Uma Estratégia de Cache baseada em Múltiplas Métricas para Redes ICN

Igor Carvalho[1], **Airton Ishimori**[1], **Antônio Abelém**[1]

[1]Grupo de Estudos em Redes e Comunicação Multimídia – GERCOM
Universidade Federal do Pará (UFPA)
Belém – PA – Brasil

{icarvalho, airton, abelem}@ufpa.br

Abstract. *Information Centric Networks (ICN) has been a new network paradigm that has been extensively researched recently. In networking-caching enables routers to cache content in the request path, in which they use a caching strategy to decide whether to store a content or not. However, most of the current strategies are only-one-criterion based in which, due to the dynamics of the network, may not be suitable, resulting in both low network hit ratio and performance. This way, this paper proposes a Multi-Criteria Caching Decision (MCCD) strategy for ICN, which considers three metrics to select the target nodes for caching contents and better distribute them over the network. Our results got a much higher cache hit ratio than the other evaluated strategies and a reasonable download time.*

Resumo. *As Redes Centradas em Conteúdo (Information Centric Networks - ICN) são um paradigma de arquitetura de rede que atraiu muitas pesquisas nos últimos anos. O cache em rede permite que roteadores possam armazenar tal conteúdo, os quais utilizam-se de uma estratégia para decidir se armazenam-no ou não. Porém, a maioria das estratégias são baseadas em apenas uma métrica o que, levando em consideração a dinamicidade da rede, pode resultar em uma baixa taxa de acertos e desempenho da rede. Desta forma, este artigo propõe uma estratégia de decisão de cache baseado em múltiplas métricas para redes ICN, no qual usamos três métricas para selecionar os nós alvos e, com isso, melhor distribuir o conteúdo na rede. Nossos resultados obtiveram uma taxa de acertos bem maior que as outras propostas avaliadas e um tempo de download razoável.*

1. Introdução

Na última década, as Redes Centrada no Conteúdo (*Information Centric Networks - ICN*) surgiram como uma das arquiteturas mais promissoras no contexto da Internet [Ahlgren et al. 2012]. Neste modelo, a obtenção de conteúdo dá-se por meio de requisições baseadas nos nomes de tais conteúdos e não por meio de identificação usando o endereço IP. Quando usuários solicitam conteúdos, uma rota é estabelecida até o servidor ou provedor de conteúdo e os nós intermediários podem armazená-los em *cache*. Isto permite que novas requisições possam ser atendidas por tais nós, reduzindo tanto a sobrecarga no servidor de conteúdo como o consumo de largura de banda, aumentando sua disponibilidade na rede [Dannewitz 2009].

De acordo com um estudo feito pela Cisco, o tráfego IP global crecerá a uma taxa anual de 23% entre 2014 e 2019. Até 2019, o tráfego total de vídeo, somando usuários comuns e empresas, será 77% de todo o tráfego da Internet [Cisco 2015]. Desta forma, uma estratégia de decisão de *cache* influencia fortemente no desempenho da rede, no sentindo de escolher os nós alvos para atender as solicitações dos usuários. Caso ela não atenda às requisições razoavelmente, consequentemente haverá uma baixa taxa de acertos *hit ratio*), resultando em alta carga no servidor e longos atrasos [Zhang et al. 2013].

A estratégia padrão em ICN é a LCE (*Leave a Copy Everywhere*). Ela armazena o conteúdo solicitado em todos os nós entre um nó solicitante e um nó que possui tal conteúdo, podendo ser um *cache* intermediário ou o servidor [Jacobson et al. 2009].Ela garante uma rápida difusão do conteúdo por toda a rede. Porém, algumas questões sobre seu desempenho foram levantadas. A primeira é sobre sua alta redundância, isto é, o mesmo conteúdo é desnecessariamente copiado em todos os nós no caminho reverso da requisição, reduzindo a variabilidade de conteúdo na rede. A outra problemática levantada é que, como o espaço reservado para *cache* dos nós é relativamente menor que o total de conteúdo disponível na rede, o LCE causa uma alta taxa de erros devido às trocas de conteúdo no *cache* do roteador [de Brito et al. 2012].

Dentro do paradigma ICN, há muitos problemas relacionados ao *cache* em rede. De acordo com [Kutscher et al. 2014], dois deles são mencionados como o problema de alocação de cache e distribuição de conteúdo. O primeiro fornece algumas características necessárias pelos nós da rede a fim de que o uso de *cache* na rede seja possível. Elas são: a centralidade dos nós, o tamanho do domínio envolvido, evitar links custosos ou inter-domínios, padrões de tráfego e localizaçao espacial dos usuários em uma parte específica na rede. O segundo está relacionado com a distribuição de conteúdo nos caches, que pode armazenar os conteúdos na rota estabelecida entre o cliente e o servidor (*on-path caching*) ou armazená-los em outros nós localizados em uma outras parte da rede (*off-path-caching*).

Muitas pesquisas sobre cache em redes foram desenvolvidas no contexto de estratégias de decisão de cache na última década. Algumas consideram a quantidade de espaço disponível para cache dos nós [Lee et al. 2013]; uma outra foi desenvolvida baseado num modelo probabilístico que considera a capacidade de cache do caminho entre o nó solicitante e o nó que possui o conteúdo [Psaras et al. 2012]; a centralidade de intermediação dos nós foi considerada por [Chai et al. 2012], isto é, a importância do nó na rede. Observa-se que cada uma delas utiliza uma métrica específica como distância, capacidade de armazenamento dos nós/caminho ou ainda características topológicas. Entretanto, como a rede e as requisições de conteúdo apresentam um comportamento dinâmico, trabalhar com uma métrica para a escolha dos possíveis nós alvos pode fazer que o conteúdo seja armazenado em poucos nós que casem com tal métrica, impedindo uma melhor distribuição do conteúdo pela rede.

Este artigo propõe uma nova estratégia de decisão de cache baseada em múltiplas métricas para redes ICN chamada MCCD - *Multi-Criteria Caching Decision*. Ela tem como base as seguintes métricas para a tomada de decisão: *centralidade de intermediação dos nós*, *tamanho de cache disponível* e a *taxa de acertos de conteúdo do nó* (Explicaremos detalhadamente cada uma delas na seção 3). Partimos da premissa de que se uma estratégia de decisão possui mais de uma métrica como base para decidir onde armazenar

os conteúdos, pode-se aumentar a taxa de acertos de conteúdos geral da rede, dado que o conteúdo é melhor distribuído pela rede por não concentrá-los em nós específicos que casam com uma métrica apenas. Avaliamos nossa proposta e comparamos seu desempenho com outras estratégias. Obtivemos uma taxa de acertos de conteúdos maior do que as demais baseadas em apenas uma métrica e conseguimos um tempo de *download* próximo ou abaixo das demais, dependendo da topologia.

O restante deste artigo está dividido da seguinte maneira: A seção 2 descreve os trabalhos relacionados sobre estratégias de decisão de cache em ICN; a seção 3 descreve a proposta MCCD; a seção 4 comenta sobre a avaliação de desempenho; a seção 5 analisa e discute os resultados obtidos e a seção 6 conclui o artigo e sugere possíveis trabalhos futuros.

2. Trabalhos Relacionados

A proposta *Cache Capacity-Aware CCN* baseia-se no conceito de *capacidade de cache* [Lee et al. 2013]. Este modelo considera a capacidade de armazenamento em cache livre em todos os nós ao longo do caminho de solicitação de conteúdo. Eles criaram uma métrica chamada de *cache capacity value*(ccv), que é a razão entre o tamanho total do cache do nó pelo número de entradas armazenadas em cache. Com isso, um número de ccv é atribuído para todos os nós, bem como o número de saltos que tal nó está do nó solicitante. O nó alvo a armazenar o conteúdo será o que possuir o maior valor de ccv. Os resultados obtidos por essa proposta se mostraram bastante satisfatório e obtiveram uma taxa de acerto 164% maior do que o resultado da segunda melhor proposta. Entretanto, eles não consideraram em sua avaliação variar o tamanho do cache dos nós a fim de ver o comportamento da proposta em cenários com muito pouco ou razoável tamanho de cache disponível para armazenamento.

Uma estratégia de decisão probabilística chamada de *ProbCache* usa como métrica a capacidade de armazenamento do caminho [Psaras et al. 2012]. Ele copia o conteúdo ao longo do caminho estabelecido na requisição com um certo valor de probabilidade baseado na quantidade de espaço de cache disponível naquele caminho. Este valor é inversamente proporcional à distância do nó que armazena o conteúdo. Quanto mais perto este estiver do nó solicitante maior será a probabilidade de armazenamento. Quando o conteúdo é enviado pelo caminho reverso, um valor de probabilidade é gerado aleatoriamente e é comparado com o valor obtido para cada nó naquele caminho. Caso o valor aleatório seja menor que o valor calculado pela proposta, o conteúdo é armazenado em tal nó. Uma vantagem da proposta é que ela pode armazenar o conteúdo mais rapidamente para próximo dos usuários. No entanto, como não considera a capacidade de armazenamentos dos nós individualmente, a estratégia pode escolher um nó alvo com pouco cache disponível e a política de descarte de conteúdo pode apagar o conteúdo rapidamente e uma requisição para este mesmo conteúdo necessite ser enviada para nós mais distantes.

A proposta denominada de *Cache Less for More - CL4M* - é baseada no conceito de centralidade de intermediação dos nós [Chai et al. 2012]. Tal métrica mede a importância dos nós na rede. Ela informa que se um nó está na maioria de dos caminhos mais curtos entre dois outros nós, ele terá uma maior probabilidade de ter um requisição de conteúdo satisfeita, quando comparado com outros nós e, por isso, é estrategicamente

melhor armazenar o conteúdo em tal nó. Desta forma, pode-se aumentar a taxa de acertos de conteúdo geral da rede e diminuir a taxa de descarte.

O trabalho de denominado de *RankCache* propõe o rankeamento dos caches de todos os nós num caminho de requisição de conteúdo [Avelar and Dias]. Para isso, calcula-se número de acertos de conteúdo requisitado cada nó obteve a partir de uma fórmula proposta. Com o resultado do *rank* calculado, um valor aleatório é gerado e comparado com este valor. Caso seja menor que o valor obtido pela proposta, o conteúdo é armazenado em tal nó. Nos testes de desempenho, obteve-se uma taxa de acertos de conteúdo de 10 a 30% melhor que a segunda melhor proposta avaliada.

Sendo assim, foi proposta uma estratégia de decisão de cache baseada em três métricas, as quais são:*centralidade de intermediação dos nós*, *tamanho de cache disponível* e a *taxa de acertos de conteúdo do nó*. Justifica-se o uso dessas três métricas por terem obtido resultados bastante satisfatórios em suas avaliações, além de considerarem importantes características topológicas bem como dos nós. Quando combinadas, acredita-se que tais características podem ainda melhorar o desempenho de nossa estratégia de decisão de cache. Diferentemente das propostas baseadas em apenas uma métrica, nossa proposta consegue melhor distribuir o conteúdo pela rede, já que não o concentra em nós que casam apenas com um tipo de métrica.

3. Proposta baseada em múltiplas métricas - MCCD

A MCCD utiliza três métricas para decidir em quais nós o conteúdo será armazenado, as quais são o valor de centralidade do nó, tamanho de cache disponível e a taxa de acertos. Para isso, a MCCD escolhe o nó com *maior valor de centralidade de intermediação* , *maior tamanho de cache disponível* e *maior número de acertos no caminho da requisição de conteúdo*.

1. *Centralidade de Intermediação do Nó:* A centralidade de *v* é dada pela seguinte fórmula:

$$Centr_v = \sum_{s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}}$$

   em que σ_{st} é o total do número de caminhos mais curtos do nó *s* to *t* e $\sigma_{st}(v)$ é o número de caminhos que passam por *v*.
   Porém, calcular a centralidade de todos os nós é muito custoso em termos de processamento e memória, principalmente em redes grandes. Por conta disso, o simulador usado calcula o valor de centralidade de todos os nós à priori e guarda tal informação em cada um deles por padrão.
2. *Capacidade de Cache Disponível:* Seja *m* o tamanho total de cache de um roteador de conteúdo em número de entradas para armazenamento. Seja *c* o total de conteúdo armazenado no roteador em número de entradas. A Capacidade de Cache Disponível (CCD) de um nó *v* é dado pela seguinte equação:

$$CCD_v = m - c$$

   Por padrão, os roteadores de conteúdos guardam tal informação sobre quantas entradas disponíveis há em seus caches e, com isso, a MCCD consegue extrair tal informação ao longo dos nós do caminho de requisição.

3. *Taxa de Acertos de Conteúdo*: Seja R o número de requisições de conteúdo que chegam a um roteador de conteúdo r. Seja H o número de acertos de requisições de conteúdos atendidas por r. Então, a Taxa de Acertos de Conteúdo (TAC) de r é dado por:

$$TAC_r = \frac{H}{R}$$

Conforme detalhado no algoritmo MCCD na figura 1, depois que o caminho de requisição do conteúdo tiver sido calculado, são criadas três listas contendo os nós deste caminho. Para cada lista, verifica-se qual nó possui o maior valor de acordo com o critério considerado. Isto é, dentro do caminho de requisição, o nó com maior valor de centralidade é selecionado numa lista, o nó com maior cache disponível em outra e o nó com maior taxa de acertos na última. Para o cálculo do valor de centralidade, utiliza-se um método que calcula este valor para todos os nós no caminho da requisição e depois é extraído aquele com maior valor. Na segunda lista, faz-se um mapeamento entre o nó e o tamanho do cache disponível. Então, a lista é varrida a fim de encontrar aquele com maior espaço disponível. Semelhantemente, para achar o nó com maior taxa de acertos, é feito um mapeamento dos nós com seus respectivos valores de taxa e encontra-se o nó com maior taxa. Tendo encontrado os nós alvos, o conteúdo volta pelo caminho reverso e então verifica-se se o nó corrente é um nó alvo. Em caso afirmativo, uma cópia do conteúdo é armazenada nele. Caso contrário, os próximos nós são verificados até que todos os nós alvos sejam encontrados.

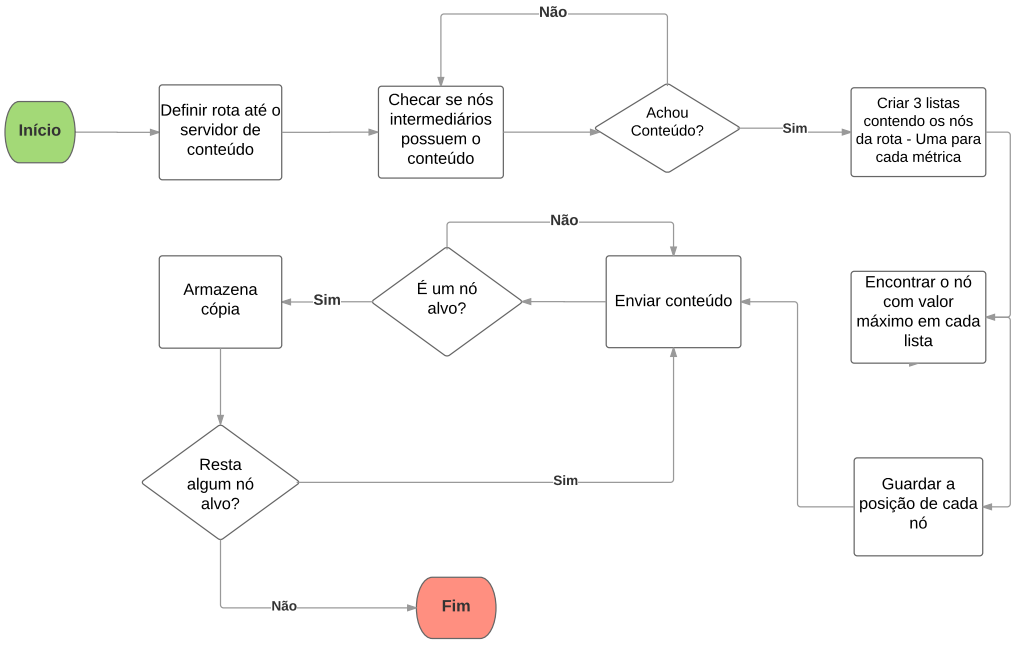


Figura 1. Algoritmo MCCD

Para melhor ilustrar o funcionamento do nosso algoritmo, considere o cenário da figura 2. Assumimos que todos os roteadores (R1 - R11) já possuem conteúdos armazenados em cache, Servidor seja o servidor permanente de conteúdo e os três usuários (1,2 e 3) estejam solicitando conteúdo da rede.

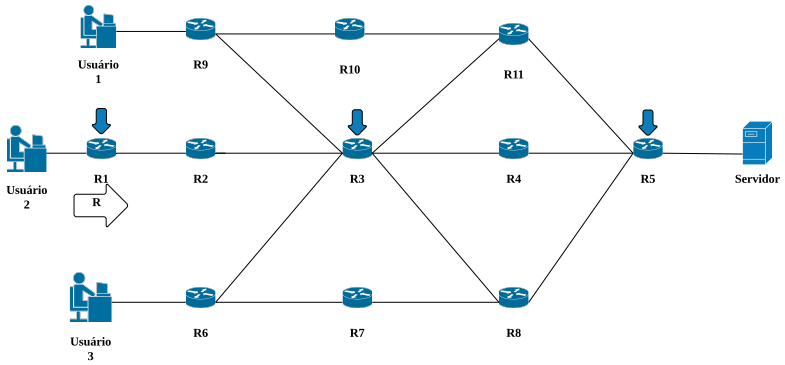


Figura 2. Funcionamento da MCCD

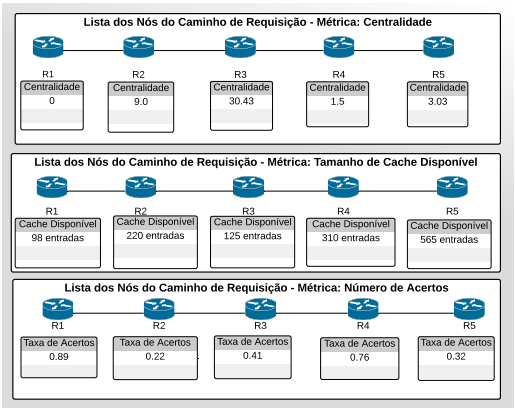


Figura 3. Obtenção das Métricas

Em um dado momento, o usuário 2 envia uma solicitação *R* de um conteúdo *c* para o roteador R1. Este verifica se possui o conteúdo em cache. Se sim, envia-o para o usuário 2. Caso contrário, repassa a *R* para o próximo roteador. Paralelamente, a MCCD vai atualizando o valor das três variáveis a cada salto. Assumimos que *c* não seja encontrado em nenhum roteador intermediário do caminho R1-R5 e *R* chegue até o servidor. Neste momento, a MCCD já selecionou os roteadores que armazenarão o conteúdo quando este voltar pelo caminho reverso. A figura 3 mostra que foram encontrados os roteadores R1, R3 e R5 sendo os nós com *maior taxa de acertos*, *maior centralidade* e *maior espaço de cache disponível*, respectivamente. Caso um nó case com mais de uma métrica, a MCCD evita que o conteúdo seja armazenado novamente neste nó, reduzindo a redundância de cópias na rede.

4. Avaliação de Desempenho

Nossa proposta foi implementada e testada utilizando o simulador Icarus 0.4.0 [9], um simulador de estratégias de de decisão de cache para redes ICN. Este simulador possui algumas implementados e escolhemos alguns deles para comparar seu desempenho e comparar com a nossa proposta.

Tabela 1. Configuração dos parâmetros da simulação

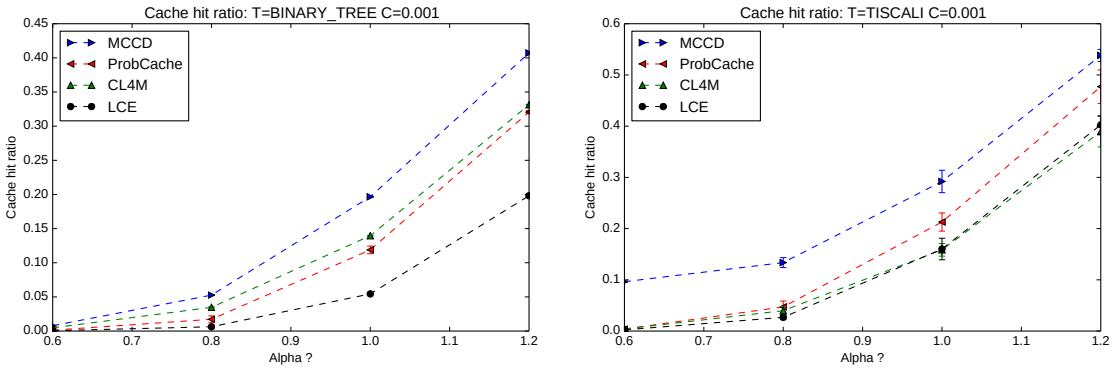
Topologias avaliadas	Árvore Binária e TISCALI
# total de conteúdos no repositório	100.000
Taxa de requisição	100/s
# de requisições para popular caches dos roteadores	100.000
# de requisições avaliadas	200.000
Política de descarte de cache	LRU
Parâmetro Zipf de popularidade de conteúdo	$\alpha = [0.6, 0.8, 1.0, 1.2]$
% de tamanho de cache para os roteadores em relação ao repositório	0.01% 1% e 10%
Estratégias avaliados	MCCD, PROBCACHE, LCE e CL4M
Métricas avaliadas	Taxa de acertos e latência

A tabela 1 mostra os parâmetros de simulação utilizados. As topologias utilizadas foram: Árvore binária com profundidade D = 7. O servidor de conteúdo estava na raiz, os usuários são os nós do último nível da árvore e os nós intermediários são os roteadores de conteúdo. Para a topologia TISCALI, que é um provedor pan-europeu, os clientes são os nós que possuem grau ¡ 4.5. O número de clientes não foi especificado pelos simulador.. Os roteadores possuem grau ¿= 6, resultando em 36 nós; Como servidores foram escolhidos os nós com grau maior que 4.5, resultando em 44 nós. No caso da topologia GEANT, que é um *backbone* europeu, os clientes são nós que possuem grau = 1, resultando em 8 nós; os roteadores são os nós com grau ¿ 2, resultando em 19 nós; os servidores possuem grau igual a 2, totalizando 13 nós. Os 100 K conteúdos foram armazenados nos servidores seguindo uma maneira uniforme. Os conteúdos são tratados como *object* e cada *object* possui 30 vezes o tamanho da sua requisição. O tempo de atraso de links internos é de 2ms e de links externos de 102ms. Foram avaliadas 200.000 requisições de conteúdo e estas requisições seguem um modelo Poisson. A distribuição de popularidade segue um modelo Zipf com um valor de α=[0,6-1,2]. Como tamanho de cache dos roteadores foram configurados três valores: 0.01%, 1% e 10% em relação ao total de conteúdos no repositório; As estratégias avaliados foram a MCCD , PROBCACHE, LCE e CL4M. As métricas avaliadas foram a taxa de acerto, que é definida como a relação entre o número de acertos de conteúdo pelo número total de requisições enviadas, e a latência, que é o tempo necessário para receber o conteúdo solicitado desde sua requisição até o seu recebimento. Todos as avaliações utilizaram um Intervalo de Confiança de 95%.

5. Análise dos Resultados

Primeiramente, analisa-se os resultados de taxa de acertos e depois os resultados do tempo de *download* para as topologias avaliadas. Eles foram analisados por tamanho de cache e, posteriormente, tais resultados serão discutidos mais detalhadamente.

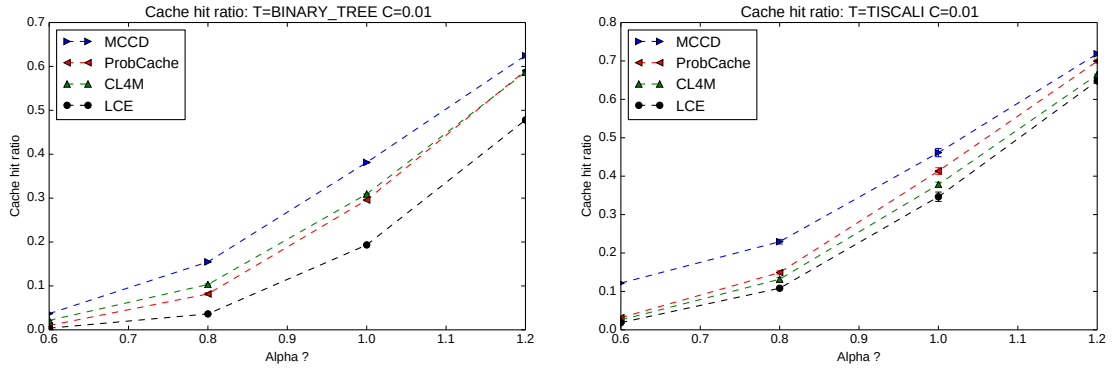
Conforme observa-se na figura 4, para um tamanho de cache de 0.1% a proposta MCCD obteve uma taxa de acerto maior que das outras propostas para ambas as topologias. No teste com árvore binária, (3a), a MCCD conseguiu uma taxa maior para todo valor de alfa, atingindo cerca de 41% de acertos com $\alpha = 1.2$ e uma taxa de acertos 124% melhor que CL4M, que ficou em segundo lugar. Em terceiro lugar ficou PROBCACHE e LCE com o pior resultado. No teste com a topologia TISCALI,(3b), a MCCD obteve uma taxa de acertos bem acima das outras. Com $\alpha = 0.6$, a MCCD obteve uma taxa de 10%, sendo 400% melhor que PROBCACHE para este valor de alfa. Nossa proposta



(a) Árvore Binária **(b)** TISCALI

Figura 4. Taxa de Acertos - Tamanho do Cache = 0.1%

obteve uma certa variação com $\alpha = 1$, mas continuou com um desempenho maior que as demais, conseguindo 53% de taxa de acertos para $\alpha = 1.2$. A proposta PROBCACHE manteve um desempenho similar a CL4M com valores de $\alpha = 0.6 - 0.8$ e superou as outras propostas para valores maiores. CL4M e LCE mantiveram resultados similares e também tiveram uma pequena variação com $\alpha = 1.0$

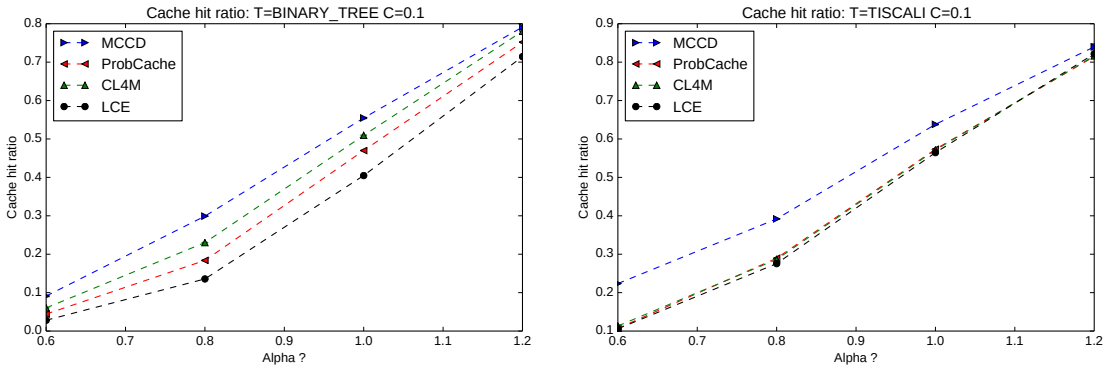


(a) Árvore Binária **(b)** TISCALI

Figura 5. Taxa de Acertos - Tamanho do Cache = 1%

No segundo teste, aumenta-se o tamanho do cache dos roteadores de conteúdo para 1%. Como observa-se na Figura 5(a) na topologia de Árvore Binária, a MCCD obteve uma taxa de acertos maior que as demais avaliadas, atingindo cerca de 62% de acertos para o maior valor de α. Com valores de $\alpha = 0.6 - 1.0$, a CL4M manteve-se acima da PROBCACHE, mas com valores acima deste intervalo as duas se mantiveram praticamente com o mesmo desempenho. A LCE ficou novamente em último lugar. Com relação à topologia TISCALI na Figura 5(b), a MCCD continuou com o melhor desempenho, tendo uma taxa de acerto 400% melhor para um valor de $\alpha = 0.6$, chegando a 70% de acertos para $\alpha = 1.2$. Neste teste, a PROBCACHE obteve um desempenho levemente superior a CL4M, ficando aquela em segundo e esta em terceiro. LCE obteve o pior desempenho novamente.

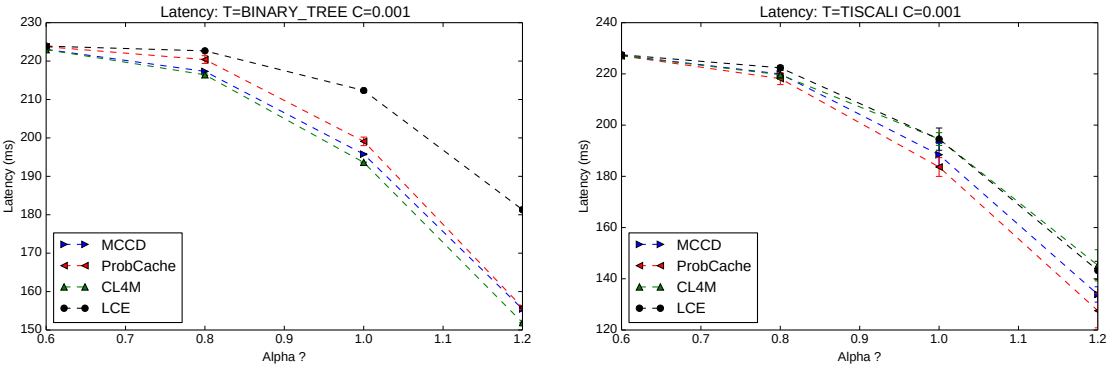
No terceiro teste com um tamanho de cache de 10% na topologia de Árvore Binária, figura 6(a), a MCCD continuou com o melhor desempenho, atingindo cerca de 80% de acertos com $\alpha = 1.2$. A CL4M ficou com o segundo melhor desempenho, seguido



(a) Árvore Binária **(b)** TISCALI

Figura 6. Taxa de Acertos - Tamanho do Cache = 10%

de PROBCACHE e LCE. Neste teste pode-se observar que elas obtiveram resultados bem próximos, mas suas curvas não apresentaram pontos de intercecção. Já na topologia TISCALI conforme a figura 6(b), o desempenho da MCCD foi bastante superior às demais. Com $\alpha = 0.6$, a MCCD foi 209% melhor que a segunda colocada. Considerando todos os valores de α, a MCCD teve uma taxa acima das outras três estratégias, obtendo cerca de 80% de acerto para os conteúdos mais populares. As outras três obtiveram uma taxa de acerto praticamente iguais neste cenário, com uma taxa levemente inferior da LCE para $\alpha = 0.6 - 1.0$.

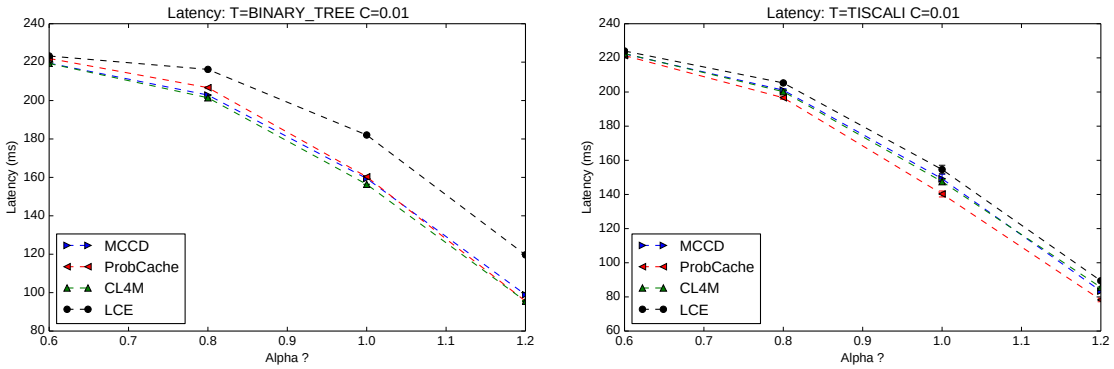


(a) Árvore Binária **(b)** TISCALI

Figura 7. Latência - Tamanho do Cache = 0.1%

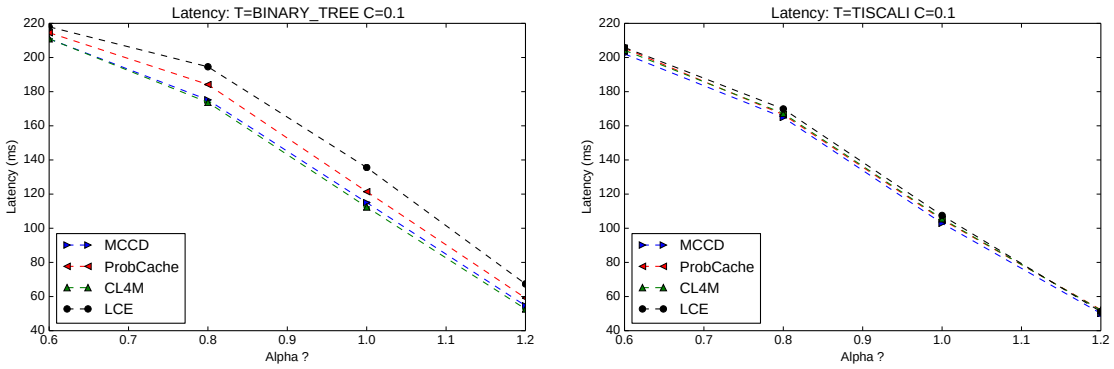
A partir de agora são analisados os resultados para os testes de latência (tempo de *download*), conforme observados na figura 7. Na topologia de Árvore Binária, Figura 7(a), CL4M e MCCD obtiveram um valor de tempo bem aproximados com $\alpha = 0.6 - 0.8$. Para valores maiores, CL4M obteve um tempo menor, sendo que MCCD obteve o segundo melhor tempo, seguido por PROBCACHE e LCE. Na Figura 7(b) para a topologia TISCALI e para $\alpha = 0.6 - 0.8$, PROBCACHE, MCCD e CL4M obtiveram um tempo equivalente. Para valores entre 0.8 - 1.2, PROBCACHE passou a ter o menor tempo, seguido por MCCD, enquanto que CL4M e LCE obtiveram praticamente o mesmo tempo.

A figura 8 mostra a latência para um tamanho de cache de 1%. Em 8(a), percebe-se que os tempos de CL4M e MCCD aproximam-se, tendo agora uma diferença menor quando comparado ao resultado anterior. Nota-se que, em um dado intervalo de $\alpha =$



(a) Árvore Binária **(b)** TISCALI

Figura 8. Latência - Tamanho do Cache = 1%



(a) Árvore Binária **(b)** TISCALI

Figura 9. Latência - Tamanho do Cache = 10%

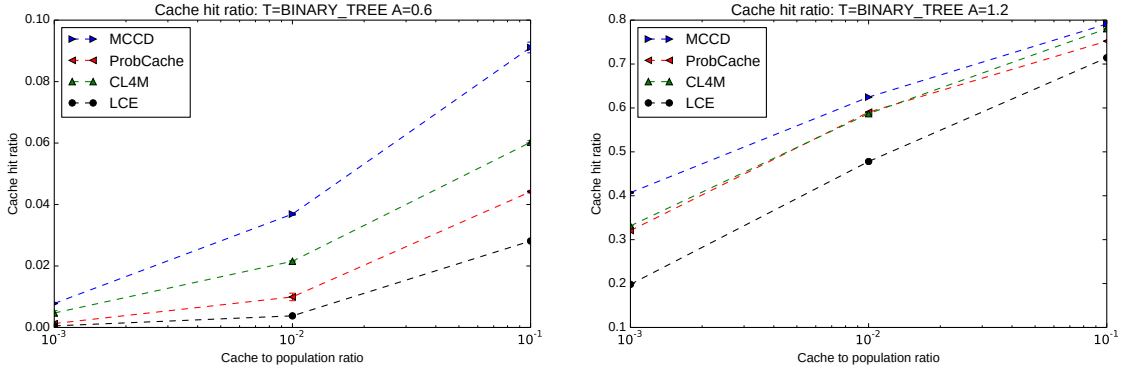
$0.8 - 1.2$, PROBCACHE e MCCD alternam seus resultados, sendo praticamente iguais na maior parte do teste. LCE obteve um tempo bem acima das demais, ficando em último lugar. Já em 8(b), verifica-se que PROBCACHE obteve o menor tempo, ficando MCCD e CL4M com tempos bem aproximados. LCE obteve o maior tempo, mas neste teste ficou bem mais próxima das demais propostas.

Finalmente, avalia-se as topologias com um tamanho de cache de 10%, de acordo com a figura 9. Em 9(a), MCCD e CL4M obtiveram os melhores desempenho, tendo CL4M um tempo um pouco menor quando $\alpha = 0.8 - 1.2$. Diferentemente do teste anterior, PROBCACHE teve um tempo maior, ficando em terceiro lugar e LCE obteve o maior tempo entre todas. Já em 9(b), MCCD obteve o menor tempo entre as propostas para todo α do intervalo considerado, mesmo que a diferença seja bastante sutil. PROBCACHE e CL4M ficaram com o segundo melhor tempo e LCE teve desempenho bem próximo destas últimas, porém, ainda assim, com o maior tempo.

5.1. Discussão dos Resultados

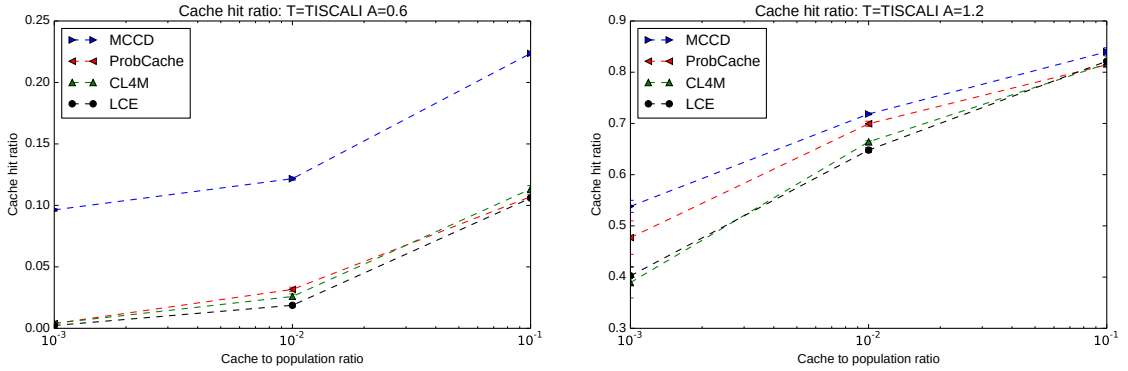
Nesta seção, os resultados são discutidos para as duas métricas avaliadas para as topologias utilizadas. Uma árvore binária é definida por dois parâmetros, os quais são: K, o fator de espalhamento e D, a profundidade da árvores a partir do nó raiz. Em árvores binárias, D impacta diretamente no comprimento dos caminhos entre os nós folhas e a

raiz e K na diversidade de caminhos. Neste tipo de topologia, os nós intermediários à raiz e os nós folhas apresentam praticamente o mesmo valor de centralidade por conta da regularidade apresentada por esta topologia. Desta forma, os nós possuem praticamente a mesma probabilidade para armazenar o conteúdo, sendo que esta probabilidade pode aumentar quando D é pequeno, fazendo com que haja uma diversidade baixa de caminhos para entrega de conteúdo.



(a) Árvore Binária ($\alpha = 0.6$) **(b)** Árvore Binária ($\alpha = 1.2$)

Figura 10. Taxa de Acertos x Tamanho do *Cache*



(a) TISCALI ($\alpha = 0.6$) **(b)** TISCALI ($\alpha = 1.2$)

Figura 11. Taxa de Acertos x Tamanho do cache

Com relação aos resultados obtidos por nossa proposta para este tipo de topologia, observou-se que a MCCD obteve uma taxa de acertos de conteúdo maior devido a possibilidade de armazenar o conteúdo em outros nós alvos que fogem à regularidade da topologia (nós que apresentam praticamente o mesmo valor de centralidade). Desta forma, a MCCD conseguiu obter um resultado de taxa de acertos maior que as demais já que, quando uma métrica não tem um desempenho satisfatório, as outras duas compensam esse baixo desempenho, fazendo com que a MCCD tenha vantagem às demais propostas por causa desse comportamento.

Considerando as figura 10(a), observa-se que a MCCD obteve um valor de taxa de acerto maior para $\alpha = 0.6$, na topologia de Árvore Binária. Isto se deve ao fato de que já que para este valor de α há uma maior quantidade de conteúdos com a mesma popularidade e, portanto, maior número de cópias mantidas em cache na rede. Entretanto,

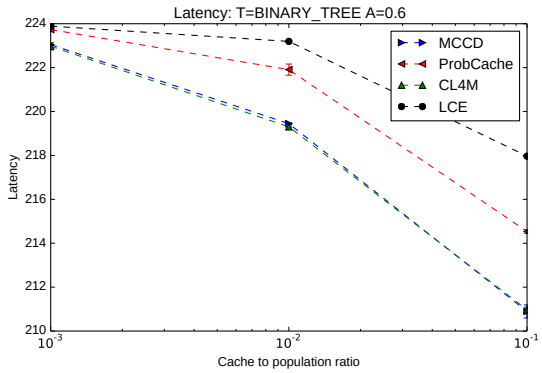
como a MCCD trabalha com mais de uma métrica, isto permite que esse conteúdo possa ser encontrado em outros nós. Ainda, a medida que o tamanho do cache aumenta, este conteúdo tende a ficar por mais tempo armazenado tanto por haver mais espaço disponível quanto também pelo fato da MCCD utilizar como métrica a quantidade de espaço de cache disponível.

Quando consideramos um valor maior de $\alpha = 1.2$, figura 10(b), há poucos conteúdos com a mesma popularidade na rede, por isso que a taxa passa a crescer quase que linearmente. Mesmo assim, a MCCD atinge uma taxa maior do que as outras propostas, dado que os nós que armazenam o conteúdo residem no caminho de requisição de outros nós que também solicitam o mesmo conteúdo, melhorando o desempenho da nossa proposta.

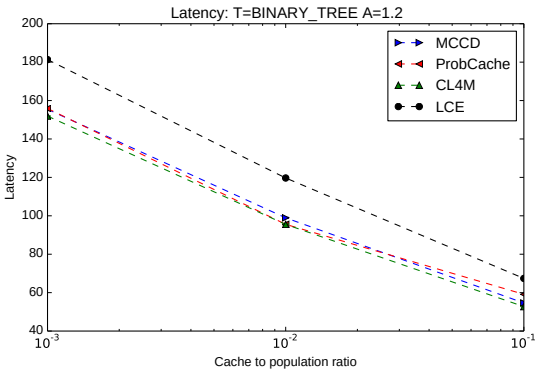
Agora, discute-se os resultados para a topologia TISCALI representado na figura 11. Com o menor valor de $\alpha = 0.6$, figura 11(a), a MCCD obteve uma taxa de acertos crescente à medida que aumenta-se o tamanho do cache dos roteadores. Da mesma forma como na árvore binária, a maior quantidade de conteúdos espalhados pela rede permite que eles sejam encontrados mais facilmente, sendo com uma taxa quase dez vezes maior para o menor valor de cache (0.01%). Isso mostra que a nossa estratégia é bem eficiente quando temos um tamanho bem pequeno alocado para cache. Quando $\alpha = 1.2$ observa-se que ainda assim a MCCD consegue uma taxa de acerto acima das demais para os tamanhos de cache considerados. Por se tratar de um *backbone* intercontinental composto por 161 nós, 328 links e nós com grau médio igual a 4.07 [Domingo-Pascual et al. 2011], essa maior diversidade de características topológicas em relação à árvore binária, a MCCD consegue distribuir o conteúdo mais uniformemente entre os nós que casam com suas métricas pela rede. Além disso, o fato de haver uma menor probabilidade de um nó casar com mais de uma métrica por causa dessa variedade faz com que o conteúdo não se concentre em poucos nós, o que resultaria em uma menor taxa de acertos de conteúdo da rede.

Também, serão discutidos os resultados do tempo de *download* para as duas topologias considerando os mesmos dois valores de α. Na árvore binária, quando $\alpha = 0.6$, figura 12(a), a MCCD obteve um tempo praticamente igual ao do esquema CL4M, que utiliza como métrica a centralidade do nó. Isto ocorre pois, considerando a regularidade da topologia, percebe-se que a métrica centralidade do nó se sobressai às demais e, portanto, as duas propostas obtiveram tempos semelhantes, para os três tamanhos de cache avaliados. Já com um valor de $\alpha = 1.2$, figura 12(b), a MCCD teve um tempo médio ora maior que CL4M e PROBCACHE ora menor que um deles ou ambos. Pode-se atribuir ao fato de que o conteúdo solicitado seja encontrado em nós que casem com as duas outras métricas base da MCCD (tamanho do cache e taxa de acertos) e tais nós podem estar mais distantes do(s) nó(s) que casa(m) com a métrica centralidade do nó.

Para a topologia TISCALI com $\alpha = 0.6$, figura 13(a), a MCCD só teve um tempo de *download* menor quando o tamanho do cache era de 10%. Como os nós apresentam diferentes graus de centralidade, links e tamanho de cache é possível que isso tenha impactado no tempo da proposta. Quando $\alpha = 1.2$, figura 13(b), o tempo a MCCD praticamente permaneceu com o mesmo desempenho de $\alpha = 0.6$. PROBCACHE obteve um tempo menor praticamente que todas as outras propostas, sendo que com o cache de 10% elas convergiram para tempos aproximados.

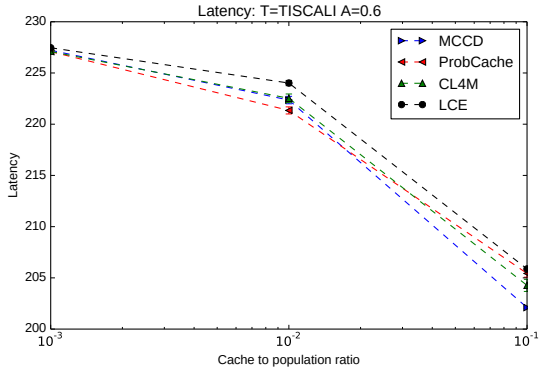


(a) Árvore Binária ($\alpha = 0.6$)

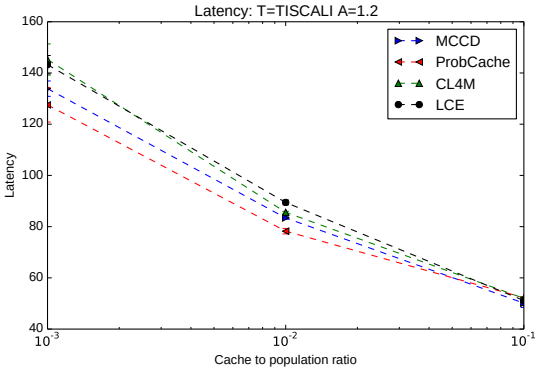


(b) Árvore Binária ($\alpha = 1.2$)

Figura 12. Latência x Tamanho do *Cache*



(a) TISCALI ($\alpha = 0.6$)



(b) TISCALI ($\alpha = 1.2$)

Figura 13. Latência x Tamanho do *Cache*

6. Conclusão e Trabalhos Futuros

A maioria das estratégias de cache em redes ICN consideram apenas em uma métrica para escolher em qual nó armazenar o conteúdo. Dependendo do dinamismo da rede e das requisições de conteúdo, ter apenas uma métrica como base de decisão pode não ser apropriado para atender as requisições dos usuários de uma forma satisfatória. Isso pode trazer uma baixa taxa de acertos de conteúdo, já que os conteúdos são armazenados em poucos nós que casam com tal métrica. Os testes feitos por este trabalho mostraram que as estratégias baseadas em uma métrica não obtiveram uma taxa de acertos maior quando comparadas com nossa proposta, baseada em três métricas.

Os resultados mostraram que tanto para uma topologia genérica quanto para um *backbone* a MCCD obteve uma taxa de acertos maior que as demais e recuperou o conteúdo com um tempo de *download* ora abaixo ora igual às demais estratégias avaliadas. Como trabalhos futuro planejamos otimizar este tempo, bem como o uso de outras métricas que possam melhorar ainda mais nossos resultados como, por exemplo, a popularidade do conteúdo, e a avaliação de outras métricas como, por exemplo, redução do número de salto para a obtenção de conteúdo e redução de solicitações de conteúdos atendidas pelo servidor, mostrando a eficiência da estratégia de cache.

7. Agradecimentos

Agradecemos à CAPES pela ajuda e suporte financeiro dados a este trabalho.

Referências

Ahlgren, B., Dannewitz, C., Imbrenda, C., Kutscher, D., and Ohlman, B. (2012). A survey of information-centric networking. *Communications Magazine, IEEE*, 50(7):26–36.

Avelar, E. A. M. and Dias, K. L. Armazenamento em redes orientadas a conteúdo baseado em ranqueamento de caches.

Chai, W. K., He, D., Psaras, I., and Pavlou, G. (2012). Cache "less for more" in information-centric networks. In *NETWORKING 2012*, pages 27–40. Springer.

Cisco (2015). Cisco visual networking index: forecast and methodology, 2014-2019. Technical report, Cisco.

Dannewitz, C. (2009). Netinf: An information-centric design for the future internet. In *Proc. 3rd GI/ITG KuVS Workshop on The Future Internet*.

de Brito, G. M., Velloso, P. B., and Moraes, I. M. (2012). Redes orientadas a conteúdo: Um novo paradigma para a internet. *Minicursos do Simpósio Brasileiro de Redes de Computadores-SBRC*, 2012:211–264.

Domingo-Pascual, J., Manzoni, P., Palazzo, S., Pont, A., and Scoglio, C. (2011). *NETWORKING 2011: 10th International IFIP TC 6 Networking Conference, Valencia, Spain, May 9-13, 2011, Proceedings*, volume 1. Springer Science & Business Media.

Jacobson, V., Smetters, D. K., Thornton, J. D., Plass, M. F., Briggs, N. H., and Braynard, R. L. (2009). Networking named content. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, pages 1–12. ACM.

Kutscher, D., Eum, S., Pentikousis, K., Psaras, I., Corujo, D., Saucez, D., Schmidt, T., and Waehlisch, M. (2014). Icn research challenges. *Work in progress*.

Lee, S.-W., Kim, D., Ko, Y.-B., Kim, J.-H., and Jang, M.-W. (2013). Cache capacity-aware ccn: Selective caching and cache-aware routing. In *Global Communications Conference (GLOBECOM), 2013 IEEE*, pages 2114–2119. IEEE.

Psaras, I., Chai, W. K., and Pavlou, G. (2012). Probabilistic in-network caching for information-centric networks. In *Proceedings of the second edition of the ICN workshop on Information-centric networking*, pages 55–60. ACM.

Zhang, G., Li, Y., and Lin, T. (2013). Caching in information centric networking: a survey. *Computer Networks*, 57(16):3128–3141.