AVALIAÇÃO DE POLÍTICAS DE PARTICIONAMENTO DE DADOS EM CACHE WEB MULTI-CLIENTE

Ana Virgínia Nery [1], **Thiago Emmanuel Pereira** [2]

RESUMO

Caches de sistemas multi-locatários apresentam o problema da interferência. Esse problema resulta da remoção de itens do cache de um locatário pela inserção de itens de outro locatário. A remoção desses itens pode afetar o desempenho do cache quando esses itens são acessados, resultando em *cache misses*. A política *Memshare* mitiga os problemas de interferência em cache multi-locatários através de um mecanismo de particionamento da capacidade do cache. Essa política foi avaliada por simulação de uma carga de trabalho de um sistema de multi-locatário de produção e apresentou bons resultados quando comparada com um cenário base, sem política de particionamento. Apesar dos bons resultados reportados, a descrição da proposta da política *Memshare* não apresenta os detalhes suficientes para sua reprodução. Por conta disso, não é possível saber se a proposta é adequada em outros cenários de uso, diferentes do cenário original. Neste trabalho, reproduzimos a avaliação da política *Memshare* para uma carga diferente da original: uma carga de trabalho de um cache web de uma plataforma de comércio eletrônico multi-locatário. Os resultados da reprodução reforçam os resultados obtidos originalmente: apesar de apresentar um desempenho global um pouco menor, a política *Memshare* melhora o desempenho da maioria dos locatários e reduz significantemente a interferência entre eles.

Palavras-chave: Cache web, Multi-locatário, Particionamento.

[1] Aluna de Ciência da Computação, Departamento de Sistemas e Computação, UFCG, Campina Grande, PB, e-mail: ana.virginia.souza.nery@ccc.ufcg.edu.br

[2] Dr., Professor, DSC, UFCG, Campina Grande, PB, e-mail: temmanuel@computacao.ufcg.edu.br

AVALIAÇÃO DE POLÍTICAS DE PARTICIONAMENTO DE DADOS EM CACHE WEB MULTI-CLIENTE

ABSTRACT

Caches in multi-tenant systems present the problem of interference. This problem arises from the removal of items from one tenant's cache due to the insertion of items from another tenant. The removal of these items can affect cache performance when these items are accessed, resulting in cache misses. The Memshare policy mitigates cache interference problems in multi-tenant caches through a cache capacity partitioning mechanism. This policy was evaluated by simulating a workload of a production multi-tenant system and showed good results when compared to a baseline scenario, without a partitioning policy. Despite the good results reported, the description of the Memshare policy proposal does not present enough details for its reproduction. Because of this, it is not possible to know if the proposal is suitable in other use cases. In this work, we reproduce the evaluation of the Memshare policy for a workload different from the original: a workload of a web cache of a multi-tenant e-commerce platform. The reproduction results reinforce the originally obtained results: although presenting a slightly lower overall performance, the Memshare policy improves the performance of most tenants and significantly reduces interference between them.

Keywords: Web cache, Multi-tenant, Partitioning.

INTRODUÇÃO

Caching é uma técnica para promover melhoria de desempenho. Essa técnica consiste em manter temporariamente dados em uma área de armazenamento mais rápida, reduzindo o tempo de acesso aos dados e a carga nos servidores. Em sistemas web, o uso de cache é importante devido ao alto volume de requisições e à necessidade de respostas rápidas. Uma abordagem comum desses sistemas é utilizar cache multi-locatário, no qual diferentes entidades compartilham o recurso de cache.

Em ambientes multi-locatário, o cache é compartilhado por diversas aplicações ou serviços de diferentes entidades, conhecidas como locatários. Empresas provedoras de plataformas de ecommerce, por exemplo, utilizam desse modelo multi-locatário para otimizar seus recursos entre seus diferentes locatários. Cada locatário tem suas próprias aplicações, que são acessadas por vários clientes. As requisições enviadas pelos locatários possuem características específicas que podem variar consideravelmente ao longo do tempo, gerando padrões de uso distintos e desafios para o gerenciamento do sistema (LIRA et al., 2024a; LIRA et al., 2024b).

Uma opção para configurar um serviço de cache em ambientes multi-locatários é utilizar uma instância de cache para cada entidade. Essa abordagem garante que a utilização de cache de um locatário não afete a utilização dos outros, uma vez que estão isolados. No entanto, o gerenciamento torna-se mais complexo, já que é necessário configurar e gerenciar uma instância de cache para cada locatário (ainda mais desafiador quando existem milhares de locatários). Além disso, como a carga dos locatários é variável no tempo, existe o risco de momentos de subdimensionamento - quando há baixa capacidade e poucos itens permanecem no cache, limitando o desempenho do locatário - ou momentos de super dimensionamento - quando é alocado mais espaço de memória do que o necessário, resultando no desperdício de recursos.

Uma outra abordagem é utilizar um único nó de cache para todos os locatários, o que simplifica a configuração e o gerenciamento do serviço de cache. Nesse tipo de ambiente, ocorre multiplexação de recursos, permitindo que locatários com diferentes demandas compartilhem e utilizem a capacidade do cache de forma mais eficiente. Por exemplo, em determinados momentos, alguns locatários podem ter alta demanda, enquanto outros apresentam menor uso de recursos. Essa variação ao longo do tempo possibilita que locatários com menor necessidade momentânea cedam espaço no cache para aqueles com maior demanda, maximizando a utilização da memória disponível e evitando desperdícios. Por esses motivos, é comum que plataformas de comércio eletrônico (com grande quantidade de locatários) optem por essa abordagem de compartilhamento de capacidade.

No entanto, o compartilhamento também apresenta desafios. Associado com

a variabilidade nas demandas, locatários com alta demanda podem gerar interferências, ao removerem itens dos demais locatários do cache. Esses itens, caso sejam requisitados novamente, não estarão mais disponíveis, impactando negativamente o desempenho do sistema e dos locatários interferidos.

Para mitigar esse problema, o cache compartilhado pode ser gerenciado utilizando políticas de particionamento. Essas políticas são usadas para definir os recursos de cache entre diferentes locatários em um ambiente compartilhado em um determinado momento. A alocação garante que cada um tenha uma porção dedicada dos recursos. Isso reduz a interferência, pois evita que locatários com alta demanda consumam recursos em excesso e interfiram no desempenho de outros.

Dessa forma, o gerenciamento do cache pode implementar políticas que busquem equilibrar o *trade-off* entre preservar os benefícios do compartilhamento de recursos e minimizar a interferência entre locatários. Um exemplo é o "Memshare: a Dynamic Multi-tenant Key-value Cache"(CIDON et al., 2017). O Memshare foca em cache web multi-locatário, implementando uma política de particionamento dinâmico, que se adapta às demandas variáveis dos locatários.

O trabalho de Cidon et al. (2017) avaliou uma carga de trabalho proveniente de um provedor comercial que utilizava o serviço Memcachier (MEMCACHIER,), um serviço de *cache-as-a-service* que atende a centenas de locatários. Nesse modelo, os clientes do Memcachier precisam especificar uma quantidade fixa de memória no cache, o que restringe a flexibilidade no uso dos recursos de memória. O estudo comparou o desempenho do cache utilizando a política de particionamento dinâmico implementada pelo Memshare com o particionamento estático do Memcachier.

Embora o Memshare apresente bons resultados em seus experimentos, há uma limitação importante: o estudo não detalha as características da carga de trabalho enviada para o cache. Isso é importante, pois algoritmos e políticas de cache podem ser sensíveis aos padrões de requisições. Uma política pode ter um desempenho adequado ao ser submetida a uma carga de cache e ter um desempenho inadequado ao ser submetida a outro tipo de carga. Um exemplo disso é o particionamento estático, que funciona bem em cargas de trabalho constantes, onde a demanda de memória dos locatários não varia. No entanto, em cenários com picos e variações frequentes, essa política pode ser ineficiente, resultando em subutilização ou falta de memória para locatários com maior demanda.

Utilizamos uma carga de trabalho diversificada, composta por locatários de diferentes tamanhos, com grandes e pequenos volumes de requisições. A carga contém picos em momentos específicos para alguns locatários, enquanto outros mantêm uma constância no número de requisições. Além disso, a repetição de itens requisitados varia bastante: alguns locatários possuem alta repetição em suas requisições, enquanto

outros apresentam baixas taxas de repetição. Com essa carga de trabalho, buscamos avaliar a eficácia da política de particionamento do Memshare em lidar com um ambiente de cache que precisa atender a essas diversas demandas.

Dessa forma, neste trabalho reproduzimos a política de particionamento implementada no Memshare utilizando um simulador. Para isso, utilizamos uma carga de cache web multi-locatário coletada do serviço de cache de uma grande empresa que fornece uma plataforma de comércio eletrônico. Cada locatário corresponde a um *e-commerce* operando de forma independente. Cada *e-commerce* atende a diferentes conjuntos de clientes, com padrões de acesso variados devido a seus nichos de mercado e localizações geográficas. Diante dessa diversidade, torna-se fundamental avaliar como a política do Memshare se comporta ao lidar com essas variações na nossa carga de requisições.

Os resultados deste trabalho mostram que, embora o desempenho global do cache com Memshare seja ligeiramente inferior ao do cenário base, a política melhora o desempenho da maioria dos locatários em todos os cenários avaliados, com até 80% dos locatários beneficiados no melhor caso. Além disso, a política reduz significativamente a interferência entre os locatários, diminuindo a média da interferência em até 70%.

O restante do documento está organizado da seguinte forma. Na seção Fundamentação Teórica, apresentamos a política de particionamento implementada pelo Memshare, explicando seus conceitos, algoritmos e resultados prévios. A seção Metodologia detalha a metodologia de avaliação das políticas de particionamento de dados em cache. Na seção Resultados, é apresentado os resultados obtidos da comparação do desempenho e da interferência no cenário sem particionamento e com o Memshare. Finalmente, na seção Conclusão e Trabalhos Futuros, são discutidas as conclusões da pesquisa, com destaque para as contribuições e implicações dos achados, além de possíveis trabalhos futuros.

FUNDAMENTAÇÃO TEÓRICA

O Memshare (CIDON et al., 2017) é um cache multi-locatário de *key-value* DRAM que gerencia dinamicamente a memória entre aplicações. Ele fornece um modelo de compartilhamento de recursos que garante uma parcela de memória reservada para cada locatário enquanto aloca dinamicamente a memória restante entre os locatários para otimizar o Hit Ratio global. A política opera em dois momentos principais: durante a remoção de itens do cache e na realocação de recursos entre locatários.

Quando o cache está cheio e uma nova solicitação chega, a política do Memshare remove o item menos recentemente utilizado (LRU) do locatário i com a menor *N*. Aqui, i representa um locatário do sistema. A métrica *N* indica o grau de utilização do cache

por cada locatário i, com base na memória alocada, e é calculada da seguinte forma:

$$N_i = \frac{T_i}{A_i} \tag{1}$$

Onde:

- A_i: quantidade de memória utilizada pelo locatário i no cache.
- T_i: quantidade de memória, em bytes, alocada para o locatário i, composta pela soma de:

$$T_i = R_i + P_i \tag{2}$$

Em que:

- R_i: quantidade de memória reservada para o locatário$_i$. O valor inicial da memória definido para o locatário não muda $(T_i \geq R_i)$.
- P_i: memória adicional do locatário, que pode ser ajustada dinamicamente durante a operação do cache com base nas necessidades do locatário.

A inicialização das R_i requer a definição de uma porcentagem da capacidade total do cache alocada como Memória Reservada, que será distribuída para as R_i. Por exemplo, se 50% da capacidade do cache for alocada como Memória Reservada, metade da capacidade do cache será repartida igualmente e alocada para as R dos locatários. Enquanto, os 50% restantes da capacidade do cache serão para P, porção do cache realocada dinamicamente entre os locatários com base nas suas necessidades.

Outro momento chave do Memshare (CIDON et al., 2017) é a realocação de recursos e mudanças na P_i. Isso ocorre quando há uma solicitação para um item que estava previamente no cache, mas foi removido. Assim, nessa situação, a política utilizada deduz que, se o locatário i do item removido tivesse mais espaço, o item não teria sido despejado anteriormente. Portanto, o locatário i poderia ter seu desempenho melhorado com memória adicional. Para que ocorra a alocação de memória adicional, sendo j um locatário diferente de i e selecionado de forma aleatória, uma parte da Pj é realocada para o P_i, em unidades C. Esse valor C é definido inicialmente e arbitrariamente no cache.

Mesmo que o locatário i já esteja utilizando toda a sua T_i, ou seja, $A_i = T_i$, se houver espaço disponível no cache, a requisição de um novo item ainda será alocada. Entretanto, isso fará com que A_i exceda T_i, e o valor de N_i diminuirá. O sistema então interpretará que o locatário está utilizando mais memória do que a indicada,

caracterizando um uso excessivo. Dessa forma, o locatário perderá prioridade no cache, uma vez que sua *N* será mais baixa, tornando-se candidato para remoções durante o processo de limpeza de itens.

METODOLOGIA

Este trabalho foi conduzido por meio de simulações, uma vez que elas proporcionam maior flexibilidade para testar diferentes cenários e parâmetros de maneira controlada, sem a necessidade de uma infraestrutura real complexa e custosa. Para avaliar o desempenho e a interferência das políticas de particionamento, utilizamos um simulador de cache customizado trace-based.

Para a avaliação, utilizamos uma carga de trabalho recebida por um cache do catálogo de uma plataforma de comercio eletrônico. Ela contém 10 horas de requisições, está amostrada em 10%. O total de mais de 56 milhões de requisições que correspondem a cerca de 5 mil locatários.

Para a análise, comparamos o cenário base, onde não foi implementada nenhuma política de particionamento, com a política de particionamento dinâmico Memshare. Variamos a capacidade do cache e metrificamos a performance com base no Hit Ratio e no nível de interferência entre os locatários, utilizando a métrica de Interferência.

MODELO DE SIMULAÇÃO

Para realização dos nossos experimentos, desenvolvemos um simulador de cache que implementa o cenário base, sem particionamento e, também, a política do Memshare.

Para implementação do cache sem utilizar políticas de particionamentos, utilizamos o algoritmo LRU(Least Recently Used) como estratégia de substituição, e a parametrização da capacidade total do cache.

Ademais, foi desenvolvido o cache gerenciado com a política de particionamento Memshare, uma vez que a implementação original não estava disponível e poucos detalhes de sua implementação foram descritos no trabalho original. Além disso, o trabalho configurava a política para um cenário especifico, utilizando segmentos de log, o que difere do nosso ambiente. Então, adaptações foram realizadas para reproduzir o comportamento do Memshare, mantendo a estratégia central da política. Para essa política, o algoritmo de substituição utilizado foi o descrito na Seção Fundamentação Teórica e a parametrização das seguintes configurações do cache:

- *cache_size*: Capacidade total do cache;
- *reserved_proportion*: Proporção da capacidade do cache destinada a Memória Reservada, distribuída para as R_i sendo i um locatário;

- *credit*: Valor de *C*;
- *num_tenants*: Quantidade de locatários na carga de trabalho.

Com base nesses parâmetros, os simuladores são capazes de reproduzir o comportamento de um sistema de cache real sob diferentes condições, permitindo a análise das políticas de particionamento e gerenciamento de memória em diferentes cenários.

O funcionamento do simulador do cache implementado com a política Memshare é da seguinte maneira: quando chega uma nova requisição, o sistema primeiro verifica se o item já está presente no cache. Se estiver, o item é movido para o início da lista LRU do locatário e um HIT é retornado. Caso o item não esteja no cache, o sistema verifica se há espaço disponível. Se o cache estiver cheio, ocorre um processo de eviction, no qual o item no final da lista LRU do locatário com o menor valor de N (métrica de necessidade) é removido, e o N desse locatário é atualizado. Esse processo se repete até que haja espaço suficiente para armazenar o novo item, que é então adicionado ao início da lista LRU do locatário correspondente, o N é atualizado, e um MISS é retornado.

CONFIGURAÇÕES DOS EXPERIMENTOS

No que se refere à capacidade do cache, buscamos avaliar o desempenho do sistema utilizando ou não políticas de particionamento sob vários cenários de capacidade do cache, incluindo casos de superprovisionamento, subprovisionamento e provisionamento ideal. Para isso, definimos a capacidade total do cache como uma porcentagem do *Footprint* da carga total, variando arbitrariamente para os valores mostrados na Tabela 1.

Capacidade do Cache (% *Footprint*)
10, 20, 25, 30, 35, 50, 100

Tabela 1 – Capacidades utilizadas nas simulações.

O *Footprint* $F(T)$ é uma métrica que mede a quantidade em bytes de dados acessados em uma específica janela de tempo T pelo somatório do tamanho das requisições de itens distintos em um certo período. Essa métrica é essencial para analisar a utilização da capacidade do sistema. Matematicamente, o *Footprint* pode ser definido como:

$$F(T) = \sum_{i=1}^{n} R_i \tag{3}$$

onde:

- n é o número de requisições de itens distintos no intervalo de tempo T,
- R_i é o tamanho (em bytes) da i-ésima requisição.

Outro aspecto importante é a proporção de memória reservada para os locatários. Experimentamos três cenários diferentes: 0%, 50% e 100% de *Reserved Proportion*. Sendo i um locatário, no cenário de 0%, toda a memória do cache é destinada às P_i, permitindo realocação dinâmica entre os locatários conforme suas necessidades. Já no cenário de 50%, metade da capacidade do cache é alocada para a R_i, e a outra metade para as P_i, balanceando realocação dinâmica e alocação fixa. Por fim, no cenário de 100%, toda a capacidade do cache é reservada, eliminando a realocação dinâmica.

Além disso, o credit *C*, que define a quantidade de memória realocada dinamicamente, foi configurado com base no tamanho médio dos itens distintos da carga de trabalho, que é de 817 bytes. Esse valor garante que, em média, cada realocação seja suficiente para acomodar pelo menos um item adicional, sem comprometer o equilíbrio entre os locatários.

Por fim, o número de locatários foi configurado com base nos dados da carga de trabalho recebida. Aproximadamente 5 mil locatários foram incluídos nas simulações, refletindo a realidade do ambiente de uma grande plataforma de comércio eletrônico. Esses parâmetros permitiram uma avaliação detalhada das políticas de particionamento sob diversas condições.

MÉTRICAS DE AVALIAÇÃO

Para avaliar o impacto das políticas de particionamento no cache do catálogo multi-locatário, utilizaremos nessa pesquisa a métrica de desempenho, Hit Ratio, que indica a proporção de requisições de dados atendidas pelo cache. Também avaliaremos a métrica de Interferência, que indica o efeito negativo sofrido na performance de um locatário.

O Hit Ratio quantifica a proporção de solicitações de dados que foram atendidas com sucesso do cache (chamadas de Hit Status) em relação ao total de solicitações feitas. Considere dois multiconjuntos: R, contendo todas as solicitações de dados feitas durante um período específico, e H, contendo apenas as solicitações atendidas pelo cache dentro do mesmo período ($H \subseteq R$). O Hit Ratio pode ser calculado usando a seguinte relação:

$$HitRatio = \frac{|H|}{|R|} \tag{4}$$

Um Hit Ratio próximo de 1 significa que quase todas as solicitações estão sendo atendidas pelo cache, reduzindo a latência e melhorando o desempenho do sistema.

Além do Hit Ratio, a Interferência entre locatários é outra métrica crítica para avaliar o impacto das políticas de particionamento. Interferência ocorre quando uma remoção indevida afeta negativamente o desempenho de um locatário.

Primeiramente, uma remoção indevida ocorre quando um item é removido do cache, mas é requisitado ao cache futuramente. Essa situação indica que talvez aquele item não devesse ter sido removido, tendo em vista que a repetição de requisição após sua remoção é uma evidência do potencial desse item de se beneficiar do recurso do cache.

Quando ocorre uma remoção indevida de um item ocasionada pela entrada de outro item de outro locatário, isso significa que ocorreu uma interferência. A ação do locatário interferiu negativamente no desempenho do outro.

Para quantificar a interferência sofrida por um locatário i, definimos I_i como todas as remoções de itens de um locatário i por um locatário j, onde j é diferente de i, e que afetaram negativamente o locatário i.

$$I_i = \frac{R_{i,\text{ret}}}{R_{i,\text{removidos}}} \tag{5}$$

Considerando as seguintes definições:

- L: Conjunto de locatários.
- R_i: Itens do locatário i removidos.
- R_{ij}: Itens removidos do locatário i pelo locatário j.
- $R_{i,\text{ret}}$: Itens do locatário i removidos que foram requisitados novamente.
- $R_{ij,\text{ret}}$: Itens removidos do locatário i pelo locatário j que foram requisitados novamente.
- $R_{i,\text{removidos}}$: Itens removidos do locatário i.

$$R_{i,\text{ret}} = \sum_{j \in L, j \neq i} R_{ij,\text{ret}} \tag{6}$$

$$R_{i,\text{removidos}} = \sum_{j \in L, j \neq i} R_{ij} \tag{7}$$

Então, a interferência é quantificada como:

$$I_i = \frac{\sum_{j \in L, j \neq i} R_{ij,\text{ret}}}{\sum_{j \in L, j \neq i} R_{ij}} \tag{8}$$

RESULTADOS

Nesta seção, os resultados foram divididos na avaliação das políticas de particionamento em cache multi-locatário baseado no Hit Ratio e na Interferência.

Hit Ratio

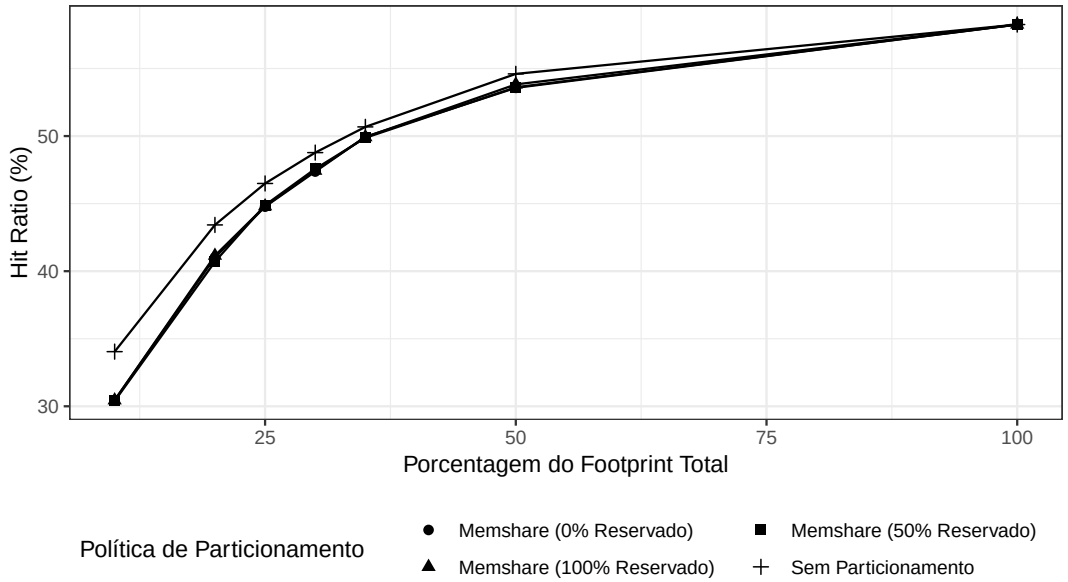

Figura 1 – Hit Ratio global do cache com e sem a política de particionamento Memshare, em diferentes capacidades. Sem utilizar particionamento obtém-se um desempenho levemente superior.

A Figura 1 apresenta no eixo Y o desempenho em Hit Ratio obtido nas simulações do cache sem particionamento e com Memshare (0%, 50%, 100% Reservado), políticas estas representadas pelas linhas com cores diferentes, e no eixo X indica cada capacidade total de cache das simulações, em porcentagem de *Footprint* total. Por exemplo, 10% da porcentagem do *Footprint* total significa que a capacidade do cache corresponde a 10% do *Footprint* total da carga.

Em todas as simulações com a variação de capacidade total do cache, observamos que sem implementar particionamento, o cache apresentou um Hit Ratio superior em comparação com as diferentes configurações do cache utilizando Memshare. No entanto, a diferença entre os cenários é baixa, a menor diferença é 0.7 pontos percentuais, para o caso do cache com Memshare (100% Reservado) e capacidade de 50% do *Footprint* total. E com a maior diferença de 3.6 pontos percentuais, no cenário do cache utilizando o Memshare (50% Reservado) e capacidade total de 10% do *Footprint* total.

Observando mais a fundo o comportamento individual dos locatários, na Figura 2 é dado no eixo Y a porcentagem dos locatários que, comparado com o cenário base do sem particionamento, tiveram seu Hit Ratio Melhorado, Piorado ou Mantido, o eixo X

representa a capacidade total do cache por porcentagem do *Footprint* total e cada sub gráfico é uma configuração de Memória Reservada da política Memshare.

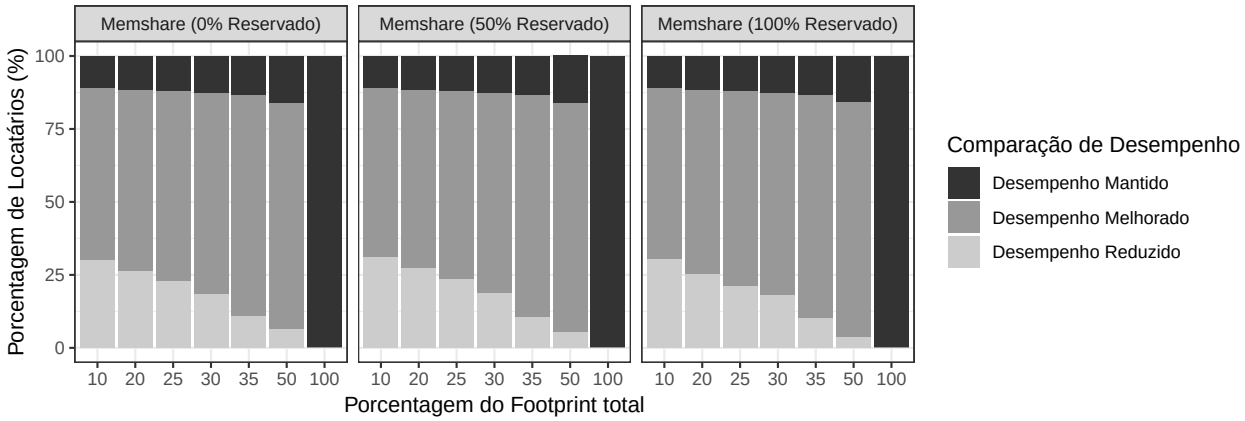

Figura 2 – Comparação do desempenho dos locatários com a política Memshare em diferentes configurações de memória reservada, em relação ao cenário sem particionamento. A política Memshare mostra melhorias de desempenho para a maioria dos locatários.

Percebemos que, embora o cenário sem particionamento de cache tenha apresentado o melhor desempenho global, como ilustrado na Figura 2, o cache utilizando a política de particionamento Memshare conseguiu, em todos os casos e capacidades, melhorar o desempenho da maioria dos locatários.

No melhor caso, aproximadamente 80,2% dos locatários melhoraram seus Hit Ratio com o cache utilizando a política de particionamento Memshare (100% Reservado) e com capacidade total equivalente a 100% do *Footprint*. Ainda assim, no pior cenário, o cache utilizando a política Memshare (50% Reservado), com apenas 10% do *Footprint* total como capacidade de cache, melhorou o desempenho de 57,6% dos locatários em comparação com o cache sem particionamento.

Por outro lado, o pior cenário em termos de locatários que tiveram seus desempenhos reduzidos, em comparação com o cache sem particionamento, ocorreu com 31,2% dos locatários no caso do cache utilizando a política de particionamento Memshare (50% Reservado) e com 10% do *Footprint* total como capacidade. No melhor cenário, apenas 3,7% dos locatários apresentaram uma piora no desempenho, no caso quando utilizado a política Memshare (100% Reservado) e capacidade de 50% do *Footprint* total.

A Figura 2 indica que, embora o cache utilizando a política de particionamento Memshare melhore o desempenho da maioria dos locatários em todos os casos, os locatários que tiveram redução no Hit Ratio tendem a sofrer impactos mais significativos. Esses impactos são suficientes para fazer com que o desempenho global fique inferior ao do cache sem particionamento, mesmo quando a maioria dos locatários se beneficia da política de particionamento.

Uma outra observação sob esses resultados envolve a variação da capacidade total do cache. À medida que aumentamos essa capacidade em relação ao *Footprint* total da carga, o Hit Ratio do sistema e dos locatários tendem a aumentar. Quando a capacidade do cache é uma baixa porcentagem do Footprint, poucos itens conseguem permanecer armazenados, resultando em constantes remoções. Isso gera menos Hit Status das requisições no cache, já que há menos espaço disponível e muitos locatários competindo pelos mesmos recursos de memória. Esse cenário caracteriza um subprovisionamento da capacidade, em que o cache se torna insuficiente para atender às demandas de todos os locatários.

Por outro lado, ao aumentar a capacidade do cache, mais itens podem ser mantidos e mais requisições podem ser atendidas pelo cache, aumentando o Hit Ratio. Quando a capacidade do cache alcança 100% do Footprint total, todas as requisições podem ser atendidas sem a necessidade de remoções, alcançando o maior Hit Ratio dessa carga. Na Figura 1, todos os casos de gerenciamento do cache alcançam o mesmo valor de Hit Ratio do sistema na capacidade de 100% *Footprint* total da carga. Essa situação também é refletida na Figura 2 em que, com 100% de *Footprint* total da carga, o desempenho de nenhum dos locatários sofre alteração entre os gerenciamentos.

No entanto, ao exceder o *Footprint* total, entramos em um cenário de superprovisionamento, onde a capacidade extra não traz mais benefícios ao sistema. Isso ocorre porque, uma vez que todos os itens da carga de trabalho já estão armazenados no cache, não há mais a necessidade de remoções. Portanto, aumentar ainda mais a capacidade do cache não resulta em melhoria adicional do Hit Ratio, pois o cache já atende todas as requisições sem perder itens e o desempenho permanece estável.

Além disso, observamos que a variação da Memória Reservada (0%, 50% e 100%) na política de particionamento Memshare não resultou em diferenças significativas no Hit Ratio, tanto no desempenho individual dos locatários quanto no sistema de cache como um todo. Isso indica que, nesse contexto, essa variável específica da política não obteve impacto considerável nos resultados obtidos.

Interferência

A diferença mais significativa entre as políticas foi observada na métrica de Interferência, conforme ilustrado na Figura 3. Essa figura apresenta, no eixo Y, a média da interferência sofrida pelos locatários, no eixo X, a variação da capacidade total do cache, expressa como uma porcentagem do *Footprint* total, e cada linha da figura é um cenário de configuração do cache.

Observamos que a média de interferência sofrida pelos locatários nos casos em que o cache utiliza a política de particionamento Memshare foi significativamente menor

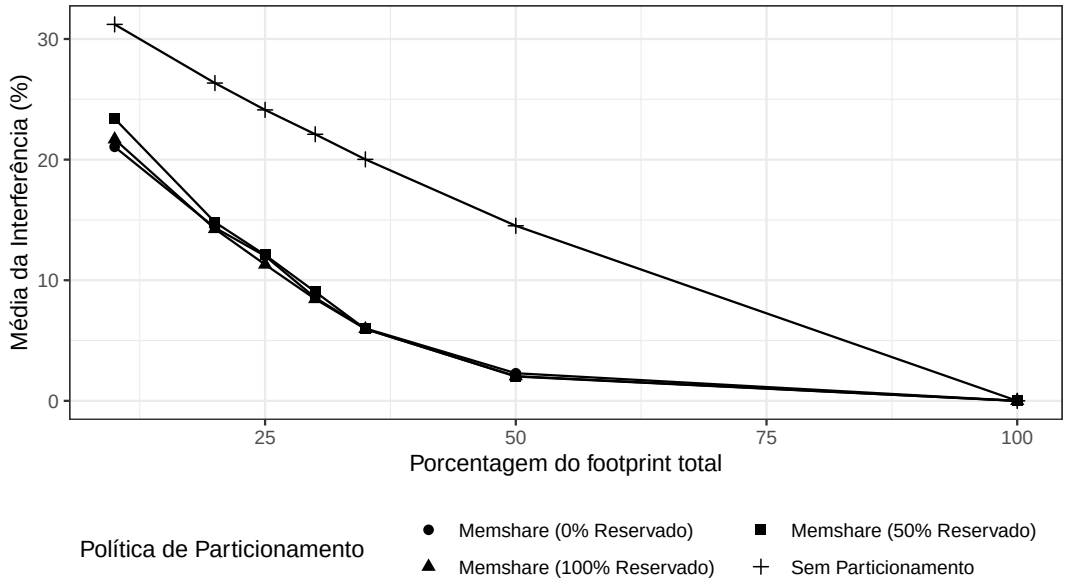

Figura 3 – Média da interferência no cache com e sem a política de particionamento Memshare, em diferentes capacidades. O Memshare reduziu significativamente a interferência entre locatários, sem impacto relevante da variação na Memória Reservada.

em comparação ao cache sem particionamento. Especificamente, quando a capacidade experimental corresponde a 35% do *Footprint*, o Memshare (100% Reservado) reduz a média da interferência em 70% em comparação com o cache sem particionamento.

Isso sugere que, embora o cache com Memshare apresente um Hit Ratio levemente inferior, ele proporciona uma redução significativa nos níveis de interferência no cache. A política busca priorizar itens que fazem uso eficiente do cache, com alta frequência de repetição, evitando remoções indevidas causadas por outros locatários, o que resulta em melhorias de desempenho nesses casos.

No entanto, a média pode não capturar todas as variações importantes da interferência. Por isso, a Figura 4 apresenta a distribuição da interferência sofrida pelos locatários. No eixo Y, temos a interferência, e, no eixo X, a variação das diferentes capacidades do cache, expressas como porcentagens do *Footprint* total. Cada sub gráfico representa uma política de gerenciamento de cache distinta.

Observamos que o cache sem particionamento apresenta a maior variabilidade de interferência em comparação com as demais políticas, exibindo valores medianos mais altos e uma maior concentração de *outliers* significativos. Muitos locatários sofrem com remoções indevidas de seus itens, o que prejudica o desempenho.

Por outro lado, à medida que o cache utilizando a política de particionamento Memshare é configurado, a interferência é progressivamente reduzida. Embora ainda existam *outliers*, sua quantidade e impacto também diminuem. Além disso, os valores medianos se aproximam progressivamente de zero, indicando que cerca de 50% dos

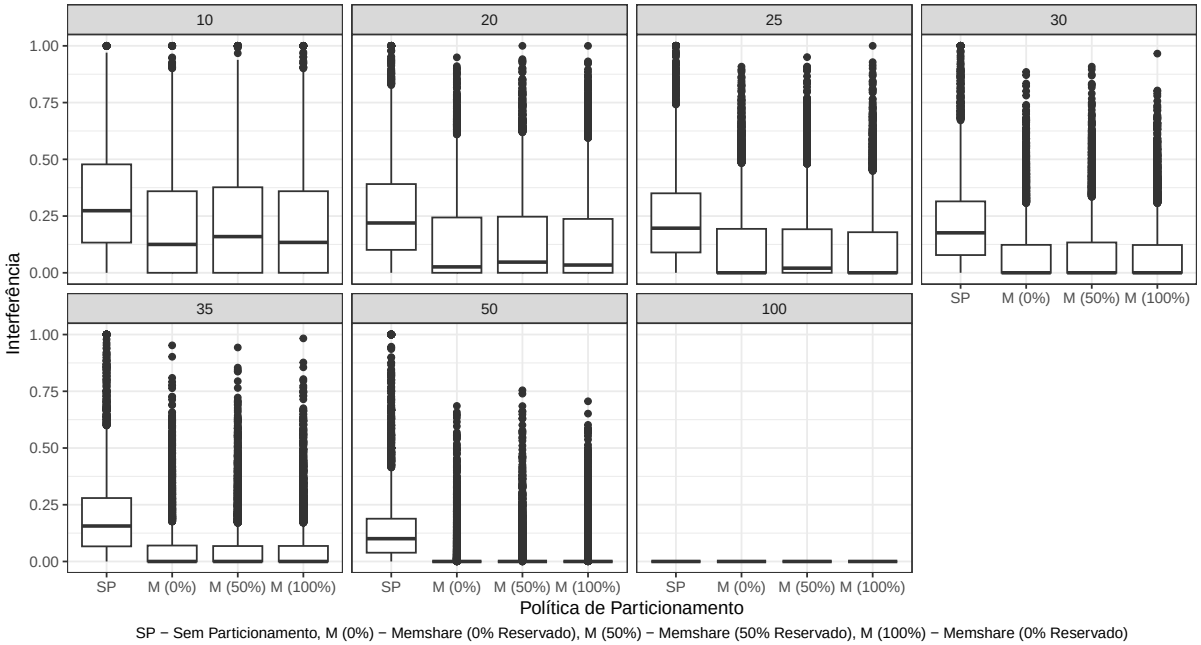

Figura 4 – Interferência entre locatários com e sem política de particionamento Memshare, em que cada sub gráfico é uma capacidade de cache, em porcentagem do *Footprint* total. No cache sem particionamento apresenta locatários com maiores interferências. A variação da Memória Reservada no Memshare tem baixo impacto na interferência.

locatários, sob a política Memshare, experimentam níveis de interferência quase nulos. A quantidade de locatários que tiveram sua interferência reduzida ou mantida, em comparação com o cache sem particionamento, varia entre 69% e 96%.

Avaliando o impacto da variação de capacidade sobre a interferência, percebemos que nos cenários de subprovisionamento — em que a capacidade do cache é uma baixa porcentagem do *Footprint* total — ocorre um alto grau de interferência. Isso acontece porque, com menos espaço disponível no cache, mais remoções são necessárias, incluindo a remoção de itens que fazem bom uso do cache (aqueles que são frequentemente requisitados). Já no cenário em que a capacidade do cache atinge 100% do *Footprint* total, todos os itens da carga de trabalho podem ser armazenados no cache sem necessidade de remoções, eliminando por completo a concorrência entre locatários pelo recurso de memória. Nessas condições, a interferência tanto em termos de média quanto em termos individuais é reduzida a zero, o que é refletido nas Figuras 3 e 4.

Além disso, a variação da Memória Reservada (0%, 50% e 100%), no uso da política de particionamento Memshare, não gerou diferenças significativas na média da Interferência do cache, nem na distribuição da Interferência sofrida pelos locatários. Isso indica que essa variação não teve impacto no contexto da carga utilizada nos experimentos, como ilustrado nas Figuras 3 e 4.

Os resultados mostram que a política Memshare reduz significativamente a interferência entre locatários, priorizando itens frequentemente requisitados. Embora o cenário sem particionamento tenha um Hit Ratio global ligeiramente superior, o Memshare equilibra melhor o desempenho e a interferência, beneficiando a maioria dos locatários, especialmente em ambientes com alta variabilidade de carga.

CONCLUSÃO E TRABALHOS FUTUROS

Neste trabalho, reproduzimos e avaliamos a política de particionamento implementada no Memshare utilizando um simulador, com base em uma carga de trabalho web multi-locatário coletada de um serviço de cache de uma grande plataforma de comércio eletrônico. Nosso objetivo foi comparar o desempenho e o nível de interferência do Memshare, em relação ao cenário base, sem utilizar uma política de particionamento, fazendo uma nova observação sobre essa política.

Os resultados indicam que, ao aplicar a política de particionamento Memshare para o gerenciamento de recursos em um sistema de cache compartilhado entre múltiplos locatários, é possível mitigar significativamente a interferência entre os locatários. A política Memshare foi projetada para lidar eficientemente com a variabilidade nas demandas dos diferentes locatários, como mostrado nos resultados. Ela gerencia dinamicamente os recursos de cache, priorizando a preservação de itens que fazem uso eficiente do cache, ou seja, aqueles que são frequentemente requisitados. Essa abordagem reduz a remoção indevida desses itens, o que seria mais comum em um cenário onde o cache opera sem uma política de particionamento.

Além disso, embora o cache sem particionamento demonstre um desempenho global levemente superior em termos de Hit Ratio, o Memshare proporciona melhorias no desempenho da maioria dos locatários, como evidenciado pela redução de interferência entre eles. Isso sugere que o Memshare é mais adequado para ambientes onde é importante garantir um desempenho mais equilibrado entre os locatários, evitando que a alta competição por recursos limite o desempenho de alguns locatários. Em última análise, a implementação do Memshare oferece um equilíbrio melhor entre desempenho e interferência, proporcionando ganhos significativos para a maioria dos locatários, mesmo em cenários de alta variabilidade de carga.

Concluímos que há um *trade-off* na escolha da política de particionamento para o gerenciamento da memória cache em ambientes multi-locatários, onde os locatários têm demandas variáveis e diferentes entre si. A escolha de uma configuração mais simples, sem particionamento, é adequada quando o objetivo é priorizar ao máximo o desempenho global do sistema, embora isso ocorra às custas de altos níveis de interferência entre os locatários.

Por outro lado, a política de particionamento Memshare equilibra melhor o uso

da memória, mitigando a interferência e proporcionando melhores desempenhos para a maioria dos locatários. A escolha entre essas políticas deve considerar as necessidades específicas do sistema, como a prioridade entre maximizar o Hit Ratio ou minimizar a interferência entre os locatários, dependendo do perfil de uso do cache.

Para trabalhos futuros, é possível considerar a avaliação de como a variação da Memória Reservada impacta a eficácia da política de particionamento Memshare. Embora os resultados apresentados não tenham mostrado grandes diferenças com a variação da Memória Reservada, é possível explorar diferentes R proporcionais para os locatários, com base em características específicas de cada um. Outra possível abordagem seria avaliar o desempenho do cache sob diferentes valores de Credit — a quantidade de memória realocada dinamicamente entre os locatários. Analisando como variações desse valor, ajustadas individualmente para cada locatário, poderiam otimizar ainda mais o uso dos recursos do cache

AGRADECIMENTOS

O presente trabalho foi realizado com o apoio do CNPq, Conselho Nacional de Desenvolvimento Científico e Tecnológico – Brasil, como parte do projeto “Avaliação de Políticas de Particionamento de Dados em Cache Web Multi-Cliente” do programa PIBIC/CNPq-UFCG, e dos projetos MCTIC/CNPq-FAPESQ/PB (EDITAL Nº 010/2021) and FairShare VTEX (EMBRAPII PCEE1911.0140).

REFERÊNCIAS

CIDON, A. et al. Memshare: a dynamic multi-tenant key-value cache. In: SILVA, D. D.; FORD, B. (Ed.). **Proceedings of the 2017 USENIX Annual Technical Conference, USENIX ATC 2017, Santa Clara, CA, USA, July 12-14, 2017**. USENIX Association, 2017. p. 321–334. Disponível em: <https://www.usenix.org/conference/atc17/technical-sessions/presentation/cidon>. páginas 4, 5, 6

LIRA, A. et al. No clash on cache: Observations from a multi-tenant ecommerce platform. In: **Proceedings of the 2024 ACM/SPEC International Conference on Performance Engineering**. [s.n.], 2024. Disponível em: <https://doi.org/10.1145/3629526.3645039>. páginas 3

LIRA, A. et al. Desafios na gerência de cache web multi-locatários. In: **Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos**. [S.l.: s.n.], 2024. p. 283–288. páginas 3

MEMCACHIER. <https://www.memcachier.com>. Accessed: 2024-09-25. páginas 4