Configuração Automática de Parâmetros em Redes Par-a-Par Sobre Redes Ad Hoc*

Diego Neves da Hora[1], **Daniel Fernandes Macedo**[2†], **José Marcos S. Nogueira**[1]

{dnhora,jmarcos}@dcc.ufmg.br,Daniel.Macedo@rp.lip6.fr

[1]Departamento de Ciência da Computação
Universidade Federal de Minas Gerais – Belo Horizonte, Brasil

[2]Laboratoire d'Informatique Paris 6
Université Pierre et Marie Curie-Paris6 – Paris, França

Resumo. *Aplicações par-a-par (P2P) devem ser configuradas tendo em vista o cenário onde serão executadas. A determinação da configuração ideal é custosa, pois requer uma caracterização da rede. Assim, as redes P2P normalmente utilizam uma configuração genérica, que provê um desempenho reduzido em relação à melhor configuração manual para cada cenário. Neste trabalho propomos algoritmos adaptativos que reconfiguram redes par-a-par não estruturadas em tempo de execução. Estes algoritmos, desenvolvidos para redes móveis ad hoc, reconfiguram o protocolo P2P de acordo com a carga da rede. Caracterizamos a carga de MANETs e desenvolvemos um algoritmo para estimar essa carga. Em seguida, propomos algoritmos de reconfiguração para redes par-a-par não estruturadas. Resultados de simulação mostram que o desempenho das abordagens adaptativas aproxima-se daquele obtido pela melhor configuração manual dos parâmetros.*

Abstract. *Peer-to-peer (P2P) applications must be configured for each scenario in which they are executed. The determination of the ideal configuration is costly, as it requires a complete characterization of the network. Thus, P2P networks usually rely on a generic configuration, which provides a reduced performance when compared to the best manual configuration. In this work we propose adaptive algorithms that reconfigure the Gnutella P2P protocol in execution time. Those algorithms, developed for mobile ad hoc networks (MANETs), reconfigure the protocol based on the network load. We characterize the load of MANETs and develop algorithms to sense the state of the network. Next we propose reconfiguration algorithms for Gnutella. Simulation results show that, on all scenarios, the performance of the adaptive solutions is as good as the performance of the best manual configuration.*

1. Introdução

Por serem independentes de infra-estrutura prévia, as redes móveis ad hoc (MANETs) possibilitam a pronta execução de aplicações de apoio a resgate em situações de

*O presente trabalho foi realizado com apoio do CNPq, uma entidade do Governo Brasileiro voltada ao desenvolvimento científico e tecnológico. Processo 55.2111/2002-3.

†Bolsista do CNPq Brasil.

desastre e de troca de informações em campos de batalha, que de outra maneira encontrariam muitos obstáculos para sua execução. [Borg 2003, Haas et al. 2002] Devido às características desses tipos de cenários, é desaconselhável o uso de arquiteturas cliente-servidor, uma vez que o servidor se torna um ponto de vulnerabilidade na rede.

As redes par-a-par (P2P) possuem uma arquitetura distribuída e são utilizadas para o compartilhamento de dados. Os participantes dessas redes são iguais em termos de funcionalidade, podendo tanto armazenar dados quanto gerar requisições. As redes par-a-par são ideais para o compartilhamento de dados em redes móveis ad hoc [Borg 2003, Talia and Trunfio 2003], devido à grande taxa de falha dos nós e dos enlaces. Nessas situações, uma solução distribuída é recomendável sobre a solução cliente-servidor.

Aplicações par-a-par possuem parâmetros que devem ser ajustados de acordo com a as características da rede física. Em uma rede Gnutella, por exemplo, devemos configurar o número de vizinhos de cada nó e o tempo de vida (TTL) das mensagens de pesquisa. O ajuste manual dos parâmetros é uma tarefa difícil e muitas vezes frustrante. Esse requer um estudo antes da implantação da rede, que deve levar em conta características como número de nós e distribuição do conteúdo na rede par-a-par. Por ser bastante complexa, essa abordagem é sujeita a erros e pode demandar um grande tempo. Uma alternativa é o uso de uma configuração genérica. No entanto, tal configuração deverá ser conservadora a fim de ser aplicável a uma grande diversidade de redes, e apresentará um desempenho inferior ao obtido pelo ajuste manual dos parâmetros da rede P2P.

Neste trabalho propomos algoritmos adaptativos que reconfiguram redes P2P não estruturadas em tempo de execução. Estes algoritmos, desenvolvidos para MANETs, ajustam os parâmetros da rede P2P de acordo com a carga da rede. Desta forma, os algoritmos propostos reduzem a quantidade de parâmetros a serem configurados pelo administrador, reduzindo o custo de implantação da rede P2P. Além disso, por serem adaptativos, os algoritmos possibilitam que a rede se adapte, em tempo real, a mudanças no tráfego e na topologia física. Utilizamos a carga da rede como entrada para os algoritmos, pois estudos anteriores mostraram que as colisões e o descarte de pacotes ocasionados por uma alta carga na rede impactam decisivamente o desempenho de redes P2P em MANETs [Oliveira et al. 2005a]. Esse trabalho é complementar ao de da Hora et al. em [da Hora et al. 2007], no qual propomos algoritmos adaptativos para redes P2P não estruturadas. Entretanto, neste trabalho ajustamos um conjunto diferente de parâmetros.

Algoritmos adaptativos provêem *autoconfiguração* a partir de um algoritmo de *autoconsciência*. Desenvolvemos, em uma primeira instância, algoritmos que identificam o estado da rede, que é definido em termos de métricas de desempenho da rede P2P e da rede ad hoc. Em seguida, desenvolvemos algoritmos que reconfiguram a aplicação Gnutella baseado no estado da rede. Estes algoritmos atuam em tempo de execução, modificando os parâmetros da aplicação para que a rede opere sob uma carga aceitável.

Propomos e avaliamos três algoritmos adaptativos através de simulações em um cenário onde variamos a carga em uma rede P2P não estruturada, Gnutella, e comparamos os resultados com diferentes configurações fixas de parâmetros. Os resultados mostram que os algoritmos se adaptam à carga da rede física, e o desempenho da rede P2P se aproxima ou é superior ao obtido pela melhor configuração manual dos parâmetros.

O restante deste trabalho está organizado da seguinte maneira. A Seção 2 mostra os trabalhos relacionados. A Seção 3 caracteriza e propõe um algoritmo para detecção de congestionamento, enquanto a Seção 4 apresenta os algoritmos adaptativos propostos. A

Seção 5 apresenta a avaliação dos mesmos e discute os resultados de simulação. A Seção 6 apresenta as conclusões e trabalhos futuros.

2. Trabalhos relacionados

A integração de MANETs e redes P2P é um tópico extensivamente pesquisado. Oliveira et al. avaliaram o desempenho de uma rede não estruturada usando diversos protocolos de roteamento (DSR, AODV, DSDV) [Oliveira et al. 2005b]. Franciscani et al. propuseram algoritmos de configuração que ajustam a topologia da rede P2P à topologia da rede ad hoc [Franciscani et al. 2005]. Conti et al. propuseram modificações ao Gnutella, utilizando o uma abordagem cross-layer, para otimizar o protocolo sobre MANETs [Conti et al. 2005].

Outros trabalhos propuseram modelos analíticos para o desempenho das redes P2P. Ge et al. usaram modelos de filas para analisar o desempenho de redes estruturadas, não estruturadas e centralizadas [Ge et al. 2003]. Ding e Bhargava fizeram uma comparação de complexidade (notação "Big-Oh") dos diversos tipos de redes P2P sobre classes de algoritmos de roteamento em MANETs: broadcast sobre broadcast; broadcast; DHTs sobre broadcast; DHTs sobre DHTs; DHTs [Ding and Bhargava 2004]. Entretanto, ambos os trabalhos não consideraram o impacto de características como erros de transmissão e mobilidade, que são freqüentes em MANETs. Oliveira et al., por sua vez, utilizaram simulações para avaliar como mobilidade, erros de transmissão, número de nós e carga da rede modificam o desempenho de redes estruturadas e não estruturadas [Oliveira et al. 2005a].

Os parâmetros das aplicações par-a-par podem ser ajustados de maneira automática, utilizando algoritmos adaptativos. Algoritmos adaptativos devem ser *autoconscientes*, *autoconfiguráveis* e *auto-otimizáveis*, como apontado por Ganek e Corbi [Ganek and Corbi 2003]. Um sistema se torna *autoconsciente* quando possui uma identidade própria e conhece seu ambiente e contexto. Em outras palavras, ele é capaz de descrever com razoável precisão o seu estado e o estado do ambiente. Um sistema se torna *autoconfigurável* quando ele é capaz de se configurar automaticamente para se adaptar às mudanças no ambiente. Um sistema é *auto-otimizável* quando ele se monitora e configura automaticamente de forma a otimizar as necessidades do usuário sem intervenção humana.

O presente documento é parte de um trabalho que procura desenvolver protocolos de controle de tráfego para MANETs com aplicações P2P. Da Hora et al. proporam modificações em alguns parâmetros das aplicações P2P para melhorar o desempenho das redes par-a-par sobre redes ad hoc [da Hora et al. 2007]. Dois protocolos par-a-par foram estudados, Chord e Gnutella, e algumas melhorias propostas. Foi proposto, para as redes estruturadas, um mecanismo de replicação do número de mensagens de pesquisas, a fim de se aumentar a percentagem de acerto das pequisas em troca de um aumento relativo do consumo de energia e latência. Foi proposto, para as redes não estruturadas, um algoritmo que ajusta dinamicamente o número de vizinhos de acordo com a carga da rede.

Neste trabalho propomos algoritmos adaptativos para configuração automática dos parâmetros de redes par-a-par não estruturadas. As redes P2P não estruturadas funcionam por inundação controlada, por algum mecanimo de Tempo-de-vida (*Time-to-live* - TTL) das mensagens de pesquisas. Propomos um algoritmo que configura automaticamente o TTL do Gnutella, uma das mais clássicas redes P2P não estrutadas. Note que, embora tenhamos implementado e avaliado nosso algoritmo utilizando o Gnutella, a mesma técnica

pode ser aplicada a qualquer rede P2P não estruturada.

3. Detecção de Congestionamento

Nesta seção iremos descrever o algoritmo de detecção da condição de congestionamento de uma rede ad hoc. Esse algoritmo constituirá a camada de autoconsciência do algoritmo adaptativo proposto. Primeiro, caracterizamos e identificamos métricas para medir um congestionamento em redes móveis ad hoc. Depois, desenvolvemos um algoritmo capaz de identificá-lo durante o funcionamento da rede.

3.1. Caracterização de Congestionamento

O desempenho de uma rede ad hoc está ligado ao desempenho da camada MAC. Em redes 802.11, a vazão máxima de um nó depende da quantidade de mensagens enviadas no nível MAC pela sua vizinhança. Esta determina indiretamente a latência ponto a ponto da comunicação e a quantidade de mensagens na fila da camada de roteamento: é conhecido que o mecanismo de controle de contenção do IEEE 802.11 influencia a vazão no meio físico [Malone et al. 2007, Tanenbaum 2002]. Esse mecanismo aumenta o tempo de transmissão de quadros ao requerer um maior número de back-offs quando o número de estações transmitindo aumentam.

Como não é possível medir diretamente o mecanismo de controle de congestionamento do nível MAC ou a carga global imposta na rede, utilizamos medições indiretas, ou seja, mensuramos parâmetros de rede que são influenciados pela carga e a contenção no meio que são facilmente aferidos. Analisamos a quantidade de pacotes esperando a transmissão, que são valores parcialmente dependentes do congestionamento na rede.

3.1.1. Caracterização do Experimento

Para caracterizar um congestionamento, avaliamos o comportamento da rede à medida que a sua carga aumenta. A carga da rede é composta somente por consultas a dados na rede P2P, não modelando tráfego de fundo ou a transferência de arquivos. Assumimos uma rede de 50 nós em uma área de $1000 \times 1000\ m^2$, onde todos os 50 nós permanecem ligados durante toda a simulação. Cada simulação teve duração de $1000\ s$. Os demais parâmetros foram configurados conforme descrito na Seção 5. Os experimentos foram feitos por meio de simulações no NS-2. Executamos 33 simulações e os resultados apresentados são a média das simulações com intervalo de confiança de 95 %.

Também avaliamos a taxa de sucesso das pesquisas, representada pela porcentagem das consultas que são respondidas corretamente, a latência fim a fim das consultas e o consumo de energia. Estas são as três métricas que desejamos otimizar utilizando os algoritmos adaptativos. Como mostraremos a seguir, o desempenho da aplicação P2P é intimamente ligado à ocupação da rede.

À medida que a rede fica carregada, o número de colisões aumenta, resultando em uma menor taxa de sucesso (Figura 1). O aumento da carga da rede também resulta em uma maior latência (Figura 2), decorrência do protocolo CSMA-CA que espera a liberação do meio para transmissão. A maior latência é causada pelo enfileiramento de pacotes, que eleva a média global de ocupação da fila da camada de roteamento. Muito embora a média, no pior caso, não passe de 50 % de ocupação da fila (Figura 3), vemos

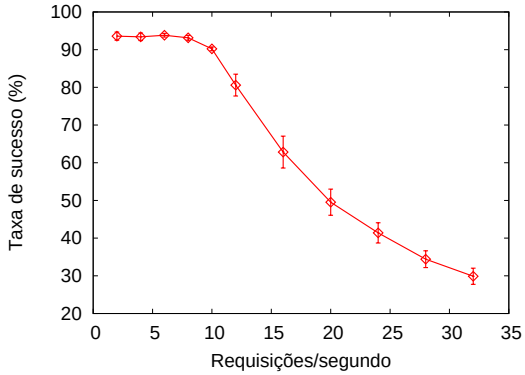

Figura 1. Taxa de sucesso das pesquisas

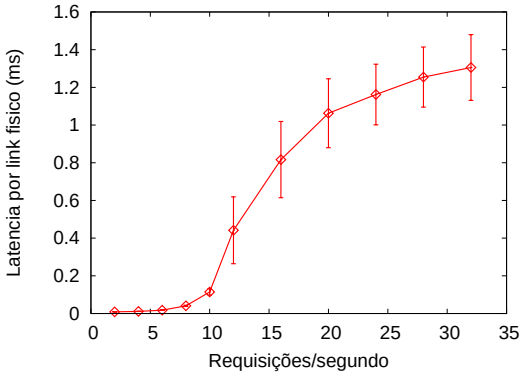

Figura 2. Latência de enlace

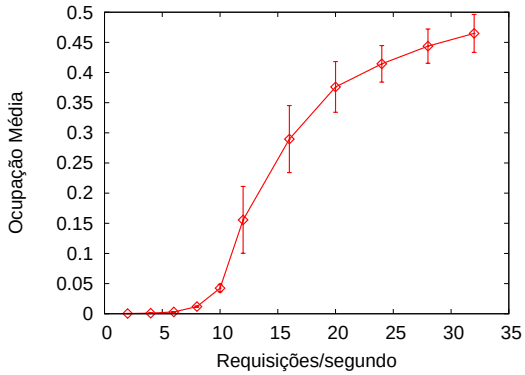

Figura 3. Ocupação média das Filas

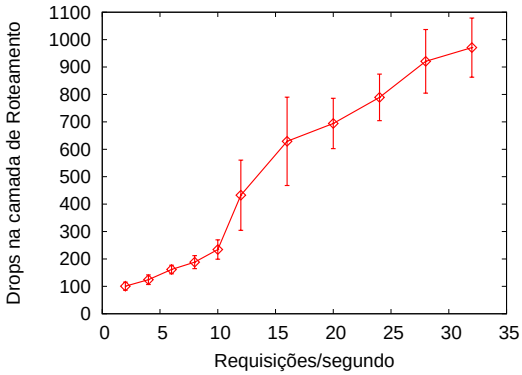

Figura 4. Descartes de pacotes

que existem descartes de pacotes (Figura 4), pois existem nós com ocupação próxima a 100 % e nós com baixa ocupação da fila.

Existem dois cenários particularmente importantes. O cenário em que são realizadas 8 requisições/segundo é o ponto de operação que precede a queda de desempenho da rede, e o cenário com 12 requisições por segundo, onde há uma perda significativa de desempenho devido ao congestionamento. Comparando os dois cenários, observamos uma queda de 13.4% no taxa de sucesso e um aumento percentual de 1200 % na ocupação média e de 900 %, para a latência.

Para identificar um congestionamento, precisaremos utilizar variáveis facilmente mensuráveis pelos nós da rede. Aprofundamos o estudo utilizando a métrica de ocupação, por ser a métrica que mais variou entre os dois cenários, e por ter uma relação intuitiva com o desempenho do sistema.

3.2. Identificando o Congestionamento a Partir de Dados Locais

A ocupação média estudada na Seção 3.1 se refere a uma média global. No entanto, cada nó possui acesso apenas aos dados sobre ele mesmo. Analisamos a dispersão da ocupação local para identificar se ela é representativa da média global. Por exemplo, caso os dados que levaram a uma média de 40% estiverem contidos no intervalo de 30% a 50%, os nós podem utilizar apenas a informação de ocupação local, uma vez que eles possuem

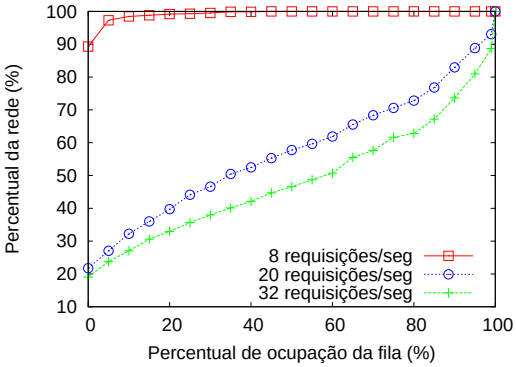

Figura 5. Ocupação média das filas

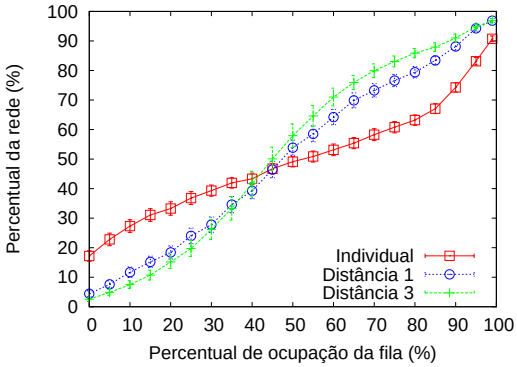

Figura 6. CDFs - 32 requisições por segundo

uma baixa variância em relação à média. No entanto, se os dados estiverem dispersos sobre uma grande faixa de valores, a ocupação local passa a ser pouco representativa da média da rede, e outra estratégia deve ser buscada.

Extraímos os dados da topologia da rede Gnutella previamente simulada e montamos um grafo direcionado cujos vértices são os nós Gnutella e as arestas modelam a relação de vizinhança. O valor de cada vértice indica o percentual de ocupação da fila na camada de roteamento. Coletamos os dados durante toda a simulação, em intervalos de $50s$.

Geramos CDFs da ocupação entre os nós da rede, para diferentes cargas de requisições, que podem ser vistas na Figura 5. Vemos na Figura 5 que, para redes pouco carregadas (8 requisições/seg), 100% dos nós possuem ocupação menor do que 10%. No entanto, para redes carregadas (20 e 32 requisições/seg) existem, igualmente, nós com baixo percentual de ocupação e com alto percentual de ocupação. Ou seja, a rede em questão, mesmo sob uma carga muito alta (32 requisições/seg), a rede possui uma característica de heterogeneidade da ocupação na fila da camada de roteamento. Assim, não podemos utilizar a ocupação individual como representativa da média global. Enquanto a média global é de $46,5\%$ (32 requisições/seg), temos $22,5\%$ dos nós com carga menor do que 10% e 25% dos nós com carga maior do que 95%. Assim, decidimos utilizar uma abordagem onde os nós irão disseminar periodicamente a sua ocupação para os seus vizinhos, como descrevemos a seguir.

3.3. Identificando Congestionamentos Usando Dados de uma Vizinhança

Nesta seção procuramos identificar congestionamentos usando as informações de um conjunto de nós, definindo uma vizinhança. Usamos a vizinhança a nível de aplicação, assim os nós que compõem a vizinhança podem estar a vários saltos de distância. Uma vizinhança da aplicação neste caso é melhor que uma vizinhança física, pois os nós terão informações de diversas regiões da rede, permitindo uma visão mais próxima do desempenho global da rede utilizando menos nós.

Definimos a ocupação local de um par j como $o_{0,j}$. A ocupação calculada por esse par j para os seus vizinhos de distância i é $o_{i,j}$ e o tamanho dessa vizinhaça é $n_{i,j}$. Cada par calcula e divulga para seus vizinhos $o_{i,j}$ e $n_{i,j}$ como descrito nas equações 1 e 2, respectivamente.

$$n_{i,j} = \begin{cases} 1 & i = 0 \\ |V_j| & i = 1 \\ \sum_{p \in V_j} n_{i-1,p} & i > 1 \end{cases} \quad (1)$$

$$o_{i,j} = \frac{\sum_{p \in V_j} (o_{i-1,p} \times n_{i-1,p})}{n_{i,j}}, i > 1 \quad (2)$$

Note que um nó j só poderá calcular $o_{i,j}$ e $n_{i,j}$ se possuir $o_{i-1,p}$ e $n_{i-1,p}$ para cada $p \in V_j$. Por isso, cada par deve divulgar para seus vizinhos esses valores calculados. Cada par propaga esses valores junto com as mensagens de *PING* e *PONG* do Gnutella, que são mensagens utilizadas pelo protocolo para garantir que os nós vizinhos ainda estão na rede ou não falharam. Escolhemos esta alternativa para reduzir o *overhead* das soluções adaptativas propostas. Nesse método de disseminação da ocupação os dados são frequentemente agregados. Por isso, passaremos a nos referir a esse algoritmo por *disseminação agregada*. Ao tentar aplicar essa técnica em um protocolo não estruturado que não possua uma troca de mensagens periódicas aos vizinhos, essa função deverá ser implementada.

Na Figura 6 vemos uma tendência central dos dados. Isso ocorre porque enquanto vários nós detectam coeficiente de ocupação alto, outros nós detectam coeficiente de ocupação baixo, o que pode ser explicado pela posição geográfica dos nós. Enquanto os nós no centro da rede estarão em constante atividade, realizando o roteamento dos pacotes de toda a rede, os nós periféricos irão rotear menos pacotes. Aumentando o nível de profundidade da pesquisa, temos um maior consenso na rede. O número de nós detectando ocupação 0% cai de 17% para 2,5%. Utilizando pesquisa de profundidade 3, 53% dos nós possuem ocupação entre 30% e 70%, enquanto que utilizando o valor local, apenas 19% dos nós possuem ocupação nesse intervalo.

Para redes pouco carregadas (Figura 7), quase todos os nós possuem ocupação baixa, mas eventualmente algum nó pode se tornar um pouco mais carregado. Neste cenário vemos que quase todos os nós detectam baixa ocupação. Com pesquisa individual, a maioria possui ocupação menor do que 10%, mas possuindo alguns nós detectando ocupação de até 40%. Com pesquisa aos vizinhos imediatos, temos uma dispersão menor. 100% dos nós possuem ocupação menor do que 15%. Utilizando pesquisa com profundidade 3, 100% dos nós possuem ocupação menor do que 10%.

Agora que caracterizamos a ocupação da rede e que temos um mecanismo de disseminação desta informação, propomos uma função que mapeia a ocupação média em um valor que mensure a condição de congestionamento da rede. Esta função será detalhada a seguir.

3.4. Medindo a Condição da Rede

Utilizamos a ocupação média da rede, obtido através do algoritmo de disseminação agregada descrito na seção anterior, para identificar a condição atual da rede. Tal função será utilizada nos algoritmos adaptativos para auxiliar no processo de reconfiguração do TTL das mensagens de consulta. Seja x a informação da ocupação média obtida através do algoritmo de disseminação agregada. Procuramos uma função F(x), tal que:

$$F(X) = \begin{cases} -1, & \text{se a rede não está congestionada} \\ 1, & \text{se a rede está congestionada} \end{cases} \quad (3)$$

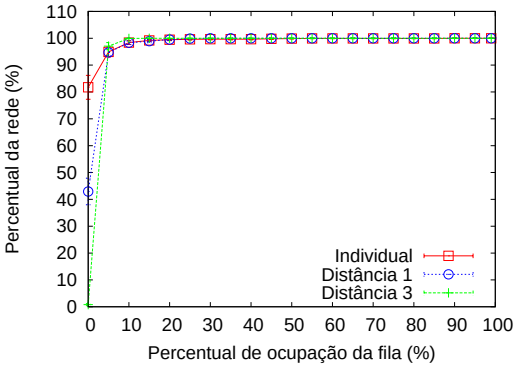

Figura 7. CDFs - 8 requisições por segundo

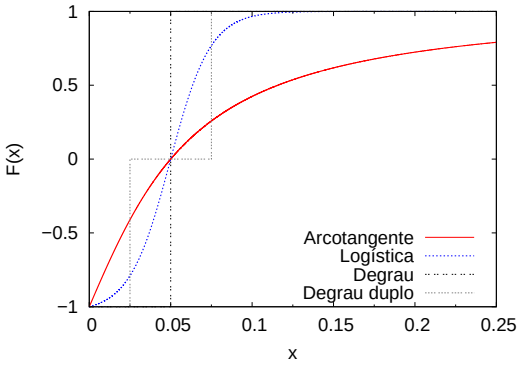

Figura 8. Funções de detecção da condição propostas

A avaliação de $F(x)$ deve nos dar uma "nota" para o estado de congestionamento da rede, e passaremos a nos referir ao resultado de $F(x)$ também dessa forma. A maneira mais simples de se implementar $F(x)$ é torná-la uma função degrau, de tal forma que todos os valores menores que um certo limite β retornam -1, e todos valores acima de β retornam $+1$. Essa abordagem não é recomendável, dado que o valor β se torna um valor de grande importância para a detecção de congestionamento, e existe uma certa incerteza de seu valor. Devemos então utilizar funções um pouco mais flexíveis, de forma a gerar o menor número possível de falsos positivos e falsos negativos.

A Figura 8 mostra as funções propostas para implementar a detecção da condição da rede. Além da função degrau proposta inicialmente, propomos uma função degrau duplo, que possui dois valores limites β_1 e β_2, retornando -1 para valores abaixo de β_1, $+1$ para valores acima de β_2 e 0 para valores intermediários. Nessa função, introduz-se a noção de um valor intermediário de congestionamento, onde a carga está alta mas o congestionamento não é significativo.

A detecção de um congestionamento utilizando um dado de entrada (ocupação) se assemelha a um perceptron de uma camada, que é o modelo mais simples de uma rede neural. As funções logística e arcotangente são frequentemente utilizadas como função de ativação em redes neurais, por isso também avaliamos essas funções para realizar a separação dos dados.

Avaliamos experimentalmente diversos valores β para a função degrau, e obtivemos o melhor resultado para $\beta = 0.05$. Para a função degrau-duplo, os melhores resultados ocorreram com $\beta_1 = 0.025$, $\beta_2 = 0.075$. Fizemos a regressão das funções arcotangente e logística de forma que $F(0) = -1$, $F(\beta) = 0$ e $F(1) = 1$. As funções arcontagente e logística são mostradas nas Equações 4 e 5. A regressão foi realizado utilizando as variáveis a e b como parâmetro.

$$F(x) = \arctan(x \times a) \times -1 \tag{4}$$

$$F(x) = 2/(1 + exp(b - x \times a)) - 1 \tag{5}$$

Após a regressão, obtivemos $a = 18$ na Equação 4, e $a = 80$ e $b = 4$ na Equação 4. Realizamos também um processo de normalização nas duas Equações, de forma a obter uma função $F(x) : [0 : 1] \rightarrow [-1 : 1]$.

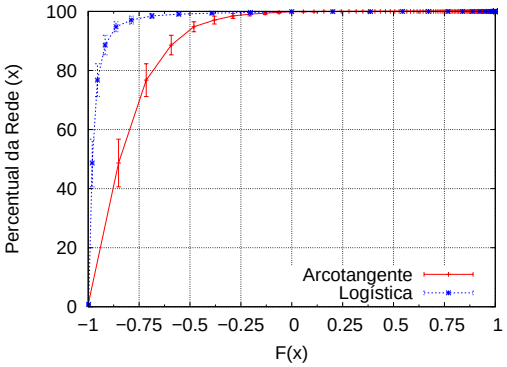

Figura 9. Detecção da condição da rede em uma rede não congestionada

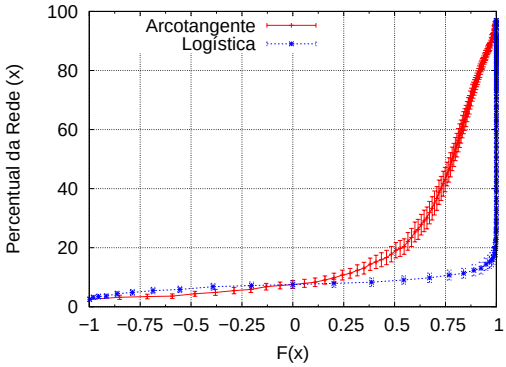

Figura 10. Detecção da condição da rede em uma rede congestionada

Nas Figuras 9 e 10 comparamos as funções logística e arcotangente em um cenário não congestionado e congestionado. As curvas apresentadas são CDFs que mostram a dispersão das "notas" que cada par identificou durante a operação da rede. A Figura 9, mostra um cenário não congestionado. Na função logística, cerca de 98% identificam uma nota menor do que -0.75, sendo que na arcotangente esse temos cerca de 70% dos pares nesse mesmo patamar. A função arcotangente é mais conservadora, evitando notas muito negativas. O mesmo acontece em um cenário congestionado (Figura 10). Na função logística, cerca de 90% dos pares identificam uma nota maior do que 0.75, enquanto que na arcotangente apenas 56% dos pares identificam uma nota maior do que 0.75, o que significa que a arcotangente é mais conservadora que a logística.

Avaliamos experimentalmente o desempenho do algoritmo adaptativo proposto na Seção 4 com essas funções e percebemos que as funções degrau e degrau-duplo apresentam desempenho inferior às funções logística e arcotangente. A função arcotangente (equação **??**) apresentou bons resultados em experimentos preliminares, e por isso a escolhemos na implementação do algoritmo adaptativo. Aparentemente, em sistemas tão complexos e incertos quanto o proposto, o ideal é sermos um conservadores.

4. Algoritmos de decisão

Esta seção descreve os algoritmos de decisão propostos. Diversos algoritmos de decisão foram experimentados e aqui mostramos os resultados daqueles que apresentaram o melhor desempenho, convergência e capacidade de adaptação. O processo de adaptação utiliza um controle de malha fechada. Utilizamos um controle de crescimento aditivo e decrescimento multiplicativo (também conhecido como AIMD em inglês), que é o mecanismo utilizado no protocolo de transporte TCP para o ajuste do tamanho da janela deslizante. Como o objetivo dos algoritmos propostos é evitar que a rede fique congestionada, a estratégia AIMD é ideal para tal tarefa. O AIMD reduz rapidamente o congestionamento devido ao decréscimo multiplicativo, enquanto o acréscimo aditivo permite um aumento conservador do parâmetro controlado. Aplicamos o algoritmo AIMD ao tempo de vida das mensagens de pesquisa (TTL), utilizando como variáveis de entrada o sucesso ou falha de uma pesquisa, bem como a ocupação da rede. A partir destes parâmetros de entrada, criamos três algoritmos, que são mostrados na Tabela 1. Iremos descrever os algoritmos propostos a seguir.

Tabela 1. Especificação do cenário de simulação padrão.

Método	Disparo	Parâmetros
AIMDQ	A cada hit/miss de pesquisa	Hit/miss
AIMDQ+O	A cada hit/miss de pesquisa	Hit/miss + ocupação
P-AIMD	Periódico	Ocupação

4.1. AIMDQ

O primeiro algoritmo proposto é chamado AIMDQ, de *AIMD with Queries*. A cada pesquisa realizada, a aplicação aguarda um tempo por respostas. Se a pesquisa encontrar resultados positivos, ela incrementa o TTL por um valor δ. Se, ao fim da espera a aplicação não receber resultados, ela decrementa o TTL, multiplicando-o por uma constante α menor do que 1. Note que esse algoritmo decide o valor do TTL sem utilizar a informação da ocupação, que foi estudada na Seção 3. Mostramos o desempenho desse algoritmo a fim de obtermos uma base comparativa.

Com o algoritmo supracitado, teremos frequentemente valores de TTL fracionários, como $3,5$. Para dar suporte a esses tipos de valores, de forma a termos uma melhor granularidade do parâmetro TTL, criamos a noção de TTL fracionário, que funciona da seguinte forma: cada nó avalia, ao receber a mensagem de pesquisa, se o TTL é maior do que 1. Se for, ele decrementa o TTL em uma unidade e repassa a pesquisa aos seus vizinhos. Quando a mensagem de pesquisa possuir TTL $t, 0 < t < 1$, ele repassa a mensagem aos seus vizinhos com uma probabilidade t, com TTL = 0.

4.2. AIMDQ+O

O segundo algoritmo melhora o desempenho do algoritmo AIMDQ ao adicionar a ocupação no processo de decisão. O AIMDQ assume que as pesquisas falharam por não conseguirem encontrar o par com informações sobre o conteúdo, devido a colisões e descartes de pacotes. Em redes P2P não estruturadas uma pesquisa também pode falhar quando o TTL utilizado foi pequeno, e assim o nó que armazena a informação procurada não foi contactado.

Assim, propomos o algoritmo AIMDQ+O, de *AIMDQ plus Occupation*. O algoritmo calcula o novo TTL a cada resultado de pesquisa recebido, da mesma forma que o algoritmo anterior. Quando uma pesquisa não retorna resultados, o algoritmo atualiza o TTL levando em conta o estado de congestionamento da rede. Seja $F(x)$ uma das funções definidas na Seção 3.4. O cálculo do novo TTL ocorre de acordo com a seguinte fórmula:

$$TTL(i+1) = TTL(i) + \begin{cases} \delta \times |F(x)|, & \text{se } F(x) \leq 0 \\ (\alpha - 1) \times TTL(i) \times F(x), & \text{se } F(x) > 0 \end{cases} \quad (6)$$

Se a pesquisa falhou mas não foi detectado um congestionamento, a falha ocorreu porque o TTL foi insuficiente, assim o algoritmo incrementa o TTL. Se a pesquisa falhou e um congestionamento está sendo acusado, o algoritmo reduz multiplicativamente o TTL de forma semelhante à abordagem anterior. Se $|F(x)| < 1$, o módulo de $F(x)$ será um multiplicador para a mudança a ser aplicada no TTL.

Tabela 2. Especificação do cenário de simulação padrão.

Parâmetro	Valor
Protocolo MAC	IEEE 802.11b
Número de nós	50
Área	$1km^2$
Distribuição dos nós	*Uniform distribution*
Modelo de mobilidade	*Random Waypoint*, $0 \leq velocidade \leq 1.0$ m/s.
Percentual de nós permanentemente *on-line*	50%
Número de arquivos	250 arquivos diferentes na rede
Número de réplicas por arquivo	5
Erros do canal de comunicação	0%

4.3. P-AIMD

Os algoritmos propostos anteriormente são executados à medida que as pesquisas são executadas. Isso significa que a adaptação dos parâmetros ocorre somente quando o usuário faz uma requisição para um arquivo, o que pode acarretar em um longo período para que a rede se adapte se poucas consultas são feitas por cada usuário. Entretanto, o algoritmo de disseminação agregada roda periodicamente, uma vez que suas mensagens trafegam com as mensagens PING e PONG. Assim, como a informação de ocupação é independente da existência de pesquisas, podemos adaptar o algoritmo de forma periódica mesmo que o usuário não faça pesquisas de conteúdo. O P-AIMD, de *Periodic AIMD*, adapta periodicamente o TTL a partir das informações de ocupação obtidas com o algoritmo de disseminação agregada. Este algoritmo utiliza a equação 6, mostrada anteriormente, para calcular o novo TTL. Além de rodar periodicamente, este algoritmo não considera o sucesso ou falha das pesquisas.

5. Avaliação

Para avaliar o desempenho dos algoritmos propostos, realizamos simulações no simulador NS-2 versão 2.29.3, utilizando o modelo de propagação de radio *two-ray-ground*. Simulamos uma rede par-a-par sobre uma rede móvel ad hoc. O protocolo P2P foi implementado utilizando mensagens UDP, uma vez que o TCP não possui um bom resultado neste tipo de ambiente [Sundaresan et al. 2005, Katabi et al. 2002]. Utilizamos o protocolo de roteamento AODV [Perkins and Royer 1999], uma vez que ele apresentou o melhor desempenho utilizando uma aplicação P2P nos cenários mais comuns em redes móveis ad hoc [Oliveira et al. 2005b]. Os nós utilizam *wifi* IEEE 802.11b, modelados a partir de um cartão Cisco Aironet [Cisco Systems 2005] utilizando uma potência de transmissão de 10dBm. Nesta configuração, o rádio consome $1,6887\ W$ no modo de transmissão e $0,6699\ W$ em modo de espera e $1,0791\ W$ na recepção. A fila de pacotes na camada de roteamento acomoda 30 pacotes.

Assumimos uma rede com 50 nós distribuídos em uma região de 1000m $\times$ 1000m seguindo uma distribuição uniforme. Os nós se movem de acordo com o modelo de mobilidade *random way-point* (uma vez que ele é frequentemente usado para movimentação individual [Broch et al. 1998]) com um tempo de pausa de 0.1s e uma velocidade média selecionada uniformemente entre 0 e 0.25 m/s. A qualquer momento da simulação, 50% dos nós estão sempre ligados, enquanto os demais entram na rede em algum momento e a deixam após algum tempo. Os tempos de entrada e saída dos nós na rede P2P seguem uma distribuição uniforme. Cada nó possui 5 arquivos diferentes, existindo assim 250

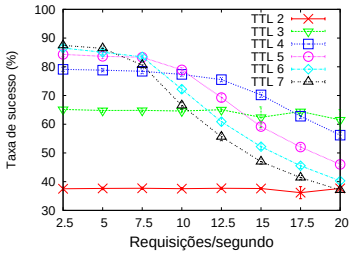

Figura 11. Diferentes configurações de TTL.

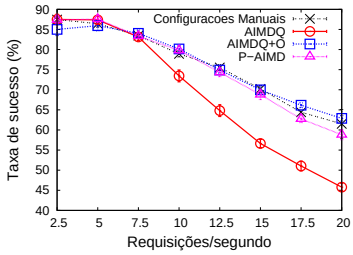

Figura 12. Taxa de sucesso.

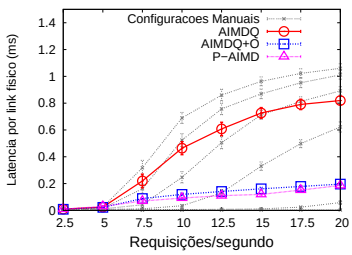

Figura 13. Latência na camada MAC.

arquivos diferentes na rede. Cada arquivo possui 5 réplicas aleatoriamente distribuídas entre os pares. No cenário padrão não consideramos perdas devidos a erros do canal de comunicação. Consideramos perdas de pacotes devido a colisões, mas não devido a interferências ou degradação momentânea do meio. A Tabela 2 sumariza os parâmetros usados no cenário padrão das simulações. Avaliamos os algoritmos propostos para diversas cargas na rede. A carga é composta somente por consultas, pois não simulamos a transferência dos arquivos encontrados ou tráfego de fundo (tráfego de outras fontes que não sejam a aplicação P2P).

Os parâmetros dos algoritmos adaptativos foram ajustados empiricamente, e os valores definidos foram: TTL inicial = 4, $\alpha = 0.9$ e $\delta = 0.1$. Nesta seção utilizamos a função F(x) baseada no arcotangente, descrita na equação 4.

A figura 11 mostra os resultados do experimento em que comparamos o desempenho de cada possível configuração manual de parâmetros. Vemos que a escolha de um parâmetro fixo terá um desempenho reduzido em relação ao ótimo, tendo em vista os diferentes possíveis cenários de carga. Na figura 12 mostramos ainda o melhor resultado obtido pelas configurações manuais e comparamos seu desempenho ao dos algoritmos adaptativos. O algoritmo AIMDQ obteve desempenho estatisticamente equivalente à melhor configuração manual de parâmetros nos cenários de baixa carga, e teve desempenho similar ao de TTL 5 nos cenários de alta carga. O algoritmo P-AIMD+O obteve um desempenho 3% inferior à melhor configuração manual nos cenário de baixa carga e obteve um desempenho 9,7% superior à melhor configuração de parâmetros. Note que é possível obter um resultado superio à melhor configuração de parâmetros, uma vez que esta utiliza apenas valores de TTL inteiros, enquanto que nos algoritmos adaptativos podemos ter TTLs fracionários. O algoritmo P-AIMD obteve um desempenho equivalente à melhor configuração de parâmetros para todos os cenários simulados.

Na Figura 13 vemos a latência da camada física. De acordo com a caracterização realizada, observamos que um aumento expressivo da latência é um indício de ocorrência de congestionamento, comportamento observado utilizando os valores 7, 6, 5 e 4 para o TTL. O aumento da latência é limitado nos algoritmos que utilizam a função que detecta congestionamento (P-AIMD e AIMDQ+O). No AIMDQ a latência aumenta significativamente, mostrando que este algoritmo produz uma alta carga na rede.

6. Conclusão

Redes P2P são frequentemente utilizadas para a disseminação e compartilhamento de informação em redes móveis ad hoc (MANETs). Aplicações par-a-par possuem parâmetros que devem ser ajustados de acordo com a as características da rede física.

Tem-se utilizado uma configuração de parâmetros estática e genérica, a fim de ser aplicável a uma grande diversidade de redes. No entanto, essa configuração apresenta um desempenho inferior ao obtido pelo melhor ajuste manual dos parâmetros. Neste trabalho propomos algoritmos que ajustam dinamicamente os parâmetros de protocolos P2P não estruturados, a fim de otimizar o funcionamento da aplicação para cada instância da rede.

Para isso, impletamos um algoritmo de detecção de congestionamento, e verificamos que é possível detectar um congestionamento utilizando o percentual de ocupação da fila na camada de roteamento, desde que esta média seja feita sobre uma vizinhança. Propomos umaa estratégia de disseminação agregada da ocupação, e implementamos esse algoritmo. Propomos três algoritmos adaptativos, que ajustam dinamicamente TTL das consultas, e avaliamos seu desempenho através de simulações em um cenário onde variamos a carga da rede em uma rede Gnutella. Comparamos os resultados com diferentes configurações fixas de parâmetros. Os resultados mostram que o desempenho do algoritmo AIMDQ é ligeiramente reduzido em relação à melhor configuração manual, pois o AIMDQ impõe uma carga alta à rede. Os algoritmos AIMDQ+O e P-AIMD se mostraram mais eficientes, obtendo resultados equivalentes à melhor configuração manual. Como trabalhos futuros, pretendemos estudar de forma ainda mais profunda conceitos de teoria de controle e inteligência artificial para melhorar o desempenho dos algoritmos propostos, e validá-los sobre diversos cenários.

Referências

Borg, J. (2003). A comparative study of ad hoc & peer to peer networks. Master's thesis, University College London.

Broch, J., Maltz, D. A., Johnson, D. B., Hu, Y.-C., and Jetcheva, J. (1998). A performance comparison of multi-hop wireless ad hoc network routing protocols. In *Proceedings of the 4th annual ACM/IEEE International Conference on Mobile Computing and Networking*, pages 85–97.

Cisco Systems (Dec., 2005). Cisco aironet 350 series client adapters. http://www.cisco.com/en/US/products/hw/wireless/ps4555/products_data_sheet09186a0080088828.html.

Conti, M., Gregori, E., and Turi, G. (2005). A cross-layer optimization of gnutella for mobile ad hoc networks. In *MobiHoc '05: Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, pages 343–354. ACM Press.

da Hora, D. N., Macedo, D. F., Nogueira, J. M., and Pujolle, G. (2007). Otimizando requisições de conteúdo em redes par-a-par sobre redes ad hoc. In *24º Simpósio Brasileiro de Redes de Computadores*.

Ding, G. and Bhargava, B. (2004). Peer-to-peer file-sharing over mobile ad hoc networks. In *2th IEEE Annual Conference on Pervasive Computing and Communications Workshops*, pages 104–108.

Franciscani, F. P., Vasconcelos, M. A., Couto, R. P., and Loureiro, A. A. F. (2005). Peer-to-peer over ad-hoc networks: (re)configuration algorithms. *Journal of Parallel and Distributed Computing (JPDC)*, 65(2):234–245.

Ganek, A. G. and Corbi, T. A. (2003). The dawning of the autonomic computing era. *IBM Systems Journal*, 42(1):5–18.

Ge, Z., Figueiredo, D. R., Jaiswal, S., Kurose, J., and Towsley, D. (2003). Modeling peer-peer file sharing systems. In *Proceedings of IEEE INFOCOM*.

Haas, Z. J., Deng, J., Liang, B., Papadimitratos, P., and Sajama, S. (2002). Wireless ad hoc networks. In Proakis, J. G., editor, *Wiley Encyclopedia of Telecommunications*. John Wiley & Sons.

Katabi, D., Handley, M., and Rohrs, C. (2002). Congestion control for high bandwidth-delay product networks. In *SIGCOMM '02: Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 89–102. ACM Press.

Malone, D., Duffy, K., and Leith, D. (2007). Modeling the 802.11 distributed coordination function in nonsaturated heterogeneous conditions. *IEEE/ACM Transactions on Networking*, 15(1):159–172.

Oliveira, L. B., Siqueira, I., Macedo, D. F., Loureiro, A. A., and Nogueira, J. M. (2005a). Evaluation of peer-to-peer network content discovery techniques over mobile ad hoc networks. In *IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 51–56.

Oliveira, L. B., Siqueira, I. G., and Loureiro, A. A. F. (2005b). On the performance of ad hoc routing protocols under a peer-to-peer application. *Journal of Parallel and Distributed Computing (JPDC)*, 65(11):1337–1347.

Perkins, C. E. and Royer, E. M. (1999). Ad hoc on-demand distance vector routing. In *Proceedings of the 2nd IEEE Workshop on Mobile Computing System and Applications*, pages 90–100.

Sundaresan, K., Anantharaman, V., Hsieh, H.-Y., and Sivakumar, R. (2005). ATP: A Reliable Transport Protocol for Ad Hoc Networks. *IEEE Transactions on Mobile Computing*, 4(6):588–603.

Talia, D. and Trunfio, P. (2003). Toward a synergy between P2P and grids. *IEEE Internet Computing*, 7(4):94–95.

Tanenbaum, A. S. (2002). *Computer networks*. Prentice-Hall, Inc., 4 ed. edition.