Universidade Federal de Juiz de Fora
Instituto de Ciências Exatas
Bacharelado em Sistemas de Informação

Gerador de Cargas Sintéticas para Avaliação de Eventos e Serviços de Grande Escala na Internet

Cássio Reis

JUIZ DE FORA
NOVEMBRO, 2019

Gerador de Cargas Sintéticas para Avaliação de Eventos e Serviços de Grande Escala na Internet

Cássio Reis

Universidade Federal de Juiz de Fora

Instituto de Ciências Exatas

Departamento de Ciências da Computação

Bacharelado em Sistemas de Informação

Orientador: Alex Vieira Borges

JUIZ DE FORA

NOVEMBRO, 2019

Gerador de Cargas Sintéticas para Avaliação de Eventos e Serviços de Grande Escala na Internet

Cássio Reis

MONOGRAFIA SUBMETIDA AO CORPO DOCENTE DO INSTITUTO DE CIÊNCIAS EXATAS DA UNIVERSIDADE FEDERAL DE JUIZ DE FORA, COMO PARTE INTEGRANTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE BACHAREL EM SISTEMAS DE INFORMAÇÃO.

Aprovada por:

Alex Vieira Borges
Doutor em Ciência da Computação

Edelberto Franco Silva
Doutor em Computação

Eduardo Pagani Julio
Doutor em Computação

JUIZ DE FORA
15 DE NOVEMBRO, 2019

Aos meus amigos, irmão e namorada.

Aos pais, pelo apoio e sustento.

Resumo

Novas formas de consumir conteúdo em vídeo pela Internet motivou pesquisas e estudos para caracterizar usuários ao redor do mundo. O foco deste trabalho é caracterizar, definir e simular usuários assistindo eventos de larga escala na Internet. Esses eventos geram alto volume de tráfego, o que acarreta em uma alta demanda por grandes recursos, os quais quando disponibilizados em nuvem garantem a distribuição mundial. Neste trabalho, analisamos o comportamento dos usuários de um sistema de vídeo ao vivo durante um grande evento na Internet, a copa do mundo de futebol da FIFA em 2014. Estudamos os dados relativos a transmissão de 64 partidas por um dos principais provedores de conteúdo Internet do Brasil e, desenvolvemos ferramental para reproduzir as cargas, de forma sintética, desse evento. Em suma, conhecer essa demanda é tarefa fundamental para avaliar a disponibilidade do serviço e consequentemente a assertividade no planejamento da disponibilização do serviço.

Palavras-chave: *Streaming*, Gerador de Cargas, Requisições, *workload*, *Ánalise de Dados*, *HTTP Live Streaming*.

Abstract

New ways to consume video content over the Internet have motivated research and studies to characterize users around the world. The focus of this scientific production is to characterize, define and simulate users watching large scale events on the Internet. These events generate high traffic volume, which results in a high demand for large resources, which when available in the cloud ensure worldwide distribution. In this paper, we analyze the behavior of users of a live video system during a large Internet event, the 2014 FIFA World Cup. We studied data related to the transmission of 64 matches by one of the main Internet content providers in Brazil, and we developed tools to reproduce the loads, synthetically, of this event. In short, knowing this demand is a fundamental task to assess service availability and, consequently, assertiveness in planning service availability.

Keywords: *Streaming*, Workload, Analytics, *HTTP* Live *Streaming*.

Agradecimentos

Aos meus pais, pelo encorajamento e apoio, amo vocês. À minha namorada, á minha família e à pequena Cecília.

Ao professor Alex Borges pela orientação, amizade e principalmente, pela paciência, sem a qual este trabalho não se realizaria.

Aos professores do Departamento de Ciência da Computação pelos seus ensinamentos e aos funcionários do curso, que durante esses anos, contribuíram de para o meu enriquecimento pessoal e profissional.

"Do. Or do not. There is no try."

Mestre Yoda

Conteúdo

Lista de Tabelas

Lista de Abreviações

CPU	Unidade Central de Processamento
DASH	*Dynamic Adaptive Streaming over HTTP*
DCC	Departamento de Ciência da Computução
DNS	Sistema de Nomes de Domínio
HDS	*HTTP Dynamic Streaming*
HLS	*HTTP Live Streaming*
HTTP	*Hypertext Transfer Protocol*
IP	Endereço de Protocolo da Internet
MPD	*Media Presentation Description*
NATs	*Network address translation*
PoP	*Pontos de presença*
P2P	*Peer-to-peer*
QoS	Qualidade de serviço
QoE	Qualidade e experiência
RAM	*Random-access memory*
RTMP	Real-Time Messaging Protocol
SSD	*Solid-State Drive*
URL	*Uniform Resource Locator*
UFJF	Universidade Federal de Juiz de Fora

1 Introdução

As principais produtoras de entretenimento do mundo têm investido esforços para transmissões de *Streaming* de vídeo em suas plataformas. Esses investimentos têm atraído a atenção da academia e da indústria, a fim de compreender as novas abordagens e os próximos passos da popularização da distribuição de vídeo ao vivo pela Internet. Essa popularidade crescente se evidencia com a consolidação de empresas como, Youtube, Netflix, HBO e Globo.com. Tais empresas destacam-se pela audiência e qualidade nas transmissões, sendo assim o aumento do número de clientes é uma demanda real e que necessita de atenção para manutenção da qualidade e da experiência dos usuários.

Neste trabalho, iremos aprofundar no protocolo utilizado pela maior provedora de *Streaming* do país, que fornece conteúdos que têm gerado um grande tráfego de rede com uma taxa de crescimento ascendente. Por exemplo, na World Cup FIFA – 2018, os servidores registraram recorde de audiência[1] e picos de até sete milhões de acessos simultâneos.

Para nosso estudo, definimos *Streaming* como a transmissão contínua de um fluxo de dados, reproduzido pelo destinatário à medida que é disponibilizado (MARQUES; BETTENCOURT; FALCÃO, 2012). Essa dinâmica representa uma abordagem mais contemporânea dada a inviabilidade de se garantir taxas de transmissões contínuas e uniformes, sobretudo em cenários de rede com mudanças repentinas e recursos escassos de transmissão de conteúdo fim-a-fim (COELHO et al., 2015), sendo assim, o ritmo de transmissão deve estar sincronizado com a capacidade da banda do cliente, evitando pausas indesejáveis na reprodução.

Portanto, é importante que provedores mobilizem recursos a fim de entender, caracterizar e estimar a capacidade de transmissão sem afetar na qualidade da experiência(QoE) dos usuários. No entanto, essa é uma grande dificuldade, visto que os limites da Internet atualmente não oferecem garantias de qualidade de serviço (QoS) para *Streaming*

[1] $HTTPs://maquinadoesporte.uol.com.br/artigo/copa-tem-altos-indices-de-audiencia-e-sportv-bate-recorde-com-brasil_34773.html$

de vídeo. Esse fator, associado à tendência de aumento de demanda de acessos, dependem de técnicas para o controle do tráfego. Portanto, a difusão de melhor esforço garante a entrega de uma mensagem por todos os processos de um sistema (RODRIGUES et al., 2018).

Outra atividade associada ao estudo das previsões de acesso é a escalabilidade no volume de tráfego de dados e a preparação para eventos futuros, este estudo gera uma dimensão do impacto no tráfego em servidores dos provedores de conteúdo. Infelizmente, os provedores de serviços são incapazes de prever com precisão a demanda desses eventos, particularmente para eventos esporádicos ou únicos, frequentemente subestimados ou superestimados (ALMEIDA et al., 2016). Sendo assim, diversos estudos estão em andamento com a proposta de caracterizar eventos de transmissão ao vivo pela Internet, principalmente eventos com alta demanda de acesso (BUSARI; WILLIAMSON, 2002; JUNIOR; ALMEIDA JUSSARA ALMEIDA; VIEIRA, 2014).

Apesar dos trabalhos de caracterização de transmissão de mídia ao vivo existentes (BORGES et al., 2012), há ainda carência de informações por parte da academia e da indústria, é necessário estudar a caracterização e o comportamento de clientes que consomem transmissões ao vivo. Esses estudos geram padrões de comportamento do cliente, que acabam se traduzindo em padrões de carga do sistema e afetam diretamente o desempenho do sistema. (BORGES et al., 2012).

Há muitos protocolos vigentes, assim para sermos mais precisos é clarificado que caracterizamos o protocolo *HTTP Live Streaming*(HLS). Neste modelo, é o programa de reprodução do cliente que controla quando e quais fragmentos devem ser enviados por parte do servidor (MARQUES; BETTENCOURT; FALCÃO, 2012). Os sistemas que atuam com base nesse modelo, necessitam de uma adaptação a taxa de dados dos fragmentos que serão solicitados ao servidor. Dessa forma o *player* encontra a qualidade que melhor se adapta a banda de Internet e altera a qualidade do vídeo impedindo que ocorram falhas na interrupção. Sendo assim, o servidor tem apenas duas tarefas, codificar o conteúdo a ser transmitido em tempo real com diferentes qualidades e enviar os fragmentos pedidos. (MARQUES; BETTENCOURT; FALCÃO, 2012).

Nesse contexto, este trabalho acadêmico examina os desafios que fazem a entrega

e reprodução simultâneas, e explora algoritmos e arquiteturas que permitem o teste de capacidade de servidores.

2 Revisão Bibliográfica

A disseminação do serviços baseados na arquitetura P2P, a partir da segunda década de 2000, acarretaram em diversos estudos com a proposta de caracterizar eventos de transmissão ao vivo pela Internet, principalmente eventos com alta demanda de acesso (BUSARI; WILLIAMSON, 2002; JUNIOR; ALMEIDA JUSSARA ALMEIDA; VIEIRA, 2014; ALMEIDA et al., 2016). Com isso, gerou-se uma oportunidade para os principais provedores do serviço de *Streaming*, visto que a demanda por serviços de vídeo *Online* tende a crescer ainda mais no mundo, principalmente pela qualidade da conexão e a portabilidade das ferramentas de vídeo. No mercado atual temos empresas como a *Netflix, YouTube, HBOGo, Globo.com*, que miram seus esforços para ganhar mercado e atrair novos usuários. Essa tarefa gera um árduo trabalho na prevenção de erros ocasionados pelo excesso de usuários, que impactam diretamente no fluxo de dados, em eventos de larga escala. Para potencializar a boa experiência dos clientes, faz-se necessário compreender e caracterizar usuários, com a missão principal de prever possíveis impactos ocasionados pela alta demanda e preparar uma infra-estrutura adequada para uma boa transmissão (ALMEIDA et al., 2016). Sendo assim, iremos abordar de maneira abrangente os cenários desenvolvidos nos trabalhos de caracterização dos clientes de um dos maiores provedores de vídeo do Brasil. Os trabalhos feitos anteriormente (ALMEIDA et al., 2016; JUNIOR; ALMEIDA JUSSARA ALMEIDA; VIEIRA, 2014) caracterizam a arquitetura de vídeo, o protocolo *HTTP Live Streaming* e o comportamento dos usuários. Iremos enriquecer esses trabalhos desenvolvendo um gerador de cargas sintéticas no intuito de fornecer um conhecimento não somente dos usuários, mas também do servidor como um todo, realizaremos testes de sobrecarga e a criação de cenários para simular as necessidades de adaptabilidade dos servidores.

Entendemos que os usuários estão cada vez mais criteriosos para a escolha do provedor de conteúdo, dada a grande variedade de empresas que atuam nesse segmento, a realização desse trabalho acadêmico impactará diretamente nos estudos de Qualidade e Experiência (QoE) (GUARNIERI et al., 2017) e na validação da estrutura sugerida em

trabalhos anteriores (JUNIOR et al., 2015).

De maneira geral, a parametrização do gerador de cargas será configurada para a análise realizada em usuários que assistiram a World Cup Fifa (2014) (JUNIOR et al., 2015; ALMEIDA et al., 2016; JUNIOR; ALMEIDA JUSSARA ALMEIDA; VIEIRA, 2014), esse monitoramento possibilita o entendimento dos problemas ocasionados pela alta demanda dos eventos e consolida uma análise crítica do modelo atual adotado pelos provedores de conteúdo (BUSARI; WILLIAMSON, 2002; JUNIOR; ALMEIDA JUSSARA ALMEIDA; VIEIRA, 2014; ALMEIDA et al., 2016).

Preocupamos em abordar as diferenças entre conceitos de técnicas,*Streaming, progressive dowloading e adaptive Streaming* (MARQUES; BETTENCOURT; FALCÃO, 2012) que auxiliam na *QoE* dos usuários. Essas técnicas ditam o ritmo da adaptabilidade do *player* e a adaptação do ritmo de transmissão a cada cliente final, tais como a velocidade da ligação e a capacidade do processamento (MARQUES; BETTENCOURT; FALCÃO, 2012). Temos em mente o potencial do serviço de disponibilização de vídeo pela Internet, sabemos que ele está se tornando cada vez mais popular, dado que em 2015 cerca de 62% do tráfego de rede consumido se deu por maioritariamente *Streaming* de vídeo. Sendo assim, pensar de maneira inovadora permitirá uma vantagem competitiva para empresas que irão se adaptar às transformações tecnológicas e de mercado(FRANÇA; DIAS; NASSAU, 2019).

3 Sistema de distribuição de vídeo

O provedor de conteúdo estudado possui dois pontos de Presença (PoPs) localizados nas duas maiores cidades do Brasil, São Paulo e Rio de Janeiro. Em cada um desses pontos de distribuição o provedor de conteúdo é conectado a um ponto de troca de tráfego local e várias redes comerciais (JUNIOR et al., 2015). Para o provedor de conteúdo um *Internet Protocol* (IP) decide para qual (PoP) encaminhar cada cliente, considerando a configuração dos protocolos de roteamento e da disponibilidade de cada servidor (CESARIO, 2012), a rede propaga os anúncios e os protocolos de roteamento escolhem as melhores rotas. As rotas escolhidas, em geral, conectam clientes ao servidor geograficamente mais próximo, levando à menor latência, maior banda, e distribuição de carga (KATABI; WROCLAWSKI, 2000). A Figura 3.1, ilustra os dois pontos de distribuição e o roteamento para do endereçamento dos pacotes de dados. Vemos que o cliente ao consumir um serviço de *Streaming* pode ser alocado no *PoP* localizado em São Paulo ou no Rio de Janeiro, dessa forma o tráfego é otimizado e a experiência do usuário é mais satisfatória.

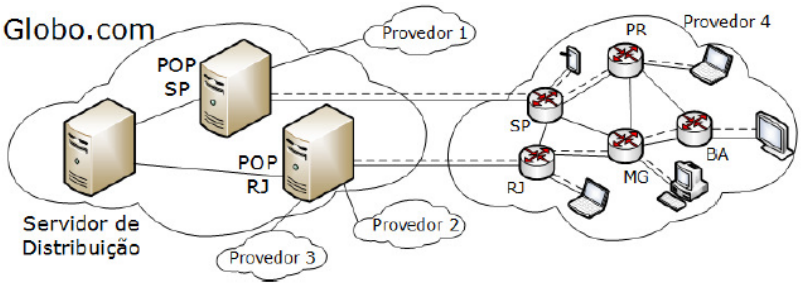

Figura 3.1: Arquitetura da transmissão de vídeo de sistemas monitorados(JUNIOR; ALMEIDA JUSSARA ALMEIDA; VIEIRA, 2014).

Na Figura 3.2 trazemos um desenho da arquitetura utilizada nas transmissões de vídeo. É possível perceber que um cliente acessa o domínio do provedor de conteúdo a partir do seu nome. Sendo assim o nome do domínio é resolvido pelo servidor de *Domain Name System* (DNS) apropriado e o Endereço de Protocolo (IP) do servidor de *DNS* que resolveu corresponde a um IP anunciado via *anycast*. Note que *anycast* divide os

clientes entre os dois pontos de distribuição, mas não garante balanceamento. (JUNIOR; ALMEIDA JUSSARA ALMEIDA; VIEIRA, 2014) .

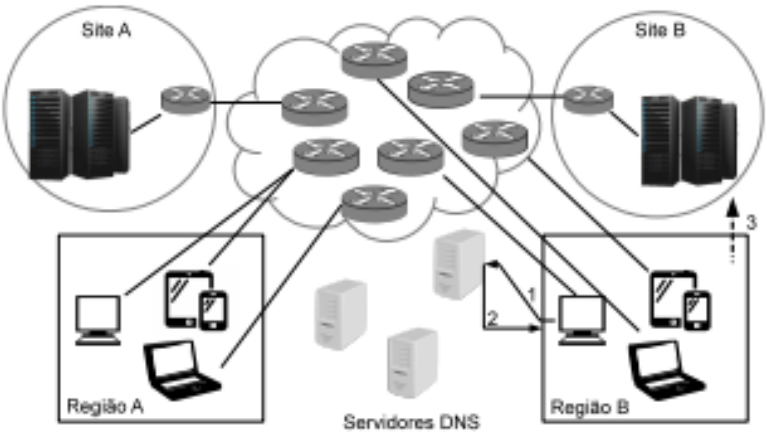

Figura 3.2: Arquitetura da transmissão de vídeo(ALMEIDA et al., 2016).

Os principais servidores de envio de mídia mundiais entregam conteúdo utilizando *HTTP*, o protocolo garante boa disponibilidade e superam os problemas encontrados na comunicação P2P. Em particular, transmissão por *HTTP* requer somente um navegador Web para visualizar conteúdos de vídeo e é desnecessário abrir portas em *firewalls* ou configurar redirecionamento de conexões externas em tradutores de endereços de rede *Network address translation* (NATs). (ALMEIDA et al., 2016).

Atualmente existem diversos protocolos para entrega de conteúdo de vídeo(*Adobe HTTP Dynamic Streaming (HDS), HTTP Live Streaming, HTTP Smooth Streaming, Real-Time Messaging Protocol (RTMP)*), neste trabalho iremos abordar o protocolo *HTTP Live Streaming - HLS*. Na próxima sessão faremos uma caracterização detalhada.

3.1 Arquitetura Live *Streaming*

O protocolo *HTTP* Live *Streaming*, um dos principais protocolos de vídeo disponíveis no mercado, foi desenvolvido pela empresa Americana *Apple Inc*, inicialmente com o propósito de que usuários enviassem conteúdos de vídeo e áudio ao vivo diretamente dos seus produtos usando um servidor Web. O protocolo ganhou notoriedade devido ao oferecimento de um meio econômico e confiável de disponibilização de vídeos de longa duração pela Internet. A adaptação a taxas de *bits* de mídia, a fim de manter a qualidade

e experiência do usuário em alto nível, é considerada um dos principais trunfos dessa arquitetura.

Na Figura 3.3 verificamos a existência de três componentes fundamentais na distribuição de vídeo[2]: servidor, distribuição e cliente.

1. Componente Servidor: O vídeo é capturado e codificado em várias taxas de transmissão (qualidade) diferentes. Cada mídia codificada e dividida em segmentos. Pequenos lotes de segmento com a mesma taxa de codificação são agrupados e indexados em "listas de reprodução";

2. Componente de Distribuição responsável por aceitar as solicitações do cliente e entregar a mídia preparada e recursos associados ao cliente[3];

3. O componente Cliente: é responsável por determinar a mídia apropriada, realizar a requisição e decodificá-la para ser apresentada em um fluxo contínuo, isto é, sem interrupções. A lista de reprodução permite ao cliente requisitar segmentos e reproduzir um trecho do vídeo (JUNIOR; ALMEIDA JUSSARA ALMEIDA; VIEIRA, 2014).

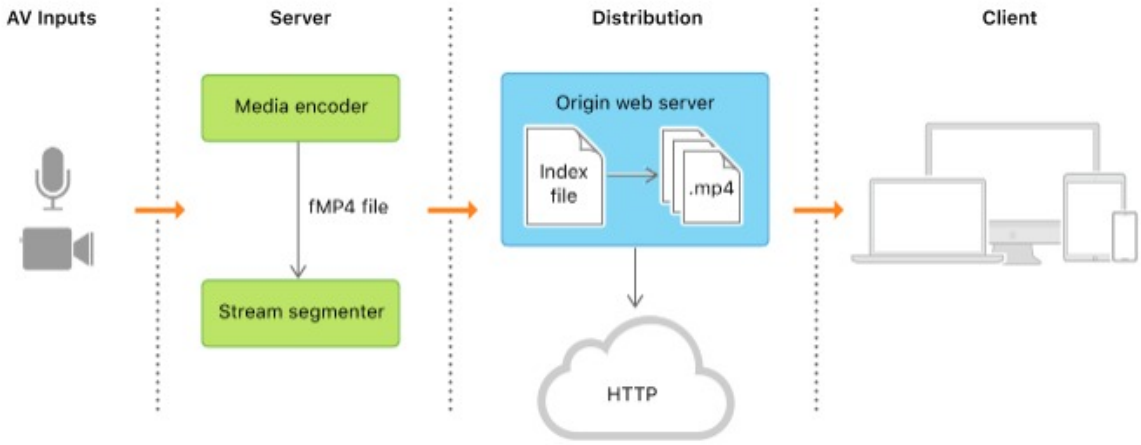

Figura 3.3: Arquitetura da distribuição de vídeo.

Em resumo, a entrada é codificada e transportada por um *hardware* disponível, segmentado e dividido por uma ferramenta de codificação. O segmentador cria o arquivo de índices que contém arquivos de mídia. Posteriormente o cliente solicita os fragmentos

[2] *HTTP*://developer.apple.com
[3] *HTTP*s://pdfs.semanticscholar.org/57d3/3cda30c2d497b694470aaa8b502613851fa5.pdf

de mídia que estão indexados em sequência, através da *Uniform Resource Locator* (URL) da lista de transmissão. Na lista de transmissão o cliente consegue definir a qualidade do *frame* de acordo com a banda de rede disponível. Ao fim de cada consumo uma nova requisição é enviada para o próximo endereço de memória da reprodução. Ainda nesse capítulo iremos abordar de maneira mais detalhada esse processo.

3.2 Dynamic Adaptive Streaming over HTTP (DASH)

Para o entendimento do sistema DASH precisamos inicialmente entender o conceito de *Streaming adaptive*. Para o *Streaming* adaptativo, são utilizados vários protocolos entre eles *push-based* e *pull-based*. Para o *push-based* o servidor gerencia a sessão do cliente, sendo responsável pela seleção das taxas de *bits* utilizadas ao longo da sessão. (COELHO et al., 2015). Já nas aplicações com protocolo *pull-based*, o cliente é quem requisita o conteúdo ao servidor. Logo, a transferência dos segmentos de vídeo dependem inicialmente das requisições realizadas pelo usuário. Estudaremos o protocolo *HTTP Live Streaming*, que utiliza o *pull-based*.

Nas requisições podem ocorrer trocas de taxas de transferência, essas trocas dependem de algoritmos implantados nos recursos do *player*, fatores como capacidade do dispositivo, largura da banda e *buffer* de reprodução são levados em consideração. Para o bom entendimento do fluxo exigido pela técnica, é necessário observar as condições de rede (instabilidade, disponibilidade e largura da banda), a capacidade do dispositivo do cliente e o contexto no qual o usuário está inserido.

Assim o *DASH* é uma abordagem técnica que viabiliza a interoperabilidade na indústria, define a organização do *Media Presentation Description* (MPD) e formatos de segmentos e guias de implementação, fornecendo assim um *framework* de *Streaming* adaptativo compatível com o protocolo *HTTP* (SEUFERT et al., 2014).

A técnica de *DASH* é utilizada para melhorar o modelo HLS, utilizando como base o *Adaptative Streaming*, a qual abordagem escolhe a qualidade da reprodução de vídeo em função da banda disponível e a capacidade da Unidade Central de Processamento (CPU) do cliente. (NUÑEZ, 2013) Os arquivos são codificados em diferentes *bitrates* e divididos em pequenos segmentos de poucos segundos. Ao iniciar a reprodução as escolhas

de qualidades podem sofrer alterações a fim de impactar a experiência do usuário. Assim o cliente é responsável direto pela qualidade da reprodução do vídeo, durante as quais ele pode enviar a informação ao servidor para melhorar ou reduzir a qualidade da reprodução, adaptando conforme a disponibilidade do tráfego de rede (NUÑEZ, 2013).

3.2.1 Listas de transmissão de vídeo .m3u8

Conforme abordado anteriormente, na reprodução do vídeo são criadas listas de transmissão de vídeo, essas listas possuem índices para os fragmentos de arquivos contendo as mídias. O cliente solicita através da URL o fragmento de vídeo (a Figura 3.4 retrata a arquitetura desse processo), as *playlists* são divididas em outras listas de transmissão com as respectivas qualidades, nas listas estão o endereçamento de cada resolução solicitada.

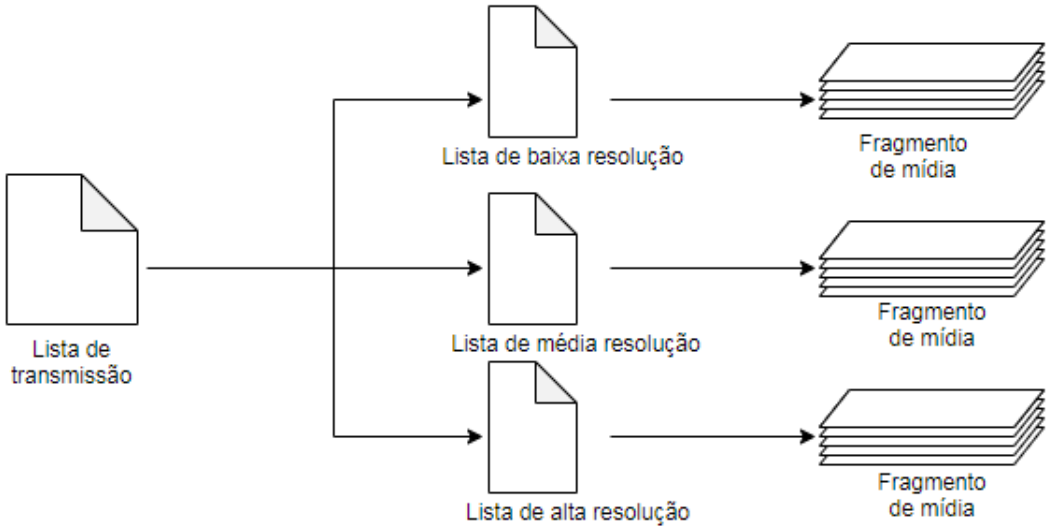

Figura 3.4: Formação das listas de reprodução de vídeo .m3u8[4]

Com o apoio da técnica de *DASH* a escolha da lista é realizada e a requisição para o fragmento correta é realizada, as listas de transmissão são criadas em sequência, para permitir o fluxo contínuo do vídeo.

Na Figura 3.5 observamos o conteúdo de uma lista de transmissão, os parâmetros mais importantes serão detalhados logo abaixo.

- #EXTM3U: *Tag* presente no cabeçalho indicando o formato do arquivo, deve ser sempre a primeira linha.

```
#EXTM3U
#EXT-X-PLAYLIST-TYPE:VOD
#EXT-X-TARGETDURATION:10
#EXT-X-VERSION:4
#EXT-X-MEDIA-SEQUENCE:0
#EXTINF:10.0,
http://example.com/movie1/fileSequenceA.ts
#EXTINF:10.0,
http://example.com/movie1/fileSequenceB.ts
#EXTINF:10.0,
http://example.com/movie1/fileSequenceC.ts
#EXTINF:9.0,
http://example.com/movie1/fileSequenceD.ts
#EXT-X-ENDLIST
```

Figura 3.5: Descrição do conteúdo do arquivo m3u8

- #EXT-X-TARGETDURATION: Duração máxima de todos os segmentos de vídeo.
- #EXT-X-VERSION: Indica a compatibilidade da versão da lista.
- #EXT-X-MEDIA-SEQUENCE: Marca a sequência de cada Lista, todas as listas possuem um identificador único e inteiro.
- #EXTINF: Tempo de execução em segundos.
- #EXT-X-BYTERANGE: Responsável por definir a variação permitida das qualidades.

  #EXT-X-ENDLIST: Marcação para o final da lista.

3.2.2 Segmentos de vídeo

Os segmentos de vídeo são criados e apontados em cada *playlist*. A junção desses segmentos compõem a apresentação como um todo. Normalmente cada pedaço faz referência ao anterior, porém com o acréscimo de uma unidade para manter a sequência do fluxo de vídeo ou outra forma para garantir a ordenação de cada pedaço. Os fragmentos são especificados por uma URL e um intervalo de bytes, caso o cliente necessite retroceder em

um determinado ponto, a marcação de onde deve-se iniciar facilita a localização de qual arquivo solicitar.

Na Figura 3.5 conseguimos identificar o arquivo *.ts* denominado como *"fileSequenceA.ts"*, esse arquivo possui 10 segundos de duração.

4 Metodologia

O trabalho foi realizado seguindo características de experimentos anteriores de geração de cargas sintéticas. Inicialmente, foi feita análise de dados e simulações referentes ao comportamentos de usuários que assistiram o evento oficial da *World Cup Fifa 2014*, tomando como base as arquiteturas propostas nos capítulos anteriores . A fim de tornar os cenários o mais próximo da realidade para as avaliações de resultados, as cargas foram construídas a partir das descrições propostas em (ALMEIDA et al., 2016), inclusive os parâmetros propostos da ferramentas foram obtidos de comportamentos reais de usuários que assistiram o jogo ocorrido entre as seleções nacionais Alemanha X Brasil neste torneio.

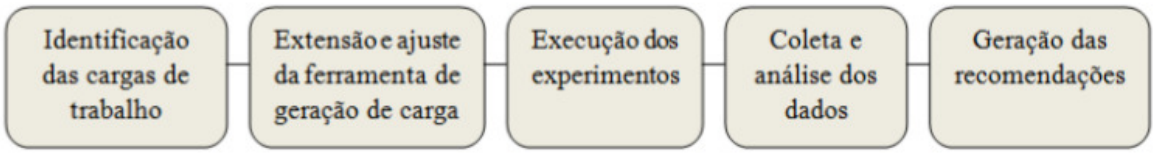

Figura 4.1: Etapas da metodologia aplicada

Nosso gerador permite a modelagem das características encontradas nos trabalhos anteriores, assim, fizemos as relações entre as requisições e o tempo de jogo, volume de dados transferidos na rede e as variações de qualidade requisitada por cada cliente. Essa análise permite encontrar dados muito próximos da realidade, sendo eles variáveis, o que torna nossa abordagem mais abrangente.

Além disso, o trabalho desenvolvido busca incrementar a literatura atual com um gerador de cargas sintéticas que reproduz fielmente os comportamentos de usuários. A pesquisa envolverá um problema crítico na preparação para o dimensionamento da demanda em eventos de grande porte. Nos basearemos em uma abordagem quantitativa, nossa pesquisa ajudará pesquisas existentes e impactará a comunidade acadêmica com um modelo atual de testes em servidores. Nosso referencial teórico será baseado nas produções mencionadas no documento e geradores de carga criados em outros trabalhos para outros problemas que envolvem a arquitetura de servidores, porém nosso universo será limitado às pesquisas que envolvem *Streaming* de vídeo e caracterização de usuários em eventos de

alta demanda.

4.1 Identificação das cargas de trabalho

Na Figura 4.2 estão presentes as métricas de interesse em cada uma das *threads* que compõe o gerador, cada *thread* representa um cliente que irá realizar as requisições de forma independente e paralela.

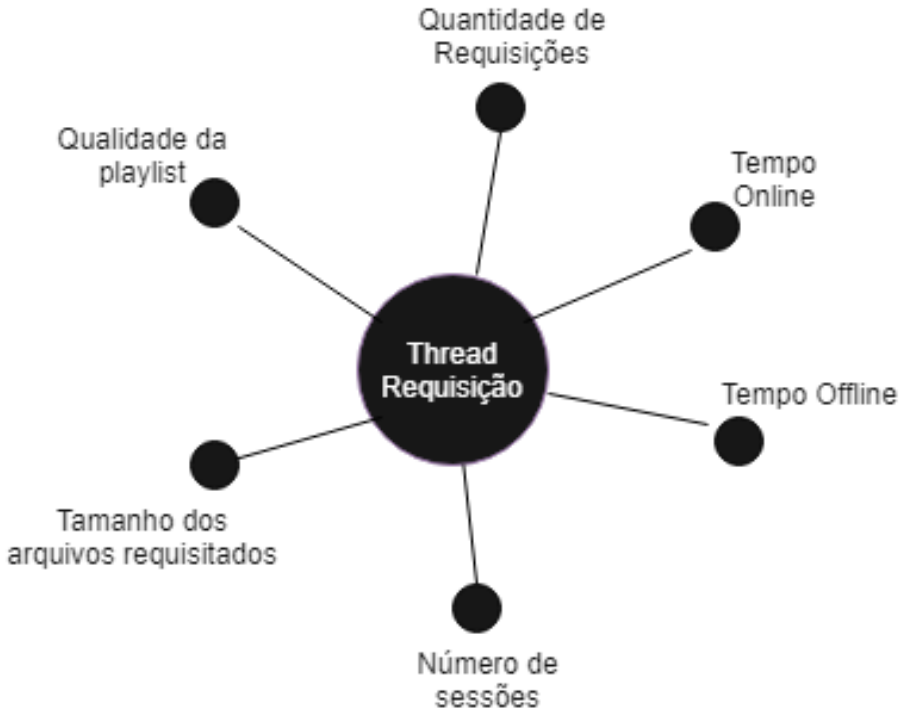

Figura 4.2: Modelo de dados e métricas de interesse

Para cada cliente iremos gerar um valor randômico de consumo de vídeo definido por nós como Tempo *Online*, para os intervalos onde o cliente não irá consumir um vídeo denominamos como Tempo *Offline*. Outras métricas de interesse são as quantidades de requisições e sessões dos usuários, o valor total de consumo de arquivos e as *playlists* consumidas pelos usuários baseado nas qualidades de vídeo solicitada por cada cliente.

4.2 Definição dos objetivos

O conhecimento do comportamento dos usuários ao consumir serviços de *Streaming* torna-se essencial para previsões de falhas na arquitetura em eventos de larga escala. Nessa linha,

definimos como nosso objetivo principal, a criação de um gerador de cargas sintéticas para simulações nos servidores *NGINX* baseado em trabalhos anteriores que observaram o comportamento de usuários na Copa do Mundo de 2014.

Nosso objetivo secundário será a caracterização da arquitetura de rede dos provedores de conteúdo e a validação das caracterizações de usuários em trabalhos referenciados. Outra tarefa importante é a análise crítica dos resultados obtidos.

4.3 Estratégias de ação

Para a avaliação da arquitetura e caracterização dos usuários, iremos analisar a literatura existente, absorver informações para embasar nossa concepção teórica do problema, em eventos de alta demanda, e implementar formas de captura de dados dos resultados pesquisados.

Para o objetivo proposto da criação do gerador de cargas, percebemos uma grande carência da literatura para modelos que simulam requisições. Nossa estratégia de ação atuará inicialmente no levantamento de requisitos, leitura de trabalhos anteriores que caracterizaram usuários em eventos de alta demanda, criação do modelo utilizando a linguagem de programação *Python 3.6.5* e a utilização de ferramentas padrões do *Unix* como *GREP* e *Shell script*. Para o processamento do gerador será feito em um computador com processador de 3.40GHz, *Random-access memory* (RAM) de 32GB e *Solid State Drive* (SSD) de 500GB. O sistema operacional utilizado será o Ubuntu versão 16.04 LTS.

4.4 Criação da Ferramenta de geração de cargas

A proposta de criação da ferramenta de geração de cargas sintéticas, surgiu a partir de um modelo proposto conforme a Figura 4.3, nesse modelo é possível perceber o ciclo de vida desde a criação dos clientes até a finalização da sessão. O modelo inicia com a criação de usuários baseada nos parâmetros definidos por um arquivo de importação, os parâmetros importados são as entradas de clientes na ferramenta de geração de cargas.

Ainda na etapa de criação de usuários, o Algoritmo 1 ilustra como foi desenvolvido a etapa de criação das *threads*, nesta etapa é determinada a geração de clientes para o

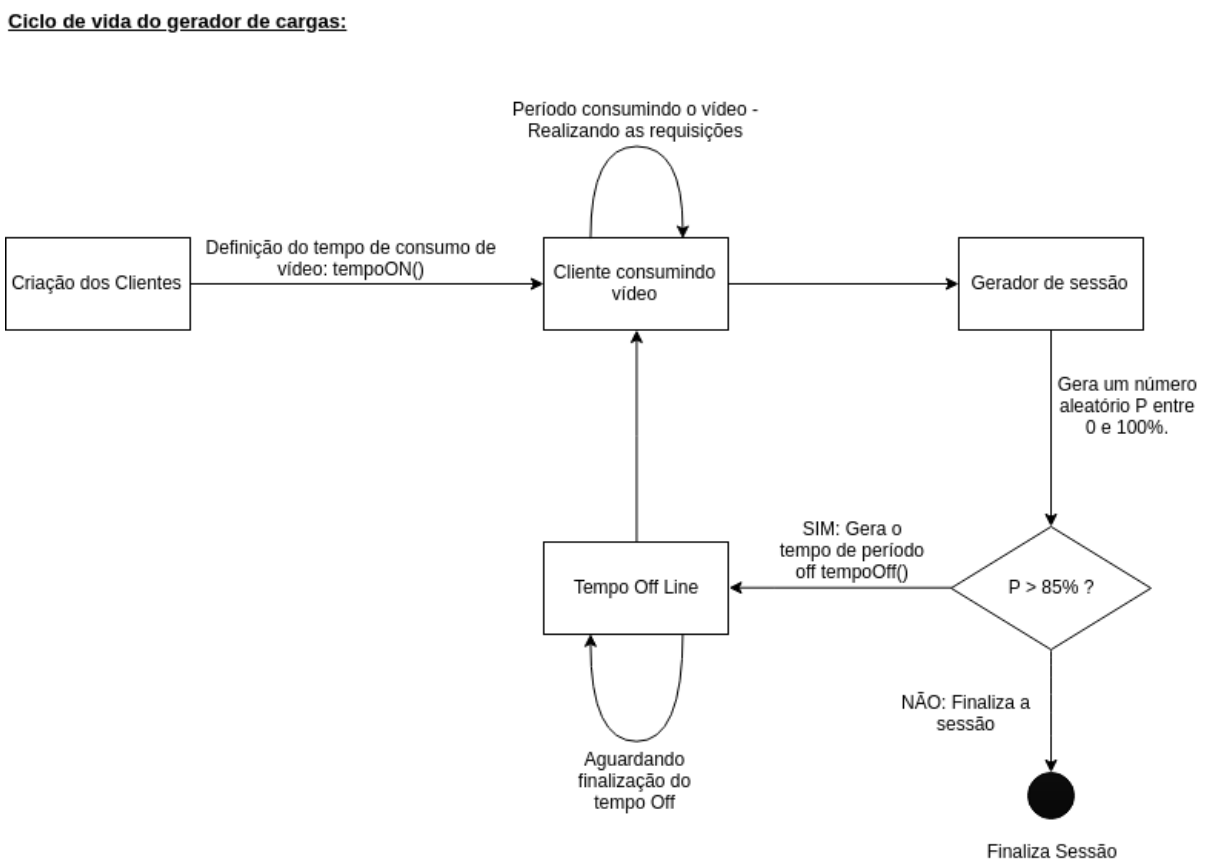

Figura 4.3: Modelo de dados e métricas de interesse

gerador. Foi utilizado a biblioteca de *threads* contida no *Python*. Por esse algoritmo é possível observar como são criados os clientes e como a taxa de criação obedece os parâmetros de configuração (por exemplo, para geração de clientes. Levamos em consideração a taxa de entrada e por quanto tempo ocorrerá a determinada taxa, desse modo conseguimos testar o nosso servidor para variações de taxas de maior e de menor intensidade).

Na sequência da Figura 4.3 observamos o estado de ”Cliente consumindo vídeo”. Nessa transição é aplicada a geração de números aleatórios baseado na função *Weibull* cuja os parâmetros desta função foram definidos com base no artigo referenciado (ALMEIDA et al., 2016) . O resultado dos números gerados definem o tempo de cada ciclo de consumo Online de vídeo, ao longo do ciclo são realizadas requisições a cada 5 segundos (ALMEIDA et al., 2016). Para cada requisição é gerada a escolha da qualidade, baseado nos *logs* do cliente, identificamos as probabilidades de escolha de cada requisição, que afeta diretamente na escolha da qualidade, porém como o objetivo do nosso trabalho é observar o modelo de forma quantitativa, abstraímos as variações que levam as escolhas das qualidades, esse fator está diretamente relacionado as experiências de cada cliente.

Algoritmo 1: Criação de Clientes

Input: Lista de Parâmetros que contêm as taxas de acesso
Output: Criação das *threads* que realizam as requisições

```
1  Início;
2  while T_i < T_f do
3  |  if not in flag_parada then
4  |  |  thread = cria_treads();
5  |  |  thread.executa()
6  |  else
7  |  |  T_i = T_i + T_f;
8  |  end if
9  end while
10
```

No algoritmo 2 as requisições são realizadas a cada 5 segundos, é importante frisar que a etapa onde é realizada a escrita no *log* foi a forma como validamos o estudo. O *log* é gerado na mesma pasta onde está instalado o gerador. O conteúdo de cada *log* contém: os nomes de cada *thread*, tempo total, tempo *Online*, tempo *Offline* e o número de sessões.

Algoritmo 2: Cliente consumindo vídeo: Realização das requisições

Input: tempo_Online

```
1  gerador = True;
2  while (x≤ tempo_Online) do
3  |  if x mod 5 ==0 then
4  |  |  realiza_requisicao();
5  |  |  escreve_log();
6  |  end if
7  |  x = x+1;
8  end while
9
```

Para chamar todos os procedimentos de forma ordenada, foi construída uma estrutura definida como controle. Nesse procedimento o tempo no qual o usuário passa consumindo o vídeo é recebido pelo retorno do procedimento no qual está inserida a função *Weibull*. Já o número de sessões por cliente pode ser computado baseando na probabilidade de existir uma transição entre os estados de *On* e *Off* conforme a Figura 4.4. Em outras palavras, a distribuição geométrica pode ser utilizada para descrever esse comportamento (ALMEIDA et al., 2016). Portanto, para cada sessão gerada, o retorno da função geométrica, definirá a mudança de estado do gerador. Sendo assim, sorteia-se um número entre 0 e 1. Se o número gerado for maior que 0,85, o gerador mantêm-se

ativo e solicita o tempo *Offline*.

Figura 4.4: Modelo "ON/Off" do comportamento de um cliente

A Figura 4.5 ilustra a possibilidade de que um único IP tenha 10 sessões ao longo do jogo. No artigo referenciado praticamente 15% dos usuários tiveram sessões que cobriam toda a partida (ALMEIDA et al., 2016). O número médio de sessões em um determinado usuário foi menor que de outros eventos, por exemplo em transmissões de evento em rede P2P, 40% dos clientes tinham uma única sessão (BORGES et al., 2012). Já no trabalho referenciado, que parametrizou o gerador (ALMEIDA et al., 2016), cerca de 80% dos clientes tem apenas uma única sessão.

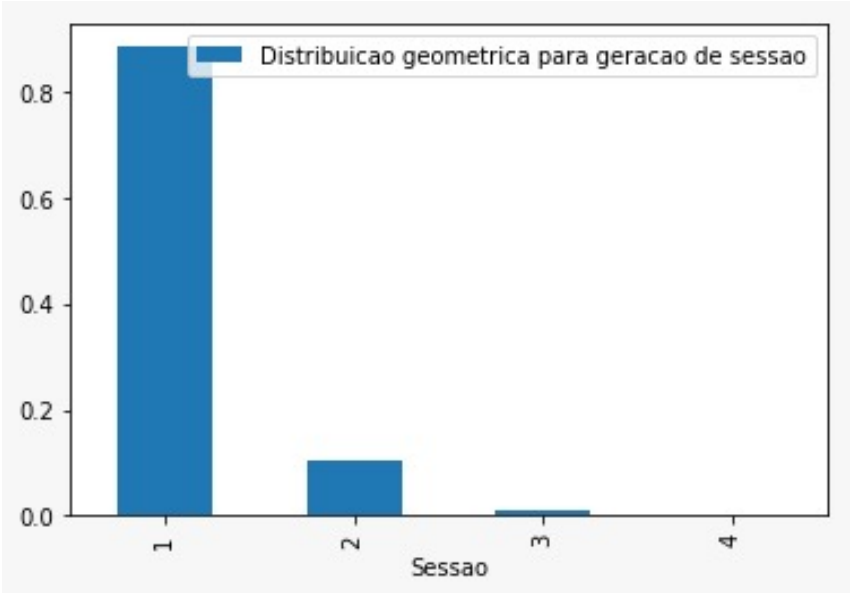

Figura 4.5: Distribuição geométrica do número de sessões

No Algoritmo 3 podemos verificar como foi estruturado o principal procedimento que dita as regras para o gerador. Esse procedimento é responsável por chamar os métodos de importação de tempo *Online* e *Offline*, além de realizar a importação de uma nova sessão ou não.

Algoritmo 3: Estrutura de controle

```
   Input: tempo_de_video,nome_thread
 1 gerador = True;
 2 while gerador do
 3 |   tempo_ Online = importa_tempo_on(); /* Recebe o resultado da função
   |   Weibullb */
 4 |   realiza_requisicao();
 5 |   if nova_sessao() then
 6 |   |   importa_tempo_Off();
 7 |   else
 8 |   |   destroi_thread();
 9 |   end if
10 | end while
```

O ciclo de vida continua até que não haja uma nova sessão, conforme visto na Figura 4.4: a máquina de estado é simples, porém muito eficiente para definir as etapas do processo de consumo de vídeo Online.

O algoritmo pode ser melhorado, principalmente se pensarmos em maneiras de aproveitar o paralelismo, tendo em vista que a quantidade de *threads* é um agente limitante de desempenho. Em trabalhos futuros poderíamos aproveitar de sistemas distribuídos e configurar outras máquinas para realizar um conjunto de requisições.

Para coleta dos dados foi criado um *script* em *AWK* e um em *Python*. O *script* reúne o *log* de todos os clientes. Em cada *log* há o tempo consumindo vídeo, a sessão de cada cliente, o tempo sem consumo de vídeo e a quantidade em *Megabytes* transferidos.

5 Análise dos resultados

5.1 Experimentos

Apresentaremos neste capítulo os experimentos realizados e as regras definidas para as cargas de trabalho. Ao longo do trabalho definimos os parâmetros do gerador baseado em análises ocorridas nos trabalhos referenciados. Dentre os jogos estudados iremos utilizar o jogo que ocorreu entre Argentina e Suíça. Queremos traçar paralelos entre os resultados obtidos pelo gerador de cargas e os resultados obtidos no artigo referenciado. A Figura 5.1, retrata a carga de trabalho imposta pelos quatro jogos aos servidores do provedor de conteúdo (ALMEIDA et al., 2016). Notamos que a curva de crescimento do número de clientes tem maior inclinação nos minutos iniciais do jogo, depois vê-se uma queda no intervalo.

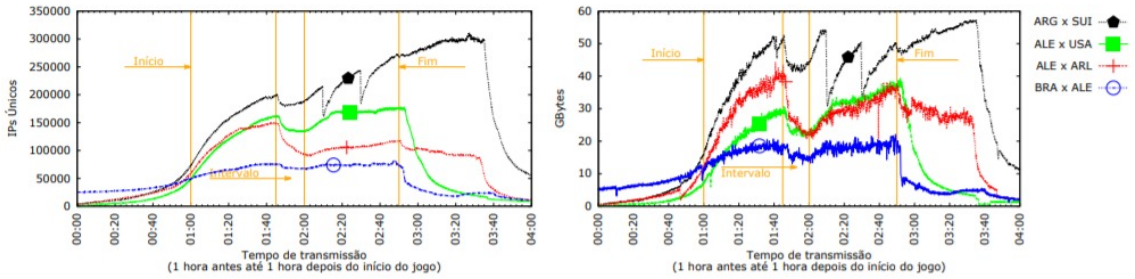

Figura 5.1: Modelo de dados e métricas de interesse (ALMEIDA et al., 2016)

A execução do gerador será realizada para dois níveis de cargas. Esperamos assim testar o servidor em dois cenários distintos: um cenário de alta intensidade de entrada de usuários por segundo durante um período de tempo e o outro cenário de crescimento lento, iremos diferenciá-los entre experimentos A e experimento B.

Na Tabela 5.1 são mostrados os experimentos e suas respectivas taxas de entradas

Tabela 5.1: Taxa de entrada de usuários por minuto

Amostra	**0 a 1min**	**1 a 2min**	**2 a 3min**	**3 a 4min**	**4 a 5min**
Experimento A	5	3	7	5	3
Experimento B	50	30	70	50	30

a cada minuto. No experimento A esperamos como resultado final um número acima de mil clientes únicos e simultâneos. No experimento B, excitaremos o servidor para uma carga de maior intensidade, esperamos obter mais de dez mil clientes simultaneamente.

5.2 Resultados

Essa sessão é responsável por apresentar e discutir os resultados obtidos nos experimentos previamente planejados. Dividiremos essa sessão por métricas de interesses e iremos traçar um paralelo entre cada amostra.

5.2.1 Número de clientes simultâneos

Uma motivação de um gerador de cargas sintéticas para eventos de *Streaming* é conhecer quantos clientes estão realizando requisições ao mesmo tempo, ou seja, o comportamento das *threads* criadas devem ser monitoradas para se ter uma dimensão da resposta da aplicação.

Na Figura 5.2, mostramos os dois comportamentos de experimentos gerados. O intuito do gráfico é mostrar a quantidade de ativos na aplicação. Além disso é importante observar que com apenas 5 minutos de entrada de clientes o modelo é capaz de gerar cargas que consomem todo o período do jogo.

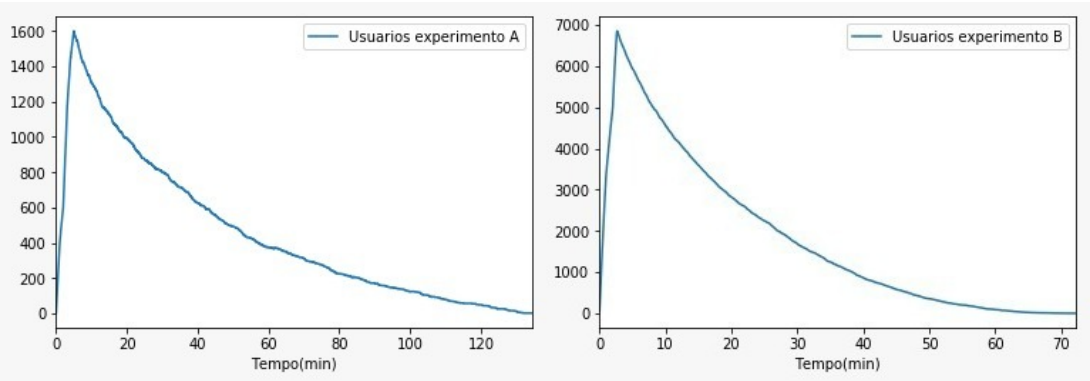

Figura 5.2: Acesso de clientes simultâneos para cada experimento

Essa análise é interessante para reproduzir picos de acessos simultâneos, por exemplo, ao longo das cargas geradas pelo experimento B, observamos um pico de acesso de 7 mil usuários nos primeiros minutos, isso se deve pela taxa de entrada ter sido concentrada nos minutos iniciais.

5.2.2 Resultados para tempo de consumo de vídeo

Na Figura 5.3, a distribuição acumulada se assemelha bastante com o trabalho referenciado (ALMEIDA et al., 2016), assim como o cenário real, o gerador de cargas criou uma demanda superior à cobertura completa do jogo.

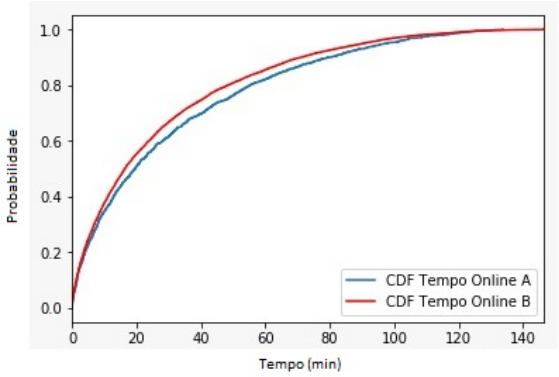

Figura 5.3: Distribuições acumuladas dos Tempos de ON

5.2.3 Resultado para tempo Offline

Para o tempo total *Offline*, os resultados encontram-se na Figura 5.4. É interessante observar que os dados gerados são muito semelhantes ao cenário real, utilizamos como parâmetro a distribuição *Lognormal*. Em aproximadamente 20% dos clientes o tempo *offline* foi superior a 15 minutos.

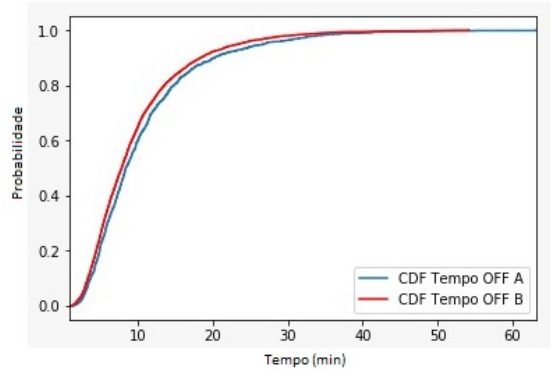

Figura 5.4: Distribuições acumuladas dos Tempos Off

5.2.4 Resultado para quantidade de requisições

Na Figura 5.5 é apresentada a distribuição acumulada de densidade para os dois experimentos. Pode-se perceber que a quantidade de requisição é proporcional ao tempo total de consumo de vídeo. As requisições foram configuradas para ocorrer aproximadamente de 5 em 5 segundos, no modelo real as requisições são realizadas de acordo com o *buffer*. A fim de reproduzir um cenário quantitativo, a configuração adotada respeitou as estatísticas para o modelo real.

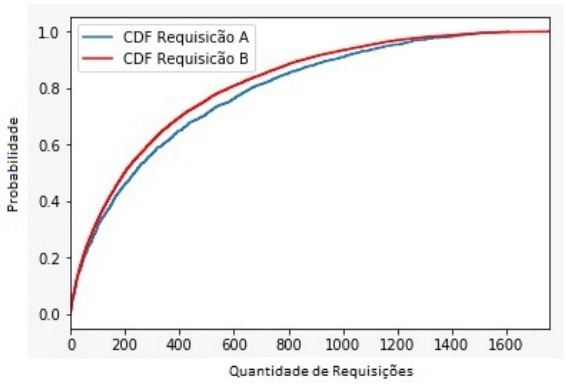

Figura 5.5: Distribuições acumuladas das quantidades de requisições

5.2.5 Resultado para quantidade de sessões e o volume de transmissão(GB)

Na Figura 5.6, está presente a distribuição acumulada do volume de dados gerados pela a aplicação. O tráfego das cargas geradas é um ponto de atenção para os provedores de conteúdo, pois em poucos segundos milhares de GB são transferidos. Nas transmissões reais o volume chegou a 2100GB em aproximadamente 1 minuto de transmissão (ALMEIDA et al., 2016).

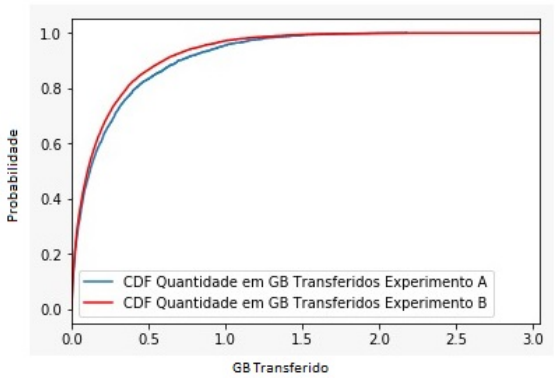

Figura 5.6: Distribuições acumuladas para tamanho total transferido

A Figura 5.7 indica a porcentagem de novas sessões para o experimento A e B. Utilizamos uma função geométrica para modelar essa métrica de interesse, a probabilidade de se ter uma nova sessão é dada por P_{off}=15, essa configuração ocasiona em um cenário onde a maioria dos usuários possui uma única sessão.

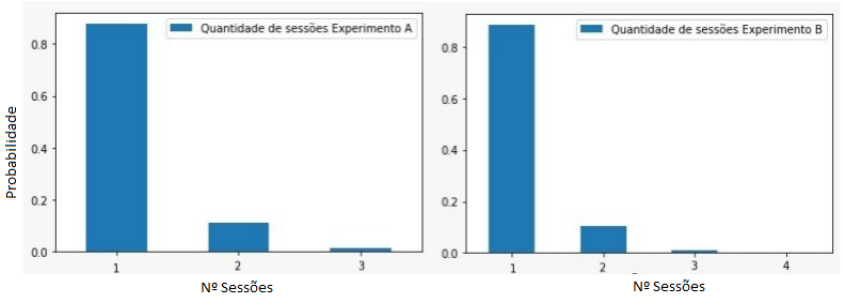

Figura 5.7: Número de sessões de cada experimento

Nas Tabelas 5.2 e 5.3, estão presentes os sumários estatísticos das métricas de interesse. Com essas tabelas queremos propor parâmetros de comparação para que a industria possa atuar de maneira preventiva nas transmissões de vídeo ao vivo. A comparação entre os experimentos mostram que o tempo de *On* variou entre o experimento A e o experimento B, isso ocorreu devido ao fato da métrica *Tempo On(s)* ser considerada como uma agregação entre de todas as sessões, e para o experimento B houve um aumento no número de sessões criadas.

Na Tabela 5.4, observamos que a perda de *threads* são proporcionais a entrada de clientes. Ou seja, quanto maior o número de clientes consumindo conteúdo de vídeo, maior será o impacto na performance do gerador, esse fator é um ponto de avaliação e de futuras

Tabela 5.2: Tabela de experimentos A

Métricas	**Média**	**Mediana**	**Desvio Padrão**	**Intervalo de Confiança**
Tempo On (s)	3845.6	1779	5801	(3271, 4420)
Tempo Off (s)	553	446	386	(514, 591)
Requisições	770	357	1160	(655, 885)
Tamanho dos Arquivos(GB)	257	119	387	(219, 296)

Tabela 5.3: Tabela de experimentos B

Métricas	**Média**	**Mediana**	**Desvio Padrão**	**Intervalo de Confiança**
Tempo On (s)	6449	4702	6826	(5813, 7085)
Tempo Off (s)	557	460	382	(521, 592)
Requisições	1291	942	1365	(1164, 1418)
Tamanho dos Arquivos(GB)	431	314	456	(389, 474)
Número de sessões	1.28	1	0.57	(1.23, 1.33)

melhorias. Outros fatores que impactaram para as perdas de *threads* foram as limitações impostas pelo sistema operacional para acesso concorrente de clientes. Nosso sistema utiliza para coleta de dados documentos de textos e esses documentos são acessados por cada *thread* simultaneamente, gerando um problema de concorrência.

Tabela 5.4: Perda de *threads* por *time out* ou concorrência

Amostra	**Taxa de Perda de Threads**
Experimento A	1%
Experimento B	5%

6 Conclusão

Prever eventos de larga escala na Internet permite que os provedores de conteúdo tenham vantagens competitivas diante dos seus concorrentes, as empresas podem preparar a infraestrutura adequada e fornecer ao cliente uma experiência agradável. O intuito do gerador de cargas é fornecer ferramentas para avaliação e testes em servidores, e, assim entendemos que esse objetivo foi alcançando, visto que devido a facilidade na parametrização, diversos cenários podem ser gerados.

Ao longo do estudo, alguns pontos de atenção foram encontrados, como o tráfego de dados que impõe à infraestrutura uma condição desafiadora para *QoE* dos clientes. Mas para melhorias desse tema a utilização de *bitrates* que se adaptam a largura da banda de rede do cliente é uma ótima opção e é fundamental para alcançar um número maior de clientes.

Os resultados mostram um grande potencial dos sistemas de *Streaming* de atrair clientes. Além disso, tivemos bons comportamentos na escalabilidade e disponibilidade, esse modo de fornecer conteúdo aos usuários quebra as fronteiras físicas e garante uma participação de forma global do conteúdo.

Os resultados gerados foram convergentes com o artigo referenciado, isso reforça que a aplicação pode ser útil para o provisionamento e alocações de recursos de forma mais precisa.

Para trabalhos futuros, podemos considerar a distribuição do gerador em outras instâncias, tornando-o mais robusto e preparado para maiores intensidades de *threads*. Outra preocupação é relacionar a entrada e saída do usuário de maneira qualitativa, relacionando a experiência do cliente com às condições de banda de rede disponíveis.

Bibliografia

ALMEIDA, B. et al. Characterizing *QoE* in large-scale live streaming. In: *34o. Simpósio Brasileiro de Redes de Computadores e Sistemas*. [S.l.: s.n.], 2016.

BORGES, A. et al. Characterizing sopcast client behavior. *Computer Communications*, v. 35, n. 8, p. 1004 – 1016, 2012. ISSN 0140-3664. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0140366412000710>.

BUSARI, M.; WILLIAMSON, C. Prowgen: A synthetic workload generation tool for simulation evaluation of web proxy caches. In: . [S.l.: s.n.], 2002, (Comput. Netw, v. 38). p. 779—-794.

CESARIO, M. Uso de anycast para balanceamento de carga na globo. com. *Talks and Tutorial, SBRC*, 2012.

COELHO, M. d. S. et al. Estratégia de adaptação de fluxo de vídeo baseada em fatores de *qoe*. Universidade Federal do Amazonas, 2015.

FRANÇA, Y. M. da S.; DIAS, E.; NASSAU, C. U. M. de. Inovação na comunicação da indústria de streaming de entretenimento1. 2019.

GUARNIERI, C. *QoE* in L.-S. L. S. T. et al. Characterizing *QoE* in large-scale live streaming. In: *IEEE Globecom*. [S.l.: s.n.], 2017.

JUNIOR, A. et al. Avaliação de transmissão ao vivo de grandes eventos pela internet. Universidade Federal de Juiz de Fora, 2015.

JUNIOR, W. de A.; ALMEIDA JUSSARA ALMEIDA, I. C. B.; VIEIRA, A. B. Caracterização do tráfego e impacto de rede da transmissão de um grande evento esportivo. In: *33o. Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*. [S.l.: s.n.], 2014.

KATABI, D.; WROCLAWSKI, J. A framework for scalable global ip-anycast (gia). *ACM SIGCOMM Computer Communication Review*, ACM, v. 30, n. 4, p. 3–15, 2000.

MARQUES, A.; BETTENCOURT, R.; FALCÃO, J. Internet live streaming. *Instituto Superior Técnico, Portugal*, 2012.

NUÑEZ, B. C. Dash: Un estandar mpeg para streaming sobre http. *Facultat d'Informatica de Barcelona, Universitat Politecnica de Catalunya*, 2013.

RODRIGUES, L. A. et al. Agregação de mensagens em uma solução hierárquica de difusão de melhor-esforço. In: SBC. *Anais do XIX Workshop de Testes e Tolerância a Falhas (WTF-SBRC 2018)*. [S.l.], 2018. v. 19.

SEUFERT, M. et al. A survey on quality of experience of http adaptive streaming. *IEEE Communications Surveys & Tutorials*, IEEE, v. 17, n. 1, p. 469–492, 2014.