UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE CIÊNCIAS EXATAS E NATURAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

IGOR FURTADO CARVALHO

UMA ESTRATÉGIA DE DECISÃO DE CACHE BASEADA EM MÚLTIPLAS MÉTRICAS PARA REDES ORIENTADAS A CONTEÚDO

BELÉM-PA

03/03/2016

IGOR FURTADO CARVALHO

UMA ESTRATÉGIA DE DECISÃO DE CACHE BASEADA EM MÚLTIPLAS MÉTRICAS PARA REDES ORIENTADAS A CONTEÚDO

Dissertação submetida à banca julgadora na Universidade Federal do Pará como parte dos requisitos para obtenção do grau de Mestre em Ciência da Computação

Orientador: Dr. Antônio Jorge Gomes Abelém

BELÉM-PA

03/03/2016

IGOR FURTADO CARVALHO

UMA ESTRATÉGIA DE DECISÃO DE CACHE BASEADA EM MÚLTIPLAS MÉTRICAS PARA REDES ORIENTADAS A CONTEÚDO

Dissertação submetida à banca julgadora na Universidade Federal do Pará como parte dos requisitos para obtenção do grau de Mestre em Ciência da Computação

Aprovada em: __/__/____

BANCA EXAMINADORA

Prof. Dr. Antônio Jorge Gomes Abelém
Universidade Federal do Pará
Orientador

Prof. Dr. Mauro Margalho
Universidade Federal da Amazônia
Membro Externo

Prof. Dr. Aldebaro Klautau
Universidade Federal do Pará

Prof. Dr. Eduardo Cerqueira
Universidade Federal do Pará

Dedico a Deus, meus pais, familiares e minha noiva

CAPÍTULO 1

Agradecimentos

Primeiramente agradeço ao Senhor Deus Todo-Poderoso pela graça de poder concluir mais uma etapa de minha vida acadêmica. Sei que sem a ajuda dEle jamais poderia ter chegado até aqui. Foram dois anos de muito aprendizado, sacrifício, dedicação e tenho plena convicção que em todos os momentos de dificuldades Ele estava comigo, ajudando-me a vencê-las. A Ele, toda Honra, Poder, Glória e Louvor!

Agradeços aos meus pais pela força e incentivo durante este período. Sou agraçiado por ter dois pais (Manoel e Hélio) abençoados que não mediram esforços para me ajudarem em tudo que precisei. Sem falar de minha amada mãe Sandra que com muito amor, carinho e paciência e, claro, muitas orações me ajudou bastante nesta caminhada. A vocês dedico esta vitória com imensa gratidão e essa conquista vossa também. Aos meus irmãos e irmãs, especialemnte a minha irmã Amanda e meu cunhado Júnior, pelo apoio e compreensão durante este período.

À minha noiva Samanta pelo amor, carinho e paciência durante estes dois anos. Você foi também muito immportante para que pudesse concretizar mais esta etapa e dedico a você também esta vitória.

Agradeço ao meu orientador professor Dr. Antônio Abelém pela ajuda, orientação, confiança e paciência, as quais foram fundamentais para a concretização desta etapa. Que Deus possa lhe abençoar grandemente. Agradeço ao meu coorientador Msc. Airton Ishimori pela ajuda, paciência e pelas discussões e orientações. Que Deus também possa lhe abençoar ricamente!

Agradeço a todos os membros do GERCOM LABTIC, que direta ou indiretamente, ajudaram-me com suas dicas, discussões e orientações e enriqueceram de alguma forma esta dissertação.

Agradeço à CAPES pelo suporte financeiro dado para a concretização deste tra-

balho.

Resumo

Resumo da Dissertação apresentada à UFPA como parte dos requisitos necessários para obtenção do grau de Mestre em Ciência da Computação.

Uma Estratégia de Decisão de Cache baseada em Múltiplas Métricas para Redes Orientadas a Conteúdo

Orientador: Dr. Antônio Jorge Gomes Abelém
Co-orientador:
Palavras-chave: Estratégia de Cache, Métricas, Redes Orientadas a Conteúdo

Rede Orientada a Conteúdo - ROC - (*Information Centric Network* - ICN) é um paradigma de arquitetura de rede que atraiu muitas pesquisas nos últimos anos. Um tópico bastante pesquisado em ROC é o cache em rede, o qual permite que roteadores intermediários entre um usuário e um servidor de conteúdo possam armazenar o conteúdo requisitado. Tal decisão é feita através de uma política de decisão de cache, a qual pode ser baseada tanto em características do conteúdo trafegado quanto da topologia ou dos nós.

Porém, a maioria delas fazem uso de apenas uma métrica para escolher os nós alvos que armazenarão o conteúdo. Considerando que a rede é um ambiente bastante dinâmico, assim como o tipo de conteúdo solicitado e trafegado, tal métrica pode acarretar em um baixo desempenho tanto da rede quanto da estratégia em si. Isso resulta em uma baixa taxa de acertos de conteúdos, longos atrasos e um maior consumo de recursos de rede, como, por exemplo, a largura de banda.

Desta forma, esta dissertação propõe uma estratégia de decisão de *cache* baseado em múltiplas métricas para ROCs. São utilizadas três métricas para selecionar os nós alvos e, com isso, o conteúdo pode ser melhor distribuído na rede. A estratégia proposta considera características dos roteadores de conteúdo (centralidade e capacidade de cache disponível), bem como de conteúdo (taxa de acertos de conteúdo registrado por cada

roteador) e, assim, melhor distribuir o conteúdo por toda a rede. Os resultados obtidos atingiram uma taxa de acertos bem maior que as outras propostas avaliadas e um tempo de *download* praticável.

Abstract

Abstract of Dissertation presented to UFPA as a partial fulfillment of the requirements for the degree of Master in Computer Science.

A Caching Strategy based on Multiple Metrics for Information Centric Networks

Advisor: Dr. Antônio Jorge Gomes Abelém
Co-advisor:
Key words: Strategy, Metrics, Information Centric Networks.

Information Centric Networks (ICN) has been a novel network architecture paradigm that changes the way of communication using a host-to-content approach. One of the most important features in ICN is in-network caching, which enables intermediate content routers to cache content between a user and a content server. This caching decision is based on some important features of the content, topology or nodes. Usually, these features are used on the basis of some caching decision strategies.

However, most of the current strategies are based on a specific metric to select the target caching nodes. Taking into account the dynamical behavior of the network and content type, such metric may decrease both network and strategy performance. This results in a low content hit ratio, longer delays and network bandwidth waste.

This way, this thesis dissertation proposes a new caching decision strategy based on multiple metrics for ICN. Three metrics are used to choose the target nodes, which are centrality, cache capacity and cache hit ratio of routers. So, content may be better spread over the network and meet user's content requests suitably. The results obtained show a higher hit ratio than the other evaluated strategies and a feasible download time.

Sumário

Lista de Abreviaturas

CCN - Content Centric Networking
CDN - Content Delivery Networks
CINC - Cooperative In-Network Caching
CS - Content Store
DHT - Distributed Hashing Table
DNS - Domain Namae System
DONA - Data Oriented Network Architecture
FIB - Forwarding Information Base
FTP - File Transfer Protocol
HTTP - HypeText Transfer Protocol
MCCD - Multi-Criteria Caching Decision
NDN - Named Data Networking
PIT - Pending Interest Table
PSIRP - Publish Subscribe Internet Routing Paradigm
ROCs - Redes Orientadas a Conteudo

Lista de Simbolos

Lista de Figuras

Lista de Tabelas

CAPÍTULO 2

Introdução

2.1 Limitações da Atual Arquitetura da Internet

Na atualidade, a Internet é considerada um complexo sistema de informação multimídia baseado na distribuição de conteúdos. Subentende-se por conteúdo dados textuais, codificados ou multimídias, como documentos, paginas web, arquivos de áudio e vídeo, e também metadados, que são usados para gerenciar, localizar e interpretar os próprios conteúdos. [1].

Atualmente, mesmo com usuários leigos, é possível publicar conteúdos na Internet, o que traz uma certa inconsistências às informações geradas. Por isso, o uso persistente de identificadores representa um grande desafio à distribuição de conteúdo na Internet. Outro ponto é a escalabilidade, ou seja, mecanismos de localização e encaminhamento de conteúdos precisam ser eficientes para o número de usuários e conteúdos disponibilizados na escala da Internet. Finalmente, é preciso permitir o acesso seguro a conteúdos , que objetiva garantir a autenticidade e o controle de acesso aos conteúdos disponíveis. Ainda não existe uma única solução que atenda a todos esses requisitos. Há apenas técnicas que atendem parcialmente ou que tentam contornar as limitações da atual arquitetura da Internet [2].

2.2 Motivação e Desafios

De acordo com um estudo feito pela Cisco, o tráfego IP global crescerá 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 [3]. 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 de conteúdo, resultando em alta carga no servidor e longos atrasos.

Como forma de acompanhar essas mudanças no uso da Internet, o paradigma orientado a conteúdo tem se mostrado muito relevante no que diz respeito a dar um melhor suporte para a obtenção de conteúdo. Pelo fato de dá suporte ao cache de conteúdo nos nós intermediários, isto ajuda a reduzir longos atrasos e melhor distribuição do conteúdo na rede.

Dentro do paradigma ICN, há muitos problemas no que diz respeito a *cache* em rede. De acordo com [4], dois deles são mencionados como o problema de alocação de cache e 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 localização espacial dos usuários em uma parte específica na rede. O outro está relacionado com a distribuição de conteúdo nos *caches*, que considera 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*) [4].

Mas para que isso ocorra, uma estratégia apropriada de decisão de cache de conteúdo deve ser bem definida e implementada a fim de decidir quais nós armazenarão o conteúdo. Por isso, é de fundamental importância que tal estratégia atenda às necessidades e dinamismo da rede e proveja o suporte para as suas diversas aplicações. Ainda, muitas ferramentas foram desenvolvidas para auxiliar tanto na implementação quanto na validação de resultados. Simuladores, ambientes de experimentação e emuladores são algumas com as quais é possível validar melhor as propostas para a área [**?**].

2.3 Objetivos

2.3.1 Objetivo Geral

O presente trabalho tem como objetivo propor e implementar uma estratégia de decisão de cache em Redes Orientadas a Conteúdo utilizando múltiplas métricas, as quais são: *centralidade dos nós, capacidade de cache disponível e taxa de acertos de conteúdo dos roteadores*. Tais métricas foram selecionadas porque consideram características relevantes tanto da rede quando dos nós e, quando combinadas, podem melhorar ainda mais o desempenho da estratégia em si. Com isso, espera-se obter um menor tempo de download dos conteúdos e uma maior taxa de acertos geral da rede, que são duas métricas bastante avaliadas dentro de ROCs [4].

2.3.2 Objetivos Específicos

Para alcançar o objeto principal deste trabalho, faz-se necessário a realização dos seguintes objetivos específicos:

- Levantamento do estado-da-arte no contexto de caching em ROCs;
- Projetar o algoritmo que será a base da estratégia proposta;
- Implementar o algoritmo em um simulador de estratégia de cache para ROCs;
- Definir cenários de testes, parâmetros de avaliação, métricas coletadas e validação da proposta.
- Discussão dos resultados e como poderiam ser aplicados em cenários reais.

2.4 Organização do texto

O restante do documento estão dividido seguindo o ordenamento descrito abaixo:

- Capítulo 3: Referencial Teórico - Descreve o estado-da-arte dentro de ROCs, as principais arquiteturas implementadas, bem como o modo de funcionamento de cada uma. Em seguida, um estudo aprofundado sobre cache em ROCs é feito, explicando as principais vantagens, usos e problemas encontrados. Por fim, o capítulo aborda algumas estratégias de decisão de cache propostas em ROCs.
- Capítulo 4 : Trabalhos Relacionados: Neste capítulos são detalhados os trabalhos relacionados sobre algumas estratégias de decisão de cache encontrados na literatura. Uma breve descrição do seu funcionamento é feita, bem como algumas vantagens e desvantagens encontradas em cada um.
- Capítulo 5: Proposta Baseada em Múltiplas Métricas - Neste capítulo, explica-se detalhadamente cada métrica, como elas são utilizadas pelo algoritmo proposto e apresenta um cenário de aplicação, a fim de um melhor entendimento do funcionamento. Ainda, são listadas algumas das estratégias de decisão de cache existentes, seu funcionamento, algumas vantagens e as principais desvantagens de cada uma. Vale ressaltar que a maioria delas trabalha apenas com uam métrica principal e o presente trabalho com múltiplas métricas a fim de obter um melhor desempenho que elas.
- Capítulo 6: Avaliação da Proposta - Aqui são listados os cenários de testes, topologias utilizadas, principais parâmetros da simulação e as métricas avaliadas. Em seguida, os resultados são discutidos e algumas observações são feitas sobre o uso da proposta em ambientes reais.

- Capítulo 7: Conclusão - Neste capítulo, descreve-se da importância do cache em rede em ROCs, assim como do uso de uma estratégia eficiente, que procure atender as requisições mais apropriadamente. Por isso, esta dissertação propôs uma estratégia baseada em múltiplas métricas a fim de prover uma solução para isso. Finalmente, cita-se alguns trabalhos futuros e outras métricas que poderão ser avaliadas.

CAPÍTULO 3

Redes Orientadas a Conteúdo

3.1 Limitação da Arquitetura centrada em sistemas finais

A Internet é uma rede de comutação de pacotes em escala global e os pacotes são encaminhados utilizando o principio do melhor esforço oferecido pelo protocolo IP . Não existe reserva de recursos para cada usuário da rede, nem diferença no tratamento dos pacotes encaminhados pelos roteadores. Desta forma, não há garantia nenhuma de desempenho ou segurança fim-a-fim para a distribuição do conteúdo na Internet atual. Como a comunicação é baseada em sistemas finais, a estação de origem precisa indicar no cabeçalho dos pacotes o endereço IP da estação destinatária com a qual deseja realizar a comunicação. Os nós intermediários são responsáveis por encaminhar os pacotes entre a origem e o destino, utilizando apenas o endereço IP para isso da estação destino. Este modelo atende às necessidades iniciais da Internet, que tinha como objetivo o compartilhamento de recursos. Entretanto, ele não é eficiente para a distribuição de conteúdo, pois, além de exigir do usuário a prévia localização do conteúdo, também necessita da sua identificação.

Uma das formas de aumentar a eficiência da distribuição do conteúdo é o redirecionamento HTTP para tratar a localização do conteúdo em virtude da sua alta volubilidade [5]. Objetos webs são solicitados através de localizadores de recursos, denominados URLs *(Uniform Resource Locator)* , existentes nos cabeçalhos de mensagens HTTP. Os redirecionamentos HTTP são eventos disparados pelo servidor que hospedava os objetos, enviando como resposta mensagens de redirecionamento com uma nova URL em seu cabeçalho. Como essa solução depende da localização do conteúdo, é necessário ter mecanismo que garantam o acesso persistente a esses conteúdos, independentemente de localização, propriedade ou qualquer característica pertinente ou associada a ele.

Esse é um exemplo de modelo de aplicação cliente-servidor utilizado por muitas aplicações de distribuição de conteúdo na atual Internet. Nesse modelo, um canal ponto-a-ponto é estabelecido entre um dado cliente e um servidor. Então, vários clientes que solicitam um dado conteúdo simultaneamente são atendidos através do estabelecimento de múltiplos canais ponto-a-ponto e do envio de uma cópia do mesmo conteúdo em cada canal. Nesse cenário, quanto mais um conteúdo for popular, maior a ineficiência do uso dos recursos, principalmente, em termos de banda passante. Por isso, é necessário o desenvolvimento de de técnicas de encaminhamento eficiente para conferir escalabilidade à distribuição de conteúdo na Internet.

3.2 Abordagem baseada em Conteúdo

As Redes Orientadas a Conteúdo [2] mudam radicalmente a forma de comunicação na Internet, pois possui uma abordagem baseada apenas no conteúdo e enfatizam o acesso à informação independente da localização, tornando a arquitetura de rede mais adequada para distribuir o conteúdo. As ROCs fazem usam conceitos como conteúdo nomeado, roteamento baseado em nomes, segurança aplicada diretamente no conteúdo, armazenamento de dados nos nós intermediários [6]. Desta forma, esta nova abordagem traz vários desafios para o desenvolvimento de ROCs, como método de nomeação e roteamento de conteúdo, técnicas de proteção de conteúdos e usuários, utilização de cachê n o núcleo da rede , entre outros. A seguir, os conceitos básicos relativos às ROCs serão detalhados, mostrando as suas vantagens e desvantagens.

3.2.1 Nomeação de conteúdo

Para se obter um determinado conteúdo na Internet atual, isso implica o conhecimento de todas as partes envolvidas na transferência de dados através do endereços IP. Dessa maneira para obter um determinado conteúdo requer o conhecimento a priori da sua localização na topologia da rede, ou seja, o conhecimento do endereço do sistema final que o hospedará, para poder estabelecer uma conexão e poder requisitar uma cópia do conteúdo. Isto amarra de forma estrita os conceitos de identificação e localização de conteúdos. Falando de identificação de conteúdos, a abordagem das ROCs baseia-se em uma premissa bastante diferente da abordagem tradicional da arquitetura centrada em estações. Por tratar os conteúdos como primitiva básica de rede, as ROCs tornam possível a obtenção de conteúdo através, apenas, de sua identificação ou nome, utilizando esquemas de nomeação de conteúdos, com propriedades bastante específicas. Deve-se ter um esquema especial para nomeação que apresente as seguintes características:

- Unicidade: O nome atribuído ao conteúdo deve ser único.
- Persistência: Nome e conteúdo devem ser correspondentes.

- Escalabilidade: Ser adotado em vários cenários, servindo a espaços de nomes de pequeno e grande porte, não impondo limitações quanto à sua natureza, local de armazenamento ou qualquer outra característica.

3.2.2 Roteamento de Conteúdo

Ao contrário das redes centradas na comunicação entre estações, as ROCs tem que ser capazes de entregar os conteúdos requisitados por nome com nenhuma informação referente à localização, tanto de usuários quanto de armazenamento de conteúdos. Para isso, os nós das ROCs precisam obter informações a respeito dos conteúdos existentes na rede para poder encaminhar, na melhor forma possível, solicitações de conteúdo e cópias de conteúdos requisitados. O roteamento de conteúdo apresenta as seguintes características:

- Orientado a conteúdo: Endereçar pacotes através de nomes de conteúdos, sem informações ou indicações de remetente e destinatário.
- Robustez: Possuir tolerância à falhas e recuperação rápida em situações de descontinuidade, evitando encaminhar dados para nós que não estão ativos.
- Eficiência: Baixo impacto na quantidade de tráfego na rede por causa das informações de controle.
- Escalabilidade: Ser utilizada em diferentes topologias, pequenas ou grandes.

3.2.3 Armazenamento em Cache

Em ROCs, trabalha-se com a premissa de utilizar os elementos de rede para armazenamento do conteúdo. Baseado na característica de acesso de conteúdos na Internet, no qual apenas uma parcela de conteúdos populares contribuem com a maior parte do tráfego na rede, a replicação dos conteúdos e a disponibilização dos mesmos por nós da rede mais próximos aos usuários implica um grande grau de redução de tráfego e melhoria dos níveis de qualidade de serviço.

Roteadores de conteúdo podem ter as suas funcionalidades estendidas para poder ter uma infraestrutura distribuída de armazenamento, nos mesmos moldes das tradicionais CDNs. À medida que encaminha conteúdo a diferentes nós da rede, um roteador pode armazenar os conteúdos mais acessados e guardar em memória, funcionando como um cache de rede .O armazenamento de conteúdos em ROCs é diferente da utilizada em CDNs. Nestas, além de avaliar a popularidade dos conteúdos através das quantidades de requisições feitas por usuários em escala global, os nós operam de forma orquestrada com um gerenciamento centralizado para otimizar a distribuição das réplicas e a utilização de recursos. A decisão de armazenamento de conteúdos nas ROCs baseia-se somente nas informações locais de conteúdo, isto é, os nós apenas utilizam as requisições e conteúdos em trânsito na determinação do armazenamento. Basicamente, qualquer nó da rede incluindo

as estações dos usuários podem atuar como um cache, a qualquer momento, possibilitando estender as vantagens atuais proporcionadas por CDNs privadas a uma rede verdadeiramente pública e global de armazenamento e distribuição de conteúdo.

Pelo fato de não tratar de localização do conteúdo diretamente, o uso de armazenamento em cache na rede acaba distribuindo cópias de conteúdos para nós distantes uns dos outros. Tal cenário pode configurar um problema para os protocolos de roteamento, já que a agregação de rotas pode ser algo bem complexa, o que irá impactar na distribuição ótima de informações de roteamento.

3.3 Arquiteturas Orientadas a Conteúdo

3.3.1 Data Oriented Network Architecture - DONA

DONA *(Data Oriented Network Architecture)* é uma arquitetura baseada nos conceitos de nomeação e localização de conteúdos para a distribuição persistente e confiável dos mesmos em uma rede hierárquica [7]. Ela propicia persistência e autenticidade através da utilização de nomes planos e auto-certificadores. Essa alta disponibilidade é dada pelo mecanismo de localização de conteúdos, levando as solicitações de conteúdo até as cópias com menor custo de obtenção, evitando nós falhos ou sobrecarregados.

Em DONA, todo nó é gerado por um outorgante, que é uma entidade associada a um par de chaves pública e privadas as quais são usadas para a identificação dos conteúdos. Isso é fundamental para a formação dos nomes em DONA. Tais nomes são do formato P:L,em que P representa o *hash* criptográfico de chave pública do outorgante e L é um rótulo arbitrário escolhido pelo mesmo, para que cada nome seja único no seu domínio. Os outorgantes tem papel de publicadores e administradores de conteúdos, já que somente nós autorizados pela chave P podem permitir acessos a objetos nomeados do tipo P:L.Já que todo usuário, ao solicitar um conteúdo P:L terá como resposta o conteúdo composto pelos dados, chave pública de P, o rótulo L, metadados e uma assinatura do conteúdo, pode-se imediatamente checar a autenticidade do publicador dos dados verificando-se que o hash da chave pública é, realmente, P e que a mesma foi utilizada na assinatura do conteúdo. O uso de nomes planos traz a dificuldade de associação dos nomes de conteúdo pelos usuários. A arquitetura considera como premissa o fato que usuários obtêm nomes por diversos mecanismos externos à rede, como sistemas de busca, comunicação privada, serviços de recomendação e a confiança utilizada pelos usuários.

O mecanismo de resolução de nomes, isto é, de roteamento de requisições de conteúdo nomeado, é implementado em nós denominados manipuladores de registros *(register handlers - RHS)*, que utilizam um protocolo bastantes simples e eficaz. Pacotes FIND (P:L)são enviados ao RH local para localizar determinado objeto P:L, que por sua vez encaminha a solicitação às cópias mais próximas dos conteúdo. Mensagens do tipo REGISTER(P:L)estabelecem o estado necessário para que RHs encaminhem os pacotes FIND de forma eficaz. O outorgante autoriza nós a enviarem pacotes do tipo REGIS-

TER(P:*)ao seu RG local. Desta maneira, independente do rótulo L utilizado, toda a requisição de conteúdos sob responsabilidade do outorgante P será encaminhada pelo RH local do nó que registrou P:* e P:L , mapeando de forma distinta os próximos saltos para cada uma das entradas. A existência de entradas na tabela de registros é importante no encaminhamento dos pacotes FIND ao RH de nível hierárquico superior, eventualmente encontrando uma entrada na tabela de registros, já que RHs de níveis hierárquicos superiores concentram as informações de roteamento dos seus RH filhos (ou subdomínios), funcionando como uma espécie de gateway padrão. Da mesma forma que CBN, DONA não prevê o rompimento completo com o IP. O pacote FIND é inserido entre os cabeçalhos IP e da camada de transporte, limitando-se a resolução dos endereços de conteúdos. Portanto, os mecanismos convencionais de transporte são adicionados para a realização de entrega dos conteúdos a partir de cópias armazenadas na rede, apenas orientando tais mecanismos a nomes, sem maiores mudanças nos protocolos e infraestrutura que comportam.

Uma das funções essenciais para qualquer sistema de distribuição de conteúdo seria a seleção automática de servidores, e isso é obtido de forma nativa em DONA. RHs podem optar pelo encaminhamento de FINDs a vizinhos de menor custo, segundo a métrica de roteamento em DONA. Multihoming e mobilidade são, também, características intrínsecas a DONA, já que FINDs podem ser encaminhados a mais de um RH por um nó multi-homed, resultando na utilização de múltiplos caminhos para a obtenção de conteúdo. O protocolo de registro de conteúdos, baseado nas mensagens REGISTER e UNREGISTER, é o responsável por dar mobilidade aos sistemas finais, já que antes da mudança de posicionamento do host na topologia da rede, o mesmo deve cancelar o registro dos seus endereços de conteúdo e registrá-los novamente em sua nova localidade. Desta forma, assim que novos registros forem divulgados e o estado de encaminhamento necessário tiver sido estabelecido, todos os FINDs serão roteado à essa nova localidade. A distribuição do conteúdo utilizando o multicast é, também, realizada de forma nativa, pois a utilização de identificadores P:L, com o rótulo arbitrário L representando a identificação de um grupo multicast do tipo (S,G), permitem a criação de grupos centrados em fontes específicas, similar à implementação da comunicação multicast SSM *(Source Specific Multicast)*.

Foram também propostas algumas extensões para DONA, intensificando o seu impacto na distribuição dos conteúdos. O uso de cache nos RHs estendem as suas funcionalidades, provendo uma infraestrutura genérica e sempre disponível nos caminhos de distribuição para o armazenamento de conteúdos, conferindo ganhos de qualidade de serviço no acesso de conteúdos. Outra extensão é o mecanismo de assinatura de conteúdos e notificação de atualizações através de FINDs de longa duração, isto é, adicionados de TTLs (time-to-live). Enquanto o FIND for válido, atualizações pertinentes ao conteúdo desejado serão enviadas em direção ao host interessado. Outra funcionalidade desejável é a capacidade de se evitar servidores falhos ou sobre carregados no provimento de conteúdos. Ao enviar REGISTERs à rede, tais nós podem incluir informações a respeito de carga e processamento, facilitando a tomada de decisões relativas ao encaminhamento de FINDs pelos RHs.

3.3.2 Content Centric Networking(CCN)/ Named Data Networking(NDN)

CCN *(Content Centric Networking)* é uma arquitetura de ROC baseada na utilização do conteúdo como objeto elementar da rede, tratando questões como alta disponibilidade e segurança de conteúdos, independendo da localização [1]. Assim como as outras propostas já citadas anteriormente, CCN mantém alguns conceitos do TCP/IP que o tornaram simples, robusto e escalável, estendendo para prover uma camada de rede flexível, com poucas restrições à camada de enlace.

Uma das principais características em CCN é a divisão do conteúdo em pedaços *(chunks)* , que são estruturas nomeadas com identificadores únicos e hierárquicos e requisitados de forma individual. Os nomes são compostos por um número variável de componentes. Cada componente é formado por um numero arbitrário de octetos, que não tem significado algum para a camada de transporte, podendo ser ainda criptografados. Uma vantagem em se utilizar nomes hierárquicos é a possibilidade de agregação de nomes fazendo referências a nós raízes da árvore hierárquica. A estruturação do nome em forma hierárquica permite, também, a expressão do posicionamento dentro da estrutura hierárquica, definindo inclusive o posicionamento relativo a outros nós da árvore de nomes. O interesse por tal conteúdo é satisfeito caso o nome do conteúdo no pacote de interesse for um prefixo do nome no pacote de dados. Isso é a mesma coisa que dizer que o pacote de dados está na subárvore de nomes especificados pelo pacote de interesse.

A arquitetura CCN é baseada em dois conceitos, que é a declaração do interesse por determinado chunk e o envio deste em resposta ao interesse. Os usuários solicitam conteúdo diretamente à rede difundindo seu interesse por determinado *chunk*, na forma de pacotes de interesses *(Interest packet, ou I-packet)*, em todas as interfaces disponíveis. Os nós vizinhos respondem ao I-packet mandando como resposta o pacote de dados (Data packet, ou D-packet) caso o tenham armazenado em memória. Do contrário, os nós encaminham o I-packet aos seus vizinhos até que o interesse encontre um nó com o dado armazenado. Os dados são apenas enviados como resposta ao interesses, consumindo o endereço equivalente em cada nó no caminho reverso,estabelecendo uma espécie de balanço entre requisições e atendimento.

O encaminhamento de pacotes é realizado pelos nós CCN parecido de como é no IP, baseado no mapeamento entre nome do conteúdo e interface associada à árvore de distribuição do pacote, com algumas características especiais. Cada roteador utiliza três estruturas distintas nas operações de encaminhamento de pacotes: o CS *(Content Store)*, a PIT *(Pending Interesting Table)* e a FIB *(Forwarding Information Base)* , mostradas na figura 1. A FIB é uma base de dados utilizada no armazenamento de informações de encaminhamento de pacotes, fazendo o mapeamento entre nomes e conteúdos em uma ou mais interfaces para o encaminhamento, permitindo o uso de múltiplas fontes de forma nativa. CS é uma estrutura de cache do roteador em CCN, é também uma área de armazenamento de chunks pelo tempo mais longo possível, usando políticas de atualização de cache similares a LRU (Least Recently Used) e LFU (Least Frequently Used). A PIT

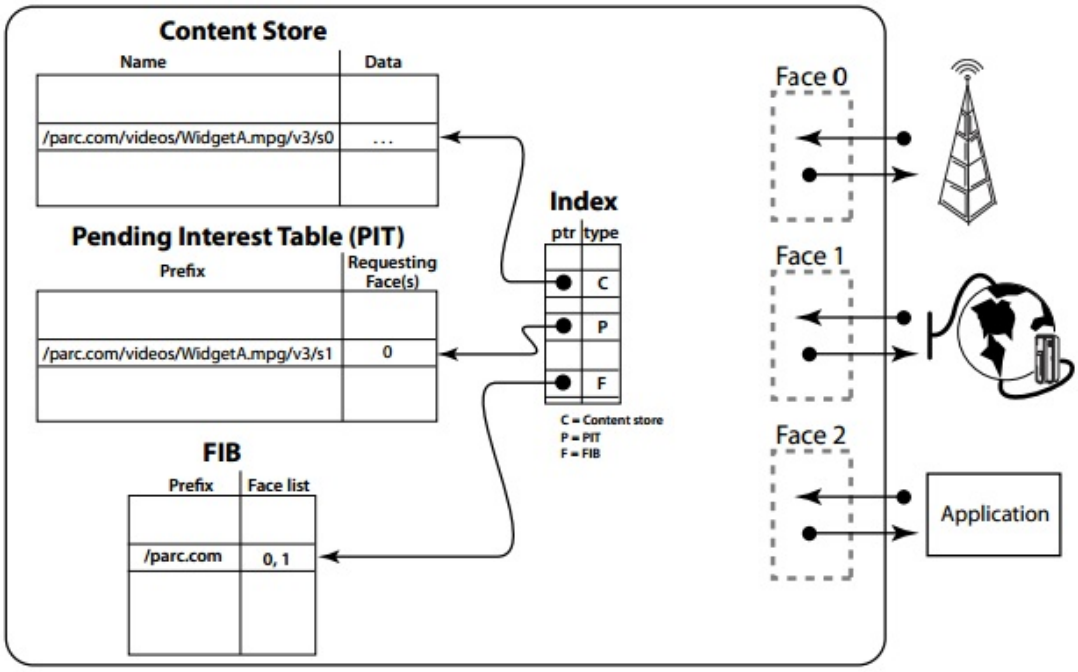

Figura 1: Funcionamento de um nó CCN

é uma estrutura que serve para armazenar os interesses sobre conteúdo passados adiante, registrando as interfaces de origem para que os dados enviados como resposta possam ser encaminhados ao solicitante do conteúdo. Ao receber um I-packet , primeiramente o roteador checa no CS se já existe um interesse na PIT pendente.

Em caso afirmativo, envia como resposta o D-packet relativo. Caso não exista nenhuma entrada armazenada no CS do roteador, verifica se já existe um interesse pendente na PIT. Em caso positivo, a interface de recebimento do I-packet é adicionada à lista de interfaces para encaminhamento de conteúdo na PIT e o I-packet é descartado. Caso não haja entrada na PIT, o roteador encaminha o pacote de acordo com regras de sua FIB, e cria na PIT um registro da interface de origem. Com esse tipo de encaminhamento difuso encontra-se, eventualmente, um nó que atender a solicitação e envie o dado pelo caminho reverso, sinalizado pelas entradas nas PITs nos nós intermediários. Somente uma entrada na PIT valida a admissão do D-packet pelo roteador, com todos os outros cenários levando a um descarte do pacote. Mas para isso , é preciso que as fontes registrem sua intenção de prover determinados conteúdos, através de uma primitiva de registro, criando o estado de encaminhamento inicial necessário para o envio dos interesses às fontes.

Em CCN existe uma camada chamada de camada de estratégia, que implementa o mecanismo de encaminhamento de pacotes que atua na FIB, determinando dinamicamente a forma como um roteador de conteúdos encaminha os pacotes de interesse. Diferentemente do TCP, em CCN cabe a camada de estratégia do receptor requisitar conteúdos corrompidos ou não entregues. O controle de fluxo é implementado pela ca-

mada de estratégia, uma vez que o envio de múltiplos pacotes de interesses em paralelo endereçados a chunks sequenciais possui função equivalente à janela TCP, controlando a quantidade de trafego que pode ser inserida na rede pela fonte de dados.

Dada as características de nomeação e encaminhamento em CCN, qualquer roteamento baseado em estado de enlace válido para IP possui um potencial de uso em CCN. O mecanismo de encaminhamento de CCN não impõe restrições quanto ao uso de múltiplas fontes ou destinos, uma vez que o uso da PIT impede a formação de laços na rede. A utilização de nomes hierárquicos, com semântica semelhante a dos endereços IP, o que confere agregabilidade aos endereços de rede em CCN, aplicando-se o mesmo mecanismo de casamento do maior prefixo *(longest prefix match)*.

Com relação à segurança, em CCN ela e aplicada diretamente no conteúdo, independente dos mecanismos de segurança adotados pelos meios de transporte. A autenticação do vínculo entre nome de dados é obtido pela assinatura do publicador do conteúdo sobre o nome e os dados do conteúdo. O vínculo entre nome e conteúdo permite que publicadores possam atribuir nomes arbitrários às suas publicações e as tornas facilmente autenticáveis, pois qualquer nó na rede pode avaliar se o vínculo entre nome e conteúdo foi assinado por determinada chave. A determinação do mecanismo de autenticação de vínculo pode variar entre diferentes conjuntos de publicadores e usuários, criando uma flexibilidade de adequação dos recursos computacionais de acordo com a necessidade de cada aplicação. Pode-se, ainda, distribuir a carga computacional de autenticação entre vários pacotes, apesar de pacotes serem pensados como autenticáveis individualmente. A validação do vínculo é simplesmente sintática, isto é, valida-se que a chave foi utilizada na assinatura do conteúdo sem inserir nenhum significado a ela, como propriedades e critérios para a confiança na chave.

A arquitetura proposta para CCN é a base do projeto NDN *(Named Data Networking)* [8] . Este projeto visa desenvolver as técnicas complementares necessárias a plena adoção das ROCs, abordadas na proposição de CCN. Questões como roteamento global, encaminhamento eficiente de conteúdo nomeado, o desenvolvimento de aplicações, as técnicas de segurança e privacidade, entre outras, fazem parte da agenda do projeto. O projeto NDN possui um testbed com 31 nós distribuídos pelos principais centros de pesquisas dos Estados Unidos. Usuários conectam-se ao testbed via túneis UDP, estendendo o alcance da ROC a todo o campus e às redes domiciliares dos usuários.

3.3.3 Publish Subscribe Internet Routing Paradigm/ Publish-Subscribe Internet Technologies

O projeto *Publish Subscribe Internet Routing Paradigm* (PSIRP), ao contrário as demais propostas de ROCs, especifica uma arquitetura de ROC sem aplicar tecnologias de conectividade e transporte existentes, como no TCP/IP [9]. Fortemente baseada nas primitivas de rede, publish/subscribe, PSIRP define as publicações (como são chamados os conteúdos) como associações persistentes entre identificadores e dados criados pelo

publicador. Identificadores autocertificáveis são utilizados neste vínculo entre nome e conteúdo, na forma de *hashes*, já que cada publicação possui um único publicador lógico.

PSIRP utiliza o conceito de *rendezvous* (que vem do francês e significa "encontro") para implementar a resolução de nomes de conteúdos. publicadores anunciam o conteúdo em redes de *rendezvous* locais, que realizam a associação de fonte de dados e assinantes interessados no conteúdo armazenado. Publicadores anunciam o escopo *(Scope ID - Sid)*, que são identificadores relacionados aos conteúdos que autorizam sua distribuição por outras fontes de dados. As fontes de dados são nós de armazenamento localizados nas extremidades da rede que utilizam o sistema de rendezvous para a divulgação dos conteúdos existentes, o identificador de rendezvous (Rendezvous ID - Rid). Assim, todo o conteúdo deve ser solicitado através do uso da dupla de identificadores Sid , que direciona a forma de distribuição autorizando determinadas fontes, e Rid , indicando a publicação desejada neste escopo. Sid e Rid usam pares de identificadores "P:L"em que P é a chave pública do proprietário do espaço de nomes e L é um rótulo arbitrário de publicação. As redes locais de *rendezvous* (Rendezvous Interconnect - RI), uma DHT hierárquica com presença em todos os domínios da arquitetura PSIRP, que permite resolver Sids e Rids de conteúdos disponíveis em dominós distintos.

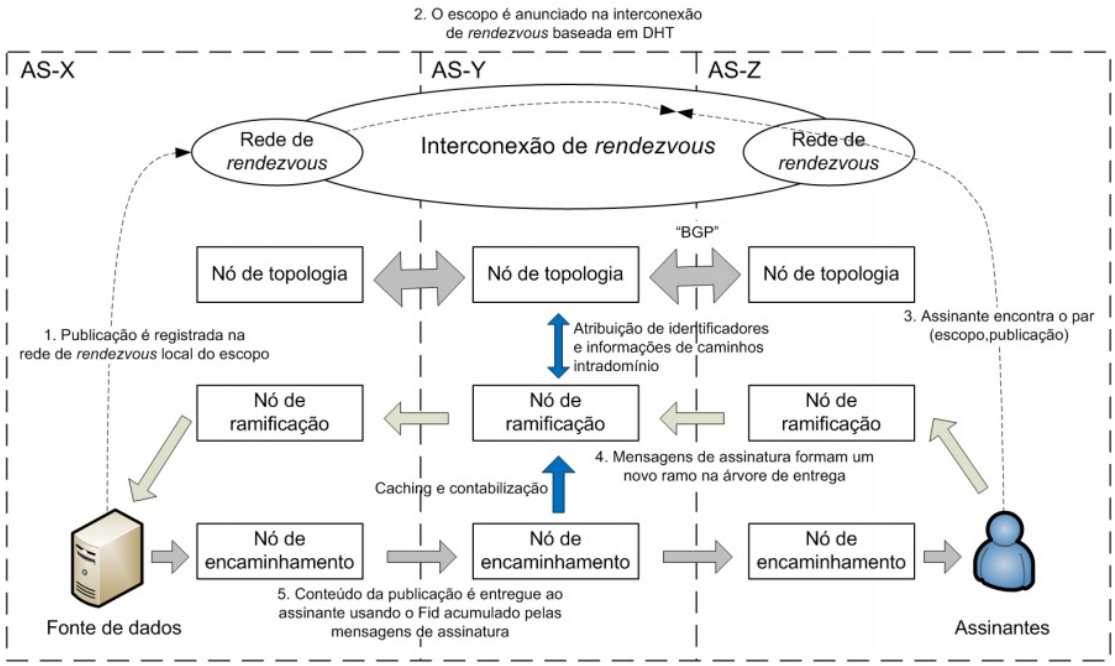

Figura 2: Arquitetura adotada em PSIRP

A resolução de identificadores de uma publicação na RI devolve ao usuário a indicação de nós de ramificação *(Branching Nodes - BNs)* da árvore de distribuição relativa à publicação. A requisição de assinatura da publicação é, então, encaminhada através dos diversos domínios de PSIRP, utilizando rotas para os BNs corretos, como na figura 2. O sistema de roteamento é responsável pela determinação e manutenção da árvore de entrega de cada publicação e pelo processo de armazenamento de conteúdos populares na rede. Os BNs selecionam as rotas para as novas assinaturas baseados nas informações de topologia da rede e de métricas obtidas a partir de medidas de intensidade de tráfego,

conseguidas por um sistema distribuído de gerenciamento topológico, além de gerenciar grandes caches e realizar a ramificação de árvores de distribuição quando há múltiplas requisições, parecidos com o multicast.

O sistema de encaminhamento encarrega-se de entregar as publicações aos assinantes através de árvores de entrega eficientes. Os nós de encaminhamento *(Forwarding nodes - FNs)* entregam pacotes através do mapeamento da árvore de distribuição da publicação de um identificador de encaminhamento (Forwarding ID - Fids) no encaminhamento. Fids são identificadores baseados em Filtros Bloom construído pelo sistema, numa espécie de roteamento pela fonte. Uma vez que todos os enlaces possuem uma identificação (Link ID), também um filtro Bloom, é possível codificar a árvore de entrega através de um filtro e utilizá-lo como Fid para o encaminhamento. Cada FN na árvore de entrega do conteúdo realiza uma simples operação lógica de AND no link ID, assume-se que o mesmo faz parte da árvore de entrega e o pacote é encaminhado pela interface correspondente. Esse mecanismo é sujeito a falsos positivos na identificação de interfaces de encaminhamento, causando o envio de pacotes a nós que, na verdade, não participam da árvore. Vejamos um exemplo é representado na Figura 3.

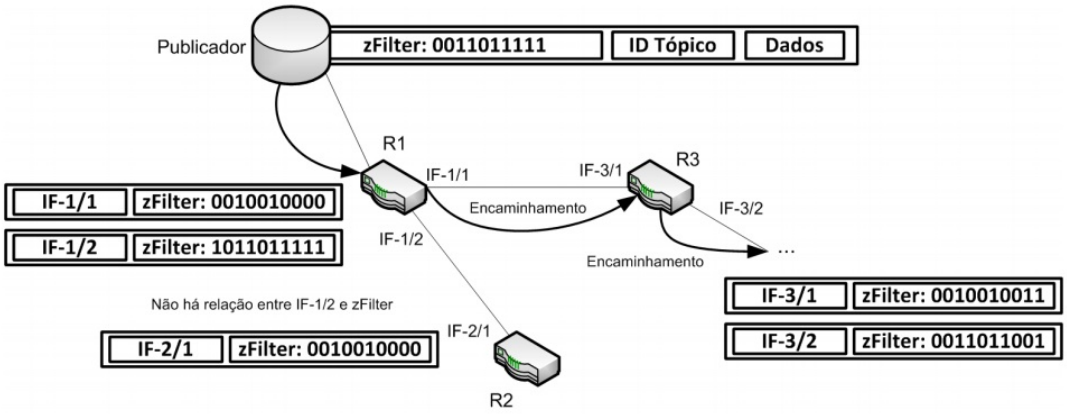

Figura 3: Exemplo de encaminhamento de dados usado em PSIRP

Baseado nos conceitos levantados pela proposição da arquitetura PSIRP, o projeto PURSUIT *(Publish/Subscribe Internet* Technologies) foi criado com o intuito de explorar e refinar ainda mais a proposta da PSIRP. Mobilidade, privacidade, armazenamento em rede e contabilização são temas de grande importância no desenvolvimento do projeto. Um dos principais objetivos do projeto PURSUIT é o desenvolvimento de soluções e mecanismos aplicáveis à criação e oferta de serviços inovadores, aproveitando todo o potencial dessas novas estruturas de informação. A identificação algoritmica de conteúdos é um conceito explorado em PURSUIT através de novas técnicas de fragmentação de conteúdo e armazenamento distribuído eficiente. A natureza de armazenamento em PURSUIT, recursiva e hierárquica, permite gerenciar mobilidade de forma nativa, orientada ao nó móvel, transparente ao núcleo da rede. Desta forma, suporte à mobilidade é um dos tópicos de interesse no desenvolvimento do projeto. Mecanismos de autenticação e contabilização devem ser estendidos para suportar os sistemas de PURSUIT. Questões relacionadas ao gerenciamento de políticas de topologia, roteamento e encaminhamento,

em todos os níveis da arquitetura, requerem o desenvolvimento de uma plataforma de gerenciamento global. O projeto PURSUIT possui um *testbed* com servidores dedicados localizados em diversas instituições americanas e europeias, executando mais de 25 nós da rede PURSUIT simultaneamente.

3.3.4 Outras arquiteturas de ROCs

Além das arquiteturas apresentadas, outras foram propostas visando a implementação de ROCs. TRIAD [10], NetInf [11] e MultiCache [12] são bom exemplos. TRIAD foi uma arquitetura pioneira a propor uma a comunicação baseada em conteúdos, utilizando as URLs presentes nas requisições HTTP como nome de conteúdos. TRIAD realiza o redirecionamento das requisições para as cópias mais próximas, conceitualmente bastante próxima das CDNs, implementando a camada de conteúdo e armazenamento em todos os roteadores de conteúdo. NetInf , proposta pelo projeto 4WARD, utiliza conceitos comuns com DONA e PSIRP/PURSUIT, como nomes planos e roteamento baseado em DHT. NetInf propicia o uso de identificadores persistentes e a separação entre publicação e dados, permitindo o uso de múltiplas versões da mesma publicação. *MultiCache* explora a comunicação multidestinatária como meio de distribuição de conteúdo, implementando uma rede sobreposta baseada em Pastry para o armazenamento de conteúdo em rede, localização de réplicas na rede e distribuição do conteúdo em si.

3.4 Cache em ROCs

O uso da Internet mudou drasticamente nos últimos anos, mudando do modelo de comunicação centrado na comunicação entre máquinas para uma abordagem centrada na obtenção de conteúdo pelo usuário. Com o objetivo de se adaptar a esse novo funcionamento, algumas arquiteturas de Redes Orientadas a Conteúdo(ROCs) foram propostas. Um ponto em comum entre ela é a utilização de cache em rede a fim de melhorar a eficiência da transmissão e disseminação de conteúdo. Comparado com outros modelos como, por exemplo, Web Cache e CDN *(Content Delivery Networks)*, as ROCs tem diferentes características: o cache é transparente para as aplicações, ele está distribuído por toda a rede já que conteúdo pode ser segmentado. Tais características introduzem novos desafios para as tecnologias de cache em ROCs.

Para melhor lidar com essa mudança no uso da Internet, algumas arquiteturas inovadoras de ROCs foram propostas, algumas mais populares como DONA, CCN , NetInf e PSIRP. Uma vantagem notável de tais propostas é que elas provêm nativamente suporte escalável e altamente eficiente para a obtenção de conteúdo, ao mesmo tempo com melhor suporte à mobilidade e segurança.

Em ROCs, os usuários não estão interessados de *onde* o conteúdo vem, mas apenas no *que* acessar. A ideia por trás das ROCs é promover o conteúdo em si como premissa fundamental na rede. Ao invés de centrar a comunicação através de endereços IPs, em

ROCs o conteúdo é localizado e roteado através dos nomes, desacoplando o conteúdo da sua localização.

Com o objetivo de aliviar a pressão que o rápido crescimento do tráfego impõe no que diz respeito ao consumo de largura de banda, uma abordagem bem comum em ROCs é prover cache em rede de forma transparente e espalhado por toda a rede, com isso acelerando a disseminação do conteúdo e melhorar utilização dos recursos de rede. Apesar do cache em rede já ser utilizado na arquitetura TCP/IP para reduzir o consumo de largura de banda, a falta de uma identificação única dos conteúdos torna difícil o aproveitamento do cache em rede [13], [14], [15]. Por exemplo, a URL é utilizada tanto como localizador como identificador de conteúdo na rede. Quando duas cópias do mesmo objeto são armazenadas em diferentes servidores de diferentes provedores de conteúdo, diferentes URLs serão usadas para identificar e acessar o conteúdo. Como resultado disso, o sistema de web cache tratará ambos como objetos distintos.

Consequentemente, embora as técnicas que tentam otimizar o cache em rede tenham sido estudada profundamente, novas características em ROCs como transparência e maior granularidade de cache tornaram teorias tradicionais sobre o assunto, técnicas e modelos de otimização desenvolvidas para Web cache e cache CDN impossíveis de serem diretamente portadas para caches em ROCs.

3.5 Características de Cache em ROCs

3.5.1 Transparência

Os tradicionais sistemas de cache são fechados, dependente de aplicações específicas e desenvolvidos para uma classe tráfego particular como, por exemplo, a Web, CDN ou aplicações P2P. Apesar do Web Cache ser baseado no protocolo HTTP, os conteúdos web seguem a convenção de nomeação baseada em domínios. Ou seja, duas cópias do mesmo objeto em diferentes domínios tem nomes diferentes. Para o sistema de cache, os objetos são logicamente segregados pelos domínios. As aplicações P2P tipicamente utilizam protocolos proprietários, tornando cada aplicação P2P um sistema fechado.

A fim de superar tal problema, pesquisadores estão tentando tornar o cache mais transparente para as aplicações. Por exemplo, uma proposta da IETF [16], [17] objetiva prover uma infraestrutura compartilhada de cache para que cada aplicação possa gerenciar o seu espaço de cache independentemente. Entretanto, a falta de uma convenção de nomeação unificada ainda torna difícil conseguir um compartilhamento de cache entre aplicações. O problema de lidar com protocolos fechados e inconsistência de nomeação pode ser resolvido de forma elegante dentro da infraestrutura de ROCs.

Primeiramente, os nomes de conteúdos em ROCs devem ser únicos e persistentes, isto é, corresponder ao conteúdo em questão. Geralmente, esses nomes são autocertificáveis [7], [18], simplificando a verificação de segurança do conteúdo. Segundo, as

ROCs fazem roteamento e decisões de alocação de cache baseados em nomes únicos de conteúdo, essencialmente fazendo com que tais nomes sejam sensíveis ao contexto da rede. Esta característica torna o cache um serviço comum, aberto, transparente e independente das aplicações.

Porém, a transparência de cache também traz muitos desafios, incluindo:

- **Inconsistência entre objetivos do uso de cache**
  O objetivo dos sistemas tradicionais de cache são geralmente simples, mas podem variar de um para ou outro. Por exemplo, Web Cache objetiva reduzir o tráfego de rede a latência percebida pelos usuários, ao passo que o cache em P2P está preocupado primariamente na redução no tráfego da rede. Como uma nova infraestrutura de rede, espera-se que as ROCs possam atender a uma variedade maior de tráfego incluindo Web, Vídeo sob Demanda*(Video on Demand - VoD)*, compartilhamento de arquivos, etc. Esses tipos de tráfegos mudaram os objetivos de cache na rede, portanto, as ROCs devem fazer uma escolha razoável sobre a escolha do seu objetivo de cache, mantendo o equilíbrio entre os diferentes tráfegos.

- **Compartilhamento de espaço de cache concorrentes entre aplicações**
  Diferentes tipos de tráfegos diferem significantemente na escala de sua população, tamanho e popularidade do objeto [19]. Por exemplo, a população de objetos Web é enorme, na ordem de bilhões, mas o tamanho do objeto é tipicamente pequeno. Por outro lado, o tamanho da população de tráfego VoD é muito pequeno, mas o tamanho do objeto é muito grande. A grande heterogeneidade desses tráfegos impõe novos desafios para os tradicionais, fechados e dedicados sistemas de cache, necessitando que as ROCs sejam capaz de compartilhar cache e recursos eficientemente entres os diferentes tipos de tráfegos.

- **Caches operando à mesma taxa de transferência**
  A transparência do cache em rede traz novos desafios no que diz respeito à velocidade de operação dos caches. Sugere-se que os roteadores de caches operem à taxa de transferência da rede, tornando o gerenciamento do cache bem diferente do tradicional gerenciamento baseado em disco [20], [21].

3.5.2 Cache Omnipresente

Em sistemas tradicionais de cache, a localização dos caches são geralmente predeterminado e tipicamente a topologia do cache é propositalmente formada por uma estrutura linear 4(a) ou uma estrutura de árvore hierárquica 4(b). A alocação do conteúdo e a coordenação entre os caches pode ser determinada quando soluciona-se o modelo analítico estabelecido à priori pela demanda do tráfego e estrutura do cache. Em ROCs, os caches são omnipresentes, ou seja, estão espalhados por toda a rede e não há mais pontos específicos para cache. A topologia da rede de caches evolui de uma árvore hierárquica para grafos arbitrários 4(c), como mostra a figura 4. Desta forma, o relacionamento pai-filho

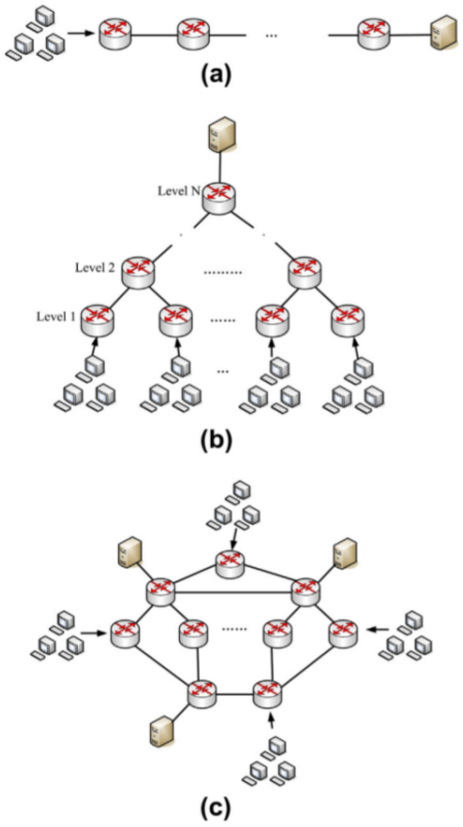

Figura 4: Topologias de Caches: (a) linear , (b) árvore (c) grafo arbitrário

deixa de existir. Tais fatores acrescentam dificuldades para a modelagem matemática e análise do sistema de cache, além de tornar a coordenação explícita mais difícil de se atingir.

A omnipresença do cache também torna sua disponibilidade mais sutil. Nos tradicionais Web caches e CDN, a disponibilidade de um conteúdo para uma requisição é clara. Em CDN, o conteúdo é proativamente alocado em servidores próximos da borda da rede, baseado no conhecimento de acessos anteriores e na estrutura da rede. Tais sistemas são baseados em mecanismos como redirecionamento e resolução pelo DNS, assegurando uma disponibilidade global das cópias de tal conteúdo. Em sistemas hierárquicos de Web cache, apenas o conteúdo alocado ao longo da rota do ponto de requisição à raiz está disponível para tal requisição, ao passo que o conteúdo em cache fora de tal rota não está disponível para tal requisição. Porém em ROCs a situação muda. O sistema de cache apresenta um grande dinamismo por conta do seu modelo genérico de topologia de caches, omnipresença do cache e sua volatilidade. Se a informação sobre a volatilidade dos conteúdos em cache será anunciada ou pelo registro global dela ou pelo sistema de roteamento da rede, a escalabilidade do sistema será degradada severamente por conta do alto volume de mensagens de atualização. Além disso, uma alta dinamicidade dificulta a consistência do sistema.

3.5.3 Maior Granularidade do Conteúdo em Cache

A maioria das propostas em ROCs utilizam técnicas para dividir grandes arquivos em pequenos "pedaços"(*chunks*)de conteúdos auto-identifiáveis, realizando operações com esses "chunks"[6], [22], [23], [12]. Essa mudança na unidade do cache introduz novos problemas:

- **Mudança na Popularidade**
  Muitos estudos foram feitos sobre a popularidade do conteúdo a nivel de arquivo. Por exemplo, sabe-se que a frequência de acesso de objetos Web e P2P seguem o modelo de distribuições Zipf [24] e Mandelbrot-Zip [25] [26], respectivamente. Entretanto, a popularidade a nível de arquivo não pode ser simplesmente extendida para popularidade a nível de pedaço de conteúdo porque pedaços diferentes de um mesmo arquivo podem ter diferentes frequências de acesso. Por exemplo, os usuários somente decidirão se assistir a um vídeo inteiro ou não dependendo das primeiras partes de tal vídeo, resultando em frequências de acesso diferenciado para diferentes pedaços. Até agora, não há nem modelagem analítica nem um estudo experimental da popularidade de conteúdo a nível de pedaço de objeto.

- **Falha na suposição de Referência Independente**
  Os tradicionais sistemas de caches baseado em arquivos são tipicamente baseados na suposição de que as requisições seguem o tão famoso modelo de referência independente, isto é, a probabilidade que um dado objeto será requisitado é somente determinado pela sua popularidade, independentemente de requisições anteriores. Esta suposição, entretanto, invalida-se para cache a nível de pedaço de conteúdo. Requisições para diferentes pedaços do mesmo arquivo são geralmente correlacionados, como, por exemplo, em ordem sequencial.

- **Oportunidade para Uso mais Eficiente do Espaço de Cache**
  Armazenamento e troca de cache a nível de pedaço em vez de nível de arquivo com nomes únicos permite que se possa obter diferentes partes do mesmo arquivo de nós diferentes, o que acelera a taxa de obtenção de conteúdo e otimiza o espaço de utilização. Apesar do P2P também utilizar uma estratégia similar, a falta de uma nomeação de conteúdo unificada entre as diferentes aplicações P2P torna o reuso de cache entre aplicações impossível.

3.6 Técnicas para Otimização de Desempenho de Cache em ROCs

Uma vez que o cache muda de uma otimização via um mecanismo de *midleware* na arquitetura atual da Internet para ser um componente fundamental em ROCs e prover uma forma eficiente para obtenção de conteúdo, é, portanto, de vital importância otimizar o seu desempenho. Transparência, omnipresença e maior granularidade do cache não só

introduz novos desafios, mas também oferece novas oportunidades para o desenvolvimento de novas técnicas de armazenamento em cache. Otimização do desempenho de cache em ROCs pode ser implementado de muitas maneiras. A seguir, será discutido tais mecanismos detalhadamente.

3.6.1 Dimensionamento de Cache

Quando outras configurações são idênticas, uma maior capacidade de armazenamento de cache significa que mais conteúdos podem ser armazenados em cache, consequentemente, maior taxa de acertos no cache. Porém, maior tamanho de cache implica numa sobrecarga adicional de busca no cache. Uma vez que um nó em ROCs deve operar a uma taxa de operação a nível de hardware (*line rate*), o tamanho do cache que pode ser instalado no nó é, por conseguinte, limitado. Há dois problemas que precisam ainda ser resolvidos:

1. A primeira pergunta é que tamanho o cache deve possuir a fim de possuir uma melhora considerável no desempenho? Em sistemas de Web cache tradicionais, não há limitação imposta no tamanho de armazenamento. No entanto, a necessidade de operações a nível de hardware limita o tamanho do cache em ROCs. Enquanto isso, espera-se que em ROCs possa ser transmitido todo tipo de conteúdo. Assim, se o tamanho do cache é muito pequeno, é possível que essa possibilidade não possa ser realizada. Então, é preferível configurar o tamanho do cache de acordo com a disparidade do desempenho do roteador, a fim de atender o requisito de operar a nível de hardware.

   Além disso, é necessário ter um entendimento completo dos detalhes de cada tecnologia do hardware das memórias, tais como, velocidade de acesso, tamanho máximo permitido, custo unitário para armazenamento e consumo energético. Essa informação pode ser útil para a escolha entre as diferentes tecnologias e aplicar em ROCs. Por exemplo, em CCN, se um nó opera a uma taxa de 1 a 100Gbps, cada pacote de interesse utiliza 40 bytes, cada entrada na tabela PIT utiliza 56 bits, cada requisição é armazenada na PIT com um tempo de vida de 80 ms, entaõ a tabela PIT deve ter entre 14M bits a 1.4G bits. Então, escolheria-se como opção RLDRAM ao invés de SRAM, já que a última não tem espaço suficiente para atender a esse requisito.

2. A segunda questão e como o recurso de armazenamento deve ser alocado através de diferentes nós para que o desempenho do sistema de cache possa ser otimizado, dado que o recurso total de armazenamento é fixo? De fato, isto é um problema de dimensionamento da rede que possui um dado custo. A alocação do espaço de cache precisa considerar tanto a topologia da rede quanto as demandas do tráfego. Sabe-se que alocar tamanho de cache baseado na centralidade do nó resulta numa pequena melhora no desempenho, isto é, a capacidade de armazenamento em cache alocado para um nó é proporcional ao seu grau [27].

Em um outro estudo, comprovou-se que alocando mais espaço de cache para nós na borda da rede do que em nós no núcleo pode melhorar o desempenho [20]. Esses estudos juntos mostram que quando a topologia da rede é considerada , o esquema de alocação de recursos não deve simplesmente ser baseado na centralidade topológica e estática do nó, mas deve ser baseado também na distância do nó com cache e os usuários. Da mesma forma, uma vez que a simples alocação de cache não pode atingir uma melhora significativa no desempenho, é necessário integrar o espaço de alocação de conteúdo com a sua localização e com políticas de consultas.

3.6.2 Mecanismos de Compartilhamento de Cache

Uma consequência direta da transparência de cache é que diferentes tipos de tráfegos/aplicações tem que competir pelo espaço de cache em um dado nó. Eles variam nas características de tráfego e objetivos de otimização de cache. Como compartilhar recursos limitados entre diferentes tráfegos eficientemente, e ao mesmo tempo prover serviços diferenciados é uma tarefa que precisa ser resolvida urgentemente em ROCs.

3.6.2.1 Serviço Diferenciado de Cache

A importância em prover um serviço diferenciado de serviços de cache é enfatizado através de um estudo[39], que pode ser resumido através de três pontos de vista:

- Do ponto de vista do usuário, o cache é mais importante e útil para usuários com rápida velocidade de acesso. Se o "gargalo"está localizado no núcleo da rede, então o cache pode reduzir o atraso de transmissão substancialmente, caso contrário, se o "gargalo"da rede está na borda, como, por exemplo, num link de acesso, então a utilidade de cache será grandemente reduzida. Portanto, os caches devem atender aos usuários com maior velocidade de acesso dando-lhes maior prioridade. Para fazer isso, depende-se da habilidade da rede em distinguir entre diferentes tipos de tecnologias de acesso, como, por exemplo, Linguagem de Marcação para Redes sem Fio (*Wireless Markup Language - WML)* desenvolvida para redes de acesso sem fio.
- Do ponto de vista do conteúdo, é possível melhorar o desempenho percebido pelo usuário ao armazenar frequentemente em cache o conteúdo "mais importante". Por exemplo, nota-se que diferentes classes de conteúdo Web contribuem diferentemente para a percepção dos usuários sobre o desempenho da rede. O desempenho percebido pelos usuários depende mais do tempo de download de páginas HTML do que o tempo de download dos objetos embutidos. Portanto, o sistema de cache deveria tratar página HTML com uma prioridade maior a fim de melhorar a qualidade de experiência (QoE) dos usuários. Isso necessita que o sistema seja capaz de distinguir entre as diferentes classes de conteúdos.
- Do ponto de vista do provedor de conteúdo, IPS podem possuir diferentes contratos com diferentes provedores de conteúdo para prover alguns sítios com melhores

serviços a um preço negociável. Isto requer que o sistema de cache diferencie entre diferentes provedores de conteúdo, o que, por exemplo, pode ser realizado pela URL or pelo principal identificador do conteúdo.

3.6.2.2 Técnicas para Compartilhamento de Cache

A fim de promover um serviço diferenciado de cache, é necessário possuir um suporte à técnicas de compartilhamento de cache. A principal escolha em desenvolver um mecanismo de compartilhamento é se o compartilhamento é fixo ou dinâmico, como discutido a seguir:

- **Compartilhamento de Cache baseado em particionamento fixo**
  Isto está relacionado em particionar o espaço de cache em partes fixas para que cada classe de aplicação obtenha sua própria cota, que não poderá ser usada por outra classe de tráfego mesmo se ela estiver sendo subutilizada. Desta forma, o desempenho de cada classe de tráfego não será interferido por outras classes. Há, no entanto, duas perguntas que ficam em aberto: Como particionar o espaço de cache através de diferentes classes de tráfego? O nó deve seguir um padrão de particionamento comum ou poderia ter diferentes taxas de partições para diferentes classes de tráfegos?

  Para o primeiro questionamento, foi proposto proposto uma abordagem para ajustar adaptativamente a taxa de partição baseado na teoria do controle de resposta dinâmica [28]. Para solucionar o segundo questionamento, observa-se que para se atingir a mesma taxa de acertos, uma aplicação de VoD necessita de um espaço de cache muito menor do que aplicações de compartilhamento de arquivos gerados por usuários. Sendo assim, seria importante dedicar os caches de mais baixo nível e mais próximos dos usuários de aplicações exclusivamente de VoD. Esta abordagem, de alguma forma, sugere diferentes taxas de particionamento para diferentes nós de acordo com a propriedade do nó.

  Embora o compartilhamento baseado em partição fixa seja de fácil implementação, ele é, entretanto, difícil de se obter um compartilhamento eficiente e dinâmico entre as diferentes classes de aplicações. Geralmente é quando os sistemas de cache definem o tempo de vida para os objetos armazenados. Uma vez que o TTL está associado com uma aplicação específica, o volume efetivo de dados para tal aplicação é próximo do produto da sua taxa de chegada e do TTL. Com resultado disso, se o particionamento fixo for utilizado, algumas aplicações não terão exaurido seu espaço de cache, enquanto que outras aplicações podem em algum momento ficar sem espaço disponível para cache.

- **Compartilhamento de Cache Dinâmico**
  O compartilhamento dinâmico permite que uma classe de tráfego use o espaço de cache se nenhuma outra classe de tráfego naquele momento necessite dele. Duas

políticas de compartilhamento dinâmico podem ser usadas: Compartilhamento baseado em prioridade e compartilhamento baseado em pesos justos. O compartilhamento baseado em prioridades dá a certas aplicações um serviço de maior prioridade.

Quando um objeto chega em um nó e precisar ser armazenado em cache mas não há espaço disponível para isso, o nó remove repetidamente do seu cache aqueles objetos que pertencem a aplicações com prioridade menor ou igual até que haja espaço suficiente para tal objeto. Porém, os esquemas puramente baseados em prioridades podem degradar severamente o desempenho das aplicações de baixa prioridade se uma grande quantidade de tráfego chegar a uma taxa constante. Da mesma forma como no particionamento fixo, o compartilhamento com pesos justos também pretende compartilhar os recursos de cache entre as diferentes classes de aplicação de acordo com pesos pré-definidos, mas diferente do particionamento fixo, o modelo com pesos justos permite que uma classe de tráfego utilize a cota de outra classe de tráfego caso o espaço dela não esteja sendo usado naquele momento, desta forma, melhorando sua utilização.

3.6.3 Política de Decisão de Cache

A política de decisão de cache determina quais objetos podem ser alocados em quais nós. Esta áreas é, possivelmente, uma das áreas mais aquecidas com relação à pesquisas dentro de ROCs. No tradicional esquema de Web cache e cache CDN, é possível chegar à alocação ótima do objeto tendo um conhecimento prévio da topologia e da demanda de tráfego. Essa abordagem é comumente conhecida como decisão de cache explicitamente coordenada. Mas em ROCs, os nós com cache não são mais fixos, as classes de tráfego são diversas e as operações nos caches precisam ser a nível de hardware. Esses fatores tornaram algoritmos de coordenação de cache explícito inadequados para ROCs devido à sua alta complexidade e sobrecarga na comunicação. ROC precisa de políticas de decisão de caches mais simplificadas, porém eficazes.

Um exemplo de política bastante simples é a LCE (*Leave a Copy Everywhere*), a qual copia o objeto em cada nó ao longo do caminho de requisição. Tal política é o modelo padrão adotado na maioria de arquiteturas de ROCs como, por exemplo, CCN. Essa abordagem, porém, causa uma alta redundância de cache, isto é, o mesmo objeto é copiado em múltiplos nós desnecessariamente, reduzindo a diversidade dos conteúdos armazenados na rede. Com o objetivo de melhorar o desempenho da utilidade do sistema como um todo, é necessário que os conteúdos populares seja rapidamente "empurrados"para a borda da rede a fim de que o tempo de*donwload* dos usuários possa ser reduzido e o uso dos recursos de rede possa ser otimizado e melhorar a diversidade de conteúdos em cache, especialmente dentro do mesmo ISP, para que as requisições dos usuários possam ser atendidas dentro do mesmo ISP, resultando numa grande redução de tráfego inter-domínio [29], [30].

Para reduzir a redundância de cache e melhorar a diversidade de conteúdo é necessário uma simples e eficaz coordenação entre os nós. Esses esquemas de coordenação podem ser classificados de várias formas. De acordo com o grau de autonomia no processo

de implementação da politica de decisão de cache, os esquemas de decisão de caches atuais podem ser classificados em *explícito* ou *implícito*. Ainda, dependendo do ponto de decisão, eles podem ser classificados em *centralizado* e *descentralizado*.

3.6.3.1 Decisão de Cache com Coordenação Explícita

Neste tipo de esquema, o padrão de acesso, a topologia da rede e o estado dos outros nós com caches são usados como entrada para calcular onde o objeto será alocado na rede. Essas informações são obtidas tanto via comunicação on-line quanto off-line. Os nós que participam na coordenação explícita podem variar de escopo. Alguams abordagens podem ser classificadas em três categorias: Coordenação global, coordenação do caminho e coordenação de vizinhança.

- **Coordenação global**
  Significa dizer que a coordenação de cache envolve todos os nós com cache na rede, o que geralmente é usado em CND. Numa CDN, o conjunto de nós com caches, a distância entre eles na rede e a frequência de acesso em cada um deles é conhecida à priori e, então, o provedor calcular a localização ótima do objeto a fim de minimizar o custo de acesso do usuário, ou de maneira centralizada ou descentralizada.

- **Coordenação de caminho**
  Involve os nós com cache ao longo do caminho entre um nó solicitante e um nó que possui o conteúdo solicitado (*caching node*). Os objetos só podem ser copiados ao longo desse caminho. Nesta abordagem a informação necessária para a coordenação no pacote de requisição do conteúdo. Tal informação tipicamente inclui o estado de cada nó e a frequência de acesso do objeto em cada um deles. Quando a requisição chega num nó que possui o conteúdo, tal nó extrai todas as informações de estado necessárias sobre o caminho, e calcula a localização ótima dentro do caminho, bem como os objetos que precisam ser trocados de posição.

- **Coordenação de vizinhança**
  Este modelo de coordenação leva em conta a vizinhança de um nó. A forma como a vizinhança é definida vaira de esquema para esquema. Por exemplo, pode-se incluir os vizinhos diretos dos nós ou vizinhos localizados a dois saltos de distância deles. A proposta *Cooperative In-Network Caching (CINC)* [30] pertence a esta categoria, que é baseada numa função *hash*. A Figura 5 mostra a operação do CINC. Quando um *chunk* chegam em um nó, este usa uma função *hash* para determinar qual dos seus vizinhos(incluindo ele mesmo) armazenará em cache o *chunk*. Desta forma, evita-se que o objeto seja duplicado dentro da vizinhança de um nó. Quando uma requisição chega, a função *hash* é também utilizada para consultar o nó responsável se ele tem ou não o conteúdo em cache.

Coordenação global, Coordenação de caminho e Coordenação de vizinhança ajudam a reduzir a redundância de objetos. Porém, todos esses esquemas de coordenação

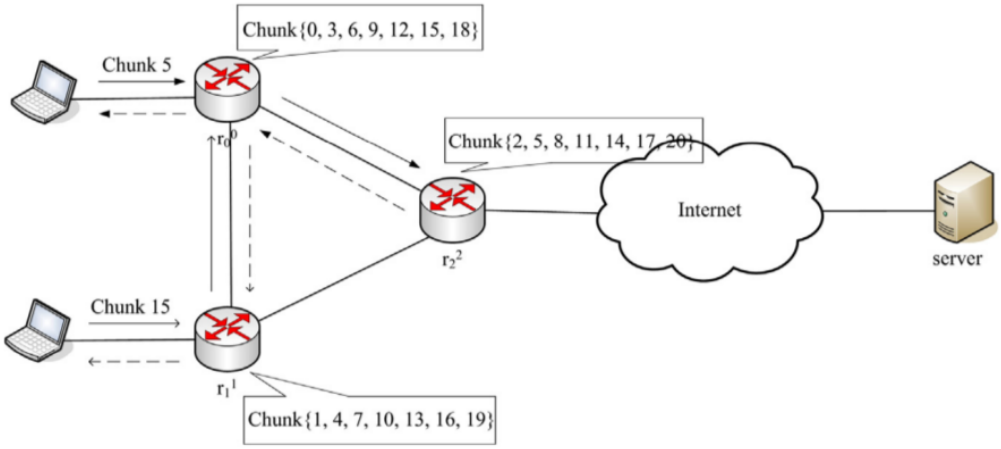

Figura 5: Operações do CINC

precisam de informações de estado dos nós e a frequência de acesso dos usuários, além de trazer uma grande troca de informações e aumento na sobrecarga de processamento para tomar a decisão final de cache, o que muita das vezes é muito custoso e complexo dentro de ROCs.

3.6.4 Coordenação Implícita de Cache

Na coordenação implícita, cada nó não precisa saber de informações de estado dos outros nós ou precisa apenas trocar bem pouca informação com outros nós, antes de tomar a decisão de armazenar o conteúdo em cache ou não.

Cache hierárquico é um exemplo de coordenação implícita. A topologia especial de cache faz com que o nó de borda identifique requisições populares e, desta forma, empurrando conteúdo com alta popularidade para a borda da rede. Por conta do efeito de filtragem, caches de altos níveis só armazenarão objetos com popularidade baixa ou moderada, o que em certa medida necessita de uma certa coordenação. Entretanto, em ROCs, a topologia de rede evolui pra um grafo arbitrário, então é difícil definir a relação de hierarquia entre os diferentes nós. Além disso, requisições externas podem ocorrer em qualquer nó. Então, tal esquema não é mais eficaz em ROCs.

Alguns esquemas de coordenação implícita foram propostos para aliviar a alta redundância de cache introduzida pelo esquema LCE. A Figura 6 ilustra alguns esquemas e, logo após, segue uma explicação detalhada de cada um deles.

- *Leave a Copy Down (LCD)* [31], [32] : Quando um acerto de cache ocorre em um dado nó, este esquema armazena uma cópia do conteúdo apenas no nó imediatamente um nível abaixo no caminho de requisição, evitando que haja um grande número de cópias do mesmo objeto espalhado pela rede. O esquema LCD necessita que muitas requisições devem ser feitas para que o conteúdo seja empurrado para a borda da red, o que implicitamente considera as frequências de acesso dos objetos.

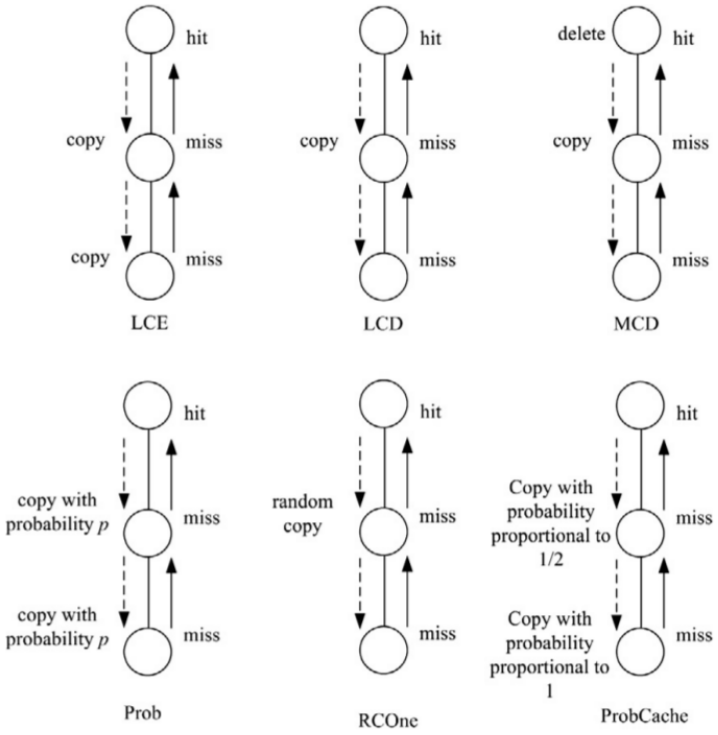

Figura 6: Operação de diferentes tipos de esquemas de decisão de cache

- *Move Copy Down (MCD)*[31], [32] : Quando um acerto de cache ocorre, o objeto é opiado para o nó imediatamente um nível abaixo no caminho de requisição e apagado do nó em que foi encontrado. Comparado com o LCD, o MCD reduz a redundância de cache de maneira mais agressiva.

- *Copy with Probability (Prob)* [31]: O objeto solicitado é copiado com uma dada probabilidade p em cada nó ao longo do caminho reverso de requisição. Esta abordagem pode ser entendida como uma generalização do LCE. Quando $p = 1$, o seu funcionamento torna-se idêntico ao do LCE.

- *Randomly Copy One (RCOne)* [33]: Este esquema copia o objeto solicitado para um dado nó escolhido aleatoriamente ao longo do caminho reverso de requisição. Para uma estrutura hierárquica de caches com n-níveis, esta proposta tem o valor de probabilidade igual a $p = 1/l$.

- *Probabilistic Cache (ProbCache)* [20]: Neste esquema, o objeto solicitado é copiado em cada nó com um dado valor de probabilidade. Mas, para cada nó, a probabilidade varia e é inversamente proporcional à distância entre o nó solicitante e o nó com acerto de cache. Portanto, se o nó com acerto está próximo do nó solicitante, o objeto será copiado com uma maior probabilidade. Por outro lado, se o nó com acerto de cache estiver longe do solicitante, o objeto tem uma pequena chance de ser copiado.

CAPÍTULO 4

Trabalhos Relacionados

Neste capítulo são abordados os trabalhos relacionados, bem como explicados como cada um usa sua respectiva métrica para escolher os nós alvos que armazenarão o conteúdo. Além disso, são citadas algumas vantagens e as principais desvantagens de cada uma.

4.0.5 Cache Capacity-Aware CCN

A proposta Cache Capacity-Aware CCN baseia-se no conceito de capacidade de cache [34]. 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 e 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 satisfatórios 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.

4.0.6 Probabilistic Cache - PROBCACHE

Uma estratégia de decisão probabilística chamada de ProbCache usa como métrica a capacidade de armazenamento do caminho [20]. 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 comparado com o valor obtido para cada nó naquele caminho. Caso o valor aleatório seja menor que o valor obtido pela proposta, o conteúdo é armazenado naquele nó. Uma vantagem da proposta é que ela armazena o conteúdo mais rapidamente para a borda da rede. No entanto, ele não considera a capacidade de armazenamentos dos nós individualmente e isso pode causar dois problemas. O primeiro deles é que a estratégia pode escolher um nó alvo com muito pouco cache disponível e o outro é que a política de descarte de conteúdo pode apagar o conteúdo antes antes do esperado e a requisição precisar ser enviar para nós mais distantes, aumentando o tempo de download.

4.0.7 Cache Less for More - CL4M

A proposta denominada de Cache Less for More - CL4M - é baseada no conceito de centralidade dos nós [35]. Tal métrica mede a importância dos nós na rede. Ela informa que se um nó está na maioria 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. Entretanto, calcular a centralidade de todos os nós para grandes topologias pode ser muito custoso em termos de processamento e memória.

4.0.8 Armazenamento de baseado em Ranqueamento de Caches - RankCache

O trabalho denominado de RankCache propõe o rankeamento dos caches de todos os nós num caminho de requisição de conteúdo [36]. Para isso, calcula-se o número de acertos de conteúdo obtido por cada nó a partir de uma fórmula por eles 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. Todavia, os autores não informaram o custo computacional para calcular o ranque dos nós.

4.0.9 Conclusões

Deste modo, é proposta uma estratégia de decisão de cache baseada em três métricas, as quais são: centralidade de intermediação dos nós, capacidade de cache disponível e a taxa de acertos de conteúdo do nó. Como pode ser observado na Tabela 1, duas estratégias utilizam como métrica a capacidade de cache (Cache Capac. Aware e

ProbCache). A primeira considera a capacidade dos roteadores individualmente e a segunda do caminho de requisição. CL4M utiliza a centralidade do nó. RankCache é a única que não como métrica base nenhuma das listadas nos trabalhos relacionados, pois utiliza um esquema de ranqueamento dos nós. A MCCD é a única que utiliza as mesma métricas de PROBCACHE e Cache Capc. Aware, além da métrica taxa de acertos de conteúdos dos nós. Quando combinadas, essas informações podem ainda melhorar o desempenho de nossa estratégia de decisão de cache. Diferentemente das propostas baseadas em apenas uma métrica, nossa proposta consegue distribuir o conteúdo pela rede melhor, já que não o concentra em nós que casam apenas com um tipo de métrica.

Tabela 1: Trabalhos Relacionados

Estratégia	Centralidade	Capac. de Cache	Taxa de Acertos
Cache Capacity Aware		x	
ProbCache		x	
CL4M	x		
RankCache			
MCCD	x	x	x

CAPÍTULO 5

Proposta baseada em Múltiplas Métricas

Este capítulo aborda sobre a proposta baseada em múltiplas métricas (*Multi-Criteria Caching Desision Scheme - MCCD*), explica-se detalhadamente cada métrica, o algoritmo base da proposta e sua aplicação através de um cenário simples para melhor entendimento da proposta. Primeiramente, são listados os trabalho relacionados à estratégias de decisão de cache em ROCs.

5.1 Métricas utilizadas pela 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, maior tamanho de *cache* disponível e maior número de acertos no caminho da requisição de conteúdo.

1. *Centralidade do Nó:* A centralidade de um nó 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 para calcular 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.

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:

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

5.2 Funcionamento da MCCD

Nesta seção detalha-se como a MCCD trabalha com as três métricas mencionadas anteriormente e como elas são utilizadas pelo algoritmo base da proposta. Assume-se que todas as requisições de conteúdos são enviadas para o servidor de conteúdo através do caminho mais curto. Ainda, os roteadores de conteúdos entre o usuário e o servidor dispõem os seus respectivos valores de centralidade (Centr), capacidade de cache disponível (CCD) e taxa de acertos de conteúdo (TAC).

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 a métrica considerada. Isto é, dentro do caminho de requisição, o nó com maior valor de centralidade é selecionado numa lista, o nó com maior capacidade de 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 de centralidade*. 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 capacidade de cache 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 de acertos*. O algoritmo base da MCCD está dividido em duas partes, as quais são: `Obter_Conteudo` e `Receber_Conteudo`. Após a explicação do pseudocódigo, mostramos um exemplo prático do seu funcionamento.

O método Inicializar_Variaveis juntamente com as três variáveis max_cent_no, max_cache_disp_no, max_tax_acertos_no guardarão, respectivamente, os nós com maior

Algorithm 1 MCCD Obter_Conteudo

```
1: procedure REQUISITAR_CONTEÚDO(c)
   Inicializa_Variaveis (max_cent_no, max_cache_disp_no, max_tax_acertos_no).
2:     for (u, v) in Caminho_Requisicao do
3:         Enviar_Requisicao(c)
4:         if c in cache(v) then
5:             v ← cache_temp
6:             Enviar_Conteudo(c)
7:         else
8:             Atualiza(max_cent_no)
9:             Atualiza(max_cache_disp_no)
10:            Atualiza(max_tax_acertos_no)
11:        end if
12:        Enviar_Requisicao(c)
13:    end for
14: end procedure
```

centralidade, com maior capacidade de *cache* disponível e maior taxas de acertos. Concomitantemente, as três listas são criadas e cada uma dessas variáveis receberá o nó com o maior valor de acordo com a métrica comparada. Uma vez encontrado o conteúdo, ele volta pelo caminho reverso e, a cada salto, o nó corrente é verificado se casa com uma das variáveis. Em caso afirmativo, uma cópia é armazenada nele e estas comparações são feitas com o nós restantes até que o conteúdo seja entregue ao usuário.

Algorithm 2 Receber_Conteudo

```
1: procedure ENVIAR_CONTEUDO(c)
2:     for (u, v) in Caminho_Reverso do
3:         Enviar_Conteudo(c)
4:         if v == max_cent_no or v == max_cache_disp_no or v == max_tax_acertos_no
   then
5:             Armazena(c , v)
6:         end if
7:         Enviar_Conteudo(c)
8:     end for
9: end procedure
```

Para melhor ilustrar o funcionamento do nosso algoritmo de decisão de *cache* descrito acima, considere o cenário na figura 7. Assumimos que todos os roteadores (R1 - R11) já possuem conteúdos armazenados em *cache*, Servidor é o servidor permanente de conteúdo e os três usuários (1,2 e 3) estejam solicitando conteúdo da rede.

Num dado momento, o usuário 2 envia uma solicitação R de um conteúdo c para o roteador R1. Então, é verificado se R1 possui c em *cache*. Se sim, envia-o para o usuário

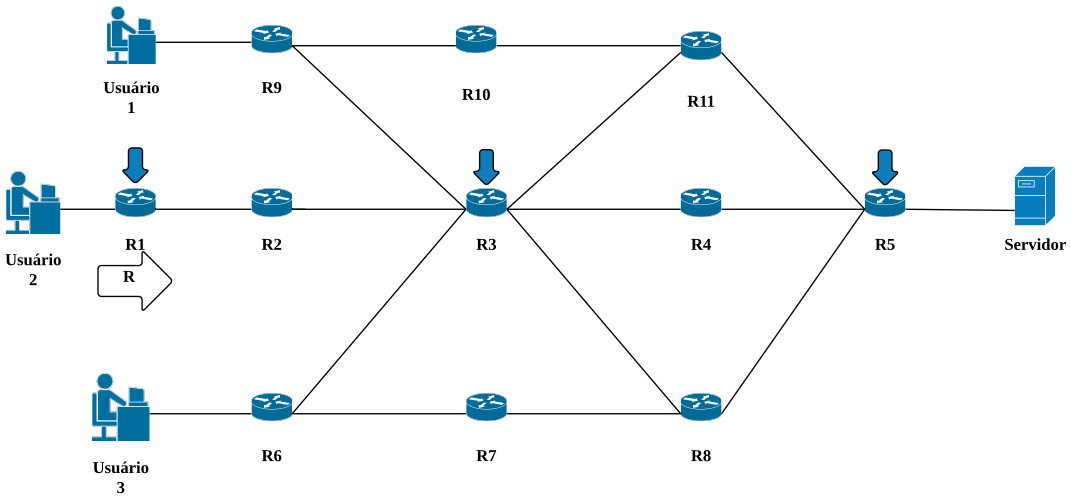

Figura 7: Funcionamento da MCCD.

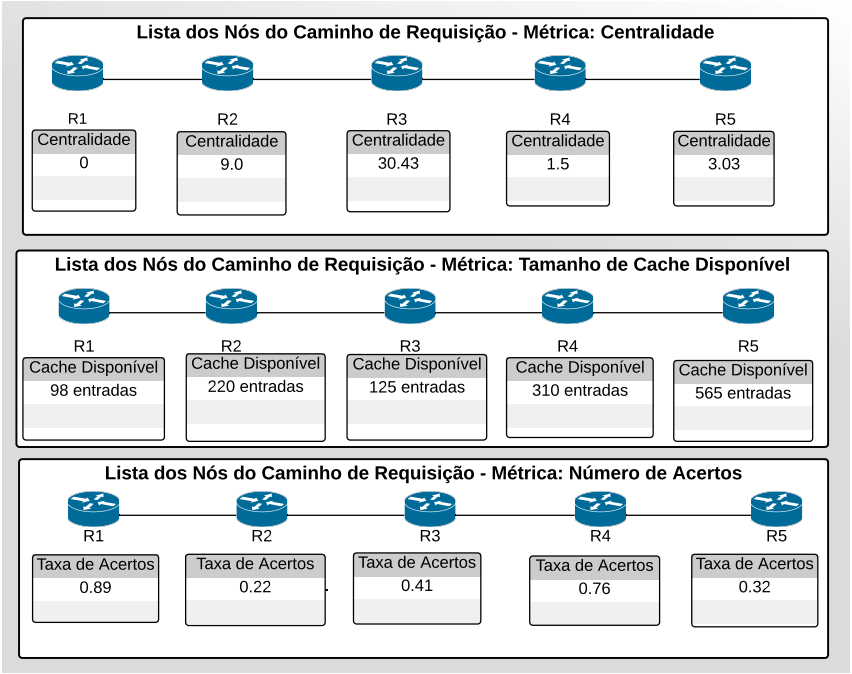

Figura 8: Obtenção das Métricas.

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 partir da figura 8, foram encontrados os roteadores R1, R3 e R5 sendo os nós com *maior taxa de acertos*, *maior centralidade* e *maior capacidade 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.

CAPÍTULO 6

Avaliação da Proposta

6.1 Definição dos Parâmetros de Simulação

A proposta MCCD foi implementada e testada utilizando o simulador Icarus 0.4.0 [9], um simulador de estratégias de decisão de *cache* para redes ICN. Este simulador possui algumas estratégias implementadas e escolhemos alguns deles para comparar seu desempenho e comparar com a nossa proposta. A Tabela 2 resume os parâmetros de avaliação e topologias utilizadas nos testes.

Tabela 2: Configuração dos parâmetros da simulação

Topologias avaliadas	Árvore Binária, TISCALI e GEANT
# total de conteúdos no repositório	50.000
Taxa de requisição	100/s
# de requisições para popular caches dos roteadores	10.000
# de requisições avaliadas	50.000
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	1% e 10%
Estratégias avaliados	MCCD, PROBCACHE, LCE e CL4M
Métricas avaliadas	Taxa de acertos e latência

As topologias utilizadas foram: Árvore binária com profundidade D = 7. O servidor de conteúdo estava na raiz, os usuários são os nós folhas e os nós restantes são os roteadores de conteúdo. TISCALI é um provedor pan-europeu; GEANT é o um *backbone* acadêmico europeu. O número de clientes, servidores e roteadores de conteúdo

utilizados foram os mesmos utilizados por padrão pelos simulador; O número total de conteúdo no repositório é de 50.000 conteúdos; a taxa de requisição de conteúdo é de 100 req/s e segue um processo de Poisson; o *cache* dos roteadores foi populado com 10.000 conteúdos na fase de inicialização e tais conteúdos foram distribuídos uniformemente entre os roteadores; Foram avaliadas 50.000 requisições de conteúdo; A política de descarte de *cache* usada foi a LRU (*Least Recently Used*); A popularidade dos conteúdos segue o modelo Zipf de distribuição de popularidade de conteúdo: Quanto menor o valor do parâmetro α, menor o número de conteúdos populares na rede; Quanto maior o valor de α, maior será a quantidade de conteúdos populares; Como tamanho de *cache* dos roteadores foram configurados dois valores: 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.

6.2 Análise dos Resultados

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

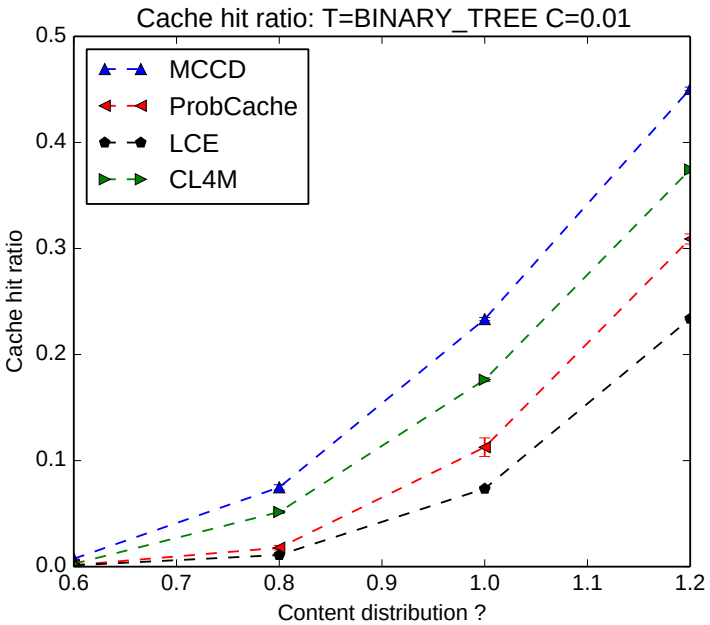

Figura 9: Taxa de Acertos - Árvore Binária - Tamanho do Cache = 1%.

Nota-se que a estratégia MCCD obteve uma taxa de acertos bem maior que das outras propostas, conforme mostrado nas figuras 9 e 10 para a topologia de árvore binária. Para um tamanho de cache de 1% e um valor de $\alpha = 0.6$, a taxa de acertos da MCCD

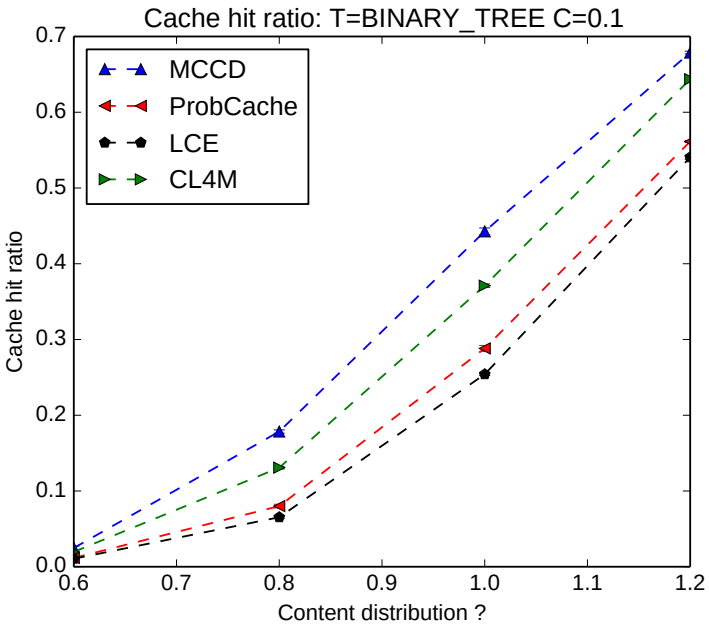

Figura 10: Taxa de Acertos - Árvore Binária - Tamanho do Cache = 10%.

maior e, aumentando-se o valor de α, MCCD obteve uma taxa de acertos de 45% com $\alpha = 1,2$. Este valor de taxa da MCCD foi 118% melhor que a segunda melhor proposta, a CL4M. PROBCACHE foi apenas um pouco melhor do que LCE e aquela não obteve um bom resultado nos dois testes com árvore binária. Considerando um tamanho de cache de 10%, a MCCD também teve desempenho melhor que as demais. Ela atingiu uma taxa de acerto de 70% para $\alpha = 1,2$. As demais propostas mantiveram as mesmas posições do teste anterior, porém com curvas um pouco mais próximas da MCCD. PROBCACHE novamente obteve um baixo desempenho, tendo curva bem próxima da LCE. O diferencial de nossa proposta é que quando uma métrica não tem um bom desempenho, a MCCd se sobressai às demais pelo fato dela trabalhar com mais métricas e poder alocar o conteúdo em outros nós estratégicos. Além disso, pelo fato de MCCD e CL4M trabalharem com as mesmas métricas (centralidade do nó), MCCD obteve uma maior taxa pois o conteúdo também será armazenado em outros dois nós, distribuindo melhor o conteúdo pela rede.

Com relação à topologia TISCALI na Figura 11, 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.Quando aumenta-se tamanho do cache dos nós para 10% como mostra a Figura 12, 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 $\alpha = 0.6 - 1.0$. Acredita-se que pelo fato da MCCD distribuir melhor o conteúdo ao longo do caminho de requisição, isto permite que outros usuários localizado em diferentes

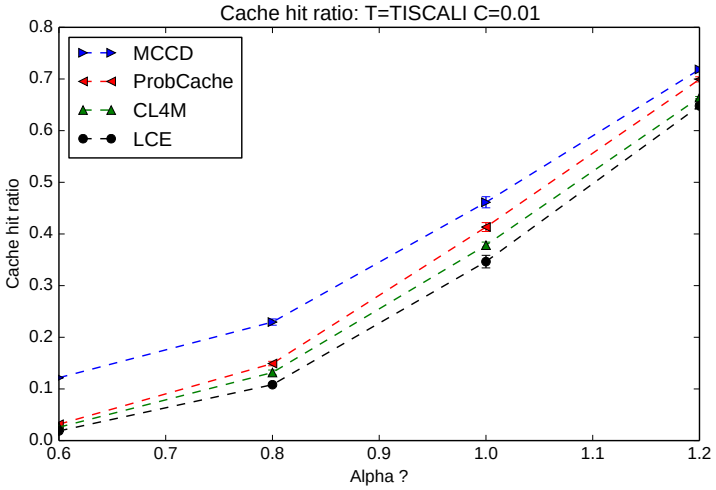

Figura 11: Taxa de Acertos - TISCALI - Tamanho do Cache = 1%.

parte os da rede conseguem recuperar o conteúdo, fazendo com que desempenho da rede aumente em termos de taxa de acertos. Tendo mais nós estratégicos para armazenar o conteúdo em grande redes permite que ele seja alcançável a partir de qualquer ponto dela.

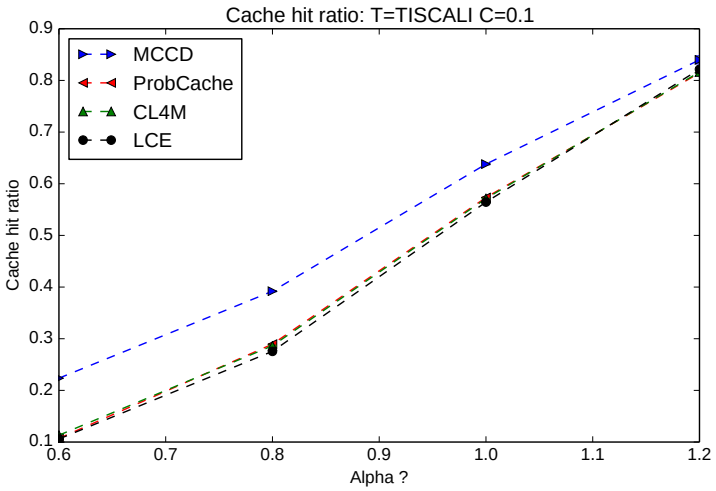

Figura 12: Taxa de Acertos - TISCALI - Tamanho do Cache = 10%.

Com relação aos resultados da topologia GEANT mostrados na Figura 13 e 14, pode-se verificar que a MCCD obteve também uma taxa de acertos muito maior que as demais. Para conteúdos menos populares, $\alpha = 0.6$ e tamanho de cache igual a 1%, MCCD obteve uma taxa de acertos 633% maior que o segundo melhor resultado, que foi obtido por PROBCACHE. Par conteúdos mais populares, $\alpha = 1.2$, a MCCD obteve uma taxa de 65%. Praticamente empatados com o segundo melhor resultado ficaram CL4M e PROBCACHE e LCE obteve o pior desempenho.

Para um tamanho de cache igual a 10% e para conteúdos menos populares, a MCCD foi 300% melhor que as outras, para toda a faixa de valores de α. Para conteúdos mais populares, a MCCD obteve uma taxa de 78% de acertos, CL4M foi um pouco melhor que PROBCACHE, que teve um desempenho quase similar ao da LCE. Observa-se que

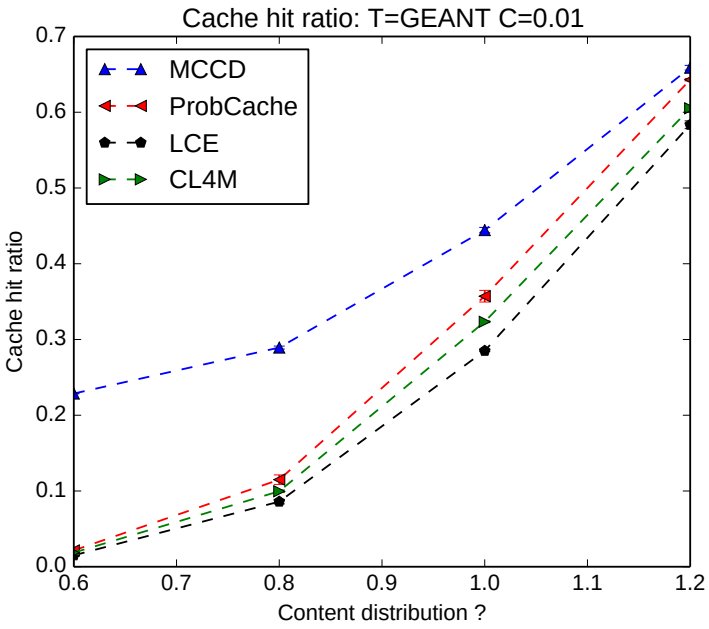

Figura 13: Taxa de Acertos - GEANT - Tamanho do Cache = 1%.

para todos os cenários avaliados, a proposta PROBCACHE teve um desempenho melhor desempenho para menores tamanho de caches (1%), mas tal desempenho foi inferior para um tamanho de cache maior (10%). Novamente, a MCCD destaca-se das demais propostas avaliadas pela melhor distribuição de conteúdo pela rede, mesmo quando a disponibilidade de espaço para cache é pouca, bem como para conteúdo menos populares.

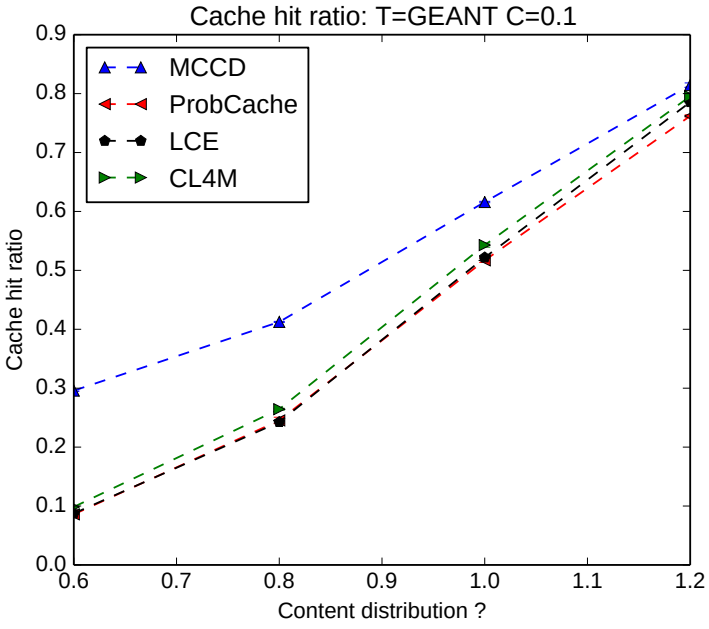

Figura 14: Taxa de Acertos - GEANT - Tamanho do Cache = 10%.

Agora, são discutidos os resultados referentes à latência. Considerando a Figura 15 para um tamanho de cache de 1% na árvore binária, a MCCD obteve o segundo melhor tempo de download, principalmente para valores de $\alpha > 0.8$. Para valores entre 0.6 - 0.8,

PROBCACHE teve praticamente o mesmo tempo que a LCE. Para valores maiores de α, PROBCACHE teve um tempo melhor que LCE.

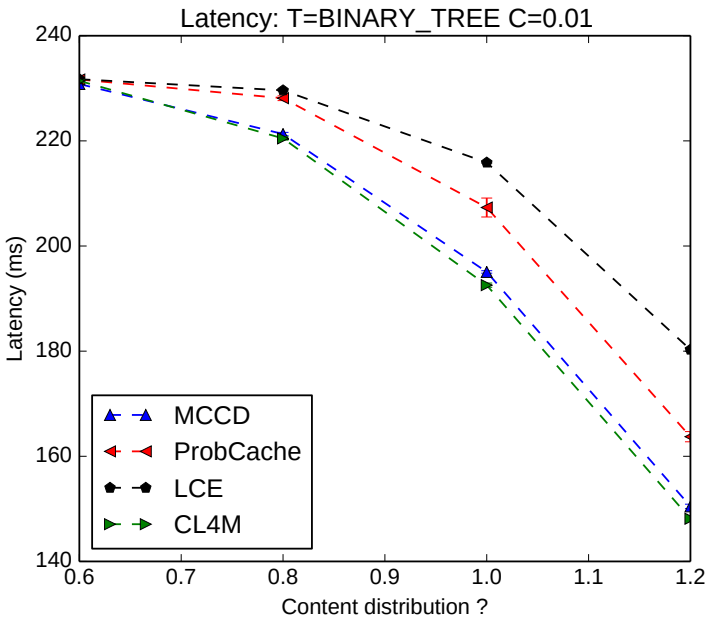

Figura 15: Latência - Árvore Binária - Tamanho do Cache = 1%.

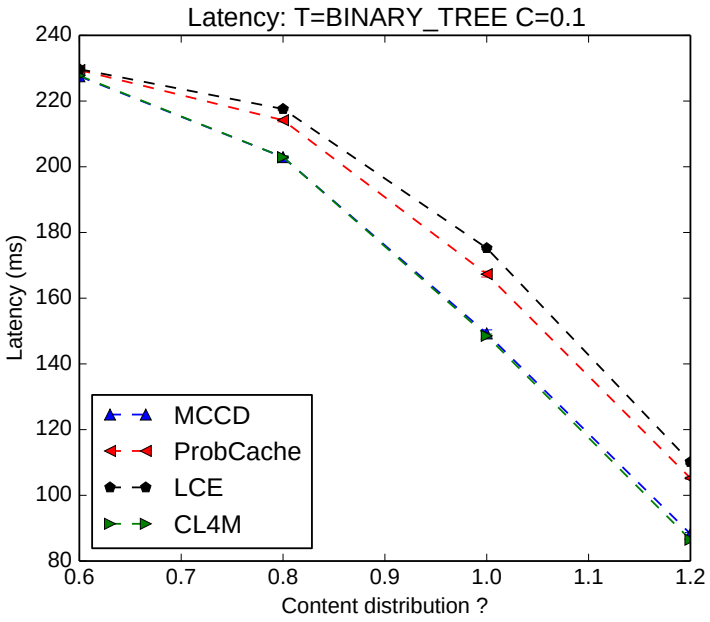

Figura 16: Latência - Árvore Binária - Tamanho do Cache = 10%.

Com 10% de tamanho de cache, a MCCD obteve praticamente o memo tempo de download que a CL4M para todo valor de α, como mostra a Figura 16. Uma possível explicação para isso de desempenhos semelhantes entre MCCD e CL4M é que ambas usam como métrica a centralidade do nó. Desta forma, isso faz com que tal métrica se sobressaia às duas outras usadas pela MCCD. Outra possível razão é que um nó alvo pode casar com mais de uma métrica, por exemplo, possuir o maior valor de centralidade e maior taxa de

acertos naquele caminho de requisição. Isso resulta que o conteúdo pode ser armazenado em tais nós, fazendo com que as duas propostas tenham tempo bem semelhantes.

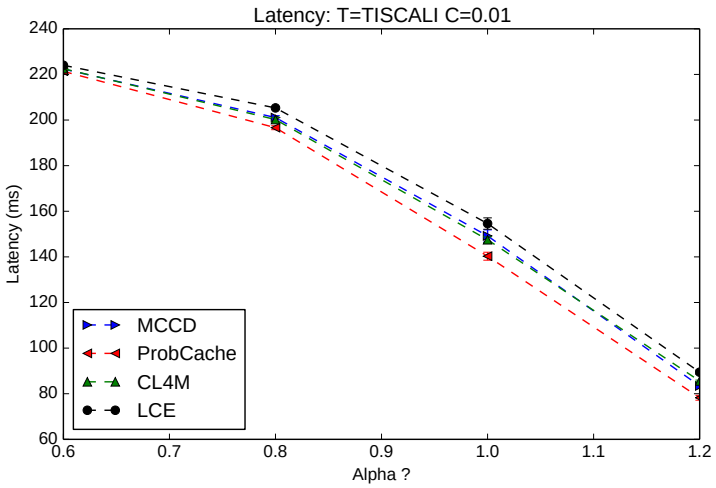

Figura 17: Latência - TISCALI - Tamanho do Cache = 1%.

As figuras 17 e 18 mostram o desempenho para a topologia TISCALI. Para um tamanho de cache igual a 1% e valores de $\alpha = 0.6 - 0.8$, MCCD, PROBCACHE e CL4M tiveram o mesmo tempo de download. Para valores maiores de α, MCCD teve um tempo um pouco menor do que as demais, seguida por PROBCACHE, CL4M e LCE. Já para um tamanho de cache igual a 10% , a MCCD obteve um tempo menor para todo valor de α, seguido novamente por PROBCACHE, CL4M e LCE. Neste teste, supõe-se que o tempo de download da MCCD foi menor devido à características topológicas como, por exemplo, número maior de nós com alto valor de centralidade, o que permitiu que o conteúdo ficasse armazenado em nós mais próximo ao usuário. Também, acredita-se que os nós alvos estavam no caminho mais curtos de outros usuários que solicitaram o mesmo conteúdo. Então, a requisição não precisou chegar ao servidor de conteúdo ou a um nó mais distante que possui o conteúdo solicitado.

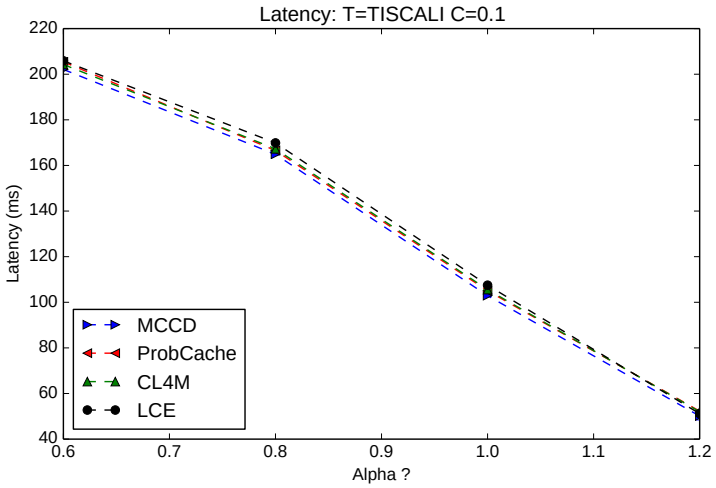

Figura 18: Latência - TISCALI - Tamanho do Cache = 10%.

As Figuras 19 e 20 mostram o desempenho para a topologia GEANT. Como observa-se para um tamanho de cache igual a 1%, a MCCD não teve um desempenho satisfatório. Seu desempenho foi apenas o terceiro melhor, próximo do tempo obtido por CL4M. Acredita-se que características topológicas possam ter influenciado nesse resultado como, por exemplo, a distância entre os nós alvos e os usuários, bem como a baixa disponibilidade de cache, haja visto que foi ofertado apenas 1% de cache para os roteadores. Além disso, PROBCACHE obteve um desempenho melhor por ter alocado o conteúdo em nós mais próximos da borda mais rapidamente quando comparado com as demais, o que reduziu o seu tempo de download consideravelmente. Porém, quando o tamanho de cache é aumentado para 10%, a MCCD obtém um tempo menor, praticamente o mesmo obtido por CL4M e melhor que LCE e PROBCACHE. Portanto, acredita-se que quando pouco cache é ofertado para os roteadores, o desempenho da MCC é bastante afetado para essa topologia.

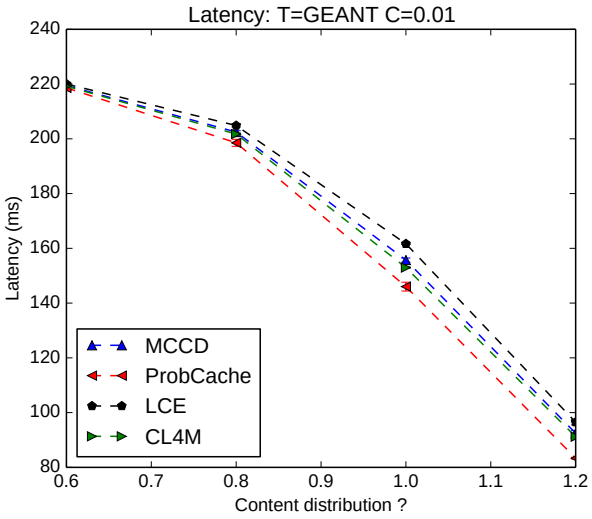

Figura 19: Latência - GEANT- Tamanho do Cache = 1%.

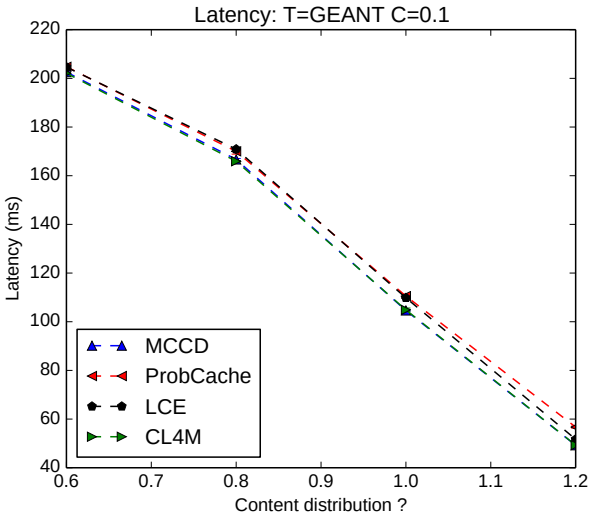

Figura 20: Latência - GEANT- Tamanho do Cache = 10%.

6.2.1 Discussão dos Resultados

Nesta seção, os resultados são discutidos para as duas métricas avaliadas (taxa de acertos e latência) nas topologias de árvore binária e TISCALI. São considerados três tamanhos de cache para os roteadores de conteúdo: 0.1%, 1% e 10% em relação à quantidade de conteúdos no repositório, conforme descrito na Tabela 2.

Primeiramente, 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.

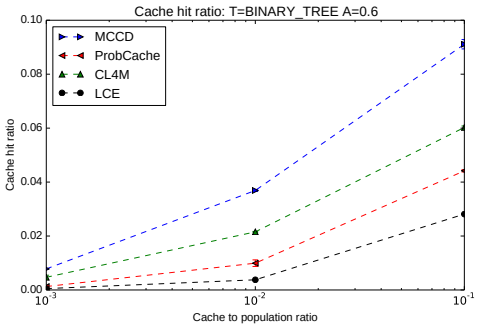

Figura 21: Taxa de Acertos x Tamanho do Cache - Árvore Binária - $\alpha = 0.6$.

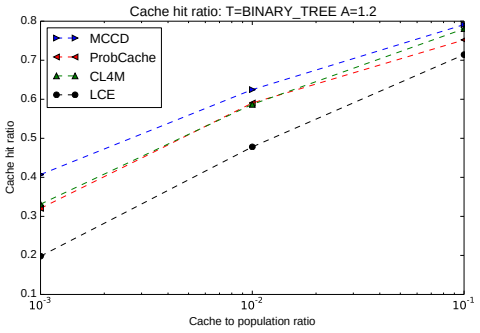

Figura 22: Taxa de Acertos x Tamanho do Cache - Árvore Binária - $\alpha = 1.2$.

Com relação aos resultados obtidos para árvore binária, 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 a Figura 21, 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 *ca-*

che disponível. Quando consideramos um valor maior de $\alpha = 1.2$, figura 22, 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.

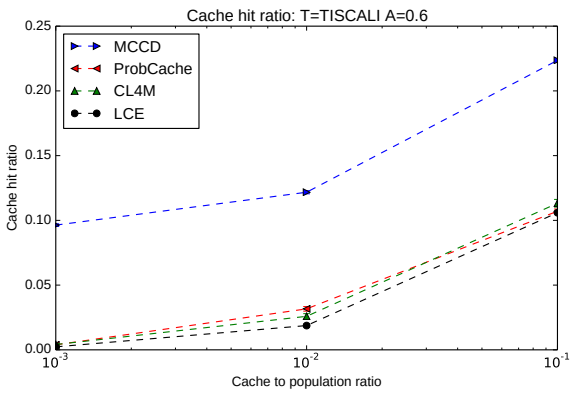

Figura 23: Taxa de Acertos x Tamanho do Cache - TISCALI - $\alpha = 0.6$.

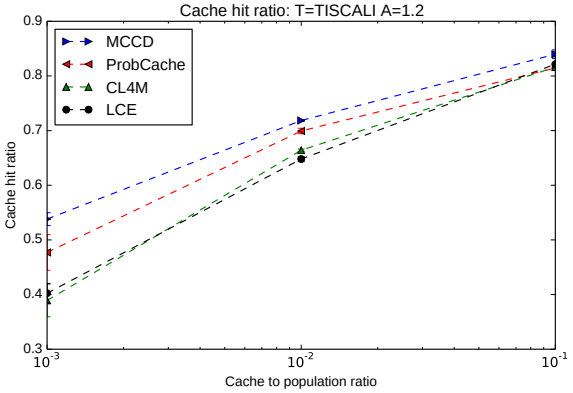

Figura 24: Taxa de Acertos x Tamanho do Cache - TISCALI - $\alpha = 1.2$.

Agora, discute-se os resultados para a topologia TISCALI representado nas figuras 23 e 24. Com o menor valor de $\alpha = 0.6$, figura 23, 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 $\alpha = 1.2$, figura 24, 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 [37], 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 25, 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 tem um melhor desempenho quando comparado às demais e, portanto, as duas propostas obtiveram tempos semelhantes, para os três tamanhos de *caches* avaliados. Já com um valor de $\alpha = 1.2$, figura 26, 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.

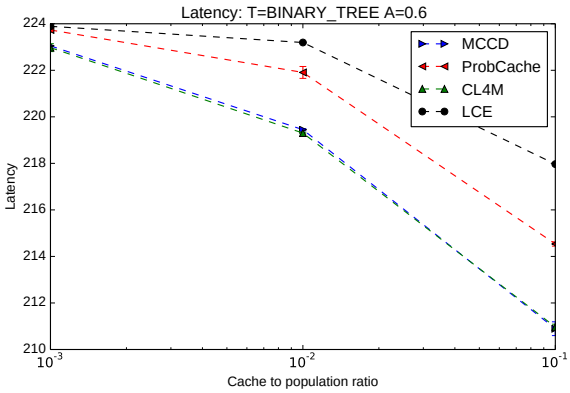

Figura 25: Latência x Tamanho do Cache - TISCALI - $\alpha = 0.6$.

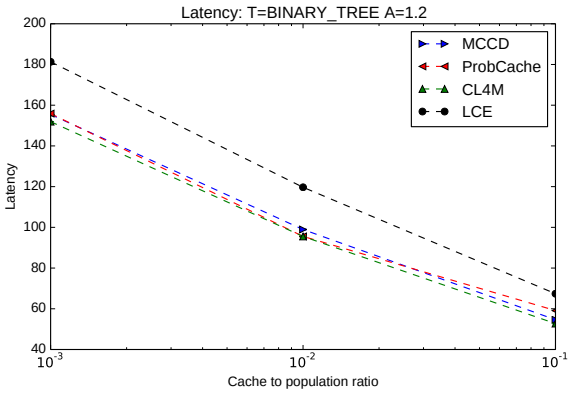

Figura 26: Latência x Tamanho do Cache - TISCALI - $\alpha = 1.2$.

Para a topologia TISCALI com $\alpha = 0.6$, figura 27, 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 28, o tempo da 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.

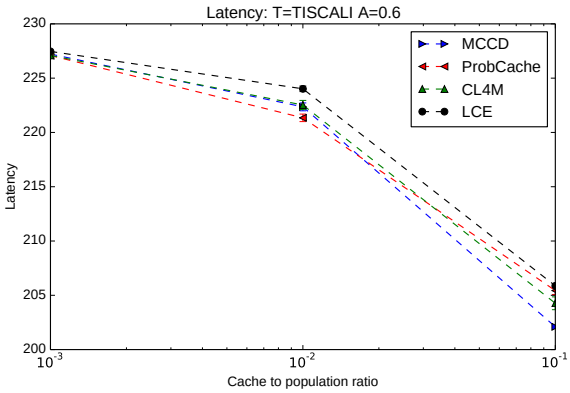

Figura 27: Latência x Tamanho do Cache - Árvore Binária - $\alpha = 0.6$.

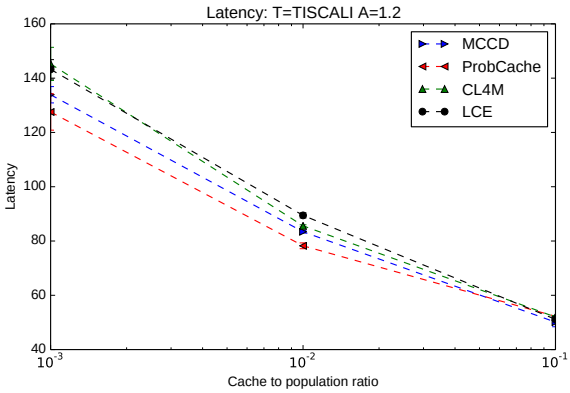

Figura 28: Latência x Tamanho do Cache - Árvoria Binária' - $\alpha = 1.2$.

6.3 Discussão sobre Uso da MCCD em Ambientes Reais

A estratégia MCCD é um modelo genérico que se encaixa dentro da arquitetura de ROCs. Os resultados obtidos mostraram que é possível trabalhar com várias métricas concomitantemente a fim de escolher os nós alvos que armazenarão o conteúdo solicitado. Mas, é importante levantar algumas questões sobre o uso da MCCD em ambientes reais e as possíveis implicações disso.

Primeiramente, a MCCD poderia ser implementada utilizando-se os modelos CCN ou NDN. Ambos trabalham com apenas dois tipos de pacotes: Interesse e Dado. Um usuário solicita um conteúdo da rede através de um pacote de interesse e pacote de dados correspondente é enviado a ele. No caso de usar a MCCD com uma delas num ambiente real, alguns campos extras precisariam ser adicionados no cabeçalho do pacote de interesses para que o pacote de dados possa extrair tais informações e armazenar o conteúdo nos roteadores selecionados. Entretanto, o custo adicional para obtenção e processamento de tais informações e o impacto de tal sobrecarga na rede são desconhecidos, considerando uma rede grande com centenas de nós.

Segundo, outra observação seria se a MCCD trabalharia com roteadores de conteúdo que tivessem tamanhos de caches homogêneos ou heterogêneos e o quanto isso impactaria no desempenho da estratégia. Uma possível saída seria considerar o tamanho relativo dos caches dos roteadores. Significa dizer que a capacidade de cache disponível considerada seria o percentual relativo de capacidade de cache disponível. Então, o conteúdo armazenado seria alocado no roteador com maior percentual relativo de espaço disponível para cache. Assim, o uso da MCCD em ambientes reais poderia ser possível uma vez que os roteadores de conteúdo disponibilizasse tal informação para ser usada pela estratégia de cache.

Além disso, é preciso analisar em quais contextos a MCCD funcionaria apropriadamente no que concerne o tipo de conteúdo que está sendo solicitado. Para conteúdos não-multimídias, acredita-se que a MCCD funcionaria normalmente, uma vez que o tamanho do conteúdo transportado seria de alguns kilobytes ou megabytes. Se considerarmos os modelos CCN e NDN, tal conteúdo seria segmentado em *chunks* (pedaços) e facilmente manipulados pela MCCD. Embora o crescimento rápido do tráfego de conteúdo multimídia, principalmente de vídeo, um estudo mais aprofundado deveria ser considerado no que diz respeito ao funcionamento da MCCD com conteúdos maiores e como eles deveriam ser alocados nos nós selecionados pela MCCD a fim de prover uma melhor Qualidade de Experiência (QoE) para os usuários finais.

Finalmente, ainda relacionado a conteúdo multimídia, algumas considerações importantes sobre o desempenho de estratégias de decisão e troca de conteúdo foram levantadas considerando tráfego de vídeo sob demanda (VoD) [38]. O estudo queria saber qual o impacto das estratégias de cache e do tamanho dos caches utilizados para armazenamento na Qualidade de Experiência. Os resultados mostraram que o tamanho dos

caches deveria ser maior que 100GB para promover um benefício significativo no tráfego PPTV (Redução de 22% do tráfego e de 34% na sobrecarga no servidor). Além disso, a combinação das estratégias de alocação e de troca de conteúdo influencia diretamente na métrica que quer se analisar. Otimizando as taxas de acertos de conteúdo ou reduzir as sobrecargas no servidor não necessariamente traz uma redução do uso de cache da rede. Isso deve ser cuidadosamente considerado pelos provedores de Acesso a Internet (ISPs) a fim de escolherem a melhor combinação de estratégias que estejam alinhadas aos seus objetivos. Portanto, o tamanho do cache tem um grande impacto no desempenho da estratégia de cache. No que diz respeito aos resultados obtidos pela MCCD, foram utilizadas 10.000 entradas disponíveis para cache em cada roteador de conteúdo, sendo, portanto, o uso da MCCD realizável em cenários como os mencionados anteriormente, certamente fazendo as adaptações necessárias para o seu correto funcionamento.

CAPÍTULO 7

Conclusões

O objetivo principal desta dissertação de mestrado foi propor uma estratégia de decisão de cache baseada em múltiplas métricas para ROCs, a MCCD. Agregando mais métricas para selecionar os nós que armazenarão o conteúdo, esperava-se obter uma taxa de acertos de conteúdo maior do que as baseadas em apenas uma métrica, levando um tempo de download praticável. Os resultados obtidos comprovaram que a MCCD conseguiu um desempenho maior que as outras propostas avaliadas, tanto para taxas de acertos quanto um tempo de download razoável.

A MCCD utilizou três métricas bases, as quais obtiveram um bom desempenho dentro dos testes as quais foram submetidas. Por isso, foram feitas adaptações necessárias que permitissem que a MCCD trabalhasse considerando três métricas ao invés de uma. Isto tornou a estratégia um tanto mais flexível que as demais, já que, como a rede é um ambiente muito dinâmico, o desempenho de uma estratégia baseada em uma métrica poderia não ser o suficiente para atender as demandas de conteúdo. A flexibilidade da MCCD permitiu que o conteúdo pudesse ser melhor distribuído pela rede em mais nós, com alto potencial de obterem uma maior taxa de acerto de conteúdo, fazendo com que ele pudesse ser obtido por usuários alocados em diferentes partes da rede. Vale lembrar que mesmo com um maior número de cópias do mesmo conteúdo na rede, a MCCD permite que o conteúdo alocado em um nó que reside num caminho de requisição de um usuário, esteja no caminho de requisição de outro usuário e, com isso, tal cópia é distribuída na rede. Certamente que um tratamento melhor para que a redundância de cópias do mesmo conteúdo precisa ser feita para que isto não afete o desempenho tanto da estratégia quanto da rede.

Os testes avaliados nesta dissertação mostraram que as estratégias baseadas em uma métrica não obtiveram uma taxa de acertos maior quando comparadas com a proposta baseada em múltiplas métricas. No caso da arvore binária, a qual apresenta um

formalismo para sua definição e dada a regularidade da disposição dos nós, a MCCD obteve uma taxa de acertos bem maior para conteúdos não-populares e populares. O tempo de download ficou praticamente igual ao obtido pela proposta CL4M, já que ambas utilizam a centralidade como métrica para decidir onde armazenar o conteúdo. No caso dos *backbones* TISCALI e GEANT, a MCCD também obteve uma taxa bem maior que as outras propostas, distribuindo melhor o conteúdo pela rede, permitindo que ele possa ser recuperado de qualquer parte. Na topologia GEANT, o tempo de download não foi o menor, ora praticamente igual ao da CL4M ora com um tempo maior que esta. Já na topologia TISCALI, a MCCD obteve um tempo menor apenas quando o tamanho de cache dos roteadores era de 10%, ficando praticamente igual a CL4M quando o tamanho de cache era de 1%.

Como trabalhos futuros, planeja-se estudar a possibilidade de se avaliar a proposta em um ambiente real, fazendo as necessárias adaptações, como discutido no capítulo 4. Um estudo mais detalhado sobre o custo computacional adicional pelo uso de múltiplas métricas deve ser considerado a fim de que sua utilização seja viável em ambientes reais. É necessário estudo mais detalhado sobre os ganhos em desempenho da MCCD para outros tamanhos de caches dos roteadores. Também, outra métrica que se espera otimizar é o tempo de download. Com uma latência menor isso melhoraria a Qualidade de Experiência (QoE) dos usuários. Além disso, o estudo de outras métricas que pudessem melhorar ainda mais o seu desempenho seria, por exemplo, a popularidade do conteúdo. Finalmente, pode-se também avaliar outras métricas de desempenho como, por exemplo, redução do número de saltos para a obtenção de conteúdo e redução de solicitações de conteúdos atendidas pelo servidor, o que evidencia a eficiência da estratégia de cache.

Referências

[1] V. Jacobson, M. Mosko, D. Smetters, and J. Garcia-Luna-Aceves, "Content-centric networking," *Whitepaper, Palo Alto Research Center*, pp. 2–4, 2007.

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

[3] Cisco, "Cisco visual networking index: forecast and methodology, 2014-2019," Cisco, Tech. Rep., 2015. [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-next-generation-network/white_paper_c11-481360.pdf

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

[5] C. Slater and K. R. Chennapragada, "Http redirection of configuration data for network devices," May 17 2005, uS Patent 6,895,433.

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

[7] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," in *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 4. ACM, 2007, pp. 181–192.

[8] L. Zhang, D. Estrin, J. Burke, V. Jacobson, J. D. Thornton, D. K. Smetters, B. Zhang, G. Tsudik, D. Massey, C. Papadopoulos *et al.*, "Named data networking (ndn) project," *Relatório Técnico NDN-0001, Xerox Palo Alto Research Center-PARC*, 2010.

[9] D. Lagutin, K. Visala, and S. Tarkoma, "Publish/subscribe for internet: Psirp perspective." *Future internet assembly*, vol. 84, 2010.

[10] D. R. Cheriton and M. Gritter, "Triad: A new next-generation internet architecture," 2000.

[11] C. Dannewitz, "Netinf: An information-centric design for the future internet," in *Proc. 3rd GI/ITG KuVS Workshop on The Future Internet*, 2009.

[12] K. Katsaros, G. Xylomenos, and G. C. Polyzos, "Multicache: An overlay architecture for information-centric networking," *Computer Networks*, vol. 55, no. 4, pp. 936–947, 2011.

[13] E. Cohen and S. Shenker, "Replication strategies in unstructured peer-to-peer networks," in *ACM SIGCOMM Computer Communication Review*, vol. 32, no. 4. ACM, 2002, pp. 177–190.

[14] A. Wierzbicki, N. Leibowitz, M. Ripeanu, and R. Wozniak, "Cache replacement policies revisited: The case of p2p traffic," in *Cluster Computing and the Grid, 2004. CCGrid 2004. IEEE International Symposium on.* IEEE, 2004, pp. 182–189.

[15] G. Zhang, M. Tang, S. Cheng, G. Zhang, H. Song, J. Cao, and J. Yang, "P2p traffic optimization," *Science China Information Sciences*, vol. 55, no. 7, pp. 1475–1492, 2012.

[16] W. K. Chai, M. Georgiades, and S. Spirou, "Towards information-centric networking: research, standardization, business and migration challenges," *Media Networks: Architectures, Applications, and Standards*, p. 163, 2012.

[17] G. Zhang, Y. Li, and T. Lin, "Caching in information centric networking: a survey," *Computer Networks*, vol. 57, no. 16, pp. 3128–3141, 2013.

[18] M. Ain, D. Trossen, P. Nikander, S. Tarkoma, K. Visala, K. Rimey, T. Burbridge, J. Rajahalme, J. Tuononen, P. Jokela *et al.*, "D2. 3–architecture definition, component descriptions, and requirements," *Deliverable, PSIRP 7th FP EU-funded project*, 2009.

[19] C. Fricker, P. Robert, J. Roberts, and N. Sbihi, "Impact of traffic mix on caching performance in a content-centric network," in *Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on.* IEEE, 2012, pp. 310–315.

[20] I. Psaras, W. K. Chai, and G. Pavlou, "Probabilistic in-network caching for information-centric networks," in *Proceedings of the second edition of the ICN workshop on Information-centric networking.* ACM, 2012, pp. 55–60.

[21] D. Perino and M. Varvello, "A reality check for content centric networking," in *Proceedings of the ACM SIGCOMM workshop on Information-centric networking.* ACM, 2011, pp. 44–49.

[22] L. Muscariello, G. Carofiglio, and M. Gallo, "Bandwidth and storage sharing performance in information centric networking," in *Proceedings of the ACM SIGCOMM workshop on Information-centric networking.* ACM, 2011, pp. 26–31.

[23] S. Arianfar, P. Nikander, and J. Ott, "Packet-level caching for information-centric networking," in *ACM SIGCOMM, ReArch Workshop*, 2010.

[24] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and zipf-like distributions: Evidence and implications," in *INFOCOM'99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 1. IEEE, 1999, pp. 126–134.

[25] O. Saleh and M. Hefeeda, "Modeling and caching of peer-to-peer traffic," in *Network Protocols, 2006. ICNP'06. Proceedings of the 2006 14th IEEE International Conference on.* IEEE, 2006, pp. 249–258.

[26] M. Hefeeda and O. Saleh, "Traffic modeling and proportional partial caching for peer-to-peer systems," *Networking, IEEE/ACM Transactions on*, vol. 16, no. 6, pp. 1447–1460, 2008.

[27] D. Rossi, G. Rossini *et al.*, "On sizing ccn content stores by exploiting topological information." in *INFOCOM Workshops*, 2012, pp. 280–285.

[28] Y. Lu, T. F. Abdelzaher, and A. Saxena, "Design, implementation, and evaluation of differentiated caching services," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 15, no. 5, pp. 440–452, 2004.

[29] G. Tyson, S. Kaune, S. Miles, Y. El-khatib, A. Mauthe, and A. Taweel, "A trace-driven analysis of caching in content-centric networks," in *Computer Communications and Networks (ICCCN), 2012 21st International Conference on.* IEEE, 2012, pp. 1–7.

[30] Z. Li and G. Simon, "Time-shifted tv in content centric networks: The case for cooperative in-network caching," in *Communications (ICC), 2011 IEEE International Conference on.* IEEE, 2011, pp. 1–6.

[31] N. Laoutaris, S. Syntila, and I. Stavrakakis, "Meta algorithms for hierarchical web caches," in *Performance, Computing, and Communications, 2004 IEEE International Conference on.* IEEE, 2004, pp. 445–452.

[32] N. Laoutaris, H. Che, and I. Stavrakakis, "The lcd interconnection of lru caches and its analysis," *Performance Evaluation*, vol. 63, no. 7, pp. 609–634, 2006.

[33] S. Eum, K. Nakauchi, M. Murata, Y. Shoji, and N. Nishinaga, "Catt: potential based routing with content caching for icn," in *Proceedings of the second edition of the ICN workshop on Information-centric networking.* ACM, 2012, pp. 49–54.

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

[35] W. K. Chai, D. He, I. Psaras, and G. Pavlou, "Cache ?less for more? in information-centric networks," in *NETWORKING 2012.* Springer, 2012, pp. 27–40.

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

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

[38] Y. Sun, S. K. Fayaz, Y. Guo, V. Sekar, Y. Jin, M. A. Kaafar, and S. Uhlig, "Trace-driven analysis of icn caching algorithms on video-on-demand workloads," in *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*. ACM, 2014, pp. 363–376.