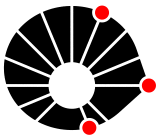

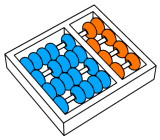

Implementação de algoritmos de adaptação de taxa de rede para transmissão de videos 360º

Diego S. Martines *César A. V. Melo* *Leonardo C. V. Filho*
Nelson L. S. da Fonseca

Relatório Técnico - IC-PFG-22-48
Projeto Final de Graduação
2022 - Novembro

UNIVERSIDADE ESTADUAL DE CAMPINAS
INSTITUTO DE COMPUTAÇÃO

Implementação de um Cliente Go/QUIC para Acesso a Vídeo 360º com Adaptação de Taxa de Bits

Diego Sepulveda Martines Leonardo Correzzola Villani Filho
César A. V. Melo Nelson L. S. da Fonseca

Resumo

Este trabalho apresenta a implementação de um servidor e cliente utilizando o protocolo HTTP/QUIC para a transmissão de vídeos 360º. Desenvolvemos algoritmos de adaptação de taxa de rede, avaliando seu desempenho com o objetivo de otimizar tanto a qualidade de transmissão quanto a eficiência no uso de recursos de rede. A proposta contribui para o aprimoramento das técnicas de transmissão de vídeos imersivos, explorando soluções que se ajustem dinamicamente às variações nas condições de rede.

Além disso, utilizamos a divisão dos vídeos em tiles, permitindo a transmissão e renderização seletiva das áreas de maior interesse do usuário, o que permite reduzir o consumo de largura de banda e melhora a experiência de visualização, mesmo em cenários de conexão instável. Isso resulta em um sistema de streaming menos suscetível a flutuações na qualidade da conexão com a internet.

1 Introdução

Nos últimos anos, o uso de vídeos 360º tem crescido exponencialmente, impulsionado pelo aumento da demanda por experiências imersivas em diversas áreas, como entretenimento, educação e turismo virtual. Esses vídeos capturam uma visão esférica completa de uma cena, permitindo que o usuário interaja e mude seu ponto de vista ao mover sua cabeça ou dispositivo. No entanto, a transmissão de vídeos 360º apresenta desafios significativos, como o alto consumo de largura de banda e a alta latência. Para mitigar esses desafios, uma técnica eficiente é a divisão do vídeo 360º em *tiles* ou esquadrilhas, que podem ser transmitidos e renderizados de forma independente, de acordo com a área de interesse do usuário. Essa abordagem não só economiza largura de banda, mas também garante uma melhor qualidade de vídeo nas regiões de maior interesse. Além disso, a implementação de algoritmos de adaptação de taxa de bits é crucial para melhorar a experiência do usuário. Esses algoritmos ajustam a qualidade do vídeo com base na largura de banda disponível e nas condições da rede, otimizando a visualização ao maximizar a qualidade do vídeo e minimizar interrupções e flutuações de qualidade. Especificamente, o algoritmo leva em consideração o campo de visão (Field of View - FoV) do usuário, determinando taxas de bits mais altas para as áreas do vídeo que estão sendo visualizadas e taxas mais baixas para as áreas que não estão sendo exibidas no momento. Isso permite uma utilização mais eficiente dos recursos de rede, garantindo uma alta qualidade de imagem apenas onde é necessário. Neste trabalho

implementamos um servidor e um cliente utilizando o protocolo HTTP/QUIC [2] para a transmissão de vídeos 360º. O objetivo do trabalho é avaliar a performance de diferentes algoritmos de adaptação de taxa de rede, utilizando um modelo eficaz de previsão do campo de visão para reduzir o consumo de dados e tornar o sistema de download menos vulnerável às instabilidades da conexão com a internet. Espera-se que este estudo contribua para o avanço das tecnologias de transmissão de vídeos 360º, oferecendo insights sobre a eficácia de diferentes abordagens de adaptação de taxa de bits e seu impacto na experiência do usuário final. A pesquisa poderá proporcionar um entendimento mais profundo sobre como otimizar a transmissão de vídeos imersivos, beneficiando tanto desenvolvedores quanto consumidores de conteúdo 360º.

Neste documento apresentamos o funcionamento e as vantagens das ferramentas utilizadas no estudo na seção 2. Também apresentamos o algoritmo de adaptação de rede utilizado na seção 3 e discutimos seu funcionamento. Além disso também apresentamos e analisamos os resultados obtidos com a transmissão de vídeos em tiles com este algoritmo (seção 5). Na seção 6 discutimos o que é possível concluir à partir dos resultados obtidos, as implicações destes resultados, as limitações do estudo e sugestões para trabalhos futuros.

2 Fundamentação teórica

Esta seção discute os conceitos básicos sobre vídeos 360º, processamento de vídeo com tiles, o protocolo QUIC e a ferramenta de emulação de rede Mininet. A seguir, detalhamos cada um desses tópicos fundamentais para o contexto deste estudo.

2.1 Vídeo 360

Os vídeos 360° estão ganhando destaque como parte essencial das aplicações multimídia, impulsionados pela popularidade crescente da Realidade Virtual (RV) imersiva. Essa tecnologia permite capturar uma visão esférica completa de uma cena, possibilitando ao usuário interagir e explorar o ambiente virtualmente movendo a cabeça ou o dispositivo. Comparados aos vídeos tradicionais, os vídeos 360° oferecem uma experiência mais imersiva e interativa, abrindo novas possibilidades em áreas como entretenimento, educação, telepresença e mais.

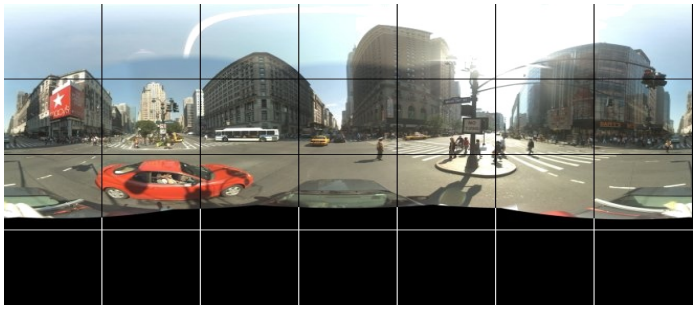

Figura 1: Imagem 360º dividida em 28 tiles

A transmissão de vídeos 360º nos trás vários desafios, como o alto consumo de largura de banda e baixa latência. Uma forma de lidar com esses desafios é dividir o vídeo 360º em tiles, que são segmentos no formato de regiões retangulares menores que cobrem uma parte do frame do vídeo. Além disso, podemos agrupar tiles sequenciais que correspondem à mesma região do vídeo mas de diferentes frames em segmentos. Cada segmento ou tile pode ser transmitida independentemente, com diferentes níveis de bitrate e resolução dependendo da área de interesse do usuário. Dessa forma, é possível que apenas os tiles que são visíveis para o usuário sejam transmitidos e renderizados em melhor qualidade, economizando em largura de banda e garantindo qualidade de vídeo. A codificação eficiente de vídeos 360° é crucial para a transmissão através de redes com capacidade limitada.

2.2 Processamento de vídeo 360 e tiling

Para que possamos transmitir o vídeo de forma adaptativa processamos o video em quatro etapas. Este processo facilita o streaming eficiente de vídeos, criando tiles que podem ser transmitidos de forma adaptativa, garantindo uma melhor experiência do usuário final, especialmente em condições variáveis de rede.

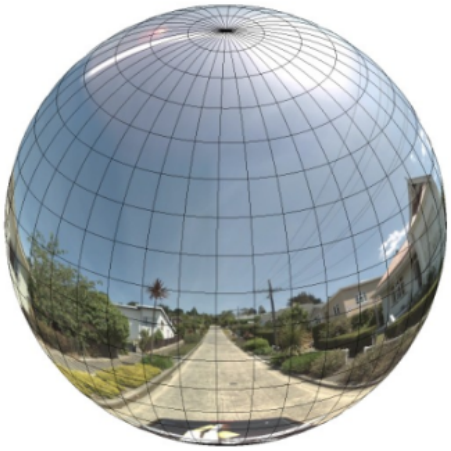

Figura 2: Exemplo de Imagem 360º dividida em Tiles

Inicialmente, o vídeo é convertido para um formato bruto YUV usando a ferramenta ffmpeg. Nesse passo o arquivo de vídeo MP4 é transformado em um arquivo YUV. Esse formato é escolhido por ser um formato de vídeo bruto, sem compressão, contendo dados de pixel que serão processados em etapas subsequentes. No segundo passo, o vídeo YUV é codificado utilizando a ferramenta kvazaar, que é um codificador HEVC (H.265). neste passo dividimos o vídeo em 10x10 tiles, o que significa que o vídeo é segmentado em blocos menores que podem ser decodificados independentemente. O vídeo YUV é assim codificado em um arquivo HEVC, com cada tile tratado como uma fatia separada. Esse passo é crucial para permitir a decodificação independente de cada tile. No terceiro passo, o vídeo codificado em HEVC é empacotado em um contêiner MP4 utilizando a ferramenta

MP4Box. Este passo envolve a criação de um contêiner MP4 que mantém a estrutura de tiles estabelecida anteriormente, facilitando a gestão dos blocos de vídeo no novo formato. No quarto passo, o arquivo MP4 é segmentado para criar uma apresentação MPEG-DASH. Utilizando a ferramenta MP4Box o vídeo contido no arquivo MP4 é dividido em múltiplos pequenos segmentos de vídeo, cada um representando uma parte do vídeo original. Além disso, é gerado um arquivo MPD (Media Presentation Description) que é um arquivo XML descrevendo a estrutura dos segmentos, incluindo informações como URLs, durações e dados de adaptação de taxa de bits. Esses segmentos de vídeo, geralmente nomeados sequencialmente, são os arquivos que permitem o streaming adaptativo, pois o player pode selecionar e baixar segmentos específicos conforme necessário, baseando-se nas condições da rede e nas capacidades do dispositivo.

2.3 Protocolo Quic

O protocolo QUIC (Quick UDP Internet Connections) foi desenvolvido pelo Google como uma alternativa mais rápida e eficiente ao tradicional TCP (Transmission Control Protocol). O QUIC funciona sobre o UDP (User Datagram Protocol), permitindo a redução do tempo de estabelecimento de conexão e oferecendo vantagens em termos de latência e recuperação de erro. Ao contrário do TCP, que requer múltiplas RTT para estabelecer uma conexão e negociar parâmetros de segurança, o QUIC combina essas etapas em um único handshake, reduzindo a latência da conexão [5] como observamos na figura 3. Isso é especialmente útil para aplicações que exigem rápidas interações, como a transmissão de vídeos 360º.

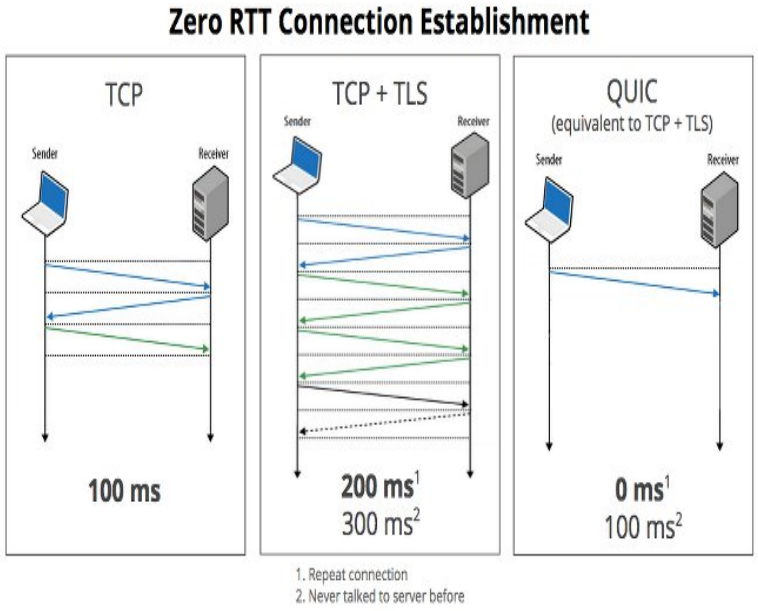

Figura 3: Comparação entre protocolos TCP e QUIC [5]

O QUIC também dá suporte para a multiplexação de múltiplos fluxos sobre uma única

conexão, evitando o head-of-line blocking (ocorrência em que o desempenho de uma conexão é prejudicado devido à espera por um único pacote ou recurso específico que está bloqueando a entrega de outros pacotes) e melhorando a vazão como observamos na figura 4.

Além disso, o QUIC incorpora a segurança nativa através do uso do protocolo TLS (Transport Layer Security) 1.3, eliminando a necessidade de uma configuração separada para a criptografia e garantindo que todos os dados transmitidos sejam seguros por padrão. O protocolo foi projetado para ser facilmente extensível, permitindo a adição de novas funcionalidades sem comprometer a compatibilidade com versões anteriores, facilitando a adaptação do QUIC para diferentes tipos de aplicações e condições de rede.

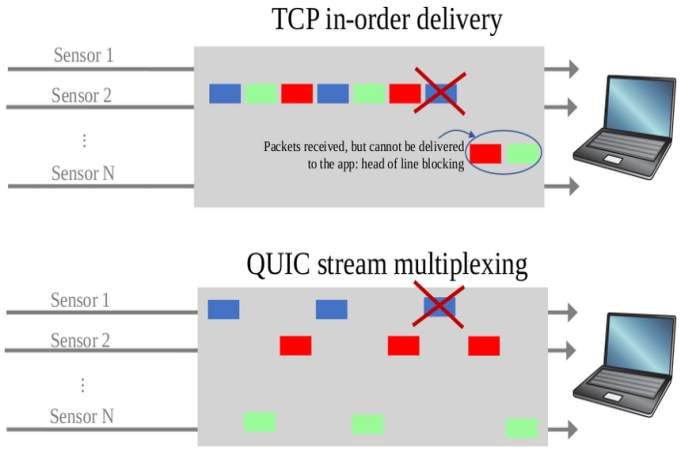

Figura 4: Head-of-line blocking

2.4 Mininet

Mininet é uma ferramenta de emulação de rede que permite a criação e teste de topologias de rede complexas em um único computador. Ele é amplamente utilizado na pesquisa e desenvolvimento de redes devido à sua flexibilidade e capacidade de simular cenários de rede realistas. O Mininet permite a criação de topologias de rede que replicam redes reais, incluindo switches, hosts e links, variando desde simples redes de dois nós até complexas infraestruturas de data centers. Isso nos permite testar como diferentes configurações de rede afetam a performance de sistemas de transmissão de vídeos 360º. Com Mininet, é possível emular diferentes condições de rede, como variações de largura de banda, latência e perda de pacotes. Isso permite avaliar a eficácia e eficiência dos algoritmos de adaptação de taxa de bits sob diversas circunstâncias, nos possibilitando testar o sistema em situações reais. Mininet também suporta a execução de pilhas de protocolo de rede reais, como o TCP/IP e o QUIC, permitindo testes precisos de desempenho e comportamento, facilitando a implementação e teste de novos protocolos ou a modificação de protocolos existentes. A ferramenta oferece suporte para scripts de automação, permitindo a criação e modificação de topologias de rede de forma programática. Mininet é projetado para ser fácil de instalar e usar, com uma interface de linha de comando intuitiva e um conjunto de ferramentas

abrangentes para a gestão de redes emuladas, podendo ser executado em qualquer máquina Linux, incluindo máquinas virtuais, tornando-o acessível para uma ampla gama de usuários. Neste trabalho utilizamos a combinação do protocolo QUIC com a emulação de rede do Mininet para criar um ambiente de teste robusto para a transmissão de vídeos 360º. O QUIC proporciona uma transmissão eficiente e segura, enquanto o Mininet permite a emulação de condições de rede variadas e realistas. Juntos, esses componentes fornecem uma plataforma poderosa para avaliar a eficácia dos algoritmos de adaptação de taxa de bits e prever o campo de visão, contribuindo para o desenvolvimento de tecnologias avançadas de transmissão de vídeos imersivos.

3 Algoritmos de adaptação de taxa

Esta seção apresenta o algoritmo de adaptação de taxa utilizados para otimizar a qualidade da transmissão de vídeo 360º. Discutimos a implementação do algoritmo baseado na vazão (throughput) e na detecção de perdas, bem como seu impacto nos cenários testados.

A escolha do algoritmo de adaptação de taxa tem um impacto significativo nos resultados da transmissão de vídeo. A decisão do algoritmo afeta a qualidade do vídeo, a estabilidade da conexão e a experiência do usuário. Neste trabalho, nos concentramos em um algoritmo baseado na vazão do canal, que ajusta a qualidade do vídeo com base na largura de banda disponível e nas condições da rede. O contexto em que o algoritmo afeta os resultados é em um ambiente de tempo real, onde a decisão deve ser tomada rapidamente para evitar problemas de bufferização e flutuações de qualidade. A escala de tempo para a tomada de decisão é de segundos, onde o algoritmo precisa avaliar as condições da rede e ajustar a qualidade do vídeo em tempo hábil.

A inspiração para o algoritmo de adaptação de taxa veio da necessidade de melhorar a experiência do usuário em aplicações de vídeo em tempo real. O algoritmo utiliza dados da vazão do canal para ajustar a qualidade do vídeo, o que permite uma adaptação mais precisa às condições da rede. A lógica do algoritmo é baseada em quatro fatores principais: prioridade do pacote, vazão instantânea (InstantFlow), vazão média (AverageFlow) e detecção de perdas (LossFlow). Com base nesses fatores, o algoritmo ajusta dinamicamente o bitrate dos pacotes de vídeo para otimizar a qualidade do serviço. A lógica básica do algoritmo pode ser descrita da seguinte forma (Figura 5):

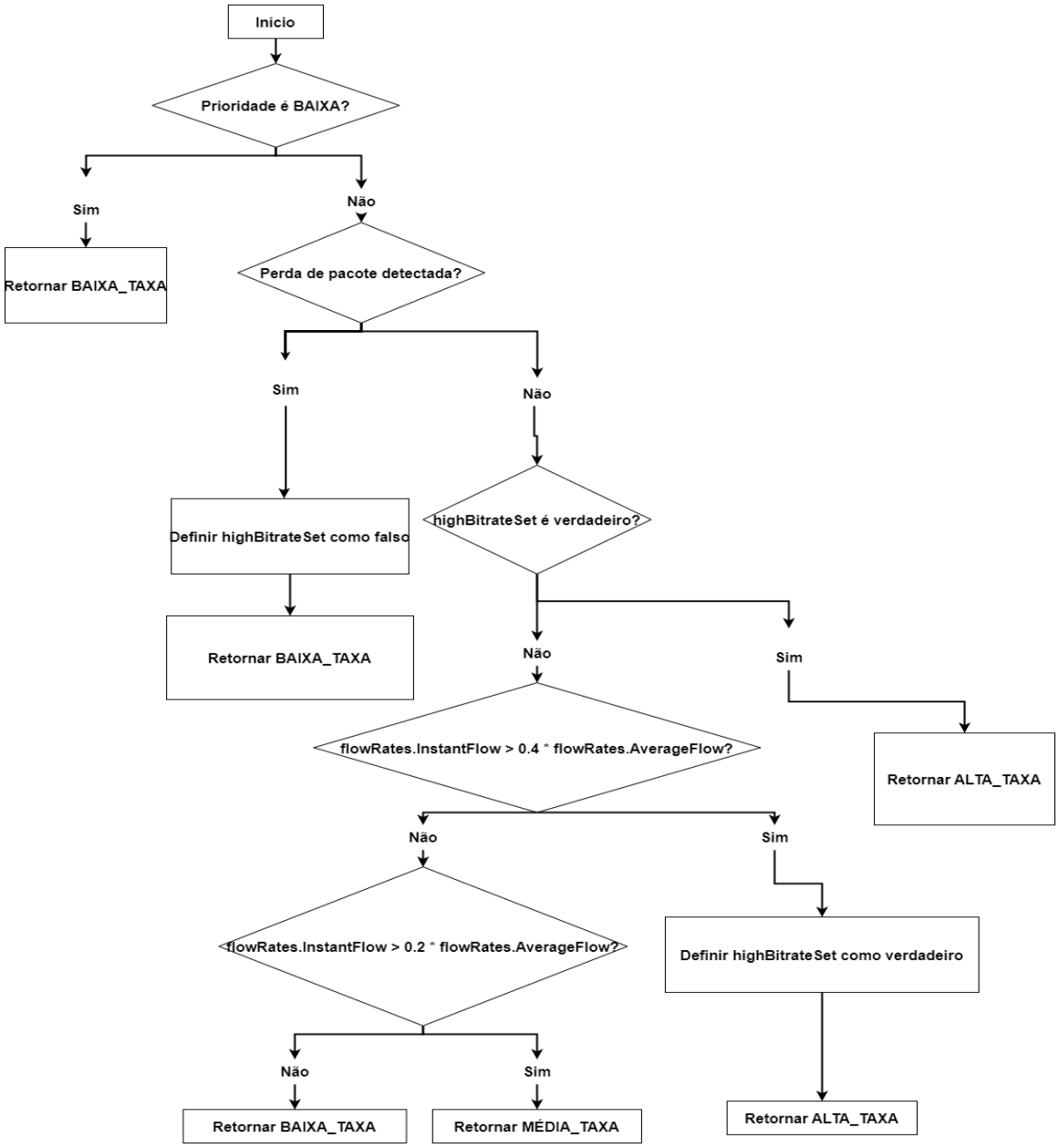

Figura 5: diagrama do algoritmo de adaptação

Para detecção de perdas, se LossFlow é verdadeiro (indicando perda de pacotes), o bitrate é ajustado para o mais baixo possível para mitigar a perda. Para pacotes de alta prioridade, se InstantFlow é maior que 60% de AverageFlow, o bitrate é ajustado para o mais alto. Se InstantFlow está entre 40% e 60% de AverageFlow, o bitrate ainda é alto. Se InstantFlow está entre 20% e 40% de AverageFlow, o bitrate é ajustado para o valor médio. Se InstantFlow é menor que 20% de AverageFlow, o bitrate é ajustado para o valor

mais baixo. Para pacotes de baixa prioridade, o bitrate é sempre ajustado para o valor mais baixo, priorizando a alocação de recursos para pacotes mais críticos.

No desenvolvimento do algoritmo, nos concentramos em um algoritmo que ajusta a qualidade do vídeo em um nível de segmento ou tile. Isso significa que o algoritmo avalia as condições da rede e ajusta a qualidade do vídeo para cada segmento ou tile do vídeo, em vez de ajustar a qualidade do vídeo como um todo. Além disso, o servidor utilizado na parte experimental é capaz de ajustar a qualidade do vídeo com base nas condições da rede. O servidor é equipado com um algoritmo de adaptação de taxa que utiliza dados da vazão do canal para ajustar a qualidade do vídeo em tempo real. Isso permite que o servidor forneça uma experiência de usuário mais estável e de alta qualidade, mesmo em condições de rede adversas.

4 Configuração Experimental

Nesta seção, apresentamos a arquitetura do cliente e do servidor implementados para a realização do estudo experimental, bem como os detalhes da simulação de rede conduzida neste trabalho.

4.1 Arquitetura do Cliente

A Figura 6 ilustra os componentes do cliente implementado. O cliente estabelece múltiplas streams com o servidor para enviar requisições de forma concorrente. As requisições são geradas e armazenadas em um buffer até que este atinja um nível de ocupação predefinido. Em seguida, o cliente envia as requisições em paralelo através das streams. As respostas recebidas são armazenadas em um buffer de respostas, e o cliente calcula a vazão média e instantânea da conexão. Com base nesses valores, o algoritmo de adaptação ajusta a taxa de bits para as próximas requisições. As novas requisições são enviadas ao servidor de acordo com a taxa de ocupação dos buffers.

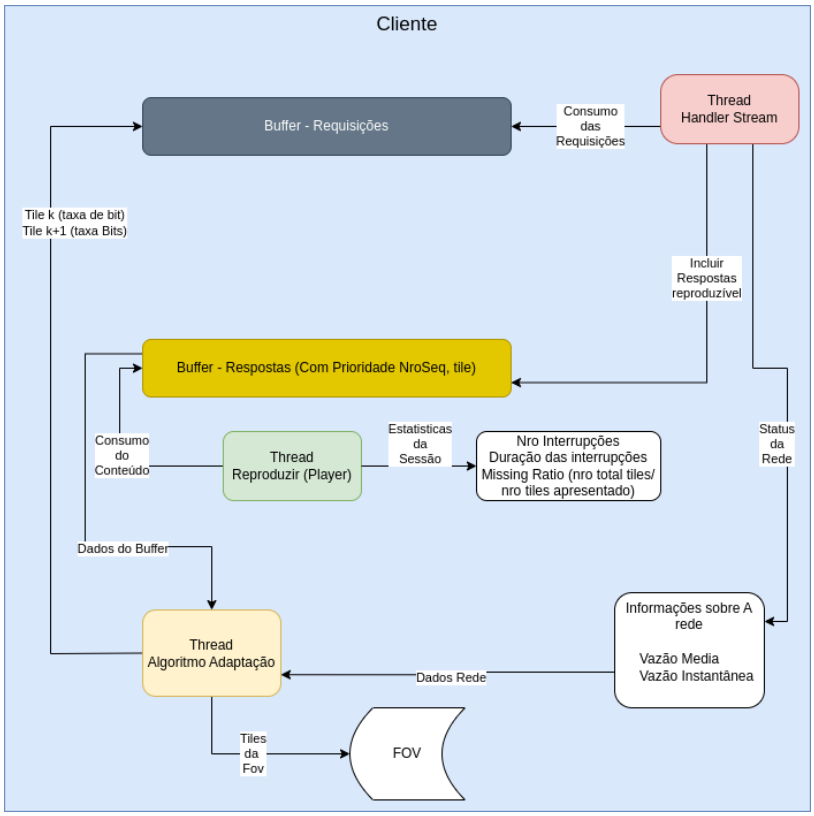

Figura 6: Diagrama apresentando a arquitetura do cliente

4.2 Arquitetura do Servidor

O servidor foi implementado utilizando a linguagem de programação Go e a biblioteca ‘quic-go‘. A Figura 7 apresenta a arquitetura do servidor. Ele utiliza um modelo de thread por conexão, onde cada conexão possui um thread dedicado para lidar com as requisições recebidas assincronamente. O thread principal do servidor é responsável por escutar novas conexões. Ao receber uma nova conexão, o servidor cria dois novos threads: um para aceitar a conexão e outro para gerenciar as streams e requisições associadas àquela conexão. O thread que aceita a conexão, por sua vez, cria um novo thread para cada stream recebida. Cada thread de stream é responsável por receber as requisições e enfileirá-las para serem processadas pelo thread que gerencia as requisições [7].

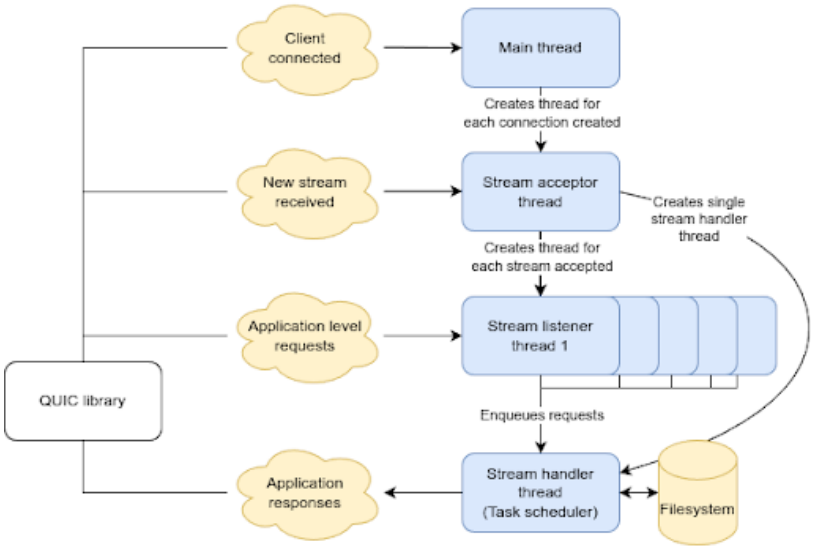

Figura 7: Arquitetura do Servidor

4.3 Simulação de Rede com Mininet

Para simular o ambiente de rede, utilizamos a ferramenta Mininet [3], que permite a criação de topologias de rede virtuais com múltiplos hosts e switches, com controle preciso sobre os parâmetros de link, como largura de banda, atraso e perda de pacotes.

A topologia de rede utilizada em nossas simulações, ilustrada na Figura 8, consiste em dois hosts, representando o cliente e o servidor, conectados por meio de um switch. Embora simplificada, essa topologia captura os principais aspectos da comunicação cliente-servidor e permite a avaliação do impacto dos parâmetros de rede no desempenho das políticas de escalonamento.

Para simular a presença de tráfego de outras aplicações na rede, utilizamos a ferramenta iPerf [4]. O iPerf gera tráfego UDP com taxa de pacotes constante (CPR) e largura de banda configurável, permitindo simular diferentes níveis de congestionamento na rede. A execução do iPerf é integrada aos scripts de teste da simulação através da API Python do Mininet, garantindo que o tráfego de fundo seja iniciado e interrompido de forma consistente em cada simulação.

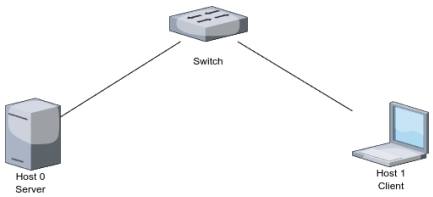

Figura 8: Topologia de Rede

5 Resultados numéricos

Esta seção analisa os resultados obtidos nos seis cenários distintos, avaliando o desempenho do algoritmo de adaptação de taxa em diferentes condições de carga, largura de banda e atraso, com foco na priorização do campo de visão (prioridade 0) em transmissões de vídeo 360º. Para validar a eficácia do algoritmo de adaptação de taxa e escolha de bitrate para vídeos 360º, foram selecionados seis cenários distintos, variando em carga de rede (background load) e atraso (delay). Isso permitirá uma análise abrangente do desempenho do algoritmo em diferentes condições. Os critérios para escolha dos cenários foram os seguintes: Dois níveis de carga de rede: baixa (10Mb) e alta (30Mb), para avaliar o comportamento do algoritmo em diferentes demandas de largura de banda.

Três níveis de atraso na rede: 10ms, 16ms e 24ms, representando uma variedade de condições de latência encontradas em ambientes diversos.

Tabela 1: Cenarios do experimento

Cenario	**Background load**	**Delay**
#1	10Mb	24 ms
#2	30Mb	24 ms
#3	10Mb	16 ms
#4	30Mb	16 ms
#5	10Mb	10 ms
#6	30Mb	10 ms

No cenário 1, cenário com alto delay e baixa carga, representado pela figura 9, observamos uma estabilidade na alta prioridade e alta oscilação entre os pacotes de baixa prioridade. A diferença na estabilidade ocorre porque no algoritmo há priorização do envio de tiles de alta prioridade e a oscilação se deve ao fato da alta sensibilidade do algoritmo à perda de pacotes.

No cenário 2, cenário com alto delay e alta carga, representado pela figura 10, observamos um comporamento similar ao cenário 1 mas com uma oscilação maior na transmissão de alta prioridade, transmissão esta que o algoritmo consegue recuperar e estabilizar rapidamente. A diferença se deve à ocupação e carga da rede, que com maior ocupação permite uma transmissão de um menor número de tiles por vez. Neste cenário também observamos que a resposta do algoritmo para o cenário de alta prioridade é mais satisfatória do que para baixa prioridade, pois o algoritmo altera somente o valor de bitrate da alta prioridade, permitindo assim a adaptação da transmissão destas tiles.

No cenário 5, cenário com baixo delay e baixa carga de rede, representado pela figura 13, observamos estabilidade na transmissões de tiles de diferentes prioridades. Com baixa perda de pacotes temos uma menor oscilação da qualidade das tiles enviadas, desta forma a qualidade das tiles se mantém alta. Neste cenário tambem observamos que a ação imediata na alteração da qualidade dos tiles de alta prioridade afeta os tiles de outras prioridades. Abaixar a qualidade dos tiles de alta prioridade aumenta o espaço para as tiles de baixa prioridade de segmentos seguintes.

No cenário 6, cenário com baixo delay e alta carga de rede, representado pela figura 14, o comportamento observado é de relativa estabilidade nas transmissões de diferentes prioridades, uma oscilação menor do que a observada em cenários com maior atraso, apesar da alta carga de rede, ja que o algoritmo é sensível às perdas aleatórias causadas pelo atraso do que à ocupação da rede.

Comparando o cenário mais distante do ideal, cenário 2, figura 10 com background load de 30MB e delay de 24ms, com o cenário mais próximo do ideal, cenário 5, 13 com background load de 10MB e delay de 10ms, podemos observar uma grande diferença na qualidade da transmissão para as duas diferentes prioridades. Devido à baixa perda de pacotes em um cenário mais próximo do ideal o algoritmo de adaptação quase não altera a qualidade das tiles enviadas. O algoritmo também considera o valor de vazão instantânea que, em uma condição ideal ou próxima do ideal, se mantém alta, mantendo a bitrate das tiles enviadas em alta qualidade.

Quando comparamos o cenário mais distante do ideal, cenário 2, figura 10 com background load de 30MB e delay de 24ms, com os cenários 1 e 6, figuras 9 e 14, com background load de 10MB e delay de 24ms e background load de 30MB e delay de 10ms respectivamente, observamos que a estabilidade da transmissão é mais afetada pela diferença de delay do que pela diferença de carga na rede. Isto se deve provavelmente à perdas aleatórias de pacote devido ao delay que, devido ao algoritmo de adaptação, causam uma redução na qualidade das tiles enviadas nos segmentos seguintes.

Nos cenários de baixa carga (figuras 9, 11 e 13), com 10 Mb, observamos uma variação menor na taxa de vazão quando comparado os cenários de alta carga (figuras 10, 12 e 14). Isso demonstra a eficiência do algoritmo em garantir a entrega prioritária do campo de visão sob condições de baixa concorrência por recursos na rede. As transmissões de prioridade 2 demonstram maior flutuação para cenários de maior atraso (figuras 9 e 10) quando comparado aos cenários de baixo atraso (figuras 13 e 14). Apesar dessas variações, a priorização do campo de visão se mantém efetiva, com os pacotes de prioridade 0 mantendo uma taxa de transferência alta e estável, com perdas pontuais.

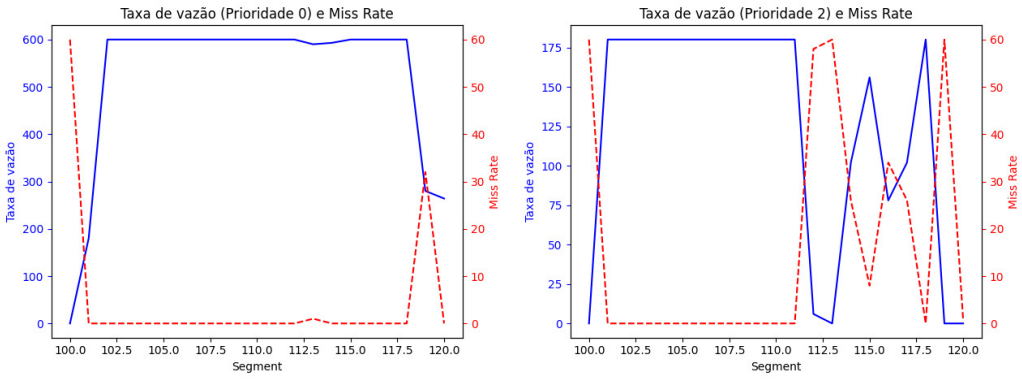

Figura 9: Gráficos de vazão da transmissão(kbps) e miss rate(números totais) por agrupamento de pacotes(100 à 120) Alta prioridade à esquerda e baixa prioridade à direita, Cenário 1

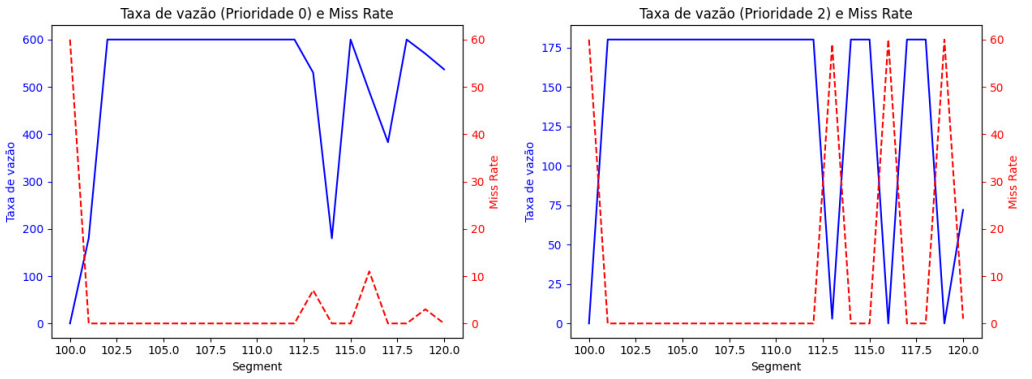

Figura 10: Gráficos de vazão da transmissão(kbps) e miss rate(números totais) por agrupamento de pacotes(100 à 120) Alta prioridade à esquerda e baixa prioridade à direita, Cenário 2

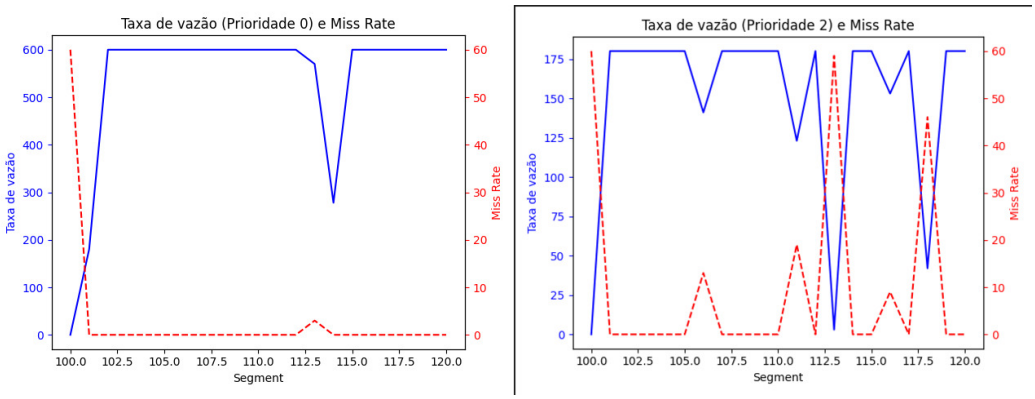

Figura 11: Gráficos de vazão da transmissão(kbps) e miss rate(números totais) por agrupamento de pacotes(100 à 120) Alta prioridade à esquerda e baixa prioridade à direita, Cenário 3

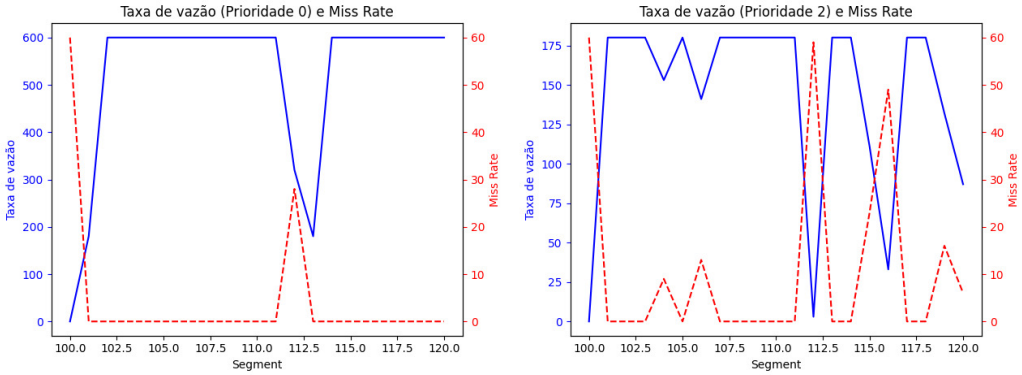

Figura 12: Gráficos de vazão da transmissão(kbps) e miss rate(números totais) por agrupamento de pacotes(100 à 120) Alta prioridade à esquerda e baixa prioridade à direita, Cenário 4

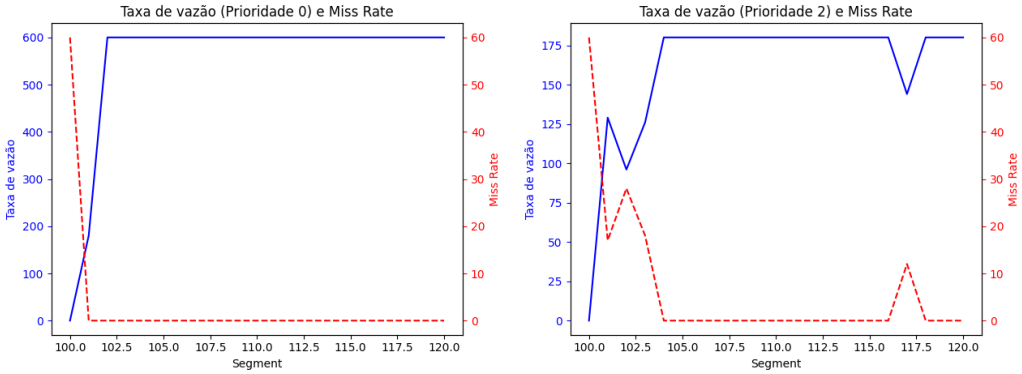

Figura 13: Gráficos de vazão da transmissão(kbps) e miss rate(números totais) por agrupamento de pacotes(100 à 120) Alta prioridade à esquerda e baixa prioridade à direita, Cenário 5

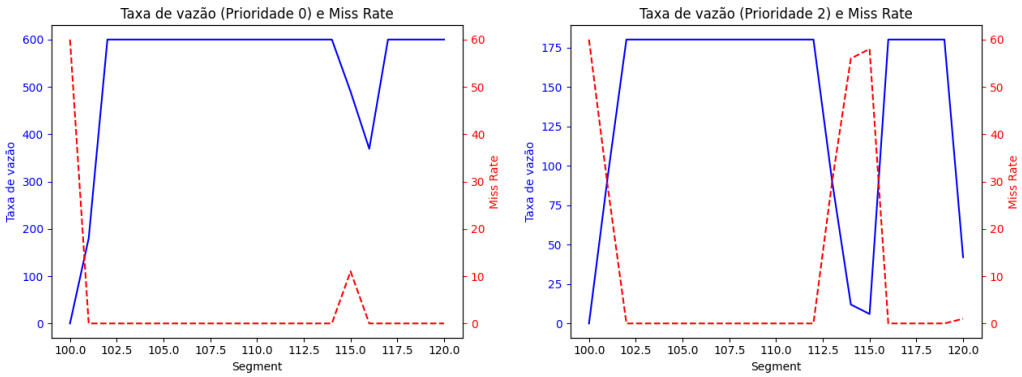

Figura 14: Gráficos de vazão da transmissão(kbps) e miss rate(números totais) por agrupamento de pacotes(100 à 120) Alta prioridade à esquerda e baixa prioridade à direita, Cenário 6

Ao comparar cenários com a mesma carga, mas diferentes atrasos (24ms vs. 16ms vs. 10ms), observa-se que o atraso tem um impacto mais pronunciado nas transmissões de prioridade 2. Comparando os cenários , verifica-se que em cenários com menor atraso, as flutuações na taxa de transferência e a perda de pacotes para prioridade 2 são menos severas, indicando um impacto considerável do atraso nas condições da transmissão. Nos cenários com o menor atraso (10ms), tanto as transmissões de prioridade 0 quanto as de prioridade 2 demonstram maior estabilidade, mesmo sob alta carga. Nos cenários 5 e 6 (figuras 8 e 9), observa-se que a redução do atraso contribui para uma melhor performance geral da rede e uma adaptação mais eficiente às variações de carga.

Em todos os cenários, a priorização do campo de visão se mostra efetiva, com os pacotes de prioridade 0 mantendo uma alta taxa de transferência e baixa perda de pacotes. O algoritmo de adaptação de taxa de fluxo e escolha de bitrate é altamente eficiente em garantir a qualidade de serviço para pacotes de alta prioridade, ajustando dinamicamente o bitrate conforme as condições de rede. A redução da latência melhora significativamente a estabilidade da rede e a eficiência do algoritmo, resultando em menores variações na vazão e taxas de perda mais baixas. Em condições de alta carga, o algoritmo continua eficaz, mas pacotes de baixa prioridade sofrem mais com variações na vazão e maiores taxas de perda, refletindo a priorização necessária para manter a qualidade de serviço dos pacotes críticos.

6 Conclusão

Este trabalho investigou a importância da priorização do campo de visão em algoritmos de adaptação de taxa de bits para streaming de vídeo 360º. Através da implementação de um sistema de streaming que utiliza um algoritmo de priorização de pacotes, demonstramos que a priorização do campo de visão tem um impacto significativo na qualidade da experiência do usuário, especialmente em condições desafiadoras de rede, como alta latência e largura de banda limitada. Os resultados obtidos em diferentes cenários de rede evidenciaram a eficácia

da priorização do campo de visão. O algoritmo desenvolvido, que ajusta dinamicamente a taxa de bits dos tiles com base na sua prioridade, garantiu a entrega dos dados essenciais para a visualização do vídeo, mesmo em condições de rede adversas

Observamos que, em cenários com alta latência, a qualidade da experiência do usuário é mais impactada, principalmente para os tiles de baixa prioridade. No entanto, mesmo nesses casos, a priorização do campo de visão minimizou a interrupção da experiência do usuário, garantindo a fluidez da visualização. Em cenários com baixa largura de banda, o algoritmo se mostrou eficaz em ajustar a taxa de bits dos tiles, priorizando a entrega dos tiles do campo de visão. A priorização do campo de visão se mostrou crucial para minimizar o impacto de perdas de pacotes e variações do bitrate do vídeo no campo de visão.

Para o futuro, várias áreas podem ser exploradas para aprimorar a transmissão de vídeos 360º. Melhorias em predição de viewport, desenvolvendo algoritmos mais precisos para prever a área de interesse do usuário, podem reduzir a quantidade de dados transmitidos desnecessariamente, aumentando a eficiência da transmissão. A criação de frameworks avançados de avaliação de QoE, que considerem todas as características do conteúdo 360º e as preferências dos usuários, pode proporcionar uma experiência de visualização mais satisfatória. A integração de tecnologias de inteligência artificial e aprendizado de máquina para otimizar a entrega e adaptação de conteúdo em tempo real pode trazer avanços significativos na qualidade e eficiência do streaming. A expansão da infraestrutura de edge computing pode reduzir ainda mais a latência e melhorar a responsividade do sistema, especialmente em redes altamente dinâmicas. Além disso, a adaptação das técnicas de streaming de vídeos 360º a novos dispositivos e plataformas, como óculos de realidade aumentada e ambientes de metaverso, será crucial para manter a relevância e a eficácia das soluções propostas. Essas perspectivas futuras prometem abrir novas possibilidades para a transmissão de vídeos 360º, oferecendo experiências imersivas ainda mais ricas e satisfatórias para os usuários finais.

Referências

[1] Langley, A., et al. (2017). The QUIC Transport Protocol: Design and Internet-Scale Deployment. Retrieved from https://research.google/pubs/the-quic-transport-protocol-design-and-internet-scale-deployment/

[2] Iyengar, J., Thomson, M. (2018). QUIC: A UDP-Based Multiplexed and Secure Transport. IETF Datatracker. Retrieved from https://datatracker.ietf.org/doc/html/draft-ietf-quic-transport-23

[3] Mininet. (n.d.). Mininet: An Instant Virtual Network on your Laptop (or other PC). Retrieved from https://mininet.org/overview/

[4] iPerf Contributors. (n.d.). iPerf - The ultimate speed test tool for TCP, UDP and SCTP. Retrieved from https://iperf.fr/

[5] Google. (2015). A QUIC update on Google's experimental transport protocol. Chromium Blog. Retrieved from https://blog.chromium.org/2015/04/a-quic-update-on-googles-experimental.html

[6] Springer. (2022). A study on the impact of 360 videos on e-learning platforms. Education and Information Technologies, 27, 1747-1766. Retrieved from https://link.springer.com/article/10.1007/s10639-022-11549-9

[7] N. Nattis Nelson L. S. da Fonseca César A. V. Melo, Relatório Técnico - IC-PFG-24-50, Projeto Final de Graduação, Request scheduling policies for 360° video applications using QUIC 2024 - Fevereiro https://github.com/DiegoSMartines/TccQuic/tree/scheduler