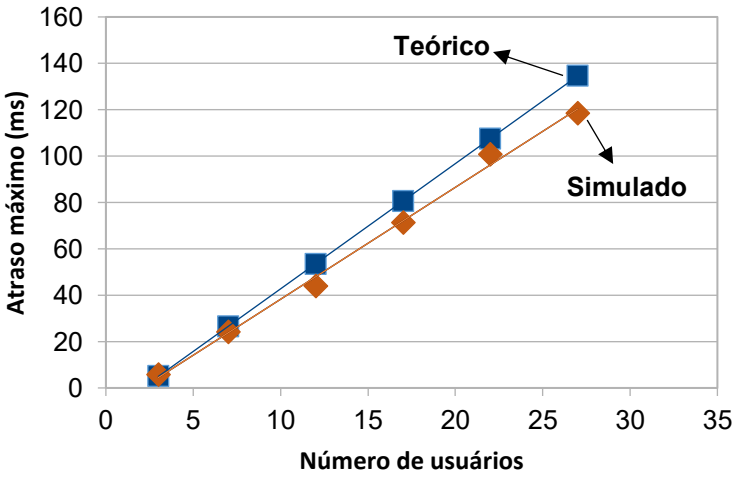
| Atraso Máximo (ms) | Tamanho da fila (Mbps) | Número de usuários |
|---|---|---|
| 6,477 | 8 | 3 |
| 27,664 | 8 | 7 |
| 55,016 | 8 | 12 |
| 83,957 | 8 | 17 |
| 113,612 | 8 | 22 |
| 139,856 | 8 | 27 |
| 5,913 | 9 | 3 |
| 24,138 | 9 | 7 |
| 44,083 | 9 | 12 |
| 71,302 | 9 | 17 |
| 100,850 | 9 | 22 |
| 118,426 | 9 | 27 |
| 4,993 | 10 | 3 |
| 21,956 | 10 | 7 |
| 42,287 | 10 | 12 |
| 65,458 | 10 | 17 |
| 88,361 | 10 | 22 |
| 108,137 | 10 | 27 |

**Fonte: Autoria própria (2023)**

O Gráfico 6 apresenta duas retas, na cor laranja os dados referentes ao modelo simulado apresentado na Tabela 3 com número de usuários 3, 7, 12, 17, 22, 27 e para

a comparação se tem na cor azul os resultados obtidos no modelo teórico através da Equação 10 apresentado na Tabela 4. O tamanho da fila definida é 9Mbps.

**Gráfico 6 – Simulação VoIP com fila de 9Mbps**




**Fonte: Autoria própria (2023)**

A reta com a simulação teve o valor de $\mathcal{R}^2$ igual a 99% que é o mesmo valor obtido no modelo teórico, isso mostra que a fila exclusiva para o tráfego VoIP tem uma relação direta entre a quantidade de usuários e o atraso máximo.

A Tabela 4 mostra o erro percentual na comparação entre o atraso máximo pelo número de usuários obtido através da Equação 10 do modelo teórico e o atraso máximo pelo número de usuários obtido através das simulações da Tabela 3.

**Tabela 4 – Erro percentual entre atraso máximo e número de usuários**

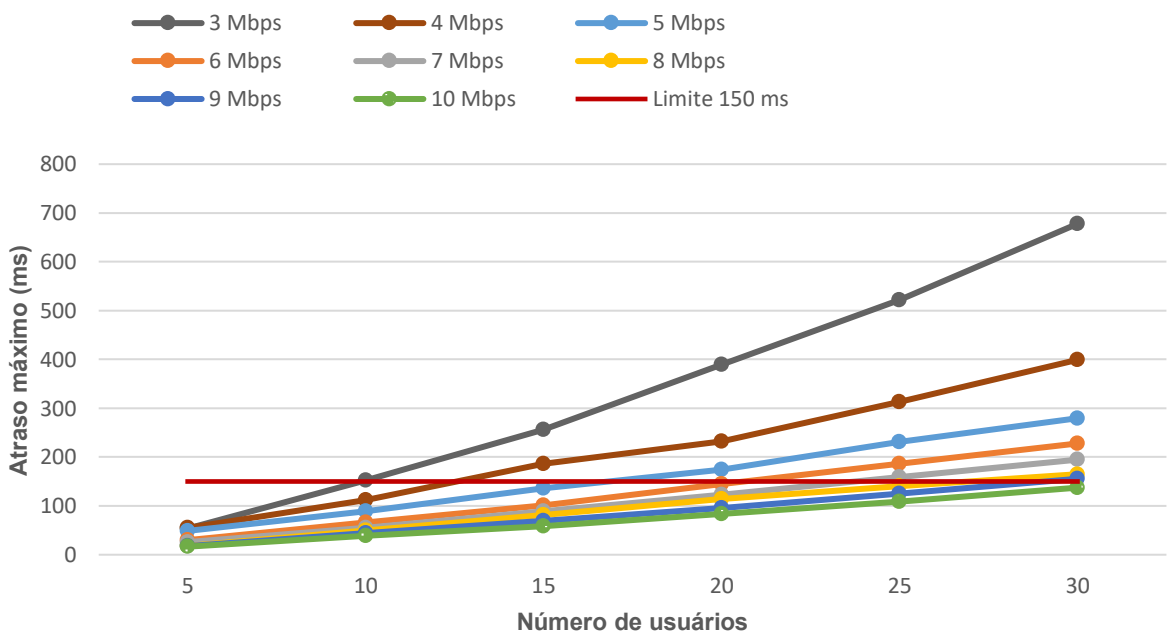
| Número de usuários | Modelo teórico (ms) | Simulado (ms) | Erro % |
|---|---|---|---|
| 3 | 5,078 | 5,913 | 16 |
| 7 | 26,463 | 24,138 | 11 |
| 12 | 53,493 | 44,083 | 19 |
| 17 | 80,524 | 71,302 | 13 |
| 22 | 107,554 | 100,850 | 7 |
| 27 | 134,585 | 118,426 | 13 |

**Fonte: Autoria própria (2023)**

O resultado do erro percentual aparentemente tem valores alto, isso se deve a quantidade de dados utilizados na elaboração da equação, para que o valor da porcentagem seja menor é necessário usar uma quantidade maior de dados.

A simulação ilustrada no Gráfico 7 mostra a correlação entre as 3 variáveis: tamanho da fila, atraso máximo e número de usuários, com o objetivo de verificar qual o tamanho da fila para que os 30 usuários não ultrapassem o limite de atraso máximo que é 150ms. Para isso, aumentou-se o tamanho da fila $x_2$ VoIP de 3Mbps até 10Mbps.

**Gráfico 7 – Simulação VoIP com fila de 3Mbps até 10Mbps**



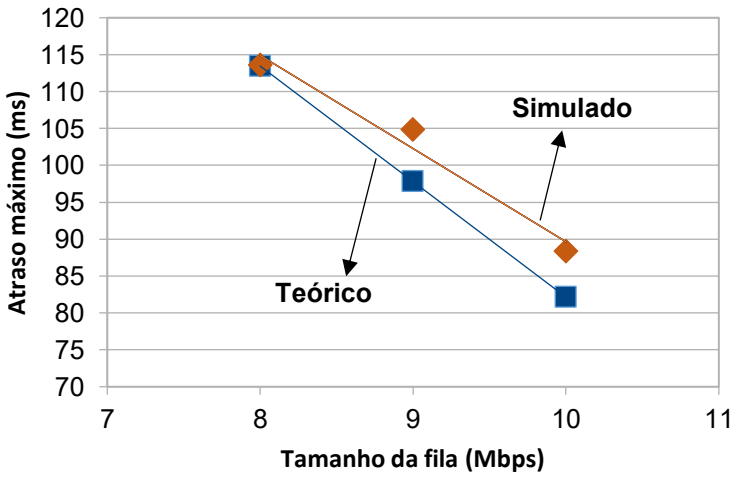

**Fonte: Autoria própria (2023)**

No Gráfico 7 se tem o limite de 150ms de atraso representado pela linha na cor vermelha, observa-se que os 30 usuários ficaram abaixo da linha com a fila em 10Mbps e pode ser visto que o atraso máximo diminui com o aumento do tamanho da fila.

Uma outra forma de visualizar os dados do Gráfico 7 é fixando a quantidade de usuários para verificar a correlação entre o atraso máximo e tamanho da fila. No Gráfico 8 tem-se duas retas para 22 usuários, na cor azul os resultados obtidos no modelo teórico através da Equação 8 apresentado na Tabela 4 e a cor laranja os

dados obtidos através das simulações apresentadas na Tabela 3. As duas retas são referentes a fila 8, 9 e 10Mbps.

**Gráfico 8 – Simulação VoIP com 20 usuários**




**Fonte: Autoria própria (2023)**

Observa-se neste resultado a correlação entre as duas variáveis, pois de acordo com a ilustração no gráfico, aumentando o tamanho da fila o atraso diminui. O valor do $\mathcal{R}^2$ obtido na reta decrescente do valor simulado é 97% e do modelo teórico é igual a 99%.

A Tabela 5 mostra o erro percentual na comparação entre o atraso máximo pelo tamanho da fila obtido através da Equação 8 do modelo teórico e o atraso máximo pelo tamanho da fila obtido através das simulações da Tabela 3.

**Tabela 5 – Erro percentual entre atraso máximo e tamanho da fila**

| Tamanho da fila (Mbps) | Modelo teórico (ms) | Simulado (ms) | Erro % |
|---|---|---|---|
| 8 | 113,454 | 113,612 | 1 |
| 9 | 97,797 | 104,850 | 7 |
| 10 | 82,140 | 88,361 | 7 |

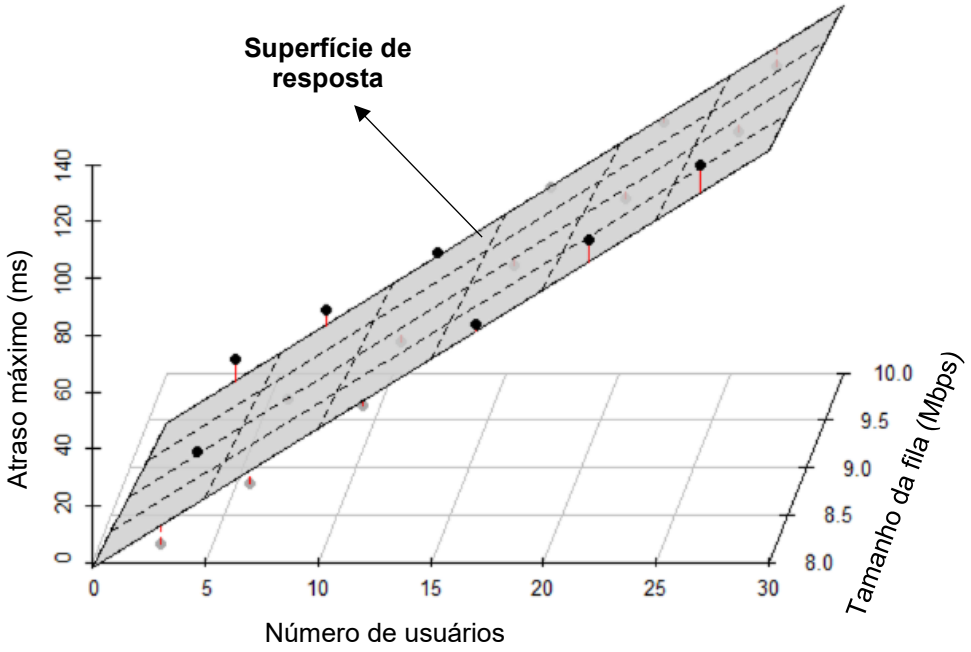
**Fonte: Autoria própria (2023)**

Para que o valor do erro percentual seja menor é necessário usar uma quantidade maior de dados para desenvolver a equação de regressão linear.

### 5.3.4 Previsão do tamanho da fila em função do atraso e número de usuários

O Gráfico 9 mostra a regressão linear múltipla para os valores da Tabela 3 e será comparado com o resultado obtido através da Equação 13 do modelo teórico e apresentado na Tabela 6.

**Gráfico 9 – Tamanho da fila em função do atraso e número de usuários**




**Fonte: Autoria própria (2023)**

A superfície de resposta é um hiperplano no espaço p-dimensional para as 3 variáveis obtido através da regressão linear múltipla. Tem-se ilustrado no gráfico os pontos na cor preto e cinza para os dados referente a Tabela 3, por exemplo para fila igual a 8Mbps e 3 usuários, o atraso é de 6,477ms e para 27 usuários, o atraso é de 139,856ms. Os valores do tamanho da fila em função do atraso máximo para 27 usuários e fila igual a 8Mbps, o atraso é de 139,856ms e para 10Mbps de fila o atraso é de 108.137ms.

A Tabela 6 mostra o erro percentual na comparação entre o tamanho da fila atraso máximo e número de usuários obtido através da Equação 13 do modelo teórico e os valores das simulações da Tabela 3.

**Tabela 6 – Erro percentual entre tamanho da fila, atraso máximo e número de usuários**

| Tamanho da fila (Mbps) | Número de usuários | Modelo teórico (ms) | Simulado (ms) | Erro % |
|---|---|---|---|---|
| 8 | 3 | 6,555 | 6,477 | 1 |
| 8 | 7 | 38,158 | 27,664 | 27 |
| 8 | 12 | 65,163 | 55,016 | 15 |
| 8 | 17 | 92,168 | 83,957 | 8 |
| 8 | 22 | 119,173 | 113,612 | 4 |
| 8 | 27 | 146,178 | 139,856 | 4 |
| 9 | 3 | 6,090 | 5,913 | 2 |
| 9 | 7 | 27,694 | 24,138 | 12 |
| 9 | 12 | 54,699 | 44,083 | 19 |
| 9 | 17 | 81,704 | 71,302 | 12 |
| 9 | 22 | 108,709 | 100,850 | 7 |
| 9 | 27 | 135,714 | 118,426 | 12 |
| 10 | 3 | 4,374 | 4,993 | 14 |
| 10 | 7 | 17,230 | 21,956 | 27 |
| 10 | 12 | 44,235 | 42,287 | 4 |
| 10 | 17 | 71,240 | 65,458 | 8 |
| 10 | 22 | 98,245 | 88,361 | 10 |
| 10 | 27 | 125,249 | 108,137 | 13 |

**Fonte: Autoria própria (2023)**

Pode-se observar também na Tabela 6 que é necessário usar uma quantidade maior de dados para que seja menor o erro percentual. Realizando uma análise de dados entre o Gráfico 3 e Gráfico 9, pode-se concluir através dos resultados que o tamanho da fila tem correlação com o atraso máximo e número de usuários, pois aumentando o tamanho da fila é possível limitar o atraso máximo e número de usuários, já o modelo teórico pode ser usado para estimar o tamanho da fila para garantir um atraso máximo para uma determinada quantidade de usuários.

A Tabela 7 mostra uma previsão de tamanho do enlace para quantidades específicas de telefones VoIP dentro de uma rede. Para calcular a previsão do enlace utilizou-se a Equação 14 com a qual é possível achar o tamanho da fila em função da quantidade de usuários e do atraso máximo que tem o limite de 150ms.

**Tabela 7 – Previsão de tamanho do enlace em uma rede**

| Telefones VoIP | Atraso máximo (ms) | Tamanho do enlace (Mbps) |
|---|---|---|
| 16 | 150 | 2 |
| 25 | 150 | 7 |
| 34 | 150 | 11 |
| 45 | 150 | 17 |
| 55 | 150 | 22 |
| 70 | 150 | 30 |

**Fonte: Autoria própria (2023)**

Calculou-se a previsão de tamanho do enlace para quantidades diferentes de telefones VoIP dentro de uma rede. Por exemplo, se em uma rede são necessários 34 telefones VoIP, o tamanho mínimo do enlace deve ser de 11Mbps para respeitar o limite de atraso máximo que é 150ms.

# 6 CONCLUSÃO

O objetivo deste trabalho foi desenvolver um modelo para determinar o tamanho da fila em função do número de usuários e do atraso máximo para conexões VoIP em uma topologia LAN utilizando rede SDN. A simulação foi realizada criando tráfegos simultâneos gerados pelo D-ITG entre TCP e VoIP com a fila FIFO padrão do *switch* e se utilizou regressão linear simples e múltiplas para verificar se existe relação entre as variáveis. O resultado desta simulação mostrou o problema, pois o $\mathcal{R}^2$ ficou em 1% e o atraso máximo em 900ms para tráfego VoIP e observou-se que os dados sofreram um atraso de enfileiramento, ou seja, atraso de fila.

Para resolver o problema foi criado uma fila exclusiva para o tráfego VoIP e para garantir o atraso máximo foi levado em consideração a largura de banda nas filas que é um mecanismo que fornece ao tráfego QoS dentro do OVS. O modelo de garantia usado neste trabalho foi o DiffServ, pois é uma proposta onde os pacotes são marcados de acordo com classes de serviços pré-determinados e é possível usar uma única fila para todos os fluxos.

Para criar o modelo teórico a simulação foi realizada com tamanhos de fila 8, 9 e 10Mbps para tráfegos exclusivo VoIP e de 5 até 30 a quantidade de usuários para obter o valor do atraso máximo. Com o resultado da simulação utilizou-se a regressão linear simples e múltipla para verificar a relação entre as variáveis e encontrar as equações de atraso máximo pelo número de usuários e pelo tamanho da fila, além da equação que é a principal contribuição deste trabalho para achar o tamanho da fila em função da quantidade de usuários e do atraso máximo.

Na simulação para verificar se as equações do modelo teórico correspondem com as simulações, foram utilizados também filas de 8, 9 e 10Mbps e o números de usuários igual a 3, 7, 12, 17, 22 e 27. A comparação do atraso máximo pelo número de usuários obtido através da equação do modelo teórico e o simulado tiveram o mesmo valor de $\mathcal{R}^2$ que foi 99%, a comparação do atraso máximo em função do tamanho da fila e o número de usuários fixando em 22 o resultado obtido do $\mathcal{R}^2$ para o modelo teórico foi igual a 99% e para o simulado igual a 97%. Também foi realizado a comparação entre o modelo teórico e simulado para o tamanho da fila em função do atraso máximo e número de usuários, com a análise dos resultados foi concluído que o tamanho da fila tem correlação com o atraso máximo e número de usuários, pois

aumentando o tamanho da fila é possível limitar o atraso máximo e número de usuários. Também foi realizada a simulação com número diferente de *switches* e o resultado mostra que aumentando a quantidade de *switches* na rede não altera o atraso máximo.

A previsão de tamanho do enlace foi calculada para quantidades diferentes de telefones VoIP dentro de uma rede respeitando o limite de atraso máximo que é 150ms, com isso, concluiu-se que a equação para achar o tamanho da fila em função da quantidade de usuários e do atraso máximo é a principal contribuição deste trabalho.

## 6.1 Trabalhos futuros

Como trabalhos futuros foram considerados alguns assuntos interessantes que foram surgindo no decorrer dos inúmeros testes realizados neste modelo e que seriam uma extensão deste trabalho.

Os principais pontos são relacionados a seguir:

- Elaborar a equação da reta de melhor ajuste com maior quantidade de dados;
- Estimar o tamanho da fila para outros tipos de tráfego, por exemplo para tráfegos de vídeo chamada que são muito utilizados hoje em dia;
- Realizar a medição de outras métricas como MOS, perda de pacotes, *jitter* e vazão.

# REFERÊNCIAS

ALALAWI, K.; AL-AQRABI, H. Quality of service evaluation of voip over wireless networks. In: **2015 IEEE 8th GCC Conference Exhibition**. [S.l.: s.n.], 2015. p. 1–6.

ALMEIDA, A. B. **Medição de qualidade de voz em Wireless utilizando Codecs G711, G729, G723 e GSM**. Dissertação de Mestrado - PUC-Campinas, Campinas, 2008.

ANDRIOLI, L. *et al.* **Analisando métodos e oportunidades em redes definidas por software (SDN) para otimização de tráfego de dados.** Revista Brasileira de Computação Aplicada, Passo Fundo, v. 9, n. 4, p. 2-14, 2017.

ANTUNES, R. S. P. **Modelo de Regressão Linear e suas Aplicações.** 2012. Dissertação (Mestrado em Ensino de Matemática) – Universidade da Beira Interior, Covilhã, 2012.

B. HUBERT, **Ubuntu manpage: tc - show / manipulate traffic control settings,** Disponível em: <http://manpages.ubuntu.com/manpages/xenial/man8/tc.8.html>. Acesso em: 23 fev. 2022.

BLACK, D.; JONES, P. Differentiated Services (DiffServ) and Real-time Communication draft-ietf-dart-dscp-rtp-03. **Internet Engineering Task Force (IETF).** United States, 2015.

BRADEN, R.; CLARK, D.; SHENKER, S. Integrated Services in the Internet Architecture: na Overview. **Network working Group.** United States, 1994.

BRAUN, W.; MENTH, M. **Software-Defined Networking Using OpenFlow: Protocols, Applications and Architectural Design Choices.** Future Internet. p. 302-336. v. 6. Jan 2014.

BERNAL, P. S. M. **Voz sobre Protocolo IP- A Nova Realidade da Telefonia**. Editora Érica. São Paulo. 1ª Edição. 2007.

BOLEY, J. M. *et al.* Adaptive QoS for Data Transfers using Software-Defined Networking. In: 2016 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), 2016, Bangalore, India. **Anais [...]** Bangalore: IEEE, 2016.

CARDOSO, R. F. F. **Uma abordagem SDN para o controlo e admissão de tráfego.** 2015. Dissertação (Mestrado em Engenharia Eletrotécnica e de Computadores) – Universidade Nova de Lisboa, Lisboa, 2015.

COSTA, L. R. *et al.* Asterisk pbx capacity evaluation. **In: IEEE International Parallel and Distributed Processing Symposium Workshop,** p. 519–524, 2015.

CUNHA, D. C. P. **Proposta de suporte à qualidade de serviço baseado em SDN: um estudo de caso para instituições de ensino federal.** 2017. Dissertação (Mestrado em Ciências da Computação) – Universidade Federal de Pernambuco, Recife, 2017.

DEVERA, M., **HTB – Hierarchy Token Bucket** Disponível em: < https://linux.die.net/man/8/tc-htb>. Acesso em: 15 fev. 2022.

D-ITG, **Manual oficial D-ITG**. Disponível em: < http://traffic.comics.unina.it/software/ITG/manual/>. Acesso em: 10 fev. 2022.

FOROUZAN, B. A. **Data Communications and Networking.** 5. ed. New York: McGraw-Hill Higher Education, 2012.

HASROUTY, C. A. *et al.* Sdn-driven multicast streams with adaptive bitrates for voip conferences. **In: IEEE International Conference on Communications (ICC),** p. 1–7. 2016.

HU, F.; HAO, Q; BAO, K. A Survey on Software-Defined Network and OpenFlow: From Concept to Implementation. **IEEE Communication Surveys & Tutorials.** v. 16, n. 4, 2014.

ITU-T, **International Telecommunication Union**. Disponível em: <https://www.itu.int/rec/T-REC-G/en>. Acesso em: 23 ago. 2022.

KARAKUS, M.; DURRESI, A. Quality of Service (QoS) in Software Defined Networking (SDN): A survey. **Journal of Network and Computer Applications.** v. 80, p. 200-218, 2017.

KARAPANTAZIS, S.; PAVLIDOU, F. N. VoIP: A comprehensive survey on a promising technology. **Computer Networks**, Elsevier B.V., p. 2050-2090. v. 53. n. 12., 2009.

KREUTZ, D. *et al.* Software-Defined Networking: A Comprehensive Survey**. IEEE Electrical Electronics Engineers Inc**, v. 103, p. 14-76 JAN-2014.

KRISHNA, H. **Providing End-to-end Bandwidth Guarantees with OpenFlow.** 2016. Thesis (Master of Science) – Delft University of Technology, Países Baixos, 2016.

KUROSE, J.; ROSS, K. **Redes de computadores e a internet uma abordagem top-down.** 6. ed. São Paulo: Pearson Education do Brasil, 2013.

KWON, D. *et al.* Qoe-based adaptive mvoip service architecture in sdn networks. **The Thirteenth International Conference on Networks.** p. 73. 2014.

LARA, A.; KOLASANI, A.; RAMAMURTHY, B. Network Innovation using OpenFlow: A Survey**. IEEE Communications Surveys & Tutorials.** p. 493-512. v. 16. Aug 2014.

LI, W; MENG, W.; KWOK, L. F. A survey on OpenFlow-based Software Defined Networks: Security challenges and countermeasures. **Journal of Network and Computer Applications.** p. 126-139. v. 68. Jun 2016.

MCKEOWN, N. *et al.* Openflow: Enabling innovation in campus networks. **ACM SIGCOMM Computer Communication Review.** Abr 2008.

MARIBONDO, P. D. S.; FERNANDES, N. C. Avoiding voice traffic degradation in ip enterprise networks using caos. **In: Proceedings of the 2016 Workshop on Fostering Latin-American Research in Data Communication Networks.** p. 34–36. 2016.

MEHTA, P.; UDANI, S. Voice over ip. **IEEE Potentials**, [New York, NY]: Institute of Electrical and Electronics Engineers, p. 36-40. v. 20. n. 4. Oct 2001.

MININET, An Instant **Virtual Network on your Laptop (or other PC).** Disponível em: <https://www.mininet.org>. Acesso em: 03 fev. 2022.

MONKS, E. M. **Planejamento de Capacidade em Redes Corporativas para implementação de Serviços VoIP.** Dissertação de Mestrado – UFRGS, Porto Alegre, 2006.

NACIMENTO, K. **Marcação de Tráfego em redes DiffServ.** 2004. Dissertacão (Mestrado em Engenharia Elétrica) - Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, 2004.

ONF, **Open Networking Foundation** – Version 1.3.5., 2015. Disponível em: < https://opennetworking.org/wp-content/uploads/2013/07/OpenFlow1.3.4TestSpecification-Basic.pdf>. Acesso em: 10 fev. 2022.

OPENDAYLIGHT. **ODL**. Disponível em: <https://www.opendaylight.org/category/announcement>. Acesso em: 02 fev. 2022.

ORADE. **Codecs**. Disponível em: < https://orade.com/los-codecs-en-la-telefonia-ip-que-son-y-como-funcionan/>. Acesso em: 10 dez. 2022.

PRADEEP, J. **End-to-end Quality-of-Service in Software Defined Networking.** 2017. Dissertation (Master of Science in Computer Science) – University of Dublin, Trinity College, Dublin, 2017.

QUININO, Roberto C.; REIS, Edna A.; BESSEGATO, Lupércio, F. **O coeficiente de determinação r² como instrumento didático para avaliar a utilidade de um modelo de regressão linear múltipla.** Belo Horizonte: UFMG, 2011. Disponível em: 96 < http://www.est.ufmg.br/portal/arquivos/rts/PD_28102011_Final.pdf>. Acesso em: 8 ago. 2022.

SCHULZRINNE, H. *et al.* **Version 2 of the protocol operations for the simple network management protocol (SNMP)**. New York, NY, USA, 2003. 1-104 p. Disponível em: <https://www.rfc-editor.org/rfc/rfc3550.txt>. Acesso em: 20 ago. 2022.

SINGH, H. P. *et al.* Voip: State of art for global connectivity—a critical review. **Journal of Network and Computer Applications**, v. 37, p. 365 – 379, 2014. ISSN 1084-8045.

TANENBAUM, A. S.; WETHERALL, D. **Redes de Computadores.** 5. ed. São Paulo: Pearson Universidades, 2011.

THORPE, C. *et al.* imos: Enabling voip qos monitoring at intermediate nodes in an openflow sdn. In: **IEEE International Conference on Cloud Engineering Workshop (IC2EW),** 2016. p. 76–81. 2016.

VOICE OVER IP, Per Call Bandwidth (2016), **VoIP**. Disponível em: <https://www.cisco.com/c/en/us/support/docs/voice/voice-quality/7934-bwidth-consume.pdf>. Acesso em: 29 ago. 2022.