UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

GABRIELA ROBERTA ALVERGA DO NASCIMENTO

DESENVOLVIMENTO DE UM SIMULADOR DE CACHE PARA ANÁLISE DE POLÍTICAS DE GERENCIAMENTO

CAMPINA GRANDE - PB

2023

GABRIELA ROBERTA ALVERGA DO NASCIMENTO

DESENVOLVIMENTO DE UM SIMULADOR DE CACHE PARA ANÁLISE DE POLÍTICAS DE GERENCIAMENTO

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharela em Ciência da Computação.

Orientador: Professor Dr. Thiago Emmanuel Pereira

CAMPINA GRANDE - PB

2023

GABRIELA ROBERTA ALVERGA DO NASCIMENTO

DESENVOLVIMENTO DE UM SIMULADOR DE CACHE PARA ANÁLISE DE POLÍTICAS DE GERENCIAMENTO

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharela em Ciência da Computação.

BANCA EXAMINADORA:

Professor Dr. Thiago Emmanuel Pereira

Orientador – UASC/CEEI/UFCG

Professora Dr.(a.) Patrícia Duarte de Lima Machado

Examinador – UASC/CEEI/UFCG

Professor Tiago Lima Massoni

Professor da Disciplina TCC – UASC/CEEI/UFCG

Trabalho aprovado em: 14 de fevereiro de 2023.

CAMPINA GRANDE - PB

RESUMO (ABSTRACT)[1]

Caching é uma estratégia tradicional para melhorar o desempenho de sistemas de computação. Sistemas de cache possuem várias configurações que podem alterar seu funcionamento e por consequência seu desempenho. É o caso, por exemplo, da parametrização das políticas de remoção de itens. Quando a escala do sistema de cache se torna muito grande, os experimentos para testes de alternativas para essas configurações se tornam difíceis. Considerando esse problema, desenvolvemos um simulador que modela um sistema de cache popular, o NGINX. Validamos esse simulador usando traces de uma implantação de um cache para um sistema web de alta escala. A validação do nosso simulador indica que ele apresenta baixo erro, cerca de 1% das requisições.

[1] Caching is a traditional strategy for improving the performance of computing systems. Cache systems have several settings that can change their operation and consequently their performance. This is the case, for example, with the parameterization of item removal policies. When the scale of the cache system becomes very large, experiments to test alternatives to these configurations become difficult. Considering this problem, we developed a simulator that models a popular caching system, NGINX. We validated this simulator using traces of a cache deployment for a high-scale web system. The validation of our simulator indicates that it presents low error, around 1% of the requests.

Desenvolvimento de um simulador de cache para análise de políticas de gerenciamento

Gabriela Nascimento
Universidade Federal de Campina Grande - UFCG
gabriela.nascimento@ccc.ufcg.edu.br

Thiago Emmanuel Pereira
Universidade Federal de Campina Grande - UFCG
temmanuel@computacao.ufcg.edu.br

Daniel Fireman
Instituto Federal de Educação, Ciência e Tecnologia de Alagoas - IFAL
daniel.fireman@ifal.edu.br

RESUMO

Caching é uma estratégia tradicional para melhorar o desempenho de sistemas de computação. Sistemas de cache possuem várias configurações que podem alterar seu funcionamento e por consequência seu desempenho. É o caso, por exemplo, da parametrização das políticas de remoção de itens. Quando a escala do sistema de cache se torna muito grande, os experimentos para testes de alternativas para essas configurações se tornam difíceis. Considerando esse problema, desenvolvemos um simulador que modela um sistema de cache popular, o NGINX. Validamos esse simulador usando traces de uma implantação de um cache para um sistema web de alta escala. A validação do nosso simulador indica que ele apresenta baixo erro, cerca de 1% das requisições.

1 INTRODUÇÃO

O desempenho sempre foi um fator crítico para o sucesso de sistemas de computação, dessa forma um método tradicionalmente empregado para a otimização desses sistemas é a utilização de um serviço de *caching*.

No contexto de aplicações web, o cache pode ser visto como uma camada intermediária, entre o cliente e o servidor, que permite guardar dados de forma temporária, para que próximas requisições que solicitem acesso a esses dados sejam respondidas sem necessidade de acessar o servidor, isto torna o tempo de resposta para o cliente menor. O armazenamento é feito através de um mapeamento chave-valor, em que a chave é a URI da solicitação e o valor é o conteúdo da resposta da solicitação.

A configuração de um serviço de cache é uma tarefa complexa, pois exige um conhecimento de como as requisições se comportam, para parametrizar configurações tais como o tamanho de cache utilizado e o tempo de vida de um item. Essas parametrizações são importantes; por exemplo, a parametrização do tamanho do cache pode levar a dois problemas.

Caso o tamanho escolhido seja muito pequeno, os itens em cache serão substituídos com muita frequência e a quantidade de requisições feitas ao servidor irá aumentar, pois novas requisições podem não encontrar seu conteúdo em cache. Por fim, isto levará a um aumento no tempo de resposta das requisições; caso a escolha de tamanho seja muito maior que a necessária, o uso de recursos computacionais para operar o cache será desnecessariamente maior (e, por consequência, seu custo).

O mesmo acontece com relação ao tempo de vida dos itens, caso a duração de tempo de vida dos itens em cache seja muito pequeno, o item ficará obsoleto rapidamente, gerando a necessidade de atualização sempre que for solicitado, consequentemente aumentando o tempo de resposta; se for muito longo, a resposta para novas solicitações pode ter o item obsoleto/desatualizado, gerando uma incompatibilidade de informações.

Dito isto, comparar diferentes configurações é uma tarefa ainda mais custosa, especialmente em um ambiente de produção, pois os problemas citados anteriormente podem ocorrer, gerando uma instabilidade no sistema e por conseguinte prejuízos. Além disso pode ser custoso realizar essas mudanças de configurações, devido ao tempo necessário para aplicá-las e verificar problemas em seu comportamento.

Dessa forma, com o intuito apoiar o processor de testes de diferentes configurações de cache, propomos a criação de um modelo de simulação de uma ferramenta de cache, NGINX, bastante usada na indústria. Criar um novo modelo de simulação foi necessário uma vez que os simuladores existentes não ofereciam a possibilidade de uma boa parametrização e apenas possuía dois status configurados. [1] [2]

Na construção desse modelo de simulação, levantamos as seguintes necessidades: 1) ter um simulador que suporte múltiplos nós de cache e configuração de seus tamanhos; 2) configuração de tempo de vida dos itens; 3) modificações na política de remoção de itens; e, 4) população do cache antes de ser executado o experimento. Sendo assim, este simulador é o principal resultado deste trabalho.

Para validar nosso modelo de simulação utilizamos dados de uma carga de registro de ações de um cache de uma grande empresa de comércio eletrônico, e parâmetros observados em um sistema real. O erro da simulação foi de apenas 1% das requisições, em um amostra com aproximadamente 920 mil requisições.

O restante deste trabalho está estruturado da seguinte forma: na Seção 2 descrevemos o sistema que utilizamos como base para a criação do simulador, e o modelo criado para a simulação. Na Seção 3 abordamos a implementação do simulador. Na Seção 4 falaremos sobre a avaliação dos resultados obtidos. Por fim, na Seção 5 apresentaremos nossas conclusões sobre os resultados obtidos com a ferramenta desenvolvida.

2 DESCRIÇÃO DO MODELO

Nesta seção iremos descrever o sistema utilizado como base para a criação do modelo do simulador, o NGINX, e o modelo gerado.

2.1 NGINX

O NGINX é um servidor HTTP de código aberto muito popular, que atualmente é utilizado como proxy reverso, **cache** e balanceador de carga [3]. Quando utilizado como serviço de cache, ele adiciona uma camada intermediária entre entre o cliente e o servidor. Esta camada guarda uma cópia do conteúdo que é acessado através de um mapeamento chave-valor, em que a chave é a URI da requisição e o valor é o conteúdo da resposta da solicitação. O NGINX utiliza

um algoritmo de hash para gerar e armazenar suas chaves. Este algoritmo é similar ao algoritmo de hash MD5, porém um pouco mais sofisticado. Caso seja solicitado um conteúdo que esteja mapeado neste armazenamento, o próprio NGINX retornará o conteúdo, sem a necessidade de acessar o servidor.

O conteúdo armazenado neste serviço tem duas marcações de tempo de vida, chamados TTL, a primeira marcação define quanto tempo o item guardado é considerado válido e pode ser retornado diretamente caso exista uma solicitação igual, a segunda marcação de tempo define por quanto tempo o item pode ser retornado mesmo estando obsoleto e será atualizado assim que o servidor responder.

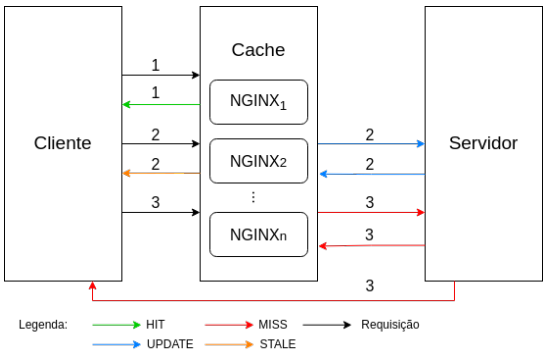

Figura 1: Modelo de funcionamento do cache NGINX

Ao atender uma requisição, o serviço de cache do NGINX pode responder com um dos seguintes status de resposta [4]:

- *MISS*: A resposta não foi encontrada no cache, portanto, foi buscada no servidor de origem.
- *BYPASS* – A resposta foi buscada no servidor de origem, em vez de atendida no cache, porque a solicitação correspondeu a uma diretiva que obriga o cache a buscar o dado no servidor antes de verificar sua existência.
- *EXPIRED* – A entrada no cache expirou. A resposta contém conteúdo novo do servidor de origem.
- *STALE* – O conteúdo está vencido porque o servidor de origem não está respondendo corretamente, e a resposta guardada no cache será retornada mesmo assim.
- *UPDATING* – O conteúdo vencido está sendo atualizado pois houve uma solicitação anterior do tipo STALE.
- *REVALIDATED* – O conteúdo do cache está sendo revalidado, o NGINX verifica se o conteúdo em cache atual ainda era válido.
- *HIT* – A resposta contém conteúdo novo e válido direto do servidor.

Para utilizar todas essas respostas, existe a necessidade de configuração. Normalmente, os serviços de cache se atém à três status principais: *HIT*, *MISS* e *STALE*. A Figura 1 elucida o fluxo desencadeado por uma requisição em cada um desses três casos: a requisição 1 representa uma solicitação em que sua resposta estava salva no cache e ainda estava válida. A requisição 2 apresenta uma requisição que possuía sua resposta salva no cache, contudo ela havia passado da primeira marcação de tempo de vida, ou seja a resposta é considera obsoleta, contudo ainda pode ser retornada, enquanto o cache espera o retorno do servidor para atualizar seu valor. Na requisição 3 temos uma requisição que não possui valor guardado no cache, logo, ela é repassada para o servidor que responde para o cliente, e envia o objeto para ser guardado para solicitações futuras para essa requisição.

Do ponto de vista do provisionamento da infraestrutura, o servidor NGINX pode utilizar uma quantidade variável de máquinas para armazenar o cache, adicionando ou removendo elas a partir da sua necessidade, cada uma dessas máquinas se torna um nó de cache.

2.2 Modelo de simulação

O simulador construído é baseado em *trace*, isto é, uma amostra do conjunto de registros do histórico de requisições de uma empresa de comércio eletrônico.

O modelo foi elaborado a partir de uma simplificação do modelo NGINX descrito anteriormente. Dentre as simplificações podemos destacar: a quantidade fixa de nós de cache, um tamanho único para os itens armazenadas e a simulação apenas das três principais respostas (*HIT*, *MISS* e *STALE*).

Com o intuito de permitir a fácil comparação de cenários de utilização, buscou-se produzir um simulador com suporte às principais configurações de um servidor NGINX tais como: quantidade de nós de cache e seus tamanhos, as duas marcações de tempo de vida de um item, população do cache antes de iniciar a simulação. Além disso, assim como o servidor NGINX, o simulador produzido utiliza hash para armazenamento das chaves e uma política de remoção de itens do cache.

A Figura 2 ilustra o fluxo simulado de uma requisição. A requisição 1 representa uma solicitação em que sua resposta estava salva no cache e ainda estava válida. A requisição 2 mostra uma requisição que possui sua resposta ainda no cache mas está com a sua primeira marcação de tempo de vida expirada, contudo ela ainda pode ser retornada enquanto o cache realiza a solicitação para o servidor retornar uma resposta atualizada, ou seja, o tempo de vida do item é atualizado, pois em nosso modelo simulado o servidor é apenas conceitual.

Na requisição 3 é apresentado uma solicitação que não possui sua resposta salva no cache ou já ultrapassou a sua segunda marcação de tempo de vida, o cache repassa a solicitação para o servidor que responde o cliente e envia a resposta também para o cache onde ela será salva para ser utilizada por solicitações futuras.

Os detalhes de implementação podem ser vistos na Seção 3.

3 IMPLEMENTAÇÃO

Nesta seção discutiremos os detalhes de implementação do simulador. O simulador foi implementado utilizando a linguagem Python, e foi dividido em dois componentes principais: o simulador, que executa a simulação propriamente dita, e o gerenciador da política de cache, que representa a implementação da política de remoção de itens que sua simulação irá utilizar

3.1 Nós de cache

Cada nó de cache foi representado por um conjunto e um dicionário. O conjunto é responsável por gerenciar as gerações do cache, o que vai decidir qual item deve ser removido quando o cache está

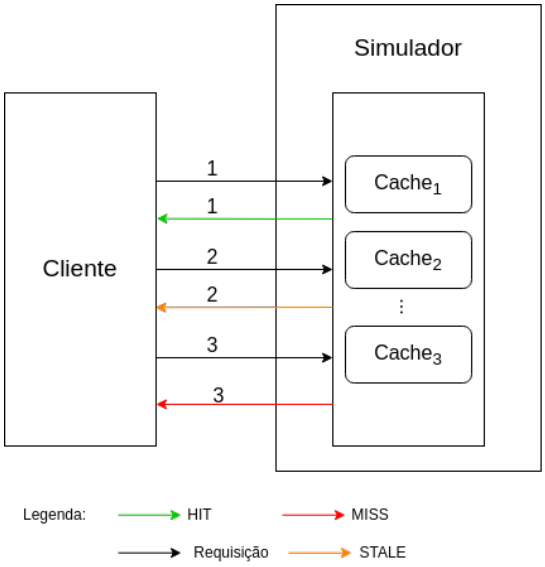

Figura 2: Modelo de funcionamento do simulador

cheio. O dicionário é a estrutura utilizada para busca, ou seja, para verificar se existe um dado no cache. Além disso, cada nó possui um tamanho, que representa a quantidade de itens que ele pode guardar.

3.2 Parâmetros de configuração do cache

Os seguintes parâmetros de configuração foram implementados no simulador:

- **Quantidade de nós de cache:** quantidade de nós de cache que será utilizada (valor mínimo: 1).
- **Tamanho dos nós:** Uma lista que irá representar o tamanho que cada nó de cache irá possuir, a lista deve ter a mesma quantidade de itens que a quantidade colocada nos nós.
- **Primeira marcação de tempo de vida:** Valor que representa quanto tempo um item leva para ficar obsoleto, mas ainda pode ser retornado como resposta, esse valor deve estar em minutos.
- **Segunda marcação de tempo de vida:** Valor que representa quanto tempo um item leva após ficar obsoleto, que ainda pode ser retornado e atualizado, caso esse tempo seja passado e não for atualizado, o item será removido do cache.
- **Arquivo de população do cache:** É uma configuração opcional, caso em sua simulação seja necessário uma população do cache, é possível passar como parâmetro um arquivo CSV com os item que você deseja que sejam adicionados.

3.3 Armazenamento dos itens de cache

Os itens são salvos em memória durante a execução utilizando as estruturas de dados que compõem os nós de cache, quando a requisição para um item chega ao cache, o item é convertido pelo algoritmo de hash MD5, após essa conversão ele irá buscá-lo no dicionário de itens para verificar se o item existe no cache e sua validade, caso não exista ou esteja vencido ele é adicionado ao cache.

3.4 Fluxo de uma requisição no simulador

A Figura 3 elucida como é o ciclo de vida de um item no cache.

(1) Ao chegar uma nova requisição, é verificado se o item solicitado esta guardado dentro do cache;
(2) Ao chegar uma nova requisição, é verificado se o item solicitado esta guardado dentro do cache, caso positivo, são verificadas as marcações de tempo de vida
(3) Se este tempo for menor que a primeira marcação, o item é retornado imediatamente
(4) Se este tempo for maior que a primeira marcação, é verificado a segunda marcação.
(5) Caso o tempo seja menor que ela, o item está desatualizado, mas pode ser retornado.
(6) O cache fará uma solicitação para que o dado guardado seja atualizado.
(7) Caso o tempo seja maior que a segunda marcação de tempo de vida, o item será removido do cache.
(8) Caso o item não seja encontrado no cache, ele será solicitado e adicionado a ele.

3.5 Política de remoção de itens

A política de remoção de item consiste em remover o item mais antigo menos utilizado. Para isso utilizamos o conceito de gerações, sempre que um item é solicitado, ele entra na primeira geração, quando ele vai sendo solicitado ao decorrer do tempo, ele vai permanecendo na geração mais nova, quando não é solicitado vai passando para as gerações velhas. Quando um item chega ao cache, ele é adicionado a geração 0, caso a geração 0 esteja cheia, algum dos itens dela deve ser realocado para geração seguinte, ou seja, geração 1, se um item já estiver no cache, e for acessado novamente, ele é movido da geração que se encontra para a geração 0 novamente, caso todas as gerações estiverem cheias, um dos item encontrado na geração mais antiga é removido do cache.

3.6 Entradas e saídas

Como arquivo de entrada, deve ser passado um arquivo CSV contendo as seguintes colunas de dados:

- **Carimbo de tempo:** O momento que a requisição foi enviada no formato EPOCH, esse formato representa o tempo em número de segundos decorridos desde 0h UTC de 1º de janeiro de 1970.
- **Produto:** Representação numérica do item a ser solicitado.
- **Loja:** Representação numérica da loja a qual item a ser solicitado pertence.

Como arquivo de saída, deve ser passado o caminho no qual será guardado o arquivo com as saídas da simulação, o arquivo é do tipo CSV contendo as seguintes colunas de dados:

- **Produto:** Representação numérica do item a ser solicitado.
- **Loja:** Representação numérica da loja a qual item a ser solicitado pertence.
- **Cache:** Geração do cache em que o item estava guardado.
- **Status:** Status da resposta do cache.

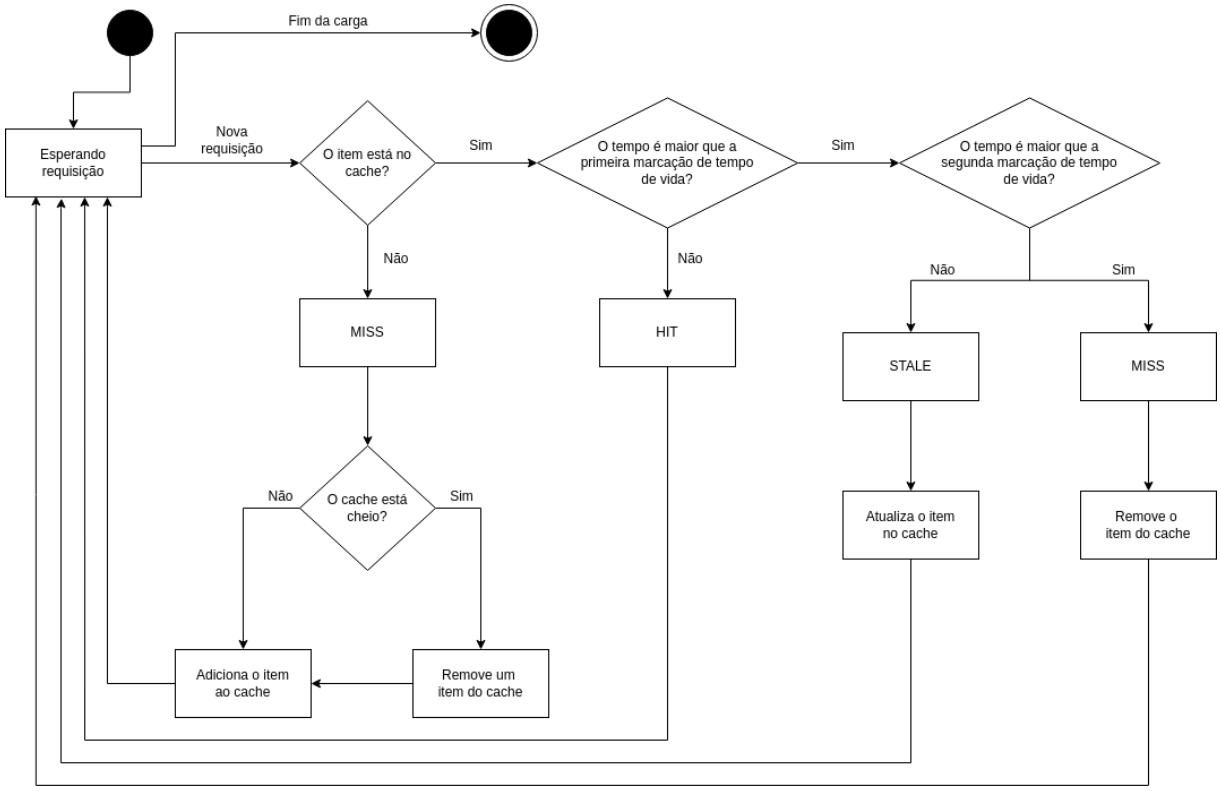

Figura 3: Máquina de estados do fluxo de requisição no simulador

4 AVALIAÇÃO

4.1 A carga

Para realizar a validação do simulador, foi utilizada uma carga real coletada do ambiente de produção de uma empresa de comércio eletrônico. Foi coletada uma amostra de 1% das requisições durante uma hora, o que totalizou 919.908 requisições. A Figura 4 mostra a taxa de chegada de requisições ao longo do tempo com um agrupamento de 1s. Nela podemos observar que a vazão média e mediana das requisições possuem valores bem próximos, a vazão dessa carga é bem definida.

4.2 Parametrização da execução

A Tabela 1 mostra a parametrização que foi escolhida para a simulação. Esses parâmetros foram escolhidos baseados na configuração original do cache de onde a carga foi coletada.

Parâmetro	Valor
Quantidade de nós de cache	2
Tamanho dos nós de cache	[165000000, 165000000]
Primeira marcação de tempo de vida	25 minutos
Segunda marcação de tempo de vida	180 minutos

Tabela 1: Parametrização da simulação

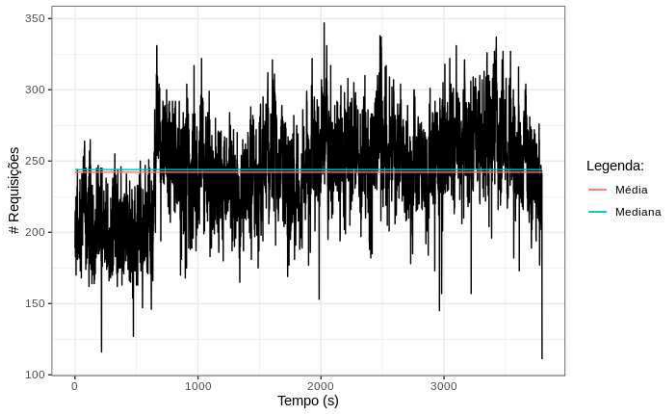

Figura 4: Gráfico de linha da quantidade de requisições por segundo

4.3 Resultados

A Figura 5 mostra uma comparação da contagem de respostas ao longo do tempo da simulação. Dessa forma, é possível ver que que a diferença os resultados da simulação e os dados originais é muito pequena, e seu comportamento é igual ao longo do tempo. Vemos que a diferença média para cada status de resposta é de:

1.295 requisições que deram HIT, 127 requisições que deram MISS e 1.349 requisições que deram STALE.

Por sua vez, a Figura 6 apresenta uma representação em forma de tabela do resultado esperado em comparação ao resultado obtido, observando assim de forma mais específica o quão pequena é a diferença entre os dois resultados, se totalizando em menos de 1% da quantidade total de requisições da carga.

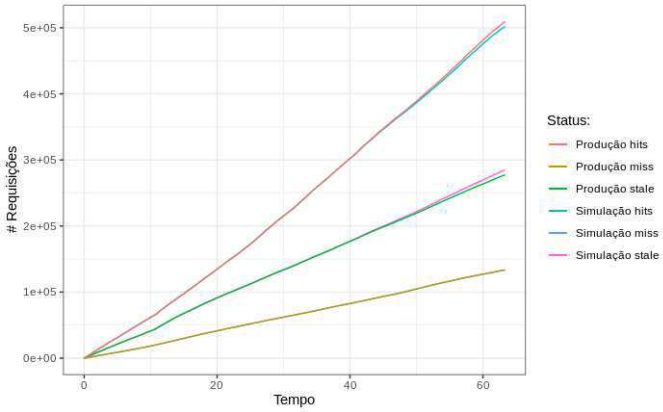

Figura 5: Gráfico de linha comparando contagem acumulada por status das repostas das requisições de produção e simulação, utilizando Second Chance

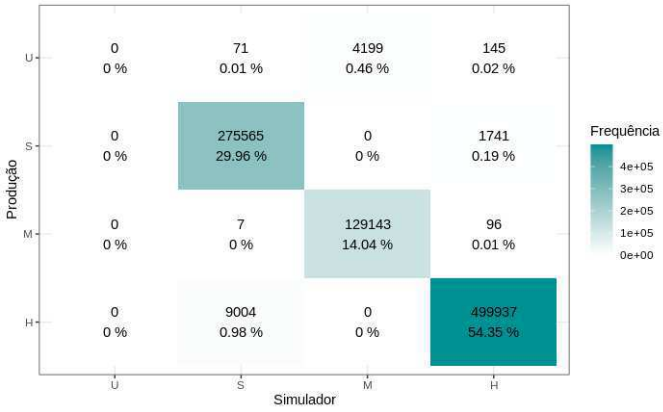

Figura 6: Matriz de confusão comparando os status das repostas das requisições de produção e simulação, utilizando Second Chance

5 CONCLUSÃO

Foi criado um simulador de cache válido com uma taxa de erro de menos de 1% e facilmente configurável. Trabalhos futuros podem fazer uso do simulador para avaliações do impacto de diferentes configurações de cache em uma mesma carga, bem como para teste de diferentes políticas de remoção de itens.

6 AGRADECIMENTO

Esse trabalho foi parcialmente financiado pelo EDITAL Nº 010/2021 - MCTIC/CNPq-FAPESQ/PB e pela VTEX BRASIL (EMBRAPII PCEE1911.0140).

REFERÊNCIAS

[1] Caleb Evans. [n. d.]. Repositório GitHub. https://github.com/caleb531/cache-simulator. Acesso 03/02/2023.
[2] Mayara Marques. [n. d.]. Repositório GitHub. https://github.com/mmrosatab/SimuladorMemoriaCache. Acesso 03/02/2023.
[3] NGINX. [n. d.]. NGINX. https://www.nginx.com. Acesso 12/01/2023.
[4] NGINX. [n. d.]. NGINX Caching guide. https://www.nginx.com/blog/nginx-caching-guide. Acesso 24/01/2023.