Uma Abordagem Colaborativa de Cache em Ambientes de Redes Móveis*

M.F. Caetano[1], Jacir L. Bordim[1], M.A.R. Dantas[2]

[1]Departamento de Ciência da Computação (CIC)
Universidade Federal de Brasília (UnB), 70910-900, Brasília, DF, Brazil.

[2]Departamento de Informática e Estatística (INE)
Universidade Federal de Santa Catarina (UFSC), 88040-900, Florianópolis, SC, Brazil.

{caetano,bordim}@cic.unb.br[1], mario@inf.ufsc.br[2]

Abstract. *The main contribution of this article is to propose a collaborative and distributed cache mecanism tailored for Ad Hoc networks. Each node's cache is partitioned into a global area and a restricted area. The proposed collaborative cache is formed by each node's global area, which is used to store contents relevant to the group, while the restricted area is used to store information relevant to its owner. Empirical results have shown that our global cache mecanism, which enables nodes to collectively decide which pages are stored in the global area, can significantly reduce the amount of network traffic, latency and energy consumption as well as the server's load.*

Resumo. *A principal contribuição deste artigo é a proposta de um mecanismo colaborativo e distribuído de cache para redes Ad Hoc. Cada área de cache, pertencente a cada nó, é dividida entre área global e área restrita. A área de cache colaborativa proposta é formada pelas áreas globais de cache de cada nó, a qual é utilizada para o armazenamento de conteúdo relevante para o grupo. Já a área restrita de cache é utilizada para o armazenamento de informações relevantes ao próprio nó ao qual essa área pertence. Resultados empíricos mostram que o mecanismo proposto permite reduzir, significativamente, a quantidade de tráfego de rede, latência, consumo de energia, assim como a carga no servidor.*

1. Introdução

A utilização de tecnologias de rede sem fio está cada vez mais presente no nosso dia-a-dia. De fato, hoje a maioria dos dispositivos já possui algum tipo de tecnologia de rede sem fio embutida, sendo encontrada em *laptops*, *PDA*, celulares, e até mesmo em *desktops*. As redes sem fio podem ser classificadas em dois grupos distintos: redes infra-estruturadas e redes sem infra-estrutura ou *Ad Hoc*. A figura 1 apresenta uma topologia de rede sem fio híbrida. De acordo com a figura, é possível observar que alguns nós não estão conectados diretamente ao ponto de acesso por não estarem próximos o suficiente do mesmo. Em um modelo de rede *Ad Hoc*, não existe a necessidade de uma infra-estrutura pré-estabelecida para a comunicação. Ou seja, os nós comunicam-se diretamente sem o intermédio de um ponto de acesso.

*Este trabalho recebeu recursos financeiros provenientes do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e da Fundação de Apoio à Pesquisa do Distrito Federal (FAPDF).

O desenvolvimento das tecnologias de comunicação sem fio, associados à redução dos custos e ao sucesso do padrão IEEE 802.11 [for WLAN Standards], possibilitaram adoção de dispositivos móveis em diversos ambientes. O padrão IEEE 802.11 especifica os mecanismos para a comunicação de um salto entre dispositivos móveis que utilizam comunicação sem fio. Para comunicação em múltiplos saltos, faz-se necessário a utilização de um protocolo de roteamento. O IETF (*Internet Engineering Task Force*) padronizou alguns algoritmos para este propósito, com por exemplo o protocolo AODV [Perkins and Royer 1999]. Desta forma, a rede passa a ter uma maior área de cobertura, pois os nós localizados na borda do sinal do ponto de acesso passam a rotear as requisições dos nós mais afastados.

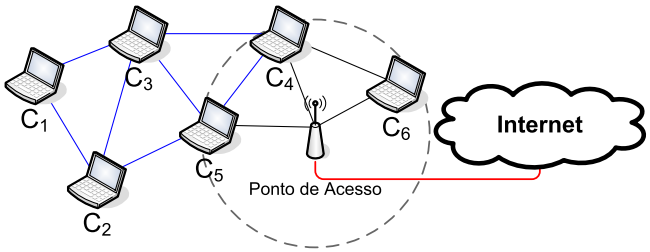

Figura 1. Topologia mista de rede sem fio. Modelo tradicional e modelo *Ad Hoc* operando juntos.

A topologia apresentada na figura 1 pode ser implementada em um ambiente real como um saguão de aeroporto, por exemplo. Neste tipo de ambiente, os usuários utilizam diversos tipos de dispositivos para obter acesso à *Internet*. Há vários problemas associados a esta forma de acesso compartilhado. Como o ponto de acesso serve como *gateway* para a Internet, todas as requisições passam por ele. Um aumento no volume de requisições consequentemente representa um aumento no tempo de resposta. O tempo de resposta pode ser ainda maior caso não exista um servidor *proxy* instalado logo após o ponto de acesso, uma vez que todas as requisições serão enviadas para o servidor onde o dado solicitado está hospedado. Entretanto, o uso da tecnologia de rede *Ad Hoc* permite que as páginas armazenadas no *cache* dos clientes sejam compartilhadas entre os demais membros da rede, fazendo com que nem toda a requisição seja submetida ao *gateway*. Desta forma é possível diminuir o tempo de resposta e minimizar as chances de sobrecarregar o *gateway*. Essa forma de cooperação é conhecida como *cache* cooperativo ou *cache* colaborativo.

Cache colaborativo tem sido largamente utilizado em redes cabeadas para diminuir o tempo de acesso, evitar gargalos, e melhorar a performance [Dahlin et al. 1994]. Atualmente, diversos trabalhos tem considerado a adoção de *cache* colaborativo no contexto de redes *Ad Hoc* [Bjornsson and Shrira 2002, Chand et al. 2006, Chow et al. 2004, Chow et al. 2007, Chow et al. 2005, Huang et al. 2006, Du and Gupta 2005, Ting and Chang 2007, Yin and Cao 2006]. O principal argumento no seu uso parte da necessidade em se obter a informação desejada em um menor número de saltos possíveis. Conforme demonstrado por Gupta [Gupta and Kumar 2000], quanto maior o número de nós envolvidos, menor será a vazão do canal. Esta característica está relacionada com os problemas conhecidos como *terminal exposto* e *terminal escondido* que afetam a performance de sistemas distribuídos que utilizam mecanismos de comunicação sem fio.

As políticas colaborativas de *cache* propostas na literatura até o momento limitam-se a disponibilizar aos demais membros da rede as informações armazenadas no *cache* de cada cliente. Cada nó mantém apenas uma política individual de *cache*, armazenando somente dados de seu interesse. Neste caso, haverá um ganho no sistema somente quando o dado requisitado encontra-se armazenado em outro membro da rede. Nesse modelo não existe uma política de cache global com objetivo de explorar o armazenamento de páginas de interesse dos demais membros da rede.

A principal contribuição deste trabalho é propor um sistema colaborativo de *cache* para redes *Ad Hoc* que explora o conceito de *cache* global. Esta área global de *cache* é formada a partir da reserva de parte do *cache* local de cada cliente para armazenar as páginas de interesse de um grupo. Os resultados obtidos mostram que a utilização de políticas de cache colaborativo, com o compartilhamento de recursos, permitem reduzir, significativamente, o volume de tráfego na rede, o consumo de energia, e a carga de requisições enviadas ao servidor.

O restante deste trabalho é organizado da seguinte forma, a próxima seção trata de definições e trabalhos correlatos. A seção 3 apresenta o modelo de *cache* colaborativo proposto neste trabalho. A seção 4 apresenta os resultados experimentais. As conclusões e propostas para trabalho futuro são apresentadas na seção 5

2. Definições e Trabalhos Correlatos

Em ambientes de rede sem fio, o emprego de políticas de *cache* eficientes permitem um melhor aproveitamento dos recursos da rede e minimizam o consumo de bateria dos dispositivos móveis [Li et al. 2007]. O conceito de *cache*, utilizado neste trabalho, é definido como uma área local e individual destinada ao armazenamento de informações de interesse do dispositivo. As informações armazenadas em *cache* são obtidas a partir de um servidor remoto, semelhante ao ilustrado na figura 1.

Quando um dado é recuperado a partir do seu *cache* local, dizemos que ocorreu um *cache-hit*. Quando o dado não encontra-se armazenado localmente, dizemos que ocorreu um *cache-miss*. Um *cache-miss* acarreta no uso do sistema de comunicação para o envio de requisições ao servidor. Além do *delay* associado a recuperação do dado, temos o envolvimento de outros dispositivos para fazer o roteamento da informação. Em redes *Ad Hoc* o custo gerado por um *cache-miss* reflete não apenas no dispositivo que fez a requisição, mas em todos os nós envolvidos com o roteamento das informações até o servidor.

2.1. Sistema de Cache

Uma *política de substituição* é acionada toda vez que o *cache* está cheio e um novo dado precisa ser armazenado. A escolha de uma *vítima* a ser substituída não é uma tarefa trivial e tem sido foco de pesquisas nos últimos anos. Dentre os propostas de políticas de substituição, podemos citar:

- LRU (*Least Recently Used*) – essa política mantém em cache apenas os dados recentemente acessados. Havendo necessidade, os dados mais antigos armazenados em cache, são selecionados para substituição [Podlipnig and Boszormenyi 2003];
- LFU (*Least Frequently Used*) – associa a cada dado armazenado em *cache* um contador, o qual é incrementado a medida que os dados são referenciados. As vítimas são selecionadas baseadas no valor do contador [Arlitt et al. 2000];

- LFU-Aging – na política de LFU original, os dados que tiveram um valor elevado de referencia no passado, porém não mais requisitados, podem permanecer no cache por um longo período. A abordagem LFU resolve esse problema com a implementação de um mecanismo de envelhecimento associado ao contador [Arlitt et al. 2000];
- SLRU (*Segmented Least Recently Used*) – divide a área para o armazenamento dos dados em duas partes: segmento protegido e segmento não protegido. A idéia básica do algoritmo é reservar uma área em disco para armazenamento dos objetos mais populares [Karedla et al. 1994];
- RAND – escolhe suas vítimas de forma aleatória.

A replicação de dados em um ambiente distribuído acarreta no problema de gerenciamento de consistência das informações. Os modelos de consistência de *cache* podem ser classificados em: modelo de consistência forte (CF), modelo de consistência Δ (CD) e modelo de consistência fraca (CFr), mais conhecida como TTL (*time-to-live*). Para maiores detalhes, direcionamos o leitor para [Cao et al. 2007].

2.2. Trabalhos Correlatos

Yin et al. [Cao et al. 2004, Yin and Cao 2006] propuseram três políticas cooperativas de *cache*, são elas: *cacheData*, *cachePath* e *HybridCache*. Independentemente da política adotada, após um *local miss*, o nó submete sua requisição ao nó servidor. Como a idéia central do trabalho é fornecer uma resposta com o menor número de saltos possível, cada requisição sempre será analisada pelos nós intermediários antes de serem roteadas. Os nós intermediários podem tomar a iniciativa de armazenar em *cache* (*i*) o dado; (*ii*) o caminho até o dado; ou (*iii*) optar para uma abordagem híbrida das duas anteriores. Ao analisar uma requisição, o nó intermediário poderá indicar um caminho alternativo, mais curto, para obter o dado. Um dos problemas dessa abordagem é a quantidade indiscriminada de cópias de dados que são armazenados. Além de tornar muito oneroso para o sistema o emprego de políticas de coerência de *cache*, o número de cópias diminui a diversidade das informações armazenadas na rede e, por conseqüência, diminui a probabilidade de consultas globais serem bem sucedidas.

Du et al. [Du and Gupta 2005] demonstraram que consultas em *cache* baseadas em *flooding* devem limitar-se a no máximo 4 saltos. Acima desse valor, o custo para encontrar o dado entre os vizinhos mais distantes é maior do que submeter a requisição diretamente ao servidor. Mesmo não implementando o conceito de *cluster*, seus resultados podem ser utilizados para definição do tamanho máximo do número de saltos que um *cluster* deve ter. Cao et al. [Cao et al. 2005] adaptaram o modelo de *cluters* apresentado em [Bjornsson and Shrira 2002] para o ambiente de rede *Ad Hoc* com o intuito de reduzir o volume de mensagens para consultas em *cache*. Aqui, o estabelecimento de *cluters* leva em consideração o *nível de energia*, *taxa de acesso* e *tempo de presença*. Desta forma, o número de mensagens necessárias para invalidar um determinado dado é menor, pois na visão do servidor a rede é formada por apenas alguns nós (*relay peers* ou *clusters head*). O foco do trabalho é no uso de *cluster* como forma de diminuir o número de mensagens geradas pelo emprego de políticas de coerência forte.

Na mesma linha, Chow et al. [Chow et al. 2007] utilizam conceito de *clusters* e assinaturas [Bloom 1970] para diminuição no número de mensagens. Os *clusters* são

formados baseados no padrão de acesso e movimentação dos nós. O controle dos grupos é feito pelo servidor, que recebe periodicamente informações sobre a posição dos nós através do uso de GPS (*Global Positioning System*). Essa abordagem trabalha com uma visão global da rede, o que torna muito onerosa a sua utilização. A tabela 1 apresenta um comparativo entre as políticas apresentadas nesta seção.

	A	**B**	**C**	**D**
Modelo de Consistência	Fraca TTL	Fraca TTL	Forte	Fraca TTL
Política de Substituição	LRU	LRU	LRU	LRU
Controle de Consistência	Stateless	Stateless	Statefull	Stateless
Uso de Cluster	Não	Não	Sim	Sim
Decisão Colaborativa	Não	Não	Não	Não
Distinção de Dado	Não	Sim	Não	Não

Tabela 1. Comparativo entre as políticas de cache apresentadas. Onde o trabalho A refere-se a [Yin and Cao 2006], B a [Du and Gupta 2005], C a [Cao et al. 2005] e D a [Chow et al. 2007].

A principal contribuição deste trabalho é a proposta de um ambiente distribuído de *cache* destinado ao armazenamento de informações de interesse de um determinado conjunto de nós. Diferente dos trabalhos descritos acima, o conceito de *cache* colaborativo abordado neste trabalho visa permitir consultas distribuídas das áreas de *caches* de forma colaborativa, onde a decisão de qual página deve ser armazenada na área global é feita de forma coordenada. O objetivo dessa abordagem é aumentar o número de *cache global hit* e, consequentemente, diminuir a penalização associada a cada *cache local miss* oriunda do modelo de *cache* tradicional. Neste modelo, os nós que formam o *cluster* contribuem com espaço local em *cache* para formar essa área global compartilhada. Maiores informações com relação ao modelo serão apresentados nas seções subseqüentes.

3. Modelo Proposto

3.1. Definição do Modelo

Neste trabalho consideramos que um mecanismo que permita a formação e manutenção do *cluster* esteja implementado. Para fins de implementação, utilizamos o mecanismo de clusterização proposto por Bjornsson *et al* [Bjornsson and Shrira 2002]. Outros mecanismos poderiam ser adaptados para este fim. Entretanto, optou-se pela escolha desse mecanismo por reunir as características básicas necessárias para a implementação do nosso modelo.

A partir da figura 2 é possível verificar os componentes desta estrutura. O *cluster* exemplificado na figura é formado por quatro nós, cuja distância entre si é de apenas um salto. Todo nó pertencente ao *cluster* distancia-se, obrigatoriamente, um salto do *cluster head* e no máximo dois saltos de um outro membro. O *cluster head* é o nó responsável por toda comunicação externa ao *cluster*, desempenhado o papel de *gateway* para os demais membros. Os nós são conhecidos como *cluster nodes* e comunicam-se entre si através de *broadcast*. Toda comunicação interna ao *cluster* é capturada pelo *cluster head*, cuja

análise fornece elementos para execução das operações de gerenciamento da área *global de cache*. De acordo com a figura 2, a área de *cache* de cada cliente é dividida em duas partes:

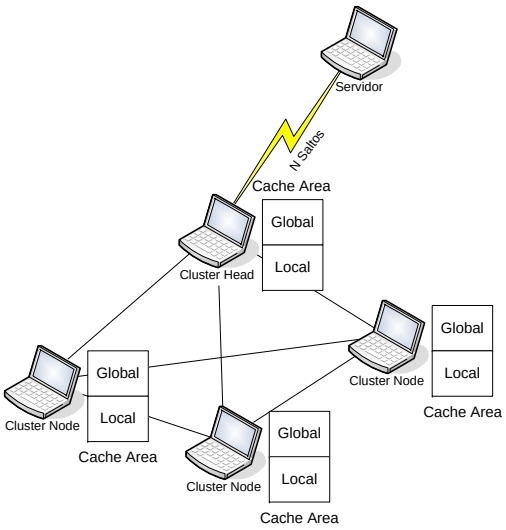

Figura 2. Disposição do Sistema de *Cache* Colaborativo Proposto Dentro do *Cluster* Formado.

- *área global* – é a área de *cache* destinada ao armazenamento de páginas de interesse dos demais membros do *cluster*. Esta área é gerenciada pelo *cluster head*. A soma das áreas globais, de todos os *cluster nodes* , compõe a área total de *cache* da política colaborativa;
- *área local* – é a área de *cache* destinada ao armazenamento das páginas de interesse do nó.

De acordo com o modelo colaborativo proposto, as páginas armazenadas em *cache* são classificadas em:

- *páginas globais* – são páginas identificadas pelo *cluster head* como sendo do interesse de dois ou mais *cluster nodes*. São armazenadas na área global de cache e o gerenciamento é feito pelo *cluster head*. O *cluster head* também define em qual porção da área global, *cluster node*, a página será armazenada;
- *páginas locais* – são páginas do interesse individual de cada *cluster node*. São armazenadas na área local e o gerenciamento é feito de maneira individual e independente por cada *cluster node*;

Deixe N_i representar um dado nó em um *cluster*, onde $0 \leq i < m$ e m é o número total de nós no cluster. A área total de cache do nó N_i é definida como sendo igual a $\tau = \alpha_i + \beta_i$, onde α_i representa a área destinada ao armazenamento das páginas de interesse local do nó e, β_i representa a área destinada ao armazenamento das páginas de interesse global. A área de *cache* global de um *cluster* é dado por $\zeta = \sum_{i=0}^{m-1} \beta_i$.

Cada nó no *cluster* pode utilizar o espaço β_i para o armazenamento de páginas de seu interesse, caso o mesmo não esteja sendo utilizado. Entretanto, nesta área existe uma precedência das páginas globais sobre as locais. Ou seja, o espaço alocado para a área global deve ser respeitado no momento em que uma página de interesse global seja

recebida para ser armazenada. Uma página será identificada como sendo de interesse global a partir do momento que o *cluster head* verifica que mais de um nó mostrou-se interessado por ela. Quando esse evento ocorrer, o *cluster head* irá notificar o *cluster* a respeito de qual nó será responsável por armazenar a página em sua área global.

Para validação do modelo, algumas restrições foram feitas com a finalidade de simplificar e facilitar a demonstração do funcionamento da proposta. No entanto, é importante ressaltar que a solução aqui proposta pode ser generalizada e as restrições eliminadas sem a perda das características de funcionamento do sistema. As restrições consideradas neste trabalho são:

- a topologia do *cluster* é definida, de maneira estática, antes do início da simulação;
- os nós membros do *cluster* estão a um saltos de distância do *cluster head*;
- toda comunicação *intra-cluster* é capturada pelo *cluster head*;
- toda a comunicação para fora do *cluster* é feita por intermédio do *cluster head*;
- os nós não ficam indisponíveis;
- é implementado o modelo de consistência fraca;

Neste trabalho utilizamos *clusters* os quais são formados a partir de requisitos mínimos, como espaço de armazenamento e localização geográfica. O comportamento do sistema de *cache* colaborativo proposto é apresentado na seqüência em forma de tratamento para casos específicos do ambiente.

3.2. Protocolo de Cache Colaborativo

As principais características do protocolo de *cache* colaborativo proposto são apresentadas conforme o tratamento dos seguintes eventos:

- **Cache Local Hit** – o sistema de *cache* verifica se existe, localmente armazenado, uma cópia válida da página solicitada. Caso exista, a mesma é identificada como tendo sido acessada recentemente e encaminhada para aplicação. Caso contrário, teremos um *cache local miss*;
- **Cache Local Miss** – os *cluster nodes* utilizam *broadcast* para submeterem suas requisições de páginas aos demais membros do *cluster*. Após um *broadcast* proveniente de um *cluster node*, caso não haja resposta, o *cluster head* irá procurar a página na área global de *cache* e repetirá o *broadcast* recebido caso a página não seja encontrada. A consulta da localização da página na área global é feita nas tabelas de controle da área global que o *cluster head* mantém atualizada em sua área de dados. O segundo *broadcast* faz-se necessário para que todas as áreas locais de *cache* sejam alcançadas. Caso algum *cluster node* responda a qualquer uma das requisições, dizemos que ocorreu um *cache global hit*. Caso contrário, o evento é classificado como *cache global miss*. Entretanto, se a página solicitada for encontrada na área global de *cache*, o *cluster head* submeterá a requisição diretamente ao *cluster node*, que armazena a página. Nesse caso, a resposta será encaminhada diretamente ao nó requisitante inicial. Por fim, se a página armazenada na área global estiver desatualizada, o *cluster head* irá recuperá-la diretamente a partir do servidor principal da rede e a resposta será enviada através de *broadcast* juntamente com instruções de atualização da área global;

- **Cache Global Hit** – ao receber a resposta do *cluster*, o *cluster head* sabe se a mesma foi proveniente da área local ou global de *cache*. As páginas provenientes dá área global serão marcadas como recentemente acessadas, já as provenientes das áreas locais serão promovidas a área global. Somente serão enviadas ao *cluster* as páginas recebidas a partir de consultas feitas pelo próprio *cluster head*. Por fim, da mesma forma como descrito no item anterior, o *cluster head* irá proceder a atualização das páginas antigas provenientes da área global de *cache*;
- **Cache Global Miss** – caso não haja resposta após o *broadcast* submetido pelo *cluster head*, torna-se claro que ocorreu um *cache global miss*. Neste caso, o *cluster head* irá submeter a consulta ao servidor de páginas da rede e a resposta será enviada através de *broadcast* aos membros do *cluster*.
- **Consistência de Cache** – assim como nos trabalhos apresentados na revisão bibliográfica, esta proposta implementa o modelo de consistência fraca TTL. Ou seja, utiliza-se uma *tag* para identificar uma previsão de quanto tempo a página armazenada continuará válida.
  Em consultas submetidas ao *cluster*, as páginas armazenadas nas áreas locais de *cache* serão excluídas, pelos respectivos *clusters nodes*, sem que o seu conteúdo desatualizado seja transmitido pela rede. As páginas armazenadas nas áreas globais de *cache* serão gerenciadas apenas pelo *cluster head*. Neste caso, o *cluster node* que armazena a página desatualizada não realizará nenhuma ação ao receber uma consulta de página.

4. Resultados e Análise

Esta seção apresenta os resultados do esquema de cache proposto. Inicialmente será detalhado o ambiente de simulação utilizado nos testes bem como a topologia analisada.

4.1. Ambiente de Simulação

A validação do modelo proposto foi feita através da implementação da proposta utilizando o simulador de redes *GloMoSim* [Zeng et al. 1998]. Optou-se pela utilização do *GloMoSim* pelo fato dele ser um simulador consolidado e largamente referenciado pelos trabalhos publicados na área. Os parâmetros utilizados na simulação estão descritos na tabela 2. A política de substituição de páginas aplicada tanto na área global, pelo *cluster head*, quanto na área privada de *cache*, pelos *cluster nodes*, é a LRU.

A proposta apresentada foi comparada com modelo tradicional e individual de *cache*. O objetivo é demonstrar o funcionamento, e possível benefício, de uma abordagem colaborativa sobre um individual. No modelo tradicional implementado, cada nó possui uma área individual de *cache* a qual é consultada toda vez que a sua aplicação deseja uma página. Neste modelo, *local miss* são tratados com o envio da consulta para o servidor central da rede.

A figura 3 apresenta a topologia de rede utilizada durante as baterias de testes de validação. O servidor de páginas da rede, representado pelo nó mais afastado do *cluster*, disponibiliza, de uma maneira transparente, todas as páginas requisitadas pelos demais nós da rede. Os nós não pertencentes ao *cluster* realizam apenas roteamento das requisições durante a execução das baterias de testes. Toda bateria foi repetida 6 vezes e os resultados obtidos tratados. Os valores discrepantes foram retirados e a média ponderada foi obtida, formando os valores finais apresentados no gráfico. O número total de

Parâmetro	Valor
NUMBER-OF-NODES	24
MOBILITY	NONE
PROPAGATION-PATHLOSS	TWO-RAY
RADIO-TYPE	RADIO-ACCNOISE
RADIO-FREQUENCY	2.4e9
RADIO-BANDWIDTH	2000000
RADIO-TX-POWER	1.1
RADIO-ANTENNA-GAIN	0.0
RADIO-RX-SENSITIVITY	-91.0
RADIO-RX-THRESHOLD	-79.0
MAC-PROTOCOL	802.11
NETWORK-PROTOCOL	IP
ROUTING-PROTOCOL	AODV

Tabela 2. Valores dos parâmetros utilizados no *GloMoSim* durante a simulação do modelo proposto.

nós na rede permaneceu sempre constante, contudo a mesma bateria de teste foi repetida para o *cluster* formado por: 2, 4, 8 e 16 nós.

A modelagem da capacidade de armazenamento do sistema de *cache* é representada pelo gráfico da figura 4. O gráfico apresenta a variação na quantidade de membros pertencentes ao *cluster* pela capacidade de armazenamento do sistema de *cache*. Quando o número de nós é igual a um, a política de *cache* utilizada é a individual, não havendo, nesse caso, colaboração e divisão da área local de *cache*. Para o ambiente de simulação, foram definidos três subconjuntos de páginas a serem utilizadas como o universo de páginas distintas existentes. Atualmente, a quantidade de páginas publicadas na *internet* está na casa das dezenas de bilhões, o que obriga a definição desses subconjuntos. Para este contexto de simulação, foram definidos os seguintes subconjuntos: 256, 512 e 1024 páginas.

Observando o gráfico da figura 4, é possível verificar que quando o número de nós é igual a um, o sistema de *cache* individual consegue armazenar 5% das 256 páginas distintas, 2.5% das 512 páginas e 1.25% das 1024 páginas. A partir de 2 nós, a proposta colaborativa de *cache* passa a ser a política de *cache* utilizada e, para este modelo, fica definido que o tamanho β_i é de 50% do tamanho de τ. Na política colaborativa, 2 nós armazenam 5% das 256 páginas, 4 nós conseguem armazenar 10%, chegando ao máximo de 40% das 256 páginas quando o número de nós é 16.

4.2. Verificação da Eficiência do Sistema de *Cache*

O objetivo dessa bateria inicial de testes foi verificar a quantidade de requisições que foram efetivamente enviadas ao servidor. Comparado com a política individual de *cache*, espera-se uma redução na quantidade de requisições submetidas ao servidor geral da rede. A figura 5 apresenta a porcentagem de mensagens submetidas ao servidor pela variação na quantidade de membros do *cluster*. Conforme apresentado na subseção anterior, o aumento no número de membros do *cluster* reflete no aumento no tamanho da área de

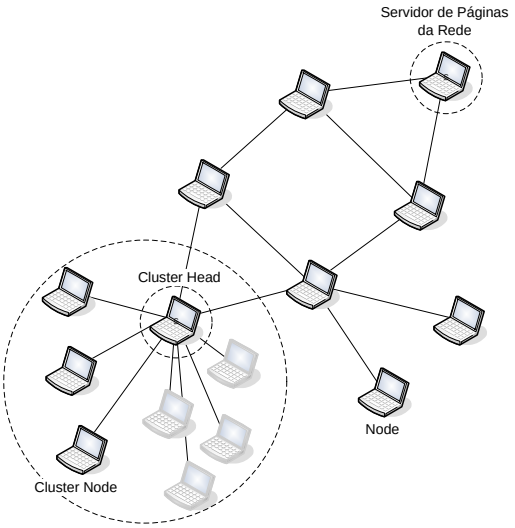

Figura 3. Topologia de Rede Utilizada no Ambiente de Simulação.

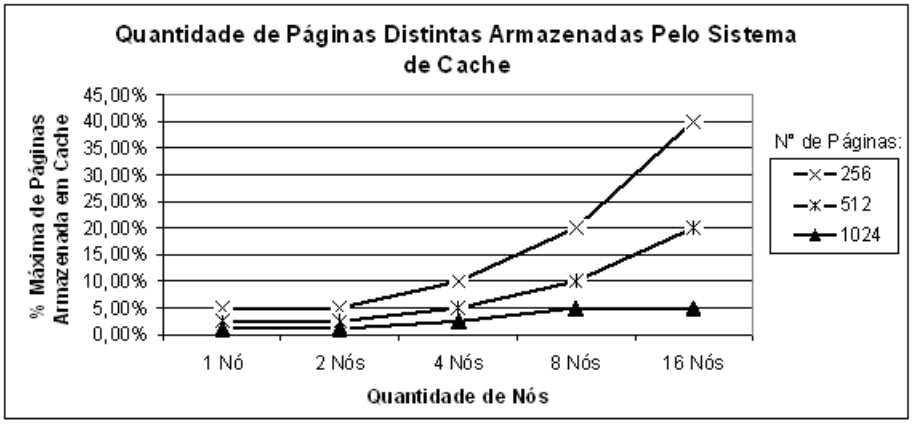

Figura 4. Quantidade de Páginas Distintas Armazenadas Pelo Sistema de *Cache*.

cache global. Quanto maior o tamanho da área global de *cache*, maior será a probabilidade de ocorrer um *cache global hit*. A quantidade de *hits* no sistema de cache é inversamente proporcional a quantidade de requisições submetidas ao servidor.

Os resultados preliminares obtidos demonstram que o melhor caso ocorre quando a quantidade de páginas distintas, armazenada no servidor central, é igual a 256. Cada nó consegue armazenar, individualmente, 5% do total de páginas (figura 4). Mesmo com 16 nós enviando requisições para o mesmo servidor, como não existe interação entre eles, 77.83% das requisições não são respondidas pelo sistema de *cache* local e, consequentemente, são enviadas ao servidor central. Enquanto que utilizando a política colaborativa de *cache* proposta, cada nó submete apenas 27.05% do total de requisições. Ou seja, dos 72,95% das requisições respondidas pelo sistema colaborativo de *cache*, aproximadamente 22,17% foram respondidas localmente, sem a necessidade de acesso a rede, e 50,78% foram respondidas dentro do *cluster*. O pior caso ocorre quando o número de páginas distintas presentes no servidor central é de 1024 páginas. A política colaborativa, para 16 nós, envia 55.75% das requisições, enquanto a política individual envia 93.77%. A piora no desempenho no sistema de *cache* é esperada, pois

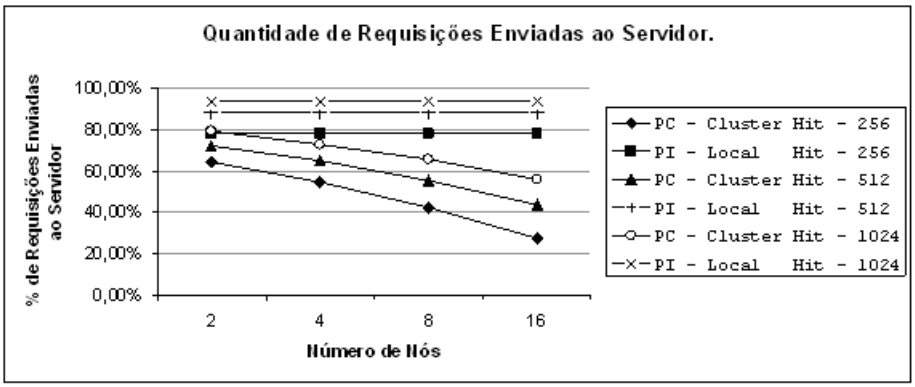

Figura 5. Quantidade de Requisições Enviadas ao Servidor.

existe um aumento no conjunto de páginas distintas acessadas. Esse aumento força uma maior rotatividade no acesso e diminui a quantidade de páginas repetidas referenciadas, o que agrava o problema conhecido como *cold-start misses* ou *first-reference misses* [Hennessy and Patterson 2007].

5. Conclusões

Este trabalho abordou o uso de *cache* colaborativo, em ambiente de rede *Ad Hoc*, como forma de reduzir as penalizações associadas as faltas de *cache* do tipo *cache local miss*. Diferente dos trabalhos referenciados na bibliografia, este propõe a criação de uma área global de *cache* distribuída destinada ao armazenamento de páginas de interesse de um determinado grupo de nós. O comportamento colaborativo estende-se para a decisão das páginas que devem ser armazenadas nesta área de *cache* global.

Resultados preliminares, do modelo proposto, demonstram uma redução em até 3 vezes no volume de requisições enviadas até o servidor central da rede. A redução no número de mensagens diminui a disputa pelo canal de comunicação e a probabilidade de colisões. Há uma economia de bateria, por parte dos nós que realizam o roteamento, e uma melhora na vazão da rede em decorrência da menor quantidade de retransmissões. A abordagem colaborativa também restringe a área de *broadcast*, o que poupa recursos dos demais nós da rede.

Como trabalhos futuros, pretende-se explorar outros mecanismos de clusterização que permitam identificar interesses comuns entre nós. Desta forma, os cluster poderiam ser formados por afinidades além da posição geográfica. Mecanismos eficientes para busca e localização de informações intra-cluster também estão sendo estudadas.

Referências

Arlitt, M., Cherkasova, L., Dilley, J., Friedrich, R., and Jin, T. (2000). Evaluating content management techniques for web proxy caches. *SIGMETRICS Perform. Eval. Rev.*, 27(4):3–11.

Bjornsson, M. E. and Shrira, L. (2002). Buddycache: high-performance object storage for collaborative strong-consistency applications in a wan. In *OOPSLA '02: Proceedings of the 17th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*, pages 26–39, New York, NY, USA. ACM Press.

Bloom, B. H. (1970). Space/time trade-offs in hash coding with allowable errors. *Commun. ACM*, 13(7):422–426.

Cao, G., Yi, L., and Das, C. R. (2004). Cooperative cache-based data access in ad hoc networks. *Computer*, 37:32 – 39.

Cao, J., Zhang, Y., Cao, G., and Xie, L. (2007). Data consistency for cooperative caching in mobile environments. *Computer*, 40(4):60–66.

Cao, J., Zhang, Y., Xie, L., and Cao, G. (2005). Consistency of cooperative caching in mobile peer-to-peer systems over manet. pages 573–579.

Chand, N., Joshi, R. C., and Misra, M. (2006). Efficient cooperative caching in ad hoc networks. In *COMSWARE*.

Chow, C.-Y., Leong, H. V., and A. Chan, A. (2004). Cache signatures for peer-to-peer cooperative caching in mobile environments. *18th International Conference on Advanced Information Networking and Applications*, 1:96–101.

Chow, C.-Y., Leong, H. V., and Chan, A. T. (2007). Grococa: group-based peer-to-peer cooperative caching in mobile environment. *IEEE Journal on Selected Areas in Communications*, 25:179–191.

Chow, C.-Y., Leong, H. V., and Chan, A. T. S. (2005). Distributed group-based cooperative caching in a mobile broadcast environment. In *MDM '05: Proceedings of the 6th international conference on Mobile data management*, pages 97–106, New York, NY, USA. ACM Press.

Dahlin, M., Wang, R., Anderson, T. E., and Patterson, D. A. (1994). Cooperative caching: Using remote client memory to improve file system performance. In *Operating Systems Design and Implementation*, pages 267–280.

Du, Y. and Gupta, S. K. S. (2005). Coop - a cooperative caching service in manets. In *ICAS-ICNS 2005. Joint International Conference on Autonomic and Autonomous Systems and International Conference on Networking and Services, 2005.*, pages 58–58. IEEE.

for WLAN Standards, I. . T. W. G. Ieee 802.11 wireless local area networks. IEEE 802.11 WIRELESS LOCAL AREA NETWORKS. Disponível em: <http://ieee802.org/11/>. Acessado em: 22/07/2008.

Gupta, P. and Kumar, P. (2000). The capacity of wireless networks. *IEEE Transactions on Information Theory*, 46:388–404.

Hennessy, J. L. and Patterson, D. A. (2007). *Computer Architecture: A Quantitative Approach*. Morgan Kaufmann Publishers Inc.

Huang, Y., Cao, J., and Jin, B. (2006). A predictive approach to achieving consistency in cooperative caching in manet. In *InfoScale '06: Proceedings of the 1st international conference on Scalable information systems*, page 50, New York, NY, USA. ACM Press.

Karedla, R., Love, J. S., and Wherry, B. G. (1994). Caching strategies to improve disk system performance. *Computer*, 27(3):38–46.

Li, W., Chan, E., and Chen, D. (2007). Energy-efficient cache replacement policies for cooperative caching in mobile ad hoc network. In *IEEE Wireless Communications and Networking Conference, WCNC.*, pages 3347 – 3352.

Perkins, C. E. and Royer, E. M. (1999). Ad-hoc on-demand distance vector routing. *wmcsa*, 00:90.

Podlipnig, S. and Boszormenyi, L. (2003). A survey of web cache replacement strategies. *ACM Comput. Surv.*, 35(4):374–398.

Ting, Y.-W. and Chang, Y.-K. (2007). A novel cooperative caching scheme for wireless ad hoc networks: Groupcaching. In IEEE, editor, *NAS 2007: International Conference on Networking, Architecture and Storage*, pages 62–68.

Yin, L. and Cao, G. (2006). Supporting cooperative caching in ad hoc networks. *IEEE Transactions on Mobile Computing*, 5:77– 89.

Zeng, X., Bagrodia, R., and Gerla, M. (1998). Glomosim: a library for parallel simulation of large-scale wireless networks. In *PADS'98: Proceedings of the 12th Workshop on Parallel and Distributed Simulations*.