Uma Arquitetura Híbrida para Buscas Complexas em Redes P2P

Péricles C. M. Lopes[1], **Ronaldo A. Ferreira**[1]

[1]Faculdade de Computação – Universidade Federal de Mato Grosso do Sul (UFMS)
Caixa Postal 549 – 79.070-900 – Campo Grande – MS – Brasil

pericles@facom.ufms.br,raf@facom.ufms.br

Resumo. *Apesar de inúmeros esforços nos últimos anos, buscas complexas eficientes em redes P2P de grande escala permanecem um problema em aberto e desafiador. Replicações massivas de dados e de mensagens de busca são duas estratégias comuns utilizadas para melhorar taxas de sucesso e tempos de resposta. Entretanto, estratégias de replicação pró-ativas podem gerar uma quantidade significativa de tráfego na rede se não forem tratadas com cuidado. Este trabalho propõe uma arquitetura híbrida que utiliza uma leve estrutura, em uma rede predominantemente não estruturada, para evitar replicações desnecessárias e acelerar a propagação de mensagens de busca em redes P2P. Resultados de simulação mostram que a arquitetura proposta possui desempenho superior à melhor solução conhecida em termos de número de mensagens, tempo de resposta, número de saltos e taxa de sucesso.*

Abstract. *Despite numerous efforts in the past few years, efficient complex queries in large-scale P2P networks remain an open and challenging problem. Massive data and query replications are two popular techniques used to improve success rates and response times. However, proactive replication strategies may lead to large amounts of traffic and low efficiency if not handled with care. This paper presents a hybrid architecture that relies on a lightweight network structure, on a predominantly unstructured network, to avoid unnecessary query replication and to speed up query propagation in P2P networks. Simulation results show that the proposed architecture outperforms the best known solution in number of messages, response time, number of hops, and success rate.*

1. Introdução

Nos últimos anos, sistemas par-a-par (P2P - *peer-to-peer*) se tornaram um poderoso paradigma de redes que permite o compartilhamento de vários recursos da Internet de maneira totalmente distribuída. Mais importante, este novo paradigma deu vida a várias outras novas aplicações, tais como mensagens instantâneas, voz sobre IP (Skype), distribuição de vídeos, computação distribuída (SETI@Home) e muitas outras. Além disso, usuários finais se transformaram em grandes produtores de informações que necessitam ser compartilhadas com o resto do mundo. Em particular, este último fato é mais evidente se considerarmos o aumento no número de blogs pessoais, redes sociais e Wikis na Internet. Para manter essas fontes de informações, os usuários ainda necessitam de infraestruturas centralizadas [Wikimedia Foundation Reports 2010], normalmente mantidas por empresas privadas. Uma solução P2P descentralizada para esse problema ainda não é possível porque, apesar

de inúmeros esforços, buscas complexas em redes P2P ainda podem ser consideradas um problema em aberto e desafiador. Exemplos de buscas complexas envolvem expressões regulares, múltiplas palavras, buscas semânticas com classificação, etc.

De maneira genérica, o propósito de uma aplicação P2P é o compartilhamento de algum tipo de recurso entre os participantes da rede. Exemplos de recurso podem ser vídeos, músicas, jogos eletrônicos, programas de computador, dentre outros. Para requisitar um determinado dado de interesse, um par deve executar algum algoritmo de busca. Isso é normalmente implementado por propagação de mensagens de busca para os demais participantes. Ao receber uma mensagem de busca, o par caso possua o item procurado, envia uma mensagem de resposta diretamente ao nó solicitante. A semântica de uma mensagem de busca varia de acordo com a aplicação e também de acordo com as facilidades oferecidas pela estrutura da rede.

O problema de busca em redes P2P estruturadas foi resolvido de forma elegante e eficiente utilizando-se tabelas de dispersão distribuídas (DHT - *Distributed Hash Table*) [Stoica et al. 2001]. Entretanto, o custo para se manter uma topologia específica, em que os pares devem manter muitos ponteiros para outros pares em uma população dinâmica, é muitas vezes uma fonte de fortes críticas. Além disso, a semântica da busca é muito limitada, já que um usuário deve saber o nome exato do objeto para determinar o par responsável por manter a réplica do objeto ou sua referência. Alguns tipos de buscas complexas podem ser implementados mas com um custo bastante alto e algumas vezes proibitivo.

Buscas exaustivas foram introduzidas recentemente para redes P2P [Ferreira et al. 2005, Terpstra et al. 2007] como um método de busca efetivo e confiável. Nesse tipo de busca, uma mensagem é avaliada em todos os dados do par que receber a mensagem de busca e cada par é livre para escolher o seu algoritmo de busca local. A semântica da busca é bem mais rica, em comparação com as buscas baseadas em *hashing*, já que algoritmos sofisticados podem ser executados localmente em cada par. Entretanto, as soluções mais eficientes se baseiam em caminhadas aleatórias [Ferreira et al. 2005, Ioannidis and Marbach 2009], ou suas variantes com vários fatores de ramificação [Terpstra et al. 2007], para prover garantias probabilísticas. Apesar da eficiência em algumas aplicações, sabe-se que caminhada aleatória é um método de busca "cego" que pode visitar o mesmo par mais de uma vez, gerando sobrecarga desnecessária e que não garante a cobertura da rede, a não ser que um número de mensagens muito grande seja gerado. Além disso, essas soluções replicam uma grande quantidade de dados em pares aleatórios com o objetivo de reduzir o tempo de resposta e aumentar a taxa de sucesso.

Neste artigo, com o objetivo de se explorar as melhores características tanto do modelo estruturado como do modelo não estruturado, é proposto e avaliado SplitQuest, um protocolo de busca exaustiva que se baseia em uma leve estrutura para evitar duplicações desnecessárias de mensagens de buscas e garantir que todo conteúdo disponível seja avaliado.

O restante do trabalho está organizado como se segue. A Seção 2 apresenta os algoritmos de busca, replicação e atribuição de identificadores de SplitQuest. É também apresentada uma análise matemática correspondente ao número máximo de saltos que uma mensagem de busca pode propagar. Na Seção 3, são apresentados os diferentes cenários

utilizados nos experimentos e os resultados das simulações para SplitQuest, contrastando seu desempenho com o BubbleStorm – a melhor solução conhecida para buscas exaustivas em redes P2P. A Seção 4 mostra os trabalhos relacionados e a Seção 5 conclui este trabalho.

2. O Protocolo SplitQuest

SplitQuest evita a sobrecarga desnecessária de métodos de buscas baseados em caminhadas aleatórias (e suas variações) com uma estrutura híbrida que permite que as mensagens de buscas sejam direcionadas para partes específicas e distintas da rede. Em adição a conexões aleatórias, normalmente presentes em redes não estruturadas, SplitQuest dispõe os pares em um anel, ou seja, cada par possui um predecessor e um sucessor identificados com valores no intervalo $[0, 1]$. Além disso, os pares armazenam localmente os identificadores (IDs) de seus vizinhos. A idéia de se utilizar um anel pode parecer semelhante às redes estruturadas, porém como será descrito nas próximas seções, as abordagens de SplitQuest para atribuição de identificadores, entrada de pares, replicação de dados e propagação de mensagens de buscas são bem diferentes. As conexões entre os pares ainda são aleatórias, não existe um roteamento bem definido e as mensagens de buscas são propagadas de maneira aleatória e livre pela rede. Mais importante ainda é que SplitQuest permite buscas de maior riqueza semântica, pois sua busca não está restrita a casamento exato de identificadores, já que os pares são livres para executar localmente os algoritmos de buscas que acharem mais conveniente.

2.1. Replicação de Índices

As hipóteses básicas do protocolo SplitQuest são que os pares estão uniformemente distribuídos no intervalo $[0, 1]$ (Seção 2.3 mostra como se pode obter essa distribuição) e que cada par é capaz de estimar o tamanho da rede (n). Estimativas de tamanho da rede são problemas bem estudados e que podem ser solucionados por diferentes abordagens como descrito em [Terpstra et al. 2007] e nas referências que lá estão.

A replicação de índices em SplitQuest é realizada em um conjunto de pares localizados em um subintervalo contíguo do intervalo $[0, 1]$. Quando pares são distribuídos de forma uniforme por todo intervalo, o número esperado de pares em um subintervalo contíguo é proporcional ao tamanho do subintervalo. Consequentemente, o problema básico neste caso é determinar o tamanho do subintervalo, ou mais especificamente, o tamanho do grupo de replicação que minimiza a replicação de índices e mensagens de buscas propagadas. Quando o valor de n é conhecido, o tamanho do subintervalo com número esperado de pares igual a d é dado por $\frac{d}{n}$. Seja então d o tamanho do grupo e q o número de grupos na rede. O número total de mensagens de replicação e busca de um objeto, $M = q + d$, é dado por:

$$M = \frac{n}{d} + d \tag{1}$$

É fácil verificar que d igual a $\sqrt{n}$ minimiza M. Quando mensagens de buscas e dados possuem tamanhos distintos, por exemplo b_1 e b_2 respectivamente, o valor de d que minimiza o total de *bytes* é dado por $\sqrt{\frac{b_1}{b_2}n}$. A Figura 1 ilustra como os pares são organizados em grupos de replicação no intervalo $[0, 1]$.

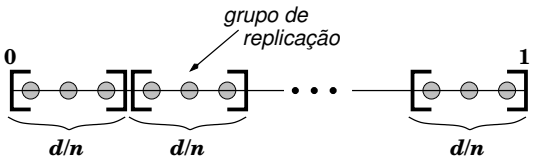


Figura 1. Grupos de replicação em SplitQuest.

Grupos em SplitQuest são numerados de 1 a $\lceil \frac{n}{d} \rceil$ e o grupo de um par pode ser computado pela fórmula $\lceil \frac{ID \times n}{d} \rceil$, sendo que o identificador do par está no intervalo $[0, 1]$. Para instalar referências para um objeto, um par i envia uma mensagem de instalação para seus vizinhos imediatos, pares com identificadores menores e maiores que i, se eles pertencerem ao mesmo grupo. Os pares conectados diretamente a i verificam e repassam as mensagens recebidas se os seus vizinhos também estão no mesmo grupo (mensagens não são repassadas para pares já visitados). O processo é repetido até que todos os pares do grupo recebam uma cópia da referência.

2.2. Algoritmo de Busca

Em SplitQuest, pares replicam seus conteúdos em outros pares que estão localizados em um mesmo grupo. Como resultado, mensagens de buscas relacionadas ao conteúdo de um grupo podem ser processadas por qualquer membro do grupo. Essa estratégia de replicação simplifica consideravelmente a propagação de mensagens de busca, na medida em que uma busca certamente será respondida, de forma positiva ou negativa, desde que essa mensagem atinja qualquer par no grupo alvo. Portanto, o algoritmo de busca deve garantir que cada grupo seja visitado pelo menos uma vez e de preferência não mais que uma vez. Para que isso seja possível, SplitQuest utiliza dois artifícios: busca por intervalo e atalhos para grupos.

Busca por intervalo significa que cada mensagem de busca contém um intervalo $[X, Y]$ que a busca deve cobrir. Quando um par i recebe uma mensagem com o intervalo $[X, Y]$, ele inspeciona suas conexões e determina quais vizinhos estão contidos no intervalo recebido. Esses vizinhos são possíveis candidatos para o repasse dessa mensagem de busca. Em seguida, o par i computa os identificadores de grupos desses possíveis candidatos e mantém apenas um par para cada grupo (caso exista mais que uma possibilidade para um mesmo grupo, a escolha é feita de maneira aleatória). Dependendo dos grupos selecionados, o par i decompõe o intervalo inicial em subintervalos e envia mensagens de busca contendo esses subintervalos para cada um dos pares selecionados.

A Figura 2 mostra um exemplo simples de propagação de uma mensagem de busca. Apenas um pequeno conjunto de pares e conexões é mostrado. A busca é iniciada em um par **a** que está conectado aos pares **e**, **b** e **g**. O par **a** é inicialmente responsável por cobrir todo o intervalo de $[0, 1]$. Assim, este par cria três mensagens com intervalos $[0, x_4]$, $[x_5, x_8]$ e $[x_8, 1]$ e as envia para os pares **b**, **e** e **g** respectivamente. O par **b** envia uma mensagem para **c** com intervalo $[0, x_2]$ e o par **e** envia uma mensagem para **f** com intervalo $[x_6, x_8]$. Finalmente, o par **c** envia uma mensagem para **d** com intervalo $[0, x1]$ e os pares **b** e **f** são responsáveis por realizar a cobertura dos subintervalos restantes ($[x_2, x_3]$ e $[x_6, x_7]$). Neste exemplo, são representadas apenas as conexões utilizadas na propagação da mensagem de busca, cada par pode possuir várias outras conexões aleatórias. Além disso, SplitQuest

pode fazer ajustes na porcentagem de pares que irão receber uma determinada mensagem, visando reduzir o número de mensagens inseridas na rede.

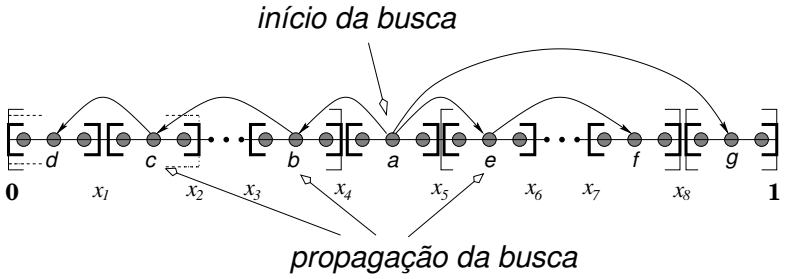


Figura 2. Propagação de uma mensagem de busca em SplitQuest.

Outro artifício utilizado por SplitQuest para melhorar o desempenho das mensagens de buscas são os atalhos para grupos. Os pares constroem atalhos (conexões diretas para outros pares) para os dois grupos adjacentes aos grupos a que eles pertencem. Esses atalhos garantem que existe sempre um caminho para um grupo, ou seja, um caminho que a cada novo salto levará a mensagem de busca para cobrir um novo grupo, evitando que uma consulta seja realizada várias vezes internamente em um mesmo grupo. Um par com identificador w pode construir atalhos para grupos adjacentes por conexões para pares com identificadores $w + \frac{1}{d}$ e $w - \frac{1}{d}$, ou para pares com identificadores mais próximos possíveis a esses valores, caso o par com o valor exato de identificação calculado não esteja presente na rede.

2.3. Atribuição de Identificadores

Uma questão-chave em SplitQuest é como os identificadores dos pares são gerados. Os identificadores dos pares devem ser distribuídos uniformemente no espaço de identificadores e os pares não devem depender de um protocolo de roteamento complexo para entrar na rede. Abordagens recentes de sistemas DHT se baseiam em funções de dispersão para gerar identificadores de forma uniforme e distribuída. Essa abordagem, entretanto, requer um protocolo eficiente de roteamento para a entrada de pares na rede. Além disso, cada par deve manter conexões para locais específicos da rede a fim de garantir limites superiores no roteamento de mensagens de busca. Em Chord [Stoica et al. 2001], por exemplo, os pares devem manter um conjunto de $O(\log n)$ conexões para um limite superior de $O(\log n)$ saltos no roteamento.

SplitQuest utiliza uma atribuição de identificadores totalmente diferente. A idéia básica é que um par seleciona dois outros pares adjacentes, mais especificamente um intervalo no anel e calcula o seu identificador como sendo o ponto médio entre os dois pares selecionados. A chave para uma distribuição uniforme de identificadores está em como o intervalo é selecionado. Uma idéia inicial poderia sugerir a seleção aleatória e uniforme de um intervalo e simplesmente dividi-lo. Entretanto, esta idéia conduz para uma distribuição muito desbalanceada de identificadores. A atribuição de identificadores de SplitQuest se baseia em resultados obtidos por [Naor and Wieder 2007] e também pela extrapolação do "poder de duas escolhas" [Mitzenmacher 2001] para k-escolhas e apresenta um algoritmo simples e eficiente.

O algoritmo desenvolvido e analisado por Naor (redução à metade) é descrito a seguir:

1. Escolha $t \log n$ pontos aleatórios no intervalo de $[0, 1]$, em que t é uma constante e selecione os segmentos em que esses pontos estão dispostos.
2. O identificador do par é o ponto médio do maior segmento selecionado no passo anterior.

O algoritmo utilizado neste trabalho é uma variação da abordagem proposta em [Naor and Wieder 2007], de forma a não alterar o resultado final. Neste caso, o primeiro passo do algoritmo é alterado de forma que um par que deseja se conectar à rede requisita a um *proxy* (um par que já está na rede) para selecionar k intervalos de maneira aleatória e uniforme. Mais especificamente, ele seleciona k pares e os maiores intervalos adjacentes a esses. O identificador do par entrante é o ponto médio do maior segmento. Um par pode colher amostras uniformes da rede utilizando algoritmos descritos em [Awan et al. 2006]. Observe que ao selecionar um intervalo, o par entrante recebe os dois pares que serão os seus vizinhos.

A Figura 3(a) mostra o quão eficiente é o algoritmo proposto para atribuição de identificadores no intervalo $[0, 1]$. Ela representa o número de pares em cada grupo da rede com população de 100 mil pares e tamanho esperado de grupos de $\sqrt{n} \approx 316$. O número de pares em cada grupo é ordenado em ordem crescente nas duas situações. Como pode ser observado, o algoritmo é mais eficiente que uma simples atribuição de identificadores por uma função de dispersão. Todos os tamanhos de grupos apresentados diferem em no máximo um desvio padrão do valor esperado. Os endereços IPs foram coletados pelo projeto "Route Views" da Universidade do Oregon [Route Views Project] e os portos foram fixados com valores constantes (1214) usados pelo Kazaa.

2.4. Análise do Protocolo

A estrutura formada pelos pares e suas conexões em SplitQuest pode ser vista como um grafo aleatório. O conjunto de nós com identificadores em um intervalo específico do anel virtual é um grupo de replicação e em SplitQuest, grupos de replicação são visitados apenas uma vez. Se forem consideradas as conexões de grupos, montadas como conexões aleatórias durante a busca, pode-se modelar a propagação dessas mensagens de buscas para os diferentes grupos como uma difusão (*broadcast*) em uma árvore aleatória. Em cada passo do algoritmo, os grupos são particionados em função das conexões do par e do intervalo que este deve cobrir. Considere G o número de grupos a serem visitados e g_k o conjunto de grupos que cobrem intervalos não sobrepostos. Inicialmente, G é igual ao número total de grupos.

Quando uma busca se inicia em um par i, as mensagens de busca devem visitar um conjunto G de grupos disjuntos. O conjunto é dividido em R subgrupos, que dependem das conexões de i, com distribuição dada por $P(R = r) = p_r$, em que p_r é a distribuição de probabilidade em $\{1, 2, \ldots\}$. Condicionado ao evento $\{R = r\}$, para $1 \leq k \leq r$, um grupo está no k-ésimo subconjunto com probabilidade $V_{k,r}$, em que $V_r = (V_{k,r}; 1 \leq k \leq r)$ é um vetor de probabilidade em $\{1, \ldots, r\}$. Se N_k é a cardinalidade do k-ésimo subconjunto, então, condicionado ao evento $\{R = r\}$ e nas variáveis aleatórias $V_{1,r}, \ldots, V_{r,r}$, a distribuição do vetor $(N_1, \ldots, N_r)$ é multinomial com parâmetros $G - 1$ e $(V_{1,r}, \ldots, V_{r,r})$ [Mohamed and Robert 2005].

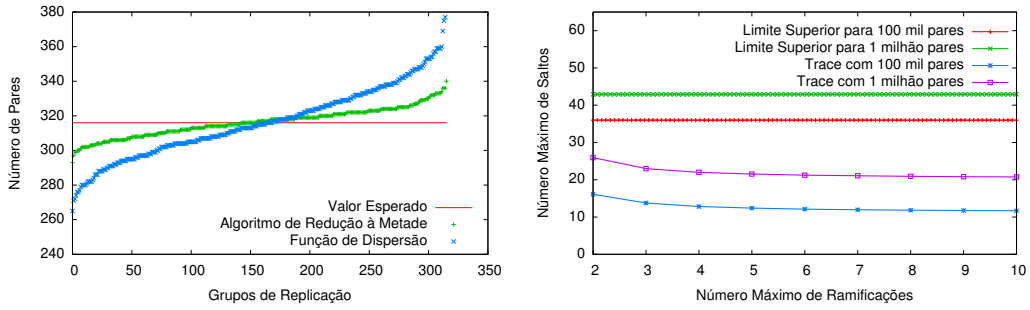
A análise da altura (H_G) da árvore aleatória para um distribuição geral de R é complexa e é geralmente obtida por técnicas de análise complexa. Um limite superior mais

conservador pode ser obtido quando r pertence ao conjunto $\{1, 2\}$, o que significa que temos no máximo duas ramificações em cada passo da difusão. Devroye [Devroye 1998] mostra que, quando r pertence a $\{1, 2\}$, a altura da árvore é limitada por $O(\log G)$, mais especificamente Devroye mostra que

$$\lim_{G \to \infty} P\{H_G > c \log G\} = 0$$

em que c é uma constante. Um resultado mais forte é derivado em [Devroye 1986], em que é mostrado que $H_G / \log G \to 4.31106\ldots$ em probabilidade.

É fácil de se verificar que o limite superior derivado acima é também um limite superior para o maior número de saltos que uma mensagem de busca é propagada em SplitQuest. Como SplitQuest pode enviar uma mensagem de busca para mais de dois grupos, o limite superior é, de fato, bastante conservador. A Figura 3(b) mostra o número máximo de saltos que uma busca se propaga desde de um par inicial em redes com 100 mil e 1 milhão de pares e quando o número máximo de ramificações é incrementado de 2 a 10. A figura mostra os resultados para uma de três diferentes topologias que serão utilizadas nos experimentos da Seção 3, os resultados para as outras duas topologias são semelhantes. As características dessas topologias serão discutidas na Seção 3. Informações mais detalhadas sobre a análise do algoritmo e dos experimentos podem ser encontradas em [Lopes and Ferreira 2009].



(a) Atribuição de IDs (b) Limite Superior

Figura 3. a) Número de pares em grupos de replicação para tamanho de rede de 100 mil pares. b) Distância máxima de propagação de uma mensagem de busca com diferentes ramificações. As linhas superiores são limites teóricos para 1 milhão e 100 mil pares respectivamente.

3. Avaliações Experimentais

Nesta seção, são apresentados os resultados de simulações para avaliar vários aspectos de desempenho do protocolo proposto. Para avaliar o desempenho de SplitQuest, foi desenvolvido um simulador na linguagem C++ que utiliza o modelo de simulação de eventos discretos. Uma simples comparação entre diferentes sistemas P2P é difícil em função das diferentes estruturas e algoritmos desenvolvidos. Redes totalmente estruturadas apresentam protocolos específicos de entrada, saída e roteamento de mensagens na rede. Além disso, ao melhor de nosso conhecimento, não apresentam abordagens para buscas complexas em redes P2P. Sistemas baseados em redes não estruturadas também nem sempre oferecem suporte a esse tipo de busca. Uma abordagem mais próxima do objetivo proposto por SplitQuest é o BubbleStorm, apresentado em [Terpstra et al. 2007] e isso justifica sua

utilização na comparação com o desempenho de SplitQuest. Todas as simulações foram executadas em um servidor com 2 processadores AMD *Quad Core Opteron* e 16GB de memória principal rodando Linux.

3.1. Métricas

A avaliação de todas as métricas possíveis de sistemas P2P reais é considerada uma tarefa difícil e até mesmo desnecessária. Existem várias métricas que podem ser utilizadas, como o consumo da largura de banda, carga de CPU, armazenamento, atrasos da rede, probabilidade de sucesso, etc. Os experimentos se concentram em sua maioria em três métricas de desempenho bem aceitas: taxa de sucesso, latência e número de mensagens.

3.2. Configuração dos Experimentos

Como mencionado anteriormente, o cenário utilizado é semelhante ao cenário proposto por [Terpstra et al. 2007], mas também são avaliados novos parâmetros e diferentes cenários para distinguir claramente as duas abordagens. A rede é formada por pares posicionados em diferentes localidades, escolhidas de maneira uniforme e aleatória, na superfície da Terra. Os atrasos de propagação das mensagens são calculados como o tempo que um sinal leva, viajando à velocidade da luz, para cobrir a distância entre par origem e destino. A distância é calculada utilizando as latitudes e longitudes dos pares. As capacidades de *download* e *upload*, que determinam o atraso da transmissão, estão diretamente relacionadas ao grau dos pares e são ajustadas de acordo com a topologia utilizada. As capacidades serão discutidas posteriormente. Em adição à propagação e o atraso de transmissão, também é introduzido um atraso como uma variável aleatória com distribuição normal entre 0 e 10 *ms* em cada transmissão de mensagem para contabilizar os atrasos de processamento.

1) Topologia da Rede

Um importante componente de uma rede P2P consiste em saber como os pares estão conectados entre eles para formar a topologia da rede. Não há consenso na literatura sobre qual topologia melhor representa uma rede P2P, alguns estudos afirmam ser um grafo aleatório com distribuição *power-law* [Adamic et al. 2001], mas outros discordam [Stutzbach et al. 2008]. Para ser o mais compreensível possível, as simulações foram feitas com três diferentes topologias e seis diferentes tamanhos de redes (10 mil, 100 mil, 250 mil, 500 mil, 750 mil e 1 milhão de pares). A primeira topologia é um grafo aleatório regular, em que cada par possui grau constante e se conecta aleatoriamente a um número fixo de outros pares, como descrito em [Terpstra et al. 2007]. As larguras de banda são as mesmas para todos os pares e iguais a 256 kB/s para *download* e 32 kB/s para *upload*.

Outra topologia utilizada corresponde a um traço de uma rede P2P obtida pelo rastreador de topologia Cruiser introduzido em [Stutzbach et al. 2008]. Na topologia de traço, a capacidade dos pares corresponde as mesmas apresentadas em [Terpstra et al. 2007]: pares com grau menor que 10 possuem largura de banda de *download* de 128 kB/s e 16 kB/s de *upload*; pares com grau entre 10 e 20 possuem largura de banda de *download* de 256 kB/s e de *upload* 32 kB/s; pares com grau entre 20 e 80 possuem capacidade de *download* e *upload* de 128 kB/s e finalmente, os pares com grau maior que 80 (super pares) possuem taxa de *download* e *upload* de 1.28 MB/s. Essas capacidades cobrem desde linhas ADSL de baixa-capacidade até altas velocidades de 10 Mbps em links dedicados. Mesmo que

essas larguras de banda não representem o estado atual das linhas ADSL, elas foram utilizadas para realizar a comparação com os resultados apresentados em [Terpstra et al. 2007]. Além disso, larguras de banda maiores não mudarão os resultados comparativos entre SplitQuest e BubbleStorm. Isso porque tanto em BubbleStorm quanto em SplitQuest mantemos as mesmas proporções de pares com larguras de banda pré-definidas e estabelecidas em [Terpstra et al. 2007].

A terceira topologia é um grafo aleatório com graus dos pares seguindo uma distribuição *power-law*. Grafos *Power-Law* foram amplamente utilizados para modelar topologias de redes. Em um grafo com esse tipo de distribuição, se os pares estiverem ordenados em ordem decrescente de seus graus, então o i^{esimo} par possui grau D/i^a, em que D é uma constante e a é um parâmetro fixado em 0.8. Este valor de a é comumente utilizado em estudos de avaliação de redes P2P. A topologia *power-law* foi gerada com o mesmo número de pares das topologias anteriores. O grau máximo de qualquer par é limitado a 800 e as larguras de banda dos pares são ajustadas de acordo com os graus e seguem as mesmas regras descritas acima para a topologia traço.

2) Cenários Dinâmicos

A natureza dinâmica dos sistemas P2P é considerada um importante parâmetro. Em sistemas envolvidos com publicação de conteúdo, esse parâmetro se torna mais importante para que se possa avaliar o impacto da variação das condições. Foram simulados dois cenários diferentes no que diz respeito ao dinamismo dos pares. O primeiro e mais simples cenário não possui eventos de saídas dos pares e a população se mantém constante durante todos os experimentos (estático). Este cenário simples é utilizado principalmente para contrastar a eficácia das duas técnicas na busca por objetos na rede. O segundo cenário considera tanto entrada quanto saída dos pares durante a publicação de conteúdo e propagação das mensagens de buscas (com *churn*). Entradas e saídas de pares são escalonadas em taxas semelhantes para manter a população próxima a um valor constante.

3) Aplicação Par-a-Par

Existem inúmeras aplicações P2P que podem se beneficiar de um substrato que ofereça buscas complexas tal como SplitQuest. Uma comparação correta entre diferentes protocolos, implica na utilização de um mesmo cenário de simulação. Assim, a aplicação utilizada neste trabalho se baseia nas descrições apresentadas em [Terpstra et al. 2007]. Foi simulada uma aplicação de Wiki em que pares publicadores injetam periodicamente novos artigos e outros pares realizam busca por esses artigos. Nesta aplicação, artigos são representados por um tipo de dado de 2060 *bytes* (2 kB para metadados mais 60 *bytes* do cabeçalho TCP/IP). Mensagens de buscas são representadas por tipos de dados de 160 *bytes* (100 *bytes* para os dados da busca mais 60 *bytes* do cabeçalho TCP/IP). A cada segundo, são inseridos 100 artigos em pares escolhidos de maneira aleatória na rede. Esses pares seguem os seus algoritmos de replicação e iniciam o processo de instalação de réplicas. No caso de SplitQuest, os pares tem que replicar os seus próprios conteúdos em seus grupos. No BubbleStorm, os pares disponibilizam seu conteúdo em pares no caminho de uma caminhada aleatória com múltiplas ramificações. Por questões de economia de memória durante as simulações, cada par representa seus próprios dados por vetores binários, em que cada valor binário representa a presença ou ausência de um dado específico.

Os tráfegos de dados e mensagens de buscas são iniciados assim que a rede atinge o tamanho esperado. Reserva-se um tempo para que a replicação seja feita e então as buscas se iniciam. Para cada artigo instalado, uma mensagem de busca é iniciada em um par escolhido aleatoriamente. Depois que a busca é iniciada, espera-se 140 segundos para analisar se o artigo procurado foi encontrado ou não.

3.3. Resultados

Nesta seção, são apresentados os resultados numéricos para comparações entre SplitQuest e BubbleStorm. São exibidos os resultados para as métricas definidas na Seção 3.1 sob três diferentes topologias e cenários. Em todas as figuras abaixo, a abscissa x representa diferentes tamanhos de redes e a ordenada y representa uma medida específica computada como média de 11 execuções independentes. Redes com diferentes tamanhos e topologias permitem comparações das duas abordagens com relação à escalabilidade e exploração da heterogeneidade dos pares. Salvo se explicitado, os parâmetros para SplitQuest são os mesmos descritos na Seção 2. Para todos os gráficos abaixo, foram calculados os intervalos de confiança para um grau de confiança de 95%. Entretanto, em situações em que os intervalos ficaram extremamente pequenos e de difícil visualização, eles não foram plotados.

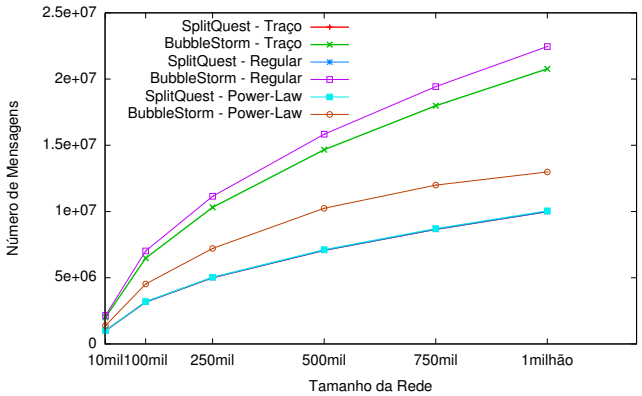


Figura 4. Número de mensagens para um cenário dinâmico.

Na Figura 4, são apresentados os resultados para o número de mensagens quando 5000 artigos são inseridos na rede. Para cada artigo, uma busca é iniciada 100 segundos depois de que o artigo é inserido. Este tempo é assim ajustado para permitir a instalação das réplicas na rede. Os resultados são analisados 140 segundos depois que se iniciam. Mensagens de buscas com sucesso que levam mais que 140 segundos não são contabilizadas e são consideradas como insucesso. Esse intervalo é importante para medir o tempo de resposta dos protocolos e mostrar o quão rápido eles são capazes de encontrar os objetos. Os resultados discutidos abaixo são para o cenário dinâmico (o cenário estático apresenta resultados semelhantes).

O número total de mensagens na Figura 4 corresponde a soma de mensagens de dados e mensagens de buscas durante todo o período de simulação. Pode-se perceber que, em todas as três topologias e tamanhos de redes, o número de mensagens para SplitQuest é quase o mesmo. Percebe-se ainda as três linhas como uma única na parte inferior da figura. Este comportamento é fácil de ser compreendido, já que SplitQuest utiliza o tamanho da rede como parâmetro para determinar o número de réplicas e mensagens de buscas.

Uma importante conclusão que pode ser tirada da figura é que o número de mensagens que SplitQuest gera é independente da topologia e de parâmetros particulares dos pares, como a distribuição dos graus. Pode-se perceber também, que o algoritmo de atribuição de identificadores, discutido na Seção 2.3, é capaz de distribuir os pares uniformemente pelo anel virtual, já que o número de mensagens para as três diferentes topologias permanece o mesmo.

A observação mais importante da Figura 4 é que SplitQuest supera o BubbleStorm em todas as três diferentes topologias. Os ganhos variam de 50% até 52% em uma topologia regular, de 53.5% até 55.5% na topologia de traço e de 22.5% até 30.5% em uma topologia *power-law*. Os maiores ganhos foram observados para redes maiores, o que torna os resultados mais significativos já que redes P2P possuem tamanho da ordem de 1 milhão de pares. Os menores ganhos, obtidos na topologia power-law, podem ser explicados de acordo com a forma que BubbleStorm calcula o número de réplicas (protocolo próprio $qd = c^2n$, em que q e d correspondem às mensagens de buscas e réplicas de dados estimadas pela rede, c um parâmetro de certeza e n o tamanho estimado da rede), o que resulta em um menor número de réplicas dependendo da distribuição do grau dos pares.

Um cenário estático é simplista e irrealista, mas permite que as idéias chaves dos algoritmos sejam avaliadas. A Figura 5(a) mostra que, em condições perfeitas de funcionamento, SplitQuest sempre alcança 100% de taxa de sucesso em todas as topologias e tamanhos de rede. BubbleStorm, que utiliza um método "cego" de busca e não oferece garantias de cobertura da rede, atinge 98% de taxa de sucesso. A taxa de sucesso do BubbleStorm pode ser incrementada pela mudança do parâmetro de certeza c. Entretanto, o número de mensagens aumentará consideravelmente, já que os valores de q e d aumentarão proporcionalmente a c^2.

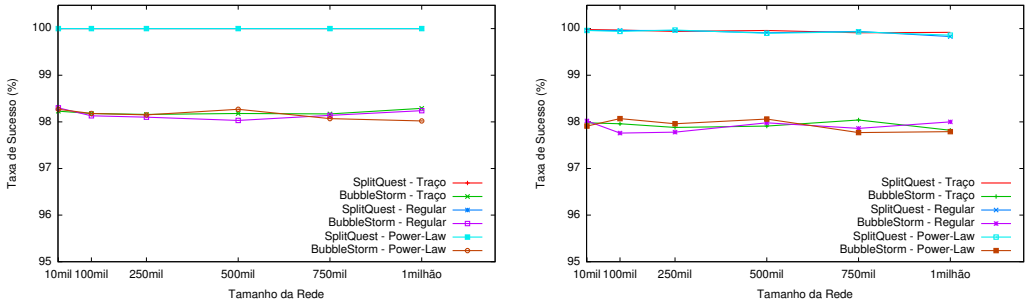
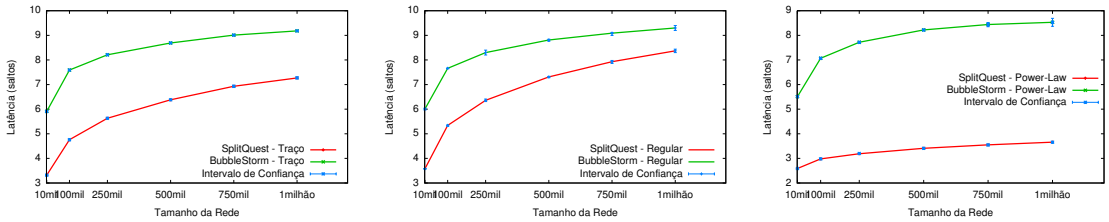


Figura 5. a) Taxa de sucesso com uma população estática de pares. b) Taxa de sucesso com uma população dinâmica de pares.

A taxa de sucesso em um cenário dinâmico e mais realista (com *churn*) pode ser observada na Figura 5(b). É possível também verificar que, para o SplitQuest, a taxa de sucesso decai aproximadamente 0.17% em comparação ao cenário estático, atingindo taxas de sucesso acima de 99.8%. Este decaimento pode ser explicado pelos pares que deixam a rede durante o momento de processamento dos dados e mensagens de buscas. Um par que recebe uma mensagem de replicação de dado deve distribuir a mensagem para outros pares no mesmo grupo. Caso esse par saia, pares ainda não cobertos talvez não recebam a mensagem. Além disso, um par que deixa a rede pode não repassar uma mensagem de busca para

outros pares no intervalo que essa deve cobrir. Embora o BubbleStorm replique mais dados e mensagens de buscas que o SplitQuest, ele não atinge taxa de sucesso superior a 98.5%. Este resultado se deve principalmente ao método de busca "cego" que não explora todas as possibilidades de pares. Os resultados mostram ainda que SplitQuest é resiliente a eventos de entradas e saídas dos pares e mantém altas taxas de sucesso mesmo sob condições de *churn*.

Uma das principais características do BubbleStorm é a baixa latência na resolução de mensagens de buscas. O seu desempenho reduziu significativamente o tempo de resposta em comparação à primeira abordagem baseada em caminhadas aleatórias [Ferreira et al. 2005]. SplitQuest também é capaz de atingir tempos baixos de resolução, como pode ser visto na Figura 6. Esta redução na latência é possível porque SplitQuest explora agressivamente a heterogeneidade dos pares e espalha as mensagens mais rapidamente que o BubbleStorm. Pares com grande número de conexões são capazes de enviar mensagens de buscas para diversos grupos diferentes em um simples passo. Além disso, as buscas direcionadas evitam *loops* e cobrem rapidamente todos os grupos da rede. A capacidade em explorar a heterogeneidade dos pares é mais evidente quando se compara as latências da topologia regular com a topologia *power-law*. Visto que a topologia *power-law* possui pares com mais que 800 conexões, SplitQuest resolve uma mensagem de busca com 4 saltos a menos que na topologia regular.



(a) Traço (b) Regular (c) Power-law

Figura 6. Latência da primeira resposta de sucesso para uma mensagem de busca em cenário dinâmico.

Para todas as topologias e tamanhos de redes, SplitQuest apresenta resultados melhores em termos de tempo e número de saltos que o BubbleStorm. O ganho mais considerável é observado na topologia power-law, em que a latência é reduzida em até 59%. O pior dos casos acontece na topologia regular, porém o SplitQuest continua superando o BubbleStorm em aproximadamente 11%. Por questões de limitação de espaço, outros resultados não foram apresentados, mas eles podem ser encontrados em [Lopes and Ferreira 2009].

4. Trabalhos Relacionados

Ferreira *et al.* [Ferreira et al. 2005] apresenta a primeira solução escalável (sublinear) que fornece garantias probabilísticas em operações de busca. O protocolo se baseia em uma variante do paradoxo do aniversário e utiliza uma estratégia de replicação pró-ativa. Um par, em uma rede de tamanho n, instala referências para seus objetos em $O(\delta\sqrt{n})$ outros pares selecionados de maneira aleatória e uniforme na rede. Para selecionar amostras uniformes de pares, o protocolo se baseia em caminhadas aleatórias modificadas. Essa modificação visa garantir que pares com poucas conexões tenham a mesma probabilidade de serem selecionados na rede que os pares com elevado número de conexões. Uma busca é realizada

enviando-se $O(\delta\sqrt{n})$ mensagens para pares selecionados aleatoriamente. A intersecção dos dois conjuntos replicados forma a base da solução. Diferente valores da constante de multiplicação δ fornecem diferentes níveis de garantia de sucesso. O principal problema da técnica é a latência no tempo de resposta das mensagens de buscas em função da demora na propagação das caminhadas aleatórias.

Terpstra *et al.*[Terpstra et al. 2007] se baseia na idéia inicialmente desenvolvida em [Ferreira et al. 2005] e apresenta o BubbleStorm, uma técnica resiliente para buscas em redes P2P. BubbleStorm tem também suas bases no paradoxo do aniversário para fornecer garantias probabilísticas de sucesso. Porém, BubbleStorm aumenta significativamente o desempenho evitando as longas caminhadas aleatórias sequênciais permitindo que várias caminhadas aleatórias sejam feitas em paralelo. Uma distinção chave do BubbleStorm é o uso de um fator de replicação (s) que permite que uma caminhada aleatória se ramifique para s pares diferentes a cada passo. Esta característica permite que várias mensagens de buscas sejam feitas em paralelo, reduzindo a latência do tempo de resposta. Um problema dessa abordagem é que, para redes com grande variância no grau dos pares, o número de mensagens gerados pelo protocolo do BubbleStorm para mensagens de buscas e réplicas de dados aumenta rapidamente.

Mais recentemente, Luo *et al.* [Luo et al. 2008] propõe uma arquitetura híbrida P2P para realizar buscas exaustivas na rede. A topologia é um misto de DHT e redes não estruturadas. Pares estáveis, que permanecem por longos períodos na rede, formam o núcleo da topologia e são posicionados em uma DHT. Esse núcleo é responsável por obter informações globais da rede (tamanho) e dos pares (grau, número de arquivos compartilhados, tamanho dos conteúdos compartilhados), etc. O protocolo de busca utiliza os pares estáveis para obter amostras uniformes.

5. Conclusão

Neste artigo é apresentado SplitQuest, um novo protocolo de busca para redes P2P que se baseia em uma estrutura híbrida para se evitar replicações desnecessárias de mensagens de buscas e para garantir cobertura da rede. Em SplitQuest, os pares são organizados em grupos e cada par replica seu conteúdo em seu próprio grupo. Mensagens de buscas utilizam um intervalo do anel virtual para representar os grupos que um par precisa cobrir. Utilizando intervalos, SplitQuest faz com que uma mensagem de busca seja comparada diante de todos os dados da rede, deixa os pares livres para escolher o algoritmo local de busca que desejar e garante que cada grupo é visitado uma única vez. Essas características fazem com que SplitQuest alcance taxas de sucesso altas, baixas latências nas mensagens de buscas e um pequeno número de mensagens em comparação com outros algoritmos de replicação pró-ativos discutidos na literatura. Os resultados numéricos mostram que SplitQuest é capaz de buscar informações em uma rede P2P de maneira bastante eficiente e que, dessa forma, possui potencial para estimular o desenvolvimento de novas aplicações completamente distribuídas, em particular aplicações que demandam baixas quantidades de dados, tais como microblogs e wikis.

Agradecimentos

Este projeto foi parcialmente financiado pelo CNPq (Proc. 483929/2007-7 e Proc. 304974/2008-0) e pela Fundect (Proc. 23/200.134/2008).

Referências

Adamic, L., Lukose, R. M., Puniyani, A. R., and Huberman, A. B. (2001). Search in Power-Law Networks. *Physical Review E*, 64.

Awan, A., Ferreira, R. A., Jagannathan, S., and Grama, A. (2006). Distributed Uniform Sampling in Unstructured Peer-to-Peer Networks. In *Proceedings of The 39th IEEE Hawaii International Conference on Systems Sciences (HICSS 2006)*, Kauai, HI.

Devroye, L. (1986). A Note on the Height of Binary Search Trees. *Journal of ACM*, 33(3):489–498.

Devroye, L. (1998). Universal Limit Laws for Depths in Random Trees. *SIAM Journal on Computing*, 28(2):409–432.

Ferreira, R. A., Ramanathan, M. K., Awan, A., Grama, A., and Jagannathan, S. (2005). Search with Probabilistic Guarantees in Unstructured Peer-to-Peer Networks. In *Proceedings of the 2005 IEEE P2P (P2P'05)*, pages 165–172, Konstanz, Alemanha.

Ioannidis, S. and Marbach, P. (2009). Absence of Evidence as Evidence of Absence: A Simple Mechanism for Scalable P2P Search. In *Proceedings of the 2009 IEEE INFOCOM (INFOCOM'09)*, Rio de Janeiro, Brasil.

Lopes, P. and Ferreira, R. (2009). SplitQuest: Controlled and Exhaustive Search in Peer-to-Peer Networks. Relatório Técnico disponível em http://ndsg.dct.ufms.br/˜pericles/splitquest.pdf. Universidade Federal de Mato Grosso do Sul, Brasil.

Luo, X., Qin, Z., Han, J., and Chen, H. (2008). DHT-assisted Probabilistic Exhaustive Search in Unstructured P2P Networks. In *Proceedings of the 2008 IEEE IPDPS (IPDPS'08)*, pages 1–9, Miami, FL.

Mitzenmacher, M. (2001). The Power of Two Choices in Randomized Load Balancing. *IEEE Trans. Parallel Distrib. Syst.*, 12(10):1094–1104.

Mohamed, H. and Robert, P. (2005). A Probabilistic Analysis of Some Tree Algorithms. *The Annals of Applied Probability*, 15(4):2445–2471.

Naor, M. and Wieder, U. (2007). Novel Architectures for P2P Applications: The Continuous-Discrete Approach. *ACM Transactions on Algorithms (*Eletronic Edition*)*, 3(3).

Route Views Project. University of Oregon Route Views Project.

Stoica, I., Morris, R., Karger, D., Kaashoek, F., and Balakrishnan, H. (2001). Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications. In *Proceedings of the 2001 ACM SIGCOMM (SIGCOMM'01)*, pages 149–160, San Diego, CA.

Stutzbach, D., Rejaie, R., and Sen, S. (2008). Characterizing Unstructured Overlay Topologies in Modern P2P File-Sharing Systems. *IEEE/ACM Transactions on Networks (*Eletronic Edition*)*, 16(2).

Terpstra, W. W., Kangasharju, J., Leng, C., and Buchmann, A. P. (2007). BubbleStorm: Resilient, Probabilistic, and Exhaustive Peer-to-Peer Search. In *Proceedings of the 2007 ACM SIGCOMM (SIGCOMM'07)*, pages 49–60, Tóquio, Japão.

Wikimedia Foundation Reports (2010). http://wikimediafoundation.org/wiki/.