# Desafios no Escalonamento de Serviços de *Cache* em Provedores de Nuvem e na Borda da Internet

**Peter L. Brendel**[1], **Guilherme P. Koslovski**[1]

[1]Universidade do Estado de Santa Catarina (UDESC) – LabP2D

***Resumo.*** *A fim de reduzir a distância de acesso a informação, provedores de nuvem e Edge Computing (EC) trabalham para carregar dados em cache e fornecê-los aos seus clientes. Contudo, a mobilidade dos usuários diminui a confiabilidade das conexões e, por este motivo, um nó de EC utilizará o enlace para carregar a cache que pode ser inutilizada pelo usuário, causando não apenas o mau uso dos enlaces, mas também má administração de memória. O presente trabalho apresenta uma abordagem para atenuar o problema exposto, discutindo um escalonamento eficiente de cache com auxílio do Transmission Control Protocol (TCP).*

## 1. Introdução

O escalonamento de serviços entre provedores de nuvem e hospedeiros nas bordas da Internet é uma etapa essencial para os usuários de dados móveis. Para esses, um escalonamento proporciona um acesso rápido e de baixo custo energético se realizado da maneira correta. Isso se deve ao fato de ser possível reduzir a distância entre o usuário e o serviço que ele busca na rede. Embora o poder de processamento dos aparelhos celulares mais recentes tenha evoluído drasticamente, pode não ser suficiente para algumas atividades que demandam alto processamento em um curto período de tempo. Sendo assim, o processamento de algumas tarefas na nuvem é necessário [Mach and Becvar 2017].

Por se tratar de aparelhos móveis, é necessário levar em consideração a mobilidade do usuário, que pode estar se afastando do fornecedor do serviço. No modelo atual de escalonamento de serviços de *cache*, não há uma garantia de que o usuário realmente queira ou seja capaz de utilizar o serviço armazenado. Desta forma, pode haver a movimentação de recursos desnecessários entre os provedores de nuvem e de borda. Quanto a comunicação de dados, os principais desafios no escalonamento são: criação de interconexões entre provedores; e manutenção da conectividade com o usuário. Numa sessão de dados, existem no mínimo dois pares de conexão TCP, do usuário à *Edge* e da *Edge* com a nuvem. Através do TCP é possível utilizar o Número de Sequência (SEQ) e de *Acknowledgement* (ACK) para verificar se a conexão do usuário está instável. Assim, um gatilho pode ser acionado informando ao escalonador a necessidade de uma adaptação, possivelmente alterando as conexões para outras *Edges* e movimentando os dados. Diante do exposto, o presente trabalho define o problema de escalonamento entre provedores de nuvem e de EC apresentando uma proposta para escalonamento baseada nas informações do TCP.

## 2. Trabalhos Relacionados

Mach e Becvar (2017) definem que a busca por *Quality of Service* (QoS) para os usuários é essencial e, para garantir a qualidade, é importante notar que ela depende de diversos aspectos. Os autores realizaram um estudo comparativo que relacionou diversos trabalhos,

focando, dentre outros aspectos, na alocação de recursos computacionais e objetivando reduzir o atraso no descarregamento da aplicação. Em outras palavras, garantindo *QoS* aos usuários conectados aos provedores de borda [Mach and Becvar 2017].

Dois aspectos essenciais foram discutidas por Li e seus colegas: O primeiro é controle de energia, é importante que a bateria do usuário seja preservada, mas também é preciso controlar o gasto energético dos provedores. Além disso, com foco em mobilidade sugerem uma alocação de recursos dinâmica de camadas. O modelo busca diminuir o consumo de energia dos dispositivos *Internet of Things* (IoT) e dos nós de borda [Li et al. 2018]. Por sua vez, o controle de congestionamento e a busca por otimização de rotas foi abordado pelo Facebook, através de um sistema baseado em *Software Defined Networking* (SDN) que busca e atualiza informações para realizar a tomada de decisões [Schlinker et al. 2017].

Na arquitetura de armazenamento e organização de repositórios *(Cloud4NetOrg)*, um componente essencial é a *Cache* [Andriani et al. 2018]. No trabalho citado, os autores apresentam duas camadas de *cache* que aceleram o acesso aos arquivos, a primeira camada local e a segunda no repositório na rede. Por fim, o real desafio deste trabalho é, além de descobrir padrões de mobilidade, garantir a eficiência no escalonamento, isto é, ser capaz de mover a *cache* e redirecionar as conexões com o usuário sem ter gastos desnecessários de recursos computacionais (processamento, comunicação e armazenamento).

## 3. Definição do Problema

Manter estabilidade nas conexões é essencial, porém, é possível avançar um passo e, a partir da instabilidade, permitir a continuidade da conexão? No estado atual dos provedores de EC isso não é possível, ao se deslocar pela cidade, por exemplo, um usuário pode ter sua conexão interrompida e perder todos os dados escalonados pelo provedor. Esse modelo não apenas dificulta a conexão para o usuário, mas também reserva recursos nos provedores, limitando a conexão de novos clientes. O objetivo é garantir a continuidade da conexão, isto é, permitir a troca de provedores de EC sem a percepção do usuário, isto pode ser alcançado se for possível manter os dados previamente coletados na nuvem.

A Figura 1 apresenta a simplificação do cenário de execução em estudo. Apenas um usuário está conectado ao provedor de EC *A*, utilizando sua *cache* com uma conexão estável. O usuário move-se de maneira a se afastar do provedor *A* e ir em direção ao provedor *D*. Ou seja, sua conexão com o provedor *A* torna-se instável ao longo do tempo, fato analisado pela conexão TCP através do aumento de ACK duplicados e do *Round-Trip Time* (RTT), ativando um evento no sistema, que realiza a movimentação da *cache* e reconexão com os provedores vizinhos, até atingir o provedor *D*, onde a conexão é a mais estável. Esses eventos (aumento da latência e detecção de ACKs duplicados) são gatilhos para acionar o escalonamento do sistema. É importante notar as interconexões entre os provedores de nuvem e de EC vizinhos, representadas através de arestas na Figura 1. Ao detectar que o usuário possivelmente está se afastando do ponto de acesso atual, é enviada uma solicitação aos provedores vizinhos para realizar os testes de conexão, vazão e capacidade, antecipando a conectividade futura. Os resultados dos testes definem o destino da conexão do usuário. No exemplo da Figura 1, o usuário move-se em direção ao provedor *D* e, ao se afastar de seu provedor atual, o gatilho será acionado pelo aumento da latência de acesso ao provedor *A*, bem como a direção futura será indicada pela constante

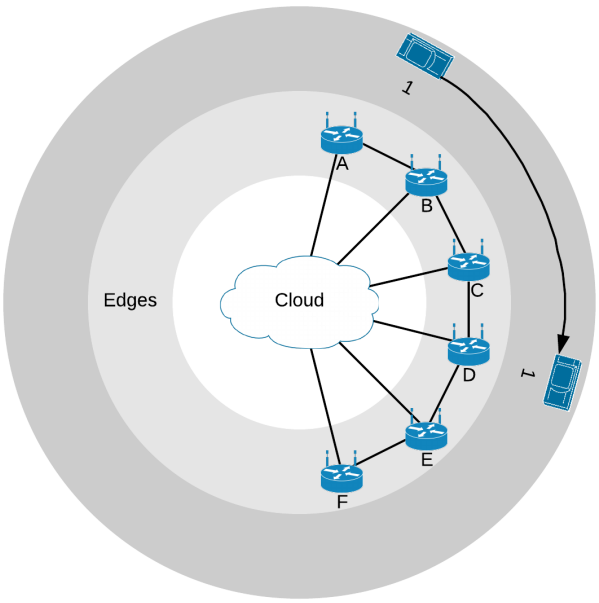


**Figura 1. Exemplo do cenário em estudo: usuário conectado ao provedor de *Edge* A, em movimento constante afastando-se do ponto de conexão.**

diminuição dos valores no vetor de direção do provedor *D*.

Considerando a mobilidade da Figura 1, armazenar *cache* no provedor de EC *A* torna-se ineficiente ao sistema, pois, além de consumir um espaço que não será acessado pelo usuário, utilizará parte da vazão do enlace de acesso até o provedor de nuvem para carregar a *cache*. Contudo, através da interconexão entre os provedores de EC, é possível mover a *cache* entre provedores de EC sem acessar o repositório na nuvem. Desta forma, garante-se um melhor aproveitamento do espaço e um acesso menor à nuvem.

Outro problema é o congestionamento do enlace entre os servidores de EC e os provedores de nuvem. Um nó de EC pode ter seu enlace sobrecarregado por um usuário. Para amenizar o problema, é possível transferir dados ao nó de destino aproveitando a interconexão com outros provedores de EC adjacentes. Ou seja, utilizando a interconexão entre os vizinhos como um caminho temporário para diminuição do gargalo.

### 3.1. Parâmetros dos Provedores

Alguns parâmetros foram identificados como essenciais para formalização do escalonamento. As **matrizes de latência e de largura de banda** entre provedores de *Edge* e nuvem são necessárias para representar a capacidade do sistema, obtida através de sistemas de monitoração. Por fim, a **capacidade de *cache***, demonstra o estado atual do recurso de armazenamento disponível para usuários.

### 3.2. Variáveis e Função Objetivo

Para formalização do problema na perspectiva dos provedores de Nuvem e de EC, é apresentado um modelo baseado na análise do ambiente, que neste caso é inferido pela conexão TCP entre dois pares: provedores de nuvem e de EC; e repositórios e usuários.

O cabeçalho do protocolo TCP fornece informações de SEQ e ACK e, a partir delas, é possível identificar a variável $e$ **= estabilidade da conexão**. Além disso, essas mesmas informações acompanhadas do RTT fornecem a variável $p$ **= *ping* (latência da comunicação)**. Na perspectiva dos usuários, uma **matriz de vazão** ($D$) define a vazão necessária (demanda) para alimentar os usuários. É importante notar que a vazão do provedor de *Edge* deve suportar os *downloads* do usuário. Nesse cenário, as variáveis definidas anteriormente serão associadas ao usuário $u$ para representar o objetivo do escalonador. Desta forma, $e_u$, $p_u$ e $d_u$ representam os valores buscados. Sendo $t$ o tempo e $t_0 < t_1$, a estabilidade da conexão tende a diminuir $(e_u(t_1) < e_u(t_0))$, devido à ocorrência de ACKs duplicados, a latência na comunicação tende a aumentar $(p_u(t_1) > p_u(t_0))$, fato que pode ser detectado através do aumento constante do *RTT*. Assim, o objetivo do escalonamento é selecionar o melhor provedor para receber os dados da *cache* do provedor atual e redirecionar a conexão do usuário e do provedor de nuvem.

## 4. Considerações e Propostas

No cenário atual podemos observar o sistema de EC como um conjunto de conexões independentes. Porém, a necessidade de reduzir o gasto de recursos computacionais e a alta mobilidade fornecida por aparelhos móveis requer que o serviço de escalonamento de *cache* seja rápido e móvel, isto é, as conexões devem ser ampliadas criando uma sub-rede entre os provedores de EC.

Conforme apresentado na seção 3, a proposta é realizar a composição de grafos e utilizar as informações concedidas pelo TCP para atualizar as tabelas necessárias e, desta maneira, garantir o funcionamento do sistema. Na próxima etapa, o problema será formalizado como *Mixed Integer Linear Programming* (MILP) uma vez que as informações podem ser representadas, em sua maioria, como valores inteiros. Posteriormente, a formulação será estudada com o *IBM ILOG CPLEX Optimization Studio* (CPLEX).

## Referências


Andriani, G., Godoy, E., Koslovski, G., Obelheiro, R., and Pillon, M. (2018). An architecture for synchronising cloud file storage and organisation repositories. *International Journal of Parallel, Emergent and Distributed Systems*, 0(0):1–17.

Li, S., Zhang, N., Lin, S., Kong, L., Katangur, A., Khan, M. K., Ni, M., and Zhu, G. (2018). Joint admission control and resource allocation in edge computing for internet of things. *IEEE Network*, 32(1):72–79.

Mach, P. and Becvar, Z. (2017). Mobile edge computing: A survey on architecture and computation offloading. *IEEE Communications Surveys Tutorials*, 19(3):1628–1656.

Schlinker, B., Kim, H., Cui, T., Katz-Bassett, E., Madhyastha, H. V., Cunha, I., Quinn, J., Hasan, S., Lapukhov, P., and Zeng, H. (2017). Engineering egress with edge fabric: Steering oceans of content to the world. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, SIGCOMM '17, pages 418–431, New York, NY, USA. ACM.