Caching em Redes com Perdas e Atrasos

Daniel S. Menasché – AD UFRJ 2020/1

May 10, 2021

Abstract

O objetivo deste trabalho é avaliar um sistema de caches em redes com perdas e atrasos. Para tal, deve-se aproveitar conhecimentos sobre cadeias de Markov tempo discreto, cadeias de Markov tempo contínuo, e simulação de eventos discretos.

1 Introdução

Considere um sistema de femtocaches, ou seja, caches colocadas próximas a antenas de telefonia celular. Vide, por exemplo, a Figura 1.

Assuma que o operador de rede tem espaço de armazenamento x conteúdos, e que todos os conteúdos tem mesmo tamanho. Temos um total de M caches, e cada cache i usa uma fração b_i do total de armazenamento x. Num cenário em que $x = 4$, por exemplo, e $b_1 = b_2 = 0.5$, cada cache tem capacidade para armazenar 2 conteúdos.

Iremos trabalhar em 3 regimes:

1. regime 1: canais sem perdas e sem atrasos
2. regime 2: canais com perdas
3. regime 3: canais com perdas e atrasos

Os cenários 1 e 2 podem ser modelados com cadeias de Markov tempo discreto. O cenário 3 pode ser modelado com cadeia de Markov tempo contínuo. Todos os cenários podem ser tratados com simulação de eventos discretos.

Os objetivos do trabalho são:

1. avaliar o impacto das políticas de cache (FIFO, LRU, Random, estática, e outras) no desempenho do sistema
2. aprender a implementar um simulador de eventos discretos
3. entender o papel contrastante da replicação e diversidade em sistemas de cache distribuído: quando é melhor replicar mais um conteúdo? e quando é melhor ter mais diversidade de conteúdo nas caches?

Seja p a probabilidade de sucesso de cada canal. Seja N o número de conteúdos no catálogo, e seja q_j a probabilidade de acesso ao conteúdo j.

Hipóteses:

1. canal broadcast: todas as requisições são enviadas para todas as caches
2. requisições chegam segundo fluxo Poisson: requisições são geradas segundo fluxo Poisson com taxa λ requisições por segundo. Cada requisição é feita para o conteúdo j com probabilidade q_j. Ou seja, o fluxo de requisições para o conteúdo j é um fluxo Poisson com taxa $\lambda_j = q_j\lambda$
3. perdas no canal: perdas em cada canal ocorrem de forma independente, com probabilidade $1 - p$
4. atualização das caches: uma cache só é atualizada se chegar uma requisição a ela. Se um canal falhar, e a requisição não atingir alguma das caches, essa cache não será atualizada

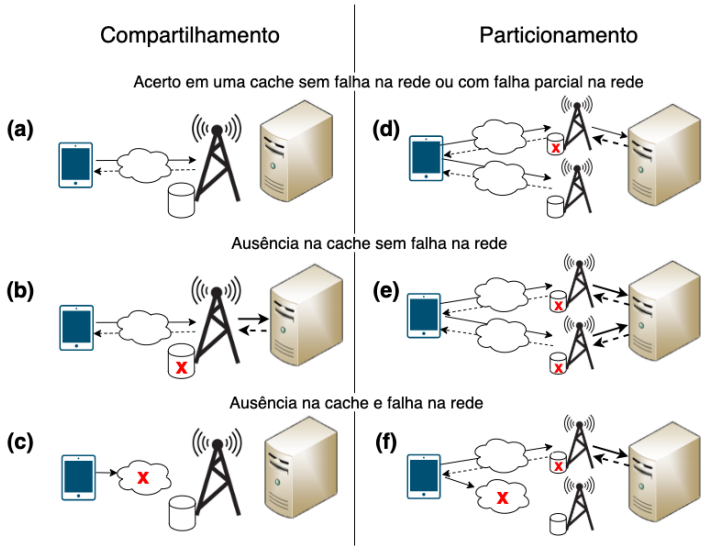

Figure 1: Alguns cenários considerados neste trabalho

Como exemplo ilustrativo, observe o sistema da Figura 1. Na Figura 1(a) temos uma única cache, o usuário faz uma requisição, e gera um acerto (cache hit). Na Figura 1(b) temos uma única cache, o usuário faz uma requisição, e gera um erro (cache miss). O cache miss gera tráfego entre o servidor e a cache, que evetualmente pode ser populada com o novo conteúdo, que é servido para o usuário. Finalmente, no cenário da Figura 1(c) temos novamente uma única cache, o usuário faz uma requisição, que é perdida pela rede. Nesse caso, o usuário irá precisar recorrer a outros meios para ter sua requisição satisfeita, porque ela não é servida de imediato.

Alternativamente, no lado direito da Figura 1 temos o caso com particionamento. Temos duas antenas de celular, e cada antena tem uma cache associada. Assumimos que toda vez que o usuário faz uma requisição, a requisição é automaticamente enviada para as duas antenas, ou seja, para as duas caches. Na Figura 1(d) a requisição gera um cache miss em uma cache, mas um hit na segunda cache. Assim, o conteúdo é servido ao usuário. Na Figura 1(e) temos dois cache misses, seguidos por tráfego para as duas caches. Na Figura 1(f) temos uma falha na rede e um cache miss.

2 Cenário 1: canais sem perdas e sem atrasos

A Figura 2 ilustra o cenário em questão. Vamos começar considerando o caso especial $p = 1$. Considere 3 conteúdos ($N = 3$), e duas caches de tamanho 2 cada. Assuma que os 3 conteúdos tem mesma probabilidade de acesso, $q_1 = q_2 = q_3 = 1/3$. Seja $\lambda = 1$, $M = 2$, $b_1 = b_2 = 0.5$.

Nesse cenário, assuma que uma requisição é dita de sucesso caso pelo menos uma das caches possua o conteúdo requisitado pelo usuário.

Qual a probabilidade de sucesso em cada um dos cenários abaixo?

1. duas caches FIFO: numa cache FIFO, se chega uma requisição a um conteúdo que não está contido na cache, o conteúdo é inserido no começo da cache, e todos os outros conteúdos fazem um shift de uma posição. O conteúdo que estava no final é removido. Se chega uma requisição a um conteúdo que já está contido na cache, nada muda.

2. duas caches LRU: numa cache LRU, se chega uma requisição a um conteúdo que não está contido na cache, o conteúdo é inserido no começo da cache, e todos os outros conteúdos fazem um shift de uma posição. O conteúdo que estava no final é removido. Se chega uma requisição a um conteúdo que já está contido na cache, tal conteúdo vai para o começo da cache.

3. duas caches Random: numa cache Random, se chega uma requisição a um conteúdo que não está contido na cache, o conteúdo é inserido e um conteúdo aleatório é removido. Se chega uma requisição a um conteúdo que já está contido na cache, nada muda.

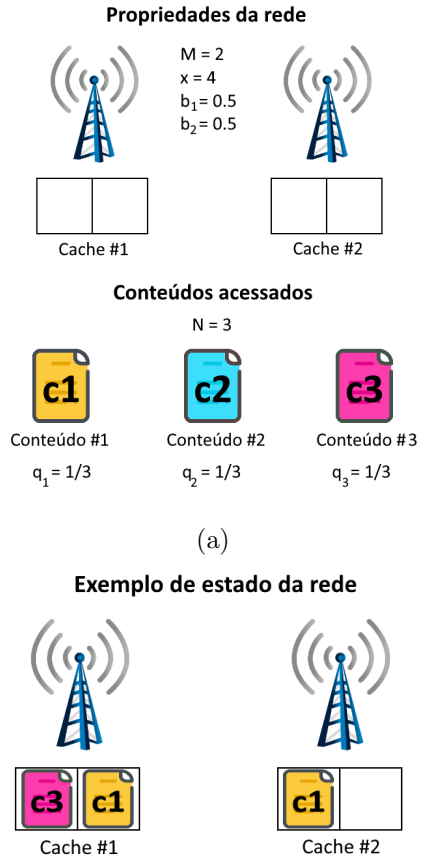

Figure 2: Cenário 1: duas caches de tamanho 2 cada. Cada conteúdo é requisitado com probabilidade 1/3. O estado da rede apresentado na parte (b) indica que uma das caches guarda os conteúdos c3 e c1. A outra cache guarda o conteúdo c1. Note que a segunda cache ainda está sendo preenchida. Uma vez completamente preenchida, ela ficará completamente preenchida para sempre. Se preferir, você pode, no seu trabalho, já inicializar as caches com conteúdos em todas as suas posições.

4. duas caches estáticas: a cache estática sempre armazena os conteúdos mais populares, e sua alocação nunca muda.

Qual das opções acima é melhor? Justifique claramente.
Cada um destes sistemas pode ser modelado com uma cadeia de Markov tempo discreto.
Classifique as cadeias em cada caso como

1. redutível ou irredutível?

2. o desempenho do sistema em estado estacionário depende do estado inicial, ou seja, depende da alocação inicial de conteúdos nas caches?

Repita a questão para o caso $N = 4$. Assuma, novamente que todos os conteúdos são acessados de fomra uniforme, ou seja, $q_j = 1/4$ para todo j. E agora? As suas respostas para as questões acima mudam?

3 Cenário 2: canais com perdas

Repita as quetões do item anterior, assumindo agora $p = 0.9$. Ou seja, cada canal tem probabilidade de perda de 10%. Do ponto de vista do usuário, assuma que o usuário vai experimentar um *user miss* se ele não conseguir acessar o conteúdo de nenhuma das duas caches. Ou seja, o *user miss* ocorre se

1. os dois canais falharem ou
2. um canal falhar e a cache do canal que não falhou não tiver o conteúdo
3. os dois canais funcionarem, mas nenhuma das caches tiver o conteúdo requisitado

Qual a probabilidade de um *user miss*, do ponto de vista do usuário, nos mesmos casos considerados no cenário 1? Responda as mesmas questões do cenário 1.

4 Cenário 3: canais com perdas e atrasos

As questões apresentadas nos últimos itens podem ser resolvidas usando cadeias de Markov tempo discreto e/ou simulação. Para as questões consideradas neste cenário, você precisará necessariamente implementar um simulador. Caso tenha tempo, considere a opção (desafio) de comparar a resposta do simulador com a resposta obtida via uma cadeia de Markov tempo contínuo.

Daqui para frente, vamos considerar parâmetros adicionais no sistema. Vamos assumir que cada requisição leva um tempo médio exponencialmente distribuído, com média $1/\mu$, para atingir cada uma das caches. Ou seja, a taxa de serviço de cada canal é μ. Além disso, o sucesso por canal continua ocorrendo com probabilidade p. Em outras palavras, a taxa efetiva de serviço de cada canal, levando em conta perdas, é de μp (você provavelmente não vai precisar se preocupar com isso).

Assim que uma requisição é feita, anota-se numa tabela o instante em que a requisição foi efetuada, e inicia-se um timer. O timer gera um alarme (timeout) após tempo exponencialmente distribuído com média $1/\alpha$. Caso a requisição não seja servida dentro do tempo estipulado (ou seja, em caso de timeout), a requisição é servida por um servidor a parte, o que leva tempo adicional exponencialmente distribuído, com média $1/\gamma$. Esse último serviço ocorre de forma independente das caches, e não afeta o estado das mesmas.

Quando uma requisição chega a uma cache, se a cache possui o conteúdo requisitado ele é servido, e retorna para o usuário depois de tempo exponencialmente distribuído com média $1/\mu$. Caso contrário, o conteúdo é baixado do servidor, para a cache, o que leva tempo exponencialmente distribuído com média $1/\theta$. Após esse tempo, ou seja, após o conteúdo popular a cache, o conteúdo retorna para o usuário depois de tempo exponencialmente distribuído com média $1/\mu$.

Assuma que não há perdas no canal downstream (da cache para os clientes). Ou seja, as perdas só ocorrem upstream (dos clientes para cache).

Um resumo é apresentado na Figura 3.

Note que pode-se ter várias requisições em aberto por parte dos clientes. Quando chega um certo conteúdo aos clientes, todas as requisições pendentes para aquele conteúdo são servidas imediatamente, os atrasos correspondentes são calculados (tempo de chegada do conteúdo menos tempo de requisição) e as tabelas de requisições pendentes são atualizadas, removendo-se as entradas referentes às requisições satisfeitas. Os timers para caracterizar os correspondentes timeouts a essas requisições pendentes também são removidos.

Seja $\mu = \alpha = \gamma = 1$ e $\theta = \infty$. Ou seja, o canal entre o servidor e a cache tem capacidade infinita (e atraso zero). Considere as 4 políticas apresentadas nos itens anteriores (FIFO, LRU, Random, estática). Qual o tempo médio para servir as requisições, nos cenários considerados?

5 Cenário 4: canais com perdas e atrasos

Repita o cenário 3, considerando agora $\mu = \alpha = \gamma = \theta = 1$.

6 Cenários adicionais

Considere cenários adicionais, ao seu gosto. Você pode considerar caches maiores (dezenas ou centenas de posições), catálogos maiores (milhares de conteúdos, seguindo, por exemplo, uma distribuição Zipf), e atrasos não exponenciais (por exemplo, uniformes).

Discuta o impacto de diferentes parâmetros do sistema no desempenho do mesmo. Compare sistemas com caches particionadas versus sistemas com cache unificada. Por exemplo, compare sistemas

com 2 caches de tamanho 2 cada versus um sistema com 1 única cache de tamanho 4. Qual sistema é melhor em cada caso considerado? Por que?

Você pode também variar os valores de p bem como de μ, γ, θ e α.

Você também pode querer responder algumas das seguintes perguntas, ou bolar suas próprias perguntas:

- Qual o valor ótimo do timeout α?
- Qual a taxa com que trafegam dados do servidor para a cache, em cada cenário considerado?
- O que os usuários devem levar em conta ao decidir a forma como irão alocar seus recursos de cache?

7 Simulação para verificar resultados analíticos

Os cenários 1 e 2 devem ser resolvidos tanto com cadeia de Markov quanto via simulação. Use a simulação para mostrar que seus resultados analíticos estão dentro do intervalo de confiança.

O cenário 3 pode ser resolvido exclusivamente via simulação.

8 Quesitos que devem constar no relatório

1. Desafios encontrados e soluções
2. Todas as soluções obtidas via simulação devem vir acompanhadas de seus intervalos de confiança
3. Decisões de projeto: breve descrição do simulador, e de como foi implementado
4. Depuração (como o simulador foi depurado). Em particular
   (a) quando possível, comparar analítico com simulação e verificar que analítico está no intervalo de confiança da simulação
   (b) simplificar o problema, e estudar casos básicos
   (c) colocar *printf* no código para rastrear o que ocorre ao longo do processo
5. Resultados
6. Discussão dos resultados

9 Como trabalho será avaliado

O trabalho será avaliado com relação a corretude, completude e criatividade.

1. cenário 1: 15 pontos
2. cenário 2: 15 pontos
3. cenário 3: 15 pontos
4. cenário 4: 15 pontos
5. cenários adicionais: 40 pontos

Em todos os passos, mostre que os seus resultados estão corretos (usando depuração adequada e indicando as estratégias usadas para depurar).

10 O que entregar?

1. relatório em PDF
2. código fonte no github, colab, ou similar.
3. slides, a serem apresentados em aula (15-20 mins de apresentação)

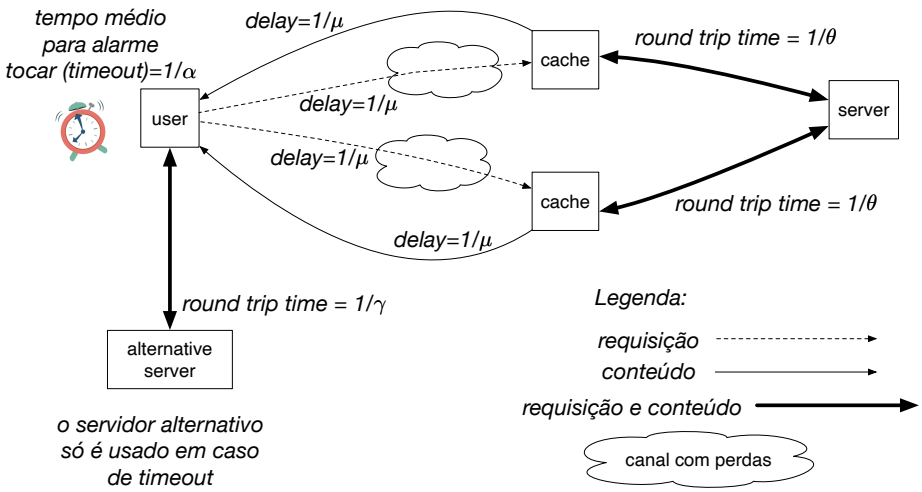

Figure 3: Atrasos em um sentido (one way delay) representados como *delay* e atrasos de ida e volta (two way delay) representados como *round trip time* (ou RTT). Note que os únicos canais que admitem perdas são os canais do usuário para as caches. Na versão estendida do trabalho, você pode considerar perdas em mais canais, caso desejar.

.

11 Trabalhos relacionados

Esse trabalho é inspirado em [SCA+20]. Em [SCA+20] os autores não consideram atrasos. O propósito deste trabalho é avaliar o que ocorre quando atrasos são levados em conta.

References

[SCA+20] Paulo Sena, Igor Carvalho, Antonio Abelem, György Dán, Daniel Menasche, and Don Towsley. Caching policies over unreliable channels. In *2020 18th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOPT)*, pages 1–8. IEEE, 2020.