Aumentando a Eficiência do *Cache* Proativo com Algoritmos de Mochilas para PoPs e *Hashes* para Servidores

Paulo Renato C. Mendes[1], **Lenise M. V. Rodrigues**[1,2], **João L. da S. Guio Soares**[1], **Arthur Serra**[1], **Yago Coelho**[1], **Nedimar Turatti**[1], **Antonio A. de A. Rocha**[3], **Daniel Sadoc Menasché**[2]

[1]Globo

[2]Instituto de Computação – UFRJ

[3]Instituto de Computação – UFF

Resumo. *Caches são elementos fundamentais para reduzir a carga nos servidores e o atraso dos usuários nas redes de entrega de conteúdo (CDNs). Este trabalho apresenta e avalia estratégias de cache proativo para aumentar a eficiência do cache em uma CDN distribuída geograficamente, com dois algoritmos para selecionar mídias a serem armazenadas em cache. O primeiro se aproveita do clássico problema das mochilas (knapsack), e acopla todos os servidores de cada PoP ao determinar como preencher cada mochila. O segundo preestabelece os servidores candidatos a servir cada mídia, por meio do uso de hashes e, então, aplica o problema da mochila em cada servidor alvo. As duas soluções foram testadas na infraestrutura da Globo, na qual foi possível aumentar o número de visualizações dos vídeos servidos em cache em 80,1%.*

1. Introdução

O crescimento da acessibilidade à Internet de maior qualidade fomentou a popularização dos serviços de *streaming*, que hoje são responsáveis por 21% do consumo de vídeo do consumidor brasileiro.[1] Com mais de 70% dos brasileiros possuindo assinaturas ativas em tais serviços, o *Globoplay*, plataforma de *streaming* da Globo, representa 30% destas assinaturas[2] e possui mais de 20 milhões de usuários,[3] assinantes e não assinantes. Este cenário cria, então, o desafio de transmitir conteúdo com uma boa qualidade de experiência para uma grande quantidade de usuários, muitas vezes, geograficamente distantes. Assim, surge a oportunidade de estudos focados na evolução da entrega de vídeos, abordando distribuição de *cache* e otimizações a ela relacionadas, como o uso da solução do problema da mochila (*knapsack* [Hasslinger et al. 2020]) para otimizar a entrega de conteúdo [Poularakis et al. 2016, Xu 2021, Shi et al. 2021, Hasan et al. 2014].

Este trabalho visa analisar e propor um melhor uso dos servidores na arquitetura de *Content Delivery Network* (CDN) da Globo, no ambiente de produção, que hoje possui mais de 120 pontos de presença (*points of presence*, ou PoPs) em sua infraestrutura, por meio da previsão de popularidade dos vídeos e do pré-posicionamento desse

[1]https://www.kantaribopemedia.com/inside-video-2022

[2]https://www.abranet.org.br/Noticias/71\%25-dos-brasileiros-assinam-ou-ja-assinaram-streaming-3847.html

[3]https://canaltech.com.br/entretenimento/globoplay-ja-tem-20-milhoes-de-usuarios-e-e-lider-nacional-de-streaming-172792/

conteúdo nos servidores da CDN: o *Precache*. Considerando as capacidades disponíveis em tais máquinas, é possível identificar pontos de melhoria e propor soluções que visem à otimização da distribuição de conteúdo, pré-posicionando vídeos em *cache*. Ao adaptar a clássica visão do problema da mochila, considerando a capacidade dos PoPs, que possuem vários servidores, e uma estratégia de distribuição por servidor como partes fundamentais na solução, foi possível encontrar uma melhora na qualidade de distribuição de conteúdo. As principais contribuições deste trabalho são:

Mochilas para PoPs: foram propostas e comparadas diferentes abordagens para alocação de conteúdos em *caches* de PoPs. Dentre as abordagens envolvendo o problema da mochila, foi associada uma mochila a cada servidor de um PoP (Seção 5.2), e comparou-se o caso em que as mochilas possuíam restrições conjuntas contra o caso em que cada mochila era independente das demais (Seção 5.3). Nossa contribuição consiste em relacionar múltiplas mochilas para cada PoP, e evidenciar que ao se considerar uma por servidor é possível resolver o problema de alocação de conteúdos, considerando a localidade da demanda e as restrições inerentes a cada PoP.

***Hashes* de conteúdos para servidores:** foram considerados *hashes* para mapear conteúdos a servidores candidatos a armazenarem tais conteúdos, garantindo que um certo conteúdo será sempre mapeado ao mesmo servidor dentro de cada PoP. Assim, simplificou-se a tarefa de busca por conteúdo dentro de cada PoP. Além disso, foi viabilizada a implementação da alocação de conteúdos por meio da solução do problema das mochilas, tendo em vista que quando os conteúdos são pré-mapeados a servidores candidatos cada mochila, associada a cada servidor, torna-se independente das demais. Isto, por sua vez, permite que se resolva um problema da mochila independente para cada servidor, ao invés de se resolver um problema mais complexo envolvendo todos os servidores.

Estrutura do artigo. Na Seção 2 é apresentada a literatura relacionada. Na Seção 3, é apresentada a arquitetura da CDN da Globo e o funcionamento do *cache* nos servidores de *Edge*. Na Seção 4, são introduzidas as estratégias clássicas utilizadas para previsão e distribuição de conteúdo. Na Seção 5, as otimizações aplicadas às estratégias de distribuição são descritas e os resultados expostos. Em particular, indicamos os ganhos advindos da solução envolvendo o problema das múltiplas mochilas, visando maximizar a quantidade de usuários assistindo conteúdos pré-posicionados, dadas as características da CDN da Globo. Finalmente, a Seção 6 conclui.

2. Trabalhos Relacionados

A popularização dos serviços de transmissão de conteúdo on-line configurou um cenário promissor para exploração e pesquisa. Os principais trabalhos que basearam o contexto de aplicação deste artigo estão listados a seguir.

Previsão de carga. Uma abordagem para previsão de audiência de vídeos, no contexto da Globo, foi apresentada na obra de [Lima et al. 2021]. Nela, discute-se sobre a plataforma de publicação de vídeos da Globo, e explora-se uma solução de predição de popularidade de novos vídeos à medida que são inseridos na plataforma, via *deep neural networks*. No presente trabalho também considera-se a previsão de audiência, mas com ênfase em vídeos já existentes na plataforma, ao invés dos novos vídeos sendo publicados, para a escolha do conteúdo a ser pré-posicionado diariamente. Também foram exploradas abordagens de alocação de conteúdos em *caches* que usam as previsões de carga.

***Hashing* para *caching*.** O uso de *hashes* para endereçar conteúdos em redes de *caches* é amplamente discutido na literatura de CDNs [Wang and Pai 2002] e *key-value stores* [Atikoglu et al. 2012]. Em geral, trata-se do problema de encontrar o conteúdo, no *cache* de diferentes servidores espalhados geograficamente na rede. Entretanto, não foi encontrado nenhum trabalho que tenha relacionado múltiplas mochilas para cada PoP. Neste trabalho, evidencia-se que ao se considerar múltiplas mochilas por PoP, uma por servidor, é possível resolver o problema de alocação de conteúdos, considerando a localidade da demanda e as restrições inerentes a cada PoP.

Problema da mochila para *caching*. Existem inúmeros trabalhos que estabelecem a conexão entre o problema da mochila e *caching* [Yan et al. 2021, Neglia et al. 2018]. De fato, é natural assumir que cada conteúdo tem um custo de armazenamento proporcional ao seu tamanho e uma recompensa pelo seu armazenamento proporcional a sua taxa de acerto (*hit rate*). Entretanto, não foi encontrado nenhum trabalho que tenha relacionado múltiplas mochilas para cada PoP. Neste trabalho, evidencia-se que ao se considerar múltiplas mochilas por PoP, uma por servidor, é possível resolver o problema de alocação de conteúdos, considerando a localidade da demanda e as restrições inerentes a cada PoP.

3. Arquitetura

Visando entregar vídeos com mais qualidade de serviço e de experiência para seus usuários, a Globo estruturou uma arquitetura que favorecesse esses ganhos. A Seção 3.1 detalha a CDN da Globo no ponto de vista do *Precache*, enquanto a Seção 3.2 detalha como o *cache* é realizado. A Seção 3.3 traz questões práticas associadas ao sistema de pré-posicionamento.

3.1. CDN na visão do *Precache*

Dentre as partes essenciais numa arquitetura CDN, destacam-se os servidores de *Origin* e de *Edge*. O primeiro é a parte responsável pela comunicação com as camadas internas, tais como APIs e bancos de dados, para disponibilizar o conteúdo gerado e que tem seu consumo requisitado. Já os servidores de *Edge* são geograficamente distribuídos. O *Precache* tem por objetivo fazer o pré-posicionamento do conteúdo nos servidores mais próximos aos usuários de forma ótima e baseada no histórico de consumo de cada região.

Na CDN da Globo, há 4 camadas de disposição dos servidores de *Edge*, tendo como base uma arquitetura *multitier*: *tier* 0, 1, 2 e 3. O *tier* 0 não possui o sistema de pré-posicionamento, pois nestes mesmos PoPs estão presentes também os servidores de *Origin*. Os *tier* 1 e 2, distribuídos pelo território nacional, realizam o pré-posicionamento de conteúdo. O *tier* 3 não foi utilizado no contexto deste trabalho.

Com essa arquitetura e com o uso do *Precache* para posicionamento do conteúdo em *cache*, é possível diminuir a latência dos usuários ao diminuir a quantidade de requisições do conteúdo ao servidor de *Origin*, impactando nas métricas de QoE, conforme será mostrado ao longo do artigo.

3.2. Estratégias de *cache*: reativo e proativo

Para a lógica de inserção no *cache*, existem duas abordagens: reativa e proativa.

Na **forma reativa**, a partir de uma requisição de um usuário conectado a um PoP, um vídeo pode ser alocado em *cache* neste PoP, caso ainda não se encontre lá. Em particular, é definido um limite máximo de visualizações que, caso ultrapassado, implica na alocação dos objetos do vídeo (i.e., manifestos e trechos) em um ou mais servidores. Pode-se usar uma função *hash* para determinar qual servidor do PoP deverá armazenar o conteúdo. Em geral, cada vídeo é armazenado em apenas um servidor em cada PoP, variando-se a quantidade de servidores contendo o mesmo vídeo apenas em casos específicos, não considerados neste trabalho (ver também Seção 5.2). Quando todo o espaço disponível para *cache* em um servidor está ocupado, segue-se a política de *Least Recently Used* (LRU) para lógica de remoção de entradas do *cache*.

No pré-posicionamento de conteúdo, em uma abordagem de colocação em *cache* de **forma proativa**, os manifestos e trechos do vídeo são requisitados por uma rotina, que simula a requisição de um usuário para adicionar o conteúdo do vídeo ao *cache*. Esta rotina gera uma alta carga nos servidores de origem, dado que é desejado disponibilizar o máximo de vídeos pré-posicionados o mais rápido possível.

3.3. Questões práticas associadas ao sistema de pré-posicionamento

Após a seleção dos vídeos a serem pré-posicionados, seus metadados são enviados para os servidores de *Edge*. Para cada um desses servidores, uma aplicação agente coloca o conteúdo dos vídeos em *cache*. Essa aplicação utiliza o NGINX como o *webserver* de *cache*, o mesmo já utilizado para o consumo normal de vídeo na CDN da Globo. Dessa forma, todos os trechos de vídeos, contemplando todas as qualidades disponíveis e todos os protocolos de transmissão utilizados na Globo (*HTTP Live Streaming*,[4] *DASH*,[5] *Microsoft Smooth Streaming*)[6], serão colocados em *cache*.

Para o conteúdo ser colocado em *cache*, é necessário consultar os servidores de *Origin*, sendo feita uma requisição para cada trecho de cada protocolo de transmissão usado, bem como seus manifestos. Dado o volume grande de vídeos a serem pré-posicionados e o alto consumo de conteúdo constante na infraestrutura da Globo, foi necessário pensar em uma estratégia condizente com este cenário para não onerar os recursos disponíveis, sobretudo, de rede e disco. Ao mesmo tempo, é desejado que um vídeo com alta audiência prevista esteja pré-posicionado quanto antes, para melhoria da experiência. Também é desejado que novas mídias inseridas na plataforma, caso tenham boa audiência, sejam pré-posicionadas mais cedo possível.

Assim, é realizado diariamente o ranqueamento, a seleção e o pré-posicionamento dos vídeos. Para alocação no *cache*, escolhemos fazer uma estratégia distribuída e concorrente para essas requisições. Emulamos o fluxo de um cliente que toca o vídeo requisitando seu conteúdo, que contempla manifestos e trechos. Tais requisições do conteúdo são feitas em paralelo para diminuir o tempo necessário para pré-posicionamento do vídeo, ao aproveitar a própria infraestrutura da CDN da Globo, que já está apta a lidar com uma alta carga de requisições desse tipo. Além disso, o pré-posicionamento conta com restrições de horário e de quantidade de *bytes* trafegados por segundo, já que não é desejado saturar

[4] `https://datatracker.ietf.org/doc/html/rfc8216`

[5] `https://www.w3.org/2010/11/web-and-tv/papers/` (em particular, submissão 64)

[6] `https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-sstr/8383f27f-7efe-4c60-832a-387274457251`

as redes, tanto de um servidor de *Origin* quanto dos servidores de *Edge*, a fim de não impactar o consumo dos usuários finais

4. Modelos de Previsão e Estratégias de Distribuição de Conteúdo

Para realizar o pré-posicionamento do conteúdo de vídeo, algumas estratégias de distribuição foram aplicadas com base em uma classificação de vídeos definida por um modelo de previsão.

4.1. Previsão do número de requisições por vídeo via *Simple Exponential Smoothing*

Primeiro, a escolha dos vídeos que seriam colocados em *cache* foi feita com base na classificação decrescente das previsões obtidas pelo *Simple Exponential Smoothing*.[7] A previsão é obtida por meio da média ponderada de observações passadas, de forma que observações mais recentes têm maior relevância no resultado final. No cenário da Globo, a série temporal é composta de valores diários de audiência e o período observado é de 30 dias. Esse modelo prevê, paralelamente, a popularidade de milhões de vídeos diariamente com um erro médio absoluto de apenas 1,83 usuários únicos, em que 99% dos vídeos possuem uma previsão com um erro absoluto máximo de 8,29 usuários únicos. Então, são escolhidos os vídeos com as k maiores audiências previstas para serem pré-posicionados nos PoPs.

4.2. Distribuição de conteúdo entre servidores de um PoP

Estratégia *Fill*: múltiplas réplicas por PoP. A estratégia inicial foi popular o *cache* de todos os servidores com área disponível para *Precache* com todos os vídeos selecionados, considerando apenas a quantidade de vídeos passível de armazenamento na área de disco de cada servidor. Essa abordagem é eficaz no que diz respeito à disponibilidade de conteúdo com alta audiência prevista, mas tem um custo alto, pois não considera características da infraestrutura combinadas à escolha do vídeo. Então, no cenário de servidores com menos espaço disponível, caso a previsão indicasse um consumo maior para vídeos com alto uso de espaço, poucos vídeos poderiam ser alocados nesse servidor.

Estratégia *Media Mod*: uma réplica por PoP. Para melhorar a distribuição de conteúdo em cada PoP, foi criada a estratégia de *Media Mod*, que consiste na distribuição dos vídeos selecionados nos servidores disponíveis de um determinado PoP, de acordo com uma estratégia modular. Assim, um dado vídeo é escolhido para ser colocado em *cache* em um único servidor do PoP, permitindo que mais vídeos sejam alocados a um PoP, já que a capacidade de disco do PoP é tratada na totalidade. Note que ao usar *hashes* para mapear vídeos a servidores, esta solução garante que um vídeo seja sempre alocado a um mesmo servidor, caso não haja alteração do número de servidores.

Limitações. As soluções apresentadas até então possuem duas limitações, por não considerarem 1) regionalização e 2) tamanho dos conteúdos. A seguir, serão apresentadas uma estratégia que considera a regionalização (Seção 5.1), e duas adaptações da mesma para considerar o tamanho dos conteúdos (Seções 5.2 e 5.3). As três estratégias são então comparadas entre si (Seção 5.4).

[7] `https://www.statsmodels.org/stable/examples/notebooks/generated/exponential_smoothing.html`

5. Regionalização, Otimização e Resultados

A seguir, foram apresentadas estratégias para otimizar a quantidade de acessos ao conteúdo pré-posicionado, que complementam as soluções previamente discutidas. A princípio, discute-se a regionalização. Em seguida, são explorados algoritmos de mochila considerando o espaço ocupado por cada vídeo (seu peso), e sua popularidade predita (seu ganho). Foram consideradas duas abordagens para o problema das mochilas e, em ambas, solucionamos o problema de otimização correspondente com o uso da biblioteca *OR-Tools*.[8]

5.1. Regionalização

Como adaptar a distribuição de conteúdos conforme a popularidade regional dos mesmos? Para responder à pergunta, foi aplicado o algoritmo de *Vector Autoregression*[9] para a previsão regional de popularidade dos vídeos. Para cada vídeo, o modelo processa sua popularidade histórica em cada região, resultando em cinco previsões de popularidade, uma para cada região. Então, para entender o impacto da regionalização na audiência e na seleção de vídeos, comparou-se o comportamento de audiência em cada região brasileira contra as previsões usando o contexto nacional.

Avaliação de desempenho. Para avaliar o efeito da regionalização das previsões, foi realizada uma análise para verificar como o uso de informações de contexto geográfico afetou a audiência em cada PoP. Dado o fluxo de requisições correspondente ao contexto nacional, aplicamos intervenções correspondentes à regionalização, e comparamos o comportamento do sistema antes e depois da intervenção. A diferença entre as métricas de interesse (por exemplo, visualizações de vídeos), representa o impacto causal.

As peculiaridades de cada PoP complicam quaisquer comparações entre os mesmos. Em vista disso, verificou-se a causalidade por meio de uma análise contrafactual, em que se estima como uma métrica desempenharia caso não houvesse ocorrido uma intervenção. Nesse caso, a intervenção é o uso da audiência prevista por região para a escolha dos vídeos a serem pré-posicionados em detrimento da audiência nacional prevista. Essa estimativa é feita a partir da audiência real que os vídeos da previsão nacional obtiveram em cada estado brasileiro. Baseado nessa audiência, estima-se como esses vídeos teriam desempenhado caso a abordagem da audiência nacional continuasse sendo usada. Dessa forma, é possível calcular o impacto da intervenção no número de visualizações alcançado pelos conteúdos preprosicionados em cada PoP.

Com base nessa análise, concluiu-se que a regionalização teve um impacto positivo para os PoPs com menor espaço disponível, ao notar que houve um aumento no número de visualizações por PoP. Entretanto, a mudança acarretou uma queda do número de visualizações para os PoPs de maior capacidade. Esse impacto se deve ao fato de que os PoPs com maior capacidade atendem não só a região em que eles estão inseridos, mas acabam por atender usuários provenientes de outras regiões. Um exemplo disso é o PoP de Brasília, que por sua grande capacidade e posicionamento estratégico, atende também boa parte dos usuários fora da região Centro-Oeste. *Considerando estas observações, no*

[8] `https://developers.google.com/optimization`
[9] `https://www.statsmodels.org/dev/vector_ar.html`

*restante do artigo denotamos por **solução envolvendo regionalização** aquela em que os PoPs de menor capacidade passam a utilizar a previsão regional de audiência, enquanto os de maior capacidade utilizam a previsão nacional.*

5.2. Problema das Múltiplas Mochilas por PoP

Em seguida, são aplicados algoritmos de otimização para alocação do conteúdo a partir das previsões obtidas dos modelos citados anteriormente. A otimização consiste na solução do problema clássico das Múltiplas Mochilas, sendo as mochilas as capacidades dos servidores de cada PoP e a função objetivo maximizar a audiência total dos vídeos selecionados para o PoP que sejam pré-posicionados em servidores de *cache*.

Considere:

- V conjunto dos vídeos selecionados para o PoP.
- S conjunto dos servidores do PoP.
- t_v o espaço ocupado em disco pelo vídeo $v \in V$.
- a_v é a audiência **prevista** para o vídeo $v \in V$ (a previsão é obtida a partir dos algoritmos descritos nas Seções 4.1 e 5.1).
- $c_s \in S$ a capacidade em disco do servidor $s \in S$ reservada para *cache*.
- $p_{v,s} = \begin{cases} 1 & \text{se vídeo } v, \text{ será pré-posicionado no servidor } s \\ 0 & \text{caso contrário} \end{cases}$

O problema de otimização é um problema de programação linear inteira:

$$\text{PROBLEMA DAS MOCHILAS: maximizar} \sum_{s \in S} \sum_{v \in V} p_{v,s} \cdot a_v \tag{1}$$

$$\left(\sum_{v \in V} p_{v,s} \cdot t_v \right) \leq c_s, \quad \forall s \in S \tag{2}$$

$$\sum_{s \in S} p_{v,s} \leq 1, \quad \forall v \in V \tag{3}$$

As duas restrições evidenciam os seguintes fatos: 1) o tamanho dos vídeos não pode exceder a capacidade de um servidor; 2)um vídeo deve estar em, no máximo, um servidor do PoP.

Sobre a restrição de um vídeo só poder ser replicado em um servidor por PoP. Verificou-se que, para a carga de trabalho atual, armazenar no máximo uma réplica de cada conteúdo por PoP é benéfico. Em trabalhos futuros, pretende-se considerar cargas alternativas, nas quais o gargalo possa estar nos servidores, motivando assim, a possibilidade de se armazenar mais de uma réplica de um mesmo conteúdo por PoP.

Dessa forma, a partir de uma relação entre o tamanho e a audiência prevista, os vídeos são distribuídos diretamente entre os servidores de um PoP através do algoritmo de múltiplas mochilas, como ilustra a Figura 1(a). Esta abordagem resultou em um aumento de 75% na quantidade de visualizações dos PoPs, em comparação com a abordagem de previsão via *regionalização* e alocação de conteúdo via *Media Mod* (sem otimização, conforme descrito na Seção 4).

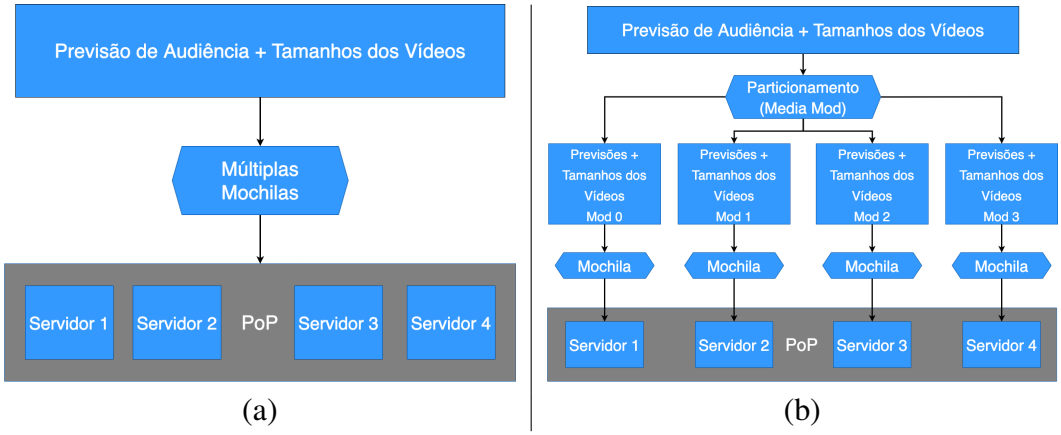

Figura 1. Fluxo do Problema de Múltiplas Mochilas por PoP, (a) com servidores acoplados e (b) com servidores desacoplados.

Problema prático de posicionamento dos conteúdos nos servidores. Contudo, dois novos problemas surgiram. Primeiro, os vídeos não são sempre alocados para os mesmos servidores em execuções consecutivas do PROBLEMA DAS MOCHILAS. Assim, vídeos que eram repetidos diariamente poderiam ser pré-posicionados em servidores diferentes, tomando mais tempo para realizar o *cache* do vídeo, pois o seu conteúdo não estava presente no servidor em que foi alocado. Segundo, a restrição (3) acopla as várias mochilas, tornando a solução do problema das mochilas potencialmente muito custosa. Para contornar tais desafios, consideramos o uso de *hashes* para mapear conteúdos a servidores candidatos para armazenar tais conteúdos, conforme discutido a seguir. O uso de *hashes* naturalmente resolve os dois desafios, por garantir que 1) cada conteúdo seja sempre mapeado para o mesmo servidor candidato e 2) haja desacoplamento entre várias mochilas, permitindo-se que se resolva vários problemas menores ao invés de um problema grande.

5.3. *Hashes* de Conteúdos para Servidores: Simplificando a Solução do Problema das Mochilas via Desacoplamento dos Vários Servidores

A seguir, apresentamos uma estratégia simples para simplificar a solução do problema das mochilas apresentado na última seção. Note que a restrição (3) gera um acoplamento entre as várias mochilas. Por outro lado, desacoplar as mochilas nos permite resolver cada um dos problemas, de cada servidor, de forma independente dos demais.

A seguir, nosso objetivo é ilustrar uma forma de desacoplar as várias mochilas. Para tal, em cada PoP, inicialmente é estabelecido um mapeamento entre cada conteúdo e o respectivo servidor candidato a potencialmente armazenar o conteúdo. Em seguida, resolvemos o problema da mochila por servidor, para determinar quais os conteúdos que cada servidor efetivamente irá armazenar. Além de simplificar a alocação de conteúdos e a solução do problema da mochila, o *hashing* dos conteúdos aos servidores também resolve um problema de ordem prática, que consiste em determinar, de forma eficiente, qual dos servidores de cada PoP que armazena qual conteúdo.

5.3.1. Predeterminando o Mapeamento Conteúdo-Servidor Conseguimos Desacoplar os Servidores

A fim de contornar os desafios da estratégia anterior, foi proposta uma extensão da mesma, ilustrada na Figura 1(b):

1. primeiro, particiona-se o grupo de vídeos V por meio do *Media Mod*, gerando $|S|$ grupos de vídeos, um grupo para cada servidor;
2. em seguida, aplica-se um algoritmo de otimização para resolver o problema de uma mochila simples para cada servidor, individualmente.

Foi definido, para cada servidor s, um problema de mochila simples. Assim, focando em um servidor s, fazem-se necessárias as seguintes definições adicionais:

- $V' \subseteq V$ como subconjunto dos vídeos selecionados para o servidor.
- $p_v = \begin{cases} 1 & \text{se vídeo } v \in V', \text{ será pré-posicionado no servidor} \\ 0 & \text{caso contrário} \end{cases}$

5.3.2. Solução do Problema da Mochila para Servidor Pré-Selecionado

Seja s um dos servidores. A seguir, apresenta-se o problema a ser resolvido pelo servidor s, responsável por armazenar alguns dos conteúdos no conjunto $V' = \texttt{conteúdos}(s)$, em que o mapa $\texttt{conteúdos}(s)$ retorna o conjunto de conteúdos que pode ser potencialmente armazenado no servidor s.

PROBLEMA DA MOCHILA PARA SERVIDOR PRÉ-SELECIONADO:

$$\text{maximizar} \sum_{v \in V'} p_v \cdot a_v \tag{4}$$

$$\sum_{v \in V'} p_v \cdot t_v \leq c_s \tag{5}$$

Note que, para esse caso, há apenas uma restrição: o tamanho dos vídeos não pode exceder a capacidade do servidor.

Pode-se verificar que, após resolver-se as várias instâncias do problema citado, uma para cada servidor, a solução final do problema é equivalente à solução do PROBLEMA DAS MOCHILAS, desde que nesse último se substitua a última restrição (3) por

$$0 \leq p_{v,s} \leq \begin{cases} 1 & \text{se o } \textit{hash} \text{ de } v \text{ o direciona para o servidor } s, \\ & \text{ou seja, se } v \in \texttt{conteúdos}(s) \\ 0 & \text{caso contrário} \end{cases} \tag{6}$$

Qual o "pulo do gato"? Note que o "pulo do gato" para converter o PROBLEMA DAS MOCHILAS no PROBLEMA DA MOCHILA PARA SERVIDOR PRÉ-SELECIONADO consistiu em estabelecer, de antemão, um mapeamento entre cada conteúdo e o eventual servidor candidato a armazenar tal conteúdo. Conforme discutido anteriormente, tal mapeamento simplifica a solução do problema da mochila, que agora pode ser dividido em

um problema da mochila independente por servidor. Além disso, simplifica também a implementação prática da busca de conteúdos nos servidores, resolvendo o problema prático de posicionamento dos conteúdos nos servidores, apontado ao final da Seção 5.2. Ressalta-se ainda que o desacoplamento da solução por servidor possibilitou que as diversas otimizações pudessem ser realizadas paralelamente, diminuindo consideravelmente o tempo necessário para a resolução do problema. Cabe destacar, no entanto, que o mapeamento entre conteúdos e servidores candidatos, dado pela função $\texttt{conteúdos}(s)$, é crítico para a operação do algoritmo, e um mapeamento inadequado poderá ter efeitos negativos sobre o desempenho do sistema. Em experimentos, verificou-se que regras simples, como o uso da função de módulo para determinar o servidor associado a cada conteúdo, são suficientes para alcançar bom desempenho, conforme discutido a seguir. Destacamos também que tal mapeamento é em geral negligenciado na literatura de *caching*, por tratar-se de uma decisão técnica possivelmente envolvendo aspectos estratégicos e proprietários, embora em alto nível o tema seja abordado, por exemplo, em [Wang and Pai 2002].

Aumento na taxa de visualização em *cache*. Essa solução alcançou um aumento de 80,1% no número de visualizações servidas por *caches* dos PoPs, em comparação com a abordagem de previsão via *Regionalização* e alocação de conteúdo via *Media Mod* (sem otimização, conforme descrita na Seção 4). Neste cenário, é possível aproveitar o pré-posicionamento de vídeos feito em dias anteriores, em um mesmo servidor, pois os vídeos pré-posicionados se repetem sempre no mesmo servidor. Dessa forma, foi possível aumentar a quantidade de vídeos pré-posicionados sem onerar a solução como um todo, otimizando a distribuição de vídeos e a complexidade de execução da solução.

5.4. Resultados

A seguir, reportamos e ilustramos os resultados obtidos. Comparamos estratégia sem otimização, usando regionalização (Seção 5.1); e com otimização, usando múltiplas mochilas por PoP, acopladas (Seção 5.2) e múltiplas mochilas por PoP, sendo cada mochila por servidor desacoplada das demais via *hashes* que mapeiam conteúdos a servidores (Seção 5.3). Cada solução foi avaliada por um período de cerca 14 dias entre os meses de março e maio de 2022, sendo que existiram intervalos entre as implementações das soluções, visto que nem sempre a troca de uma solução no ambiente de produção ocorre de maneira suave, sendo necessários eventuais ajustes. *Os resultados apresentados correspondem às soluções aplicadas no ambiente de produção da Globo, impactando diretamente no funcionamento atual da plataforma.*

A Figura 2 ilustra os resultados obtidos, para 10 PoPs. Nesta figura, é possível acompanhar a evolução das distribuições e a variabilidade das visualizações para cada PoP em escala logarítmica. Note que o uso da otimização impactou positivamente os PoPs maiores (do 1 ao 5). Aplicando o teste de hipótese t-test, verificamos um p-valor menor ou igual a 0.005 para o ganho no número de visualizações para os PoPs 1, 3, 4 e 5, o que indica que o ganho é estatisticamente significativo nesses casos. Entretanto, para os PoPs menores (do 6 ao 9), o aumento do número de visualizações não é estatisticamente significativo, já que a média e o desvio-padrão tiveram pouca alteração. Como explicado no parágrafo "Análise de Desempenho" na Seção 5.1, a partir de uma análise

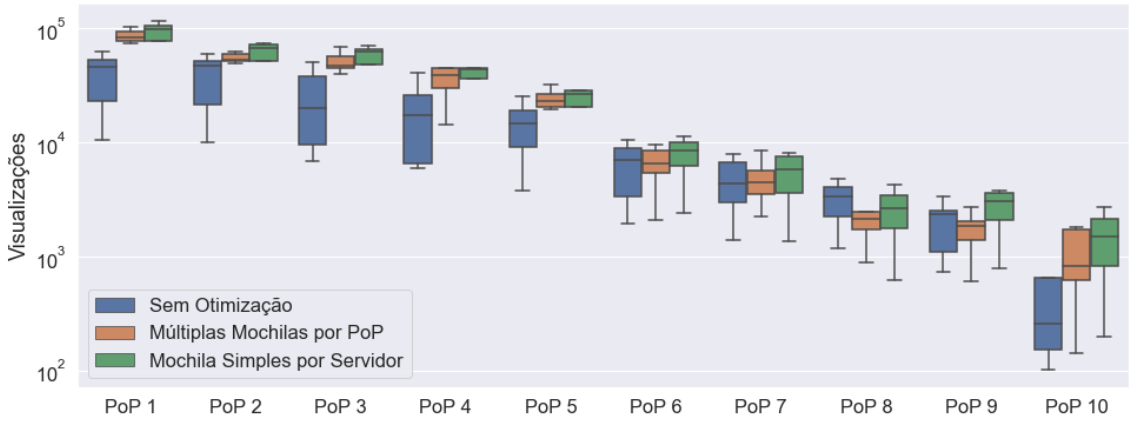

Figura 2. Visualizações obtidas por vídeos pré-posicionados nos PoPs.

contrafactual, os principais ganhos para os PoPs menores vieram com a regionalização (já contemplada no caso ao qual nos referimos aqui como "sem otimização").

Na maioria dos PoPs, para as duas soluções com otimização (em laranja e verde), é possível notar um comportamento estatisticamente similar em termos de número de visualizações. A diferença de um otimizador para o outro está, principalmente, na complexidade, conforme discutido na Seção 5.3.2. Vale ressaltar que o PoP 10 apresentou problemas no período analisado e seus resultados podem não refletir o uso da otimização.

Por meio das otimizações realizadas, notou-se um aumento da quantidade de visualizações para conteúdos pré-posicionados. Foi observado que até 40% das visualizações totais de um PoP eram provenientes do consumo de vídeos pré-posicionados, os quais ocupavam apenas 10% do espaço em cada servidor. Os demais 90% eram ocupados por conteúdos armazenados de forma reativa (Seção 3.2).

Com o vídeo completo disponível mais próximo do usuário, foi possível também ver um aumento na qualidade de experiência por meio do indicador de *bitrate* médio, que indica com qual qualidade o vídeo foi assistido. Considerando todos os vídeos transmitidos pela CDN, incluindo aqueles que envolvem *cache* reativo e *cache* proativo, foi observado um aumento de até 10% no indicador. A Tabela 1 mostra o aumento percentual do *bitrate* para as duas abordagens em relação ao período sem otimização.

Contrastando a Figura 2 contra a Tabela 1, percebemos que o aumento do bitrate expresso nesta última é complementar ao crescimento do número de visualizações. Vale ressaltar que em todos os PoPs percebemos um acréscimo de bitrate com o uso da otimização, cujo aumento varia de 2% e 5%.

6. Conclusão

Neste trabalho foram apresentados algoritmos para pré-posicionamento de conteúdos em *caches* de PoPs de uma CDN. Ademais, foi indicado como usar o histórico de visualizações de um vídeo, a fim de implementar o *cache* proativo de conteúdo considerando o tamanho dos vídeos, predição de visualizações e características regionais de demanda. Usando a infraestrutura de CDN da Globo, obteve-se ganhos expressivos, e.g., da ordem de 80% em termos de número de visualizações suportadas pelos espaços de *cache*, resolvendo-se uma instância do problema das mochilas por cada servidor de PoP.

Este artigo abre inúmeras direções para trabalhos futuros. Outras funções objetivo podem ser exploradas, por exemplo, visando maximizar métricas de QoE como extensão à maximização da quantidade de visualizações servidas pelos *caches*. A decisão de quais vídeos alocar em *cache* pode também considerar outros parâmetros de entrada, como categoria e duração dos vídeos. Além disso, para demonstrar a significância estatística nas visualizações entre as soluções testadas, pretendemos aumentar o período de observação para cada uma das soluções, assim como o detalhamento temporal, considerando, por exemplo, os dias da semana e reexecutar as observações em paralelo, de modo a comparar diretamente os alcances de visualização. Por fim, é necessário analisar cada PoP independentemente, já que possuem diferentes particularidades e configurações.

Tabela 1. Acréscimo percentual de *bitrate* suportado para serviço aos clientes, fruto da otimização, contra caso base com regionalização sem otimização.

PoP	Otimização (%)	PoP	Otimização(%)	PoP	Otimização(%)
1	2.132540	4	4.460024	7	4.399732
2	3.275138	5	3.839312	8	4.660927
3	2.588841	6	5.842910	9	4.721776
				10	2.050560

Referências

Atikoglu, B., Xu, Y., Frachtenberg, E., Jiang, S., and Paleczny, M. (2012). Workload analysis of a large-scale key-value store. In *ACM SIGMETRICS/PERFORMANCE Measurement and Modeling of Computer Systems*, pages 53–64.

Hasan, S., Gorinsky, S., Dovrolis, C., and Sitaraman, R. K. (2014). Trade-offs in optimizing the cache deployments of cdns. In *IEEE INFOCOM 2014-IEEE conference on computer communications*, pages 460–468. IEEE.

Hasslinger, G., Ntougias, K., Hasslinger, F., and Hohlfeld, O. (2020). General knapsack bounds of web caching performance regarding the properties of each cacheable object. In *2020 IFIP Networking Conference (Networking)*, pages 821–826. IEEE.

Lima, D. S., Oliveira, B. G., Mendes, P. R. C., Costa, L., and Coelho, Y. (2021). An ML-Based Approach for Near Real-Time Content Caching. In *Workshop on Design, Deployment, and Evaluation of Network-assisted Video Streaming*, pages 8–14.

Neglia, G., Carra, D., and Michiardi, P. (2018). Cache policies for linear utility maximization. *IEEE/ACM Transactions on Networking*, 26(1):302–313.

Poularakis, K., Iosifidis, G., Argyriou, A., et al. (2016). Caching and operator cooperation policies for layered video content delivery. In *INFOCOM*, pages 1–9. IEEE.

Shi, F., Fan, L., Lai, X., Chen, Y., and Lin, W. (2021). A hierarchical caching strategy in content delivery network. *Computer Communications*, 179:92–101.

Wang, L. and Pai, V. (2002). The Effectiveness of Request Redirection on CDN Robustness. In *5th Symposium on Operating Systems Design and Implementation (OSDI)*.

Xu, Q. (2021). 0-1 knapsack problem driven resource scheduling in caching-enabled network: A case study on sports video. *Internet Technology Letters*, 4(5):e296.

Yan, G., Li, J., and Towsley, D. (2021). Learning from optimal caching for content delivery. In *CONEXT*, pages 344–358.