Relatório Técnico

TAQUARA - Tecnologia, Aplicações e Qualidade de Serviço em Redes Avançadas no Projeto GIGA

Projeto Giga/RNP 2460

2 de junho de 2007

1. Equipe de Professores

- Alfredo Goldman - IME/USP
- Antônio Jorge Gomes Abelém - DI/UFPA
- Edmundo Roberto Mauro Madeira - Unicamp
- Edson dos Santos Moreira - ICMC/USP
- Eleri Cardozo - Unicamp
- Fábio Kon - IME/USP
- José Ferreira de Rezende - UFRJ
- Luís Henrique Maciel Kosmalski Costa - UFRJ
- Marcelo Gonçalves Rubinstein UERJ
- Nelson Luís Saldanha da Fonseca - Unicamp
- Tereza Cristina M. B. Carvalho - Poli-USP

2. Instituições

Universidade Federal do Rio de Janeiro - UFRJ
Escola Politécnica - Departamento de Eletrônica e de Computação
COPPE - Programa de Engenharia Elétrica
Grupo de Teleinformática e Automação - GTA

Universidade Estadual de Campinas - Unicamp
Departamento de Ciência da Computação
Faculdade de Engenharia Elétrica e de Computação

Universidade de São Paulo - USP
Instituto de Matemática e Estatística

Universidade do Estado do Rio de Janeiro - UERJ
Faculdade de Engenharia

Universidade Federal do Pará - UFPA
Departamento de Informática

2.1. Instituições Intervenientes

Cobra Tecnologia, Proderj e Netcenter.

3. Introdução

Este relatório trata das atividades realizadas durante o projeto TAQUARA.

4. Redes EPONS

As redes com elementos ópticos passivos do tipo Ethernet (EPONs) são redes ópticas ponto-multiponto que transportam dados em quadros Ethernet IEEE 802.3 e que têm em seu interior apenas elementos ópticos passivos unidirecionais tais como combinadores, divisores e acopladores ópticos. As transmissões dos fluxos de subida e de descida se fazem por meio de comprimentos de onda independentes, transportadas por fibras monomodo [Kramer et al. 2001, Takeuti 2005].

As redes EPONs são uma alternativa interessante para redes de acesso pois permitem levar a fibra óptica diretamente aos usuários finais, aumentando a banda disponível e reduzindo os custos de implantação e de manutenção [Kramer et al. 2001, Pereira 2006]. Em uma rede de acesso EPON, os usuários se conectam por meio de unidades de rede óptica (*Optical Network Units*, ONUs) exclusivas ou compartilhadas, que são conectadas a um terminal de linha óptica (*Optical Line Terminal*, OLT) disposto junto ao provedor de serviços. Um exemplo de rede de acesso EPON é apresentado na Figura 1.

As transmissões em redes EPON são feitas segundo a especificação da norma IEE 802.3. No fluxo de descida, a OLT transmite os quadros para as ONUs por difusão e cada ONU seleciona os quadros que lhe são destinados com base em campos de endereçamento lógico específicos. Nos fluxos de subida, a rede emprega um protocolo de *polling* denominado Protocolo de Controle Multiponto (*Multipoint Control Protocol*, MPCP) para controlar o acesso das ONUs ao meio compartilhado. A norma não estabelece, contudo, qual seria a política de escalonamento requerida para arbitrar este acesso nem para regular a concorrência dos fluxos de cada ONU pelos recursos da rede.

4.1. Motivacão

Políticas de escalonamento para redes EPON são hoje um tema de intensa pesquisa [Kramer et al. 2002, McGarry et al. 2004, Naser and Mouftah 2006]. Dados os elevados valores de largura de banda e de tempo de propagação envolvidos, as políticas quadro-a-quadro tradicionais não podem ser implementadas de maneira eficiente em uma rede EPON. Por isso, usualmente consideram-se políticas baseadas em janelas que distribuem as tarefas de escalonamento entre a OLT e as ONUs. Há, então, duas etapas que são denominadas escalonamento inter-ONU e intra-ONU. No escalonamento inter-ONU, a OLT atribui às ONUs janelas de tempo cuja duração é calculada com base nas demandas que elas informam ao final da sua última transmissão. Durante a sua janela de tempo, a ONU ocupa sozinha o canal compartilhado e pode então transmitir os quadros dos seus fluxos. A divisão da capacidade da janela de tempo entre os fluxos é realizada pela própria ONU por meio do escalonamento intra-ONU.

A maioria das propostas na literatura pressupõe que os escalonamentos intra-ONU e inter-ONU são realizados de maneira independente. Neste caso, entretanto, as ONUs

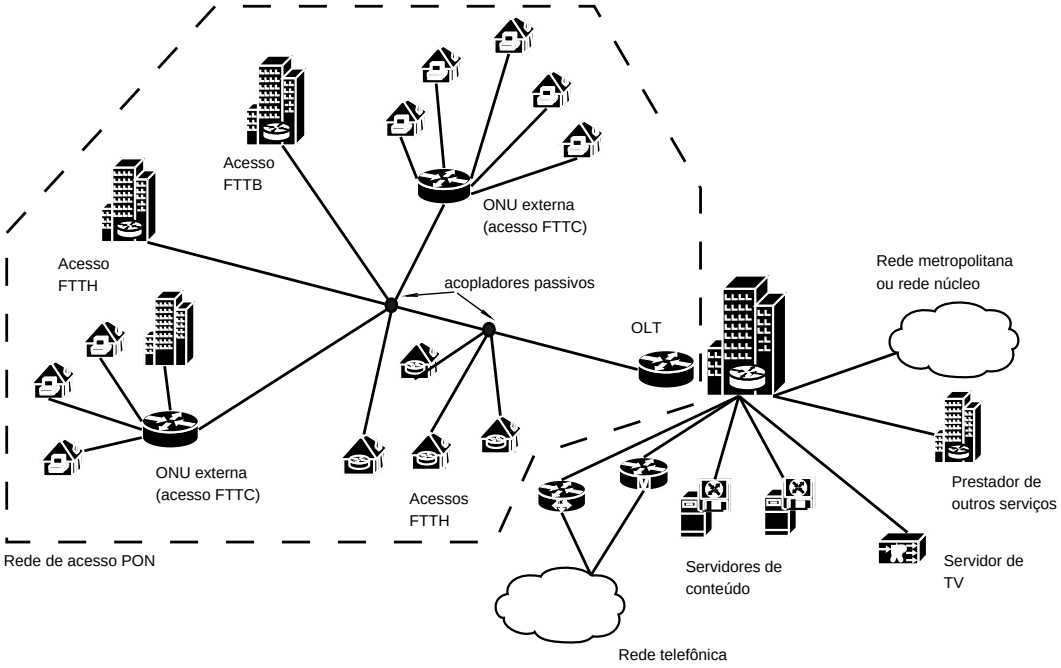


Figura 1. Exemplo de rede de acesso EPON.

não compartilham qualquer informação sobre a divisão da capacidade das suas janelas entre seus fluxos, o que pode levar a uma divisão desbalanceada dos recursos entre fluxos de ONUs diferentes. Com isso, a prioridade dos fluxos não seria respeitada e os usuários seriam tratados de maneira injusta [Kramer et al. 2004].

Para contornar esse problema, Kramer et al. [Kramer et al. 2004] propuseram uma política de escalonamento descentralizada denominada Enfileiramento Justo com Curvas de Serviço (*Fair Queueing with Service Envelopes*, FQSE). A política FQSE pressupõe uma relação hierárquica entre os escalonamentos inter-ONU e intra-ONU, tal que a OLT forneça às ONUs um indicador que lhes permita determinar o volume de tráfego de cada fluxo que se pode transmitir tal que os recursos da rede sejam divididos entre os fluxos de maneira balanceada e, portanto, justa.

A motivação para a pesquisa realizada foi a concepção de uma política de escalonamento que fosse capaz de prover justiça entre os fluxos da rede de forma a superar as deficiências das políticas existentes, assegurando, assim, que fluxos com o mesmo nível de prioridade porém pertencentes a ONU's diferentes receberão a mesma quantidade de banda passante. Tem-se também, como objetivo, que a política garanta limitantes para o atraso, permitindo, assim, que a infraestrutura forneça garantias de Qualidade de Serviço fim-a-fim.

O presente relatório relata uma proposta para uma nova política de escalonamento denominada Compartilhamento Proporcional com Reserva de Carga (*Proportional Sharing with Load Reservation*, PSLR), que também assegura a distribuição justa dos recursos entre os fluxos de diferentes ONUs. Assim como a política FQSE, a política PSLR permite que a rede reserve uma certa quantidade de recursos, na forma de taxa de serviço mínima, a cada um dos fluxos e atribua-lhes uma dada prioridade na concorrência pelos recursos ociosos. A diferença entre as duas políticas está na forma como a OLT e as

ONUs regulam essa concorrência, o que leva à política PSLR a apresentar uma complexidade computacional inferior àquela imposta pela política FQSE. Além dessa vantagem, a política PSLR apresenta limitantes analíticos para a justiça relativa entre fluxos e para o retardo introduzido nos fluxos. Tais resultados ainda não foram obtidos para a política FQSE.

4.2. Aspectos preliminares

A seguir serão descritos conceitos fundamentais para que se poss entender a política proposta. Descreve-se o sistema considerado, bem como introduz-se os conceitos de justiça e a política FSQE, na qual este trabalho foi baseado.

4.2.1. Notação e descrição do sistema

Neste trabalho, considera-se que o enlace de subida de uma rede EPON tem capacidade r e é compartilhado por N fluxos, que estão agrupados em J ONUs. Dentro das ONUs, cada fluxo tem uma fila exclusiva para armazenar os seus quadros em espera. O i-ésimo fluxo da j-ésima ONU é identificado pelo sub-índice ij.

Supõe-se que a OLT emprega um mecanismo de *polling* para atribuir janelas de tempo para que cada uma das ONUs transmita uma certa quantidade de quadros dos seus fluxos. No n-ésimo ciclo de *polling*, a duração (em bits) da janela de tempo atribuída pela OLT à j-ésima ONU é representada por $C_j[n]$. Durante essa janela, a ONU transmite $c_{ij}[n]$ bits de cada fluxo, quantidade esta que inclui os bits de preâmbulo e os bits entre quadros. O volume de tráfego (em bits) do ij-ésimo fluxo que esteja em espera dentro da ONU no instante em que se inicia a sua n-ésima janela para transmissão é dado por $Q_{ij}[n]$. Considera-se que o tempo de passagem entre dois fluxos de uma mesma ONU é desprezível.

A duração total do n-ésimo ciclo de polling é então dada por $b[n] = hr + \sum_{ij} c_{ij}[n]$, sendo $h = \sum_j h_j$, em que a constante h_j representa o maior tempo de passagem entre a j-ésima ONU e a seguinte, que inclui o tempo de processamento necessário para o chaveamento.

4.2.2. Critério de justiça

Neste trabalho, supõe-se que uma política de escalonamento é justa se ela é capaz de reservar uma taxa mínima de serviço de ρ_{ij} a cada fluxo ij e, ao mesmo tempo, de compartilhar o restante da capacidade da rede entre os fluxos com tráfego em espera segundo sua prioridade. Seja $\mathcal{B}(t)$ o conjunto de fluxos que têm tráfego em espera no instante t. Uma política justa ideal é aquela que destina a cada fluxo $ij \in \mathcal{B}(t)$ uma taxa de

$$s_{ij}(t) = \rho_{ij} + \frac{w_{ij}}{\sum_{lm \in \mathcal{B}(t)} w_{lm}} \left[r - \sum_{lm \in \mathcal{B}(t)} \rho_{lm} - \sum_{hk \notin \mathcal{B}(t)} s_{hk}(t) \right], \qquad (1)$$

em que w_{ij} é o fator de ponderação que representa a prioridade do fluxo ij e r é a capacidade da rede.

Como foi dito na Seção 4, as redes EPON normalmente requerem políticas baseadas em janelas em que as tarefas de escalonamento sejam distribuídas entre a OLT e as ONUs. Neste caso, é difícil obter um algoritmo de escalonamento que efetivamente se aproxime de (1) ao longo dos ciclos. Alternativamente, contudo, é possível propor um critério de justiça menos rigoroso e mais adequado ao caso de políticas distribuídas baseadas em janelas. Suponha que a duração dos ciclos de *polling* seja constante e dada por B bits. Suponha ainda que a rede destine a cada fluxo ij uma janela de transmissão cujo tamanho seja dado por

$$c_{ij}[n] = \begin{cases} \tilde{Q}_{ij}[n], & ij \notin \mathcal{B}[n] \\ \dfrac{\rho_{ij}B}{r} + w_{ij}\xi[n], & ij \in \mathcal{B}[n], \end{cases} \tag{2}$$

em que $\tilde{Q}_{ij}[n] = Q_{ij}[n-1] - c_{ij}[n-1]$ é o tráfego em espera do fluxo ij no final da $(n-1)$-ésima janela de transmissão da ONU j. Note que $\tilde{Q}_{ij}[n]$ é o volume de tráfego em espera que a ONU j utiliza para requerer recursos da OLT no final desse ciclo. O conjunto $\mathcal{B}[n]$ é o conjunto de fluxos que já teriam algum tráfego em espera ao final do n-ésimo ciclo mesmo que nenhum novo quadro chegue entre este ciclo e o anterior. Matematicamente, este conjunto é dado por $\mathcal{B}[n] = \left\{ij \mid c_{ij}[n] < \tilde{Q}_{ij}[n]\right\}$. A variável $\xi[n]$ representa o volume adicional de serviço (por unidade de peso) que deve ser destinado a um fluxo $ij \in \mathcal{B}[n]$ no n-ésimo ciclo, que é dado por

$$\xi[n] = \frac{1}{\sum_{lm\in\mathcal{B}[n]} w_{lm}} \left\{ B - hr - \sum_{lm\in\mathcal{B}[n]} \frac{\rho_{lm}B}{r} - \sum_{hk\notin\mathcal{B}[n]} \tilde{Q}_{hk}[n] \right\}. \tag{3}$$

Seja agora $\mathcal{B}[\cdot]$ o conjunto de fluxos que têm tráfego em espera durante todos os ciclos de $n+1$ a $n+q$. O total de serviço a ser destinado a um fluxo $ij \in \mathcal{B}[\cdot]$ neste intervalo é então dado por

$$\begin{aligned} S_{ij}[n+1; n+q] &= \sum_{k=n+1}^{n+q} c_{ij}[k] \\ &= q\left(\frac{\rho_{ij}B}{r}\right) + \frac{w_{ij}}{\sum\limits_{lm\in\mathcal{B}[\cdot]} w_{lm}} \left\{ q\left(B - hr - \frac{B}{r} \sum_{lm\in\mathcal{B}[\cdot]} \rho_{lm} \right) - \right. \\ &\quad \left. \sum_{hk\notin\mathcal{B}[\cdot]} \sum_{k=n+1}^{n+q} \tilde{Q}_{hk}[n] \right\}. \end{aligned} \tag{4}$$

A taxa média de serviço destinada ao fluxo ij naquele intervalo é dada por

$$
\begin{aligned}
s_{ij}[\cdot] &= \left(\frac{r}{qB}\right) S_{ij}[n+1; n+q] \\
&= \rho_{ij} + \frac{w_{ij}}{\sum\limits_{lm \in \mathcal{B}[\cdot]} w_{lm}} \left\{ r\frac{B - hr^2}{B} - \sum_{lm \in \mathcal{B}[\cdot]} \rho_{lm} - \frac{r}{qB} \sum_{hk \notin \mathcal{B}[\cdot]} \sum_{k=n+1}^{n+q} \tilde{Q}_{hk}[n] \right\} \quad (5)
\end{aligned}
$$

que é parecida com a taxa destinada pela política ideal, dada por (1). Assim, neste trabalho consideramos que uma política é justa se (4) vale para todo fluxo $ij \in \mathcal{B}[\cdot]$.

4.2.3. A política FQSE

A política FQSE visa a dividir os recursos da rede entre os fluxos de maneira justa e independente da ONU a que pertençam. Esta política permite que a rede reserve uma certa quantidade de recursos, na forma de taxa de serviço mínima, a cada um dos fluxos e atribua-lhes uma dada prioridade na concorrência pelos recursos ociosos.

Para atingir esse objetivo, a política FQSE utiliza funções denominadas curvas de serviço, que expressam a quantidade de tráfego de cada fluxo que a rede deve transmitir em um dado ciclo. Tais funções são dadas em termos de um índice não-negativo denominado Parâmetro de Satisfação (PS), que por sua vez representa o quanto a rede é capaz de atender às demandas dos fluxos [Kramer et al. 2004].

Na política FQSE, cada ciclo de *polling* é dividido em uma fase de requisição e uma fase de permissão. Na fase de requisição, a OLT consulta as ONUs para obter a curva de serviço agregada de seus fluxos, que é representada por um conjunto de pontos que são os vértices da sua aproximação por uma função linear por partes. A OLT agrega as curvas de serviço das ONUs para calcular o índice PS da rede para o ciclo em questão, com o que também calcula a duração da janela de transmissão de cada ONU. Na fase de permissão, a OLT atribui as janelas de transmissão por meio de mensagens de permissão, que carregam o índice PS calculado. As ONUs, por sua vez, utilizam esse índice para dividir a capacidade da sua janela de transmissão entre os seus fluxos.

Em [Kramer et al. 2004], estuda-se a complexidade computacional da política FQSE. À parte do custo de se calcular o índice PS, a política FQSE tem complexidade computacional de $\mathcal{O}(n \log n)$ tanto em termos do número de ONUs como do número de fluxos por ONU. Além disso, essa política pode apresentar algum grau de injustiça na divisão de recursos devido a mecanismos utilizados para prevenir bloqueios por quadros que não caibam na janela de transmissão [Kramer et al. 2004, Sect. IV.A] e para aumentar sua utilização [Kramer et al. 2004, Sect. IV.B]. Considerando-se apenas o primeiro mecanismo, os resultados indicam que a diferença entre o volume de tráfego transmitido em um ciclo para fluxos de igual prioridade e igual reserva de taxa mínima é inferior a $2(L-1)$, em que L é o tamanho máximo de um quadro Ethernet.

4.3. Compartilhamento proporcional com reserva de carga

Apesar da simplicidade de (2)–(5), não é trivial elaborar um algoritmo computacional que atinja essa solução. Isto porque não há como se determinar a priori quais são os fluxos que compõem o conjunto $\mathcal{B}[n]$.

Lapsley e Low [Low and Lapsley 1999] estudaram algoritmos de controle de fluxo no tempo contínuo que envolvem problemas de otimização semelhantes àqueles de que se originam (2) e (3). Esses autores propõem a solução do problema por meio do algoritmo adaptativo baseado no gradiente projetado, demonstrando a sua convergência sempre que o conjunto de alocações factíveis for fechado e, nele, a função utilidade for contínua, $\cap$- convexa e monotonicamente crescente [Low and Lapsley 1999]. Dado que (2) e (3) advêm de um problema de otimização com função objetivo logaritmica, que satisfaz estas propriedades, pode-se adotar a mesma estratégia para elaborar um algoritmo de escalonamento para redes EPON.

No algoritmo proposto, denominado Compartilhamento Proporcional com Reserva de Carga (*Proportional Sharing with Load Reservation*, PSLR), cada ONU envia à OLT a demanda total dos seus fluxos e a soma dos fatores de ponderação dos fluxos que têm tráfego em espera no final da sua janela de transmissão. Esses valores são utilizados pela OLT para obter a demanda global da rede e corrigir o tamanho das janelas de transmissão de cada ONU do próximo ciclo. Assim, cada ONU recebe uma janela que corresponde à sua taxa mínima adicionada da parcela da capacidade ociosa que deve ser destinada aos seus fluxos. Para que a ONU possa realizar internamente a divisão justa desses recursos, a OLT envia junto com a mensagem de permissão um indicador que expressa o tamanho da janela por unidade de ponderação que foi utilizado para estimar a parcela adicional de banda a ser destinada a cada fluxo com demanda reprimida. Este indicador é atualizado a cada ciclo com base justamente nas informações enviadas pela ONU, tal que a alocação de banda aos fluxos convirja iterativamente à alocação justa.

4.3.1. Algoritmo computacional

O algoritmo, que é ilustrado na Figura 2, é dividido em uma fase de inicialização e uma fase operacional. Na fase de inicialização, a OLT prepara as ONUs para as transmissões e mede o seu tempo de ida e volta (*round-trip time*, RTT). Na fase operacional, a OLT atribui janelas de tempo para que cada ONU transmita os seus quadros.

Durante a fase de inicialização do algoritmo, a OLT envia a cada ONU uma mensagem inicial para prepará-la para transmissão e para medir o seu RTT. Esta mensagem pode ser encapsulada em uma mensagem GATE do protocolo MPCP ou ainda transmitida diretamente no enlace utilizando seqüências de escape, como proposto por Kramer et al. para a política IPACT [Kramer et al. 2002]. Quando uma ONU recebe aquela mensagem, ela marca os primeiros quadros em espera de cada fluxo ij até um total de $\rho_{ij}B/r$ bits. Os quadros marcados serão transmitidos no primeiro ciclo de *polling* da fase operacional. Vale notar que a ONU não marca os quadros que excederiam o dado limite de bits. Para cada fluxo ij, o total de bits marcados durante a fase de inicialização é representado por $c_{ij}[1]$.

Após marcar os quadros, a ONU envia à OLT uma mensagem de requisição contendo os seguintes campos:

- a identidade lógica da ONU j;
- a demanda total da ONU para o primeiro ciclo de *polling* da fase operacional, que é dado por $C_j[1] = \sum_i c_{ij}[1]$;

- o total de pesos dos fluxos que ainda terão tráfego a transmitir após o primeiro ciclo de *polling* da fase operacional, dado por $W_j[1] = \sum_i w_{ij} \mathbb{I}\left\{\tilde{Q}_{ij}[2] > 0\right\}$, em que $\mathbb{I}\{z\}$ é uma função indicadora que vale 1 quando a condição z é verdadeira e 0 caso contrário.

A mensagem de requisição pode ser encapsulada em uma mensagem REPORT do protocolo MPCP ou enviada diretamente pelo enlace utilizando seqüências de escape. Como mostra a Figura 2, a OLT espera a requisição de cada ONU antes de enviar a mensagem de inicialização para a ONU seguinte. Isto é necessário pois a OLT ainda desconhece o RTT de cada ONU e, por isso, não pode antecipar o envio das mensagens de modo a minimizar os tempos de passagem entre ONUs consecutivas.

Ao receber a mensagem de requisição da última ONU, a OLT inicia a fase operacional do algoritmo enviando uma mensagem de permissão para que a primeira ONU possa transmitir. A OLT calcula ainda o instante de tempo em que deve enviar as mensagens de permissão das demais ONUs para o ciclo corrente. Note que isto só é possível pois a OLT conhece tanto o RTT como a demanda total $C_j[1]$ de cada uma delas. Assim como ocorre no algoritmo IPACT, esse cálculo é feito de modo a antecipar as mensagens e, assim, minimizar o tempo de passagem entre ONUs consecutivas [Kramer et al. 2002].

Seja t_0 o instante de tempo em que a OLT envia a mensagem de permissão da primeira ONU para o n-ésimo ciclo de *polling* . O instante de tempo em que a OLT deve enviar a permissão da ONU j nesse ciclo é dado por

$$t_j = \max \begin{cases} t_0 \\ t_{j-1} + r^{-1} C_{j-1}[n] + G - R_j \end{cases} \tag{6}$$

em que R_j é o tempo de ida e volta da ONU j e G é o intervalo mínimo de tempo necessário para preparar a camada física para as transmissões e para compensar pequenas variações no tempo de ida e volta [Kramer et al. 2002, Kramer et al. 2001, Kramer 2005]. Note que, ao contrário do que ocorre no algoritmo IPACT, a OLT não envia permissões para um novo ciclo de *polling* antes que todas as requisições tenham sido recebidas. Isto é necessário porque a OLT precisa da demanda de todas as ONUs para calcular o tamanho de janela a que corresponde a parcela justa de recursos de cada ONU.

As mensagens de permissão que a OLT envia às ONUs contêm os seguintes campos:

- a identidade lógica da ONU, j;
- o serviço adicional por unidade de peso a ser destinado a cada fluxo, $\xi[n+1]$; e
- a duração, em bits, do ciclo de *polling* anterior, $b[n-1]$.

Assim como as demais mensagens de controle, a mensagem de permissão pode ser encapsulada em uma mensagem GATE do protocolo MPCP ou enviada diretamente pelo enlace utilizando seqüências de escape. Como condição inicial, supõe-se que $b[0] = B$ e $\xi[1] = 0$. Estes valores estão de acordo com o total de bits de cada fluxo que foi marcado na fase de inicialização do algoritmo. Para $n > 1$, $b[n]$ é dado pela diferença entre o valor de t_0 no ciclo de *polling* atual e o seu valor no ciclo anterior, multiplicada pela taxa r. O cálculo do valor de $\xi[n+1]$ é discutido mais adiante.

Quando a ONU j recebe a mensagem de permissão para o n-ésimo ciclo, ela calcula a quantidade de tráfego de cada fluxo ij que será transmitida durante o ciclo

seguinte. Caso a quantidade calculada resulte na fragmentação de um quadro, a ONU pressupõe a sua transmissão integral e utiliza um contador para registrar quantidade extra de bits, que é descontada nos ciclos seguintes para que, em média, garanta-se a alocação justa dos recursos. Note que este mecanismo é similar ao que é utilizado pela política *Deficit Round Robin* [Shreedhar and Varghese 1995].

A quantidade de bits que cada fluxo ij pode transmitir durante o $(n+1)$-ésimo ciclo de *polling* é então dada por

$$c_{ij}^{e}[n+1] = \left[\frac{\rho_{ij} b[n-1]}{r} + w_{ij}\xi[n+1] - d_{ij}[n]\right]^{+}, \tag{7}$$

em que $0 \leq d_{ij}[n] < L$ é o contador do fluxo ij e $[z]^{+} = \max(0; z)$. A ONU então marca os quadros a serem transmitidos durante o $(n+1)$-ésimo ciclo, incluindo o que seria fragmentado se exatos $c_{ij}^{e}[n+1]$ bits fossem transmitidos. O total de tráfego que a ONU j marca para o fluxo ij é dado por $c_{ij}[n+1]$.

Após marcar os quadros de todos os fluxos, a ONU atualiza os contadores dos fluxos fazendo

$$d_{ij}[n+1] = \left[c_{ij}[n+1] + d_{ij}[n] - \frac{\rho_{ij} b[n-1]}{r} - w_{ij}\xi[n]\right]^{+}. \tag{8}$$

Supõe-se que $d_{ij}[n+1] = 0$ sempre que não houver quadros desmarcados do fluxo ij no instante em que os contadores são atualizados. Finalmente, a ONU envia à OLT os quadros que já haviam sido marcados na fase de inicialização (se for o primeiro ciclo de *polling*) ou durante o $(n-1)$-ésimo ciclo da fase operacional. A ONU termina sua transmissão enviando uma nova mensagem de requisição à OLT contendo os seguintes campos:

- a identidade lógica da ONU, j;
- a demanda total da ONU para o próximo ciclo de *polling*, dada por $C_j[n+1] = \sum_i c_{ij}[n+1]$;
- o total de pesos dos fluxos ij que ainda terão tráfego a transmitir após o ciclo seguinte, i.e., $W_j[n] = \sum_i w_{ij} \mathbb{I}\left\{c_{ij}[n+1] < \tilde{Q}_{ij}[n+1]\right\}$.

Após receber a requisição da última ONU, a OLT atualiza o valor de $\xi[n]$ por meio de

$$\tilde{\xi}[n+1] = \begin{cases} \tilde{\xi}[n], & \sum_j W_j[n] = 0, \\ \left[\tilde{\xi}[n] + \eta\left(B - b[n]\right)\right]^{+}, & \sum_j W_j[n] > 0 \end{cases} \tag{9}$$

$$\xi[n+1] = \frac{1}{\sum_j W_j[n]} \tilde{\xi}[n+1], \tag{10}$$

em que o parâmetro $\eta > 0$ corresponde ao passo do algoritmo. Finalmente, a OLT calcula a duração (em bits) do ciclo de *polling* e utiliza (6) para calcular o instante de tempo em que deve enviar uma nova mensagem de permissão a cada ONU. O processo de *polling* então se reinicia.

Em [Pereira 2006, Teorema B.1], demonstra-se que o algoritmo da política PSLR converge para a alocação justa (4) e (5) se o parâmetro η em (9) satisfizer a desigualdade

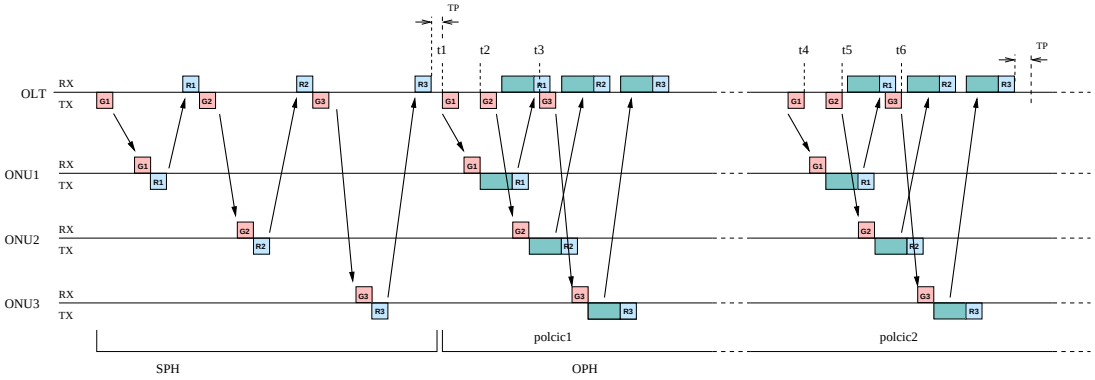


Figura 2. Operação do algoritmo PSLR.

$\eta < 1 + \frac{1}{r}\sum_{ij\in\mathcal{B}[\cdot]}\rho_{ij}$. Dado que o conjunto $\mathcal{B}[\cdot]$ varia durante a operação da rede, supõe-se que $\eta < 1$ é capaz de assegurar a convergência sob qualquer condição.

É fácil ver que a complexidade do algoritmo apresentado é $\mathcal{O}(n)$ tanto em relação ao número de ONUs como em relação ao número de fluxos por ONU [Pereira 2006, Teorema B.2]. Por isso, a política PSLR se mostra bastante atrativa em termos de escalabilidade. Vale notar ainda a sua simplicidade, dado que o algoritmo requer apenas operações aritméticas triviais para atingir a alocação justa de recursos aos fluxos. Ambas as características constituem importantes vantagens em relação à política FQSE, que tem complexidade $\mathcal{O}(n \log n)$ e que requer a execução de um complexo algoritmo de aproximação para obter as curvas de serviço de cada ONU.

4.3.2. Limitante de justiça

Para determinar o quão justa é a divisão de recursos promovida por uma política de escalonamento real, é necessário definir alguma medida que reflita a discrepância entre o serviço que ela oferece aos fluxos e o que deveria ser oferecido no caso ideal. Na literatura, uma das métricas mais utilizadas é o limitante de justiça relativa (*Relative Fairness Bound*, RFB) [Zhou 2003]. No caso de uma política que reserve uma banda mínima para os fluxos, o RFB deve ser calculado de modo a desconsiderar o serviço correspondente à banda reservada aos fluxos ij e lm, já que esta parcela não advém da divisão dos recursos ociosos. Assim, o presente trabalho redefine o RFB como

$$\text{RFB} = \max_{ij,lm\in\mathcal{B}(\tau;t)} \max_{\tau;t} \left| \frac{S_{ij}(\tau;t) - \phi\rho_{ij}}{w_{ij}} - \frac{S_{lm}(\tau;t) - \phi\rho_{lm}}{w_{lm}} \right|, \tag{11}$$

em que ρ_{ij} e ρ_{lm} representam a banda reservada pelos fluxos ij e lm, respectivamente, e $\phi \geq 0$ é o maior valor para o qual as frações são ambas positivas.

No caso da política PSLR, o limitante de justiça relativo satisfaz a desigualdade [Pereira 2006, Teorema B.4]

$$\text{RFB} \;\leq\; \frac{1}{\min_{ij} w_{ij}} \max_{ij} \left[\rho_{ij} \frac{B + 2N(L-1)}{r - \sum_{lm}\rho_{lm}} + B - h + 2(L-1) \right]. \tag{12}$$

A demonstração deste resultado é omitida por restrições de espaço. A partir de (12), é fácil perceber que a diferença entre o serviço total oferecido a um dado fluxo e o que é oferecido a outro é sempre limitada, mesmo que os fluxos pertençam a ONUs diferentes.

Cabe observar ainda que é possível obter um limitante para o RFB menor do que (12) se o intervalo de interesse $(\tau; t)$ compreender n ciclos de *polling* inteiros. Conforme é demonstrado em [Pereira 2006, Teorema B.5], $\mathrm{RFB}_{\mathrm{CC}} \leq 2\,(L-1)\,/\,\min_{ij} w_{ij}$, ou seja, a diferença entre os serviços totais oferecidos a dois fluxos é sempre inferior ao tamanho de dois pacotes, ponderados pelo menor peso atribuído pela rede aos fluxos. Se forem considerados fluxos de igual prioridade e taxa mínima, o valor de $\mathrm{RFB}_{\mathrm{CC}}$ indica que a diferença de serviço entre dois fluxos sob a política PSLR é sempre inferior a $2(L-1)$, valor este similar ao obtido no caso da política FQSE quando se considera apenas o mecanismo para prevenir bloqueios por quadros maiores do que a janela de transmissão.

4.3.3. Latência e atraso

Stiliadis e Varma [Stiliadis and Varma 1998] definiram uma classe de políticas de escalonamento denominada servidores latência-taxa para a qual podem ser obtidos limitantes determinísticos para o atraso e para a latência de fluxos individuais. Uma política pertence à classe dos servidores latência-taxa com parâmetros $(\rho_{ij}; \theta_{ij})$ se o serviço oferecido a qualquer fluxo ij que esteja latente em todo o intervalo $[\tau; t]$ satisfaz a relação $S_{ij}\,(\tau; t) \geq [\rho_{ij}\,(t - \tau - \theta_{ij})]^{+}$, em que ρ_{ij} é a taxa de serviço de longo prazo proporcionada pela política ao fluxo ij e θ_{ij} é o seu parâmetro de latência temporal. Este parâmetro representa o maior tempo necessário para que o escalonamento passe a atender o fluxo ij continuamente à taxa ρ_{ij}.

Para uma política que pertença à classe dos servidores latência-taxa, é possível obter limitantes de pior caso para a latência e para o atraso dos fluxos se o tráfego for policiado pelo algoritmo do balde furado. Tais limitantes são dados por $Q_{ij}(t) \leq \sigma_{ij} + \rho_{ij}\theta_{ij}$ e $D_{ij}(t) \leq \sigma_{ij}/\rho_{ij} + \theta_{ij}$ [Stiliadis and Varma 1998].

É possível demonstrar que a política PSLR pertence à classe dos servidores latência-taxa [Pereira 2006, Teorema B.7]. Nesse caso, o valor de θ_{ij} é dado pelo seguinte teorema:

Teorema 1. *Seja ρ_{ij} a banda reservada a um fluxo ij pela política PSLR. Neste caso, o escalonamento pode ser representado por um servidor latência-taxa $(\rho_{ij}; \theta_{ij})$ em que*

$$\theta_{ij} \leq \frac{1}{r}\left[3B - h + 4N(L-1) + \frac{B + 2N(L-1)}{r - \sum_{kp}\rho_{kp}}\left(r + 2\sum_{lm \neq ij}\rho_{lm}\right)\right]. \qquad (13)$$

Por restrições de espaço, omite-se a prova detalhada deste teorema, que é apresentada em [Pereira 2006, Teorema B.7]. Destaca-se, contudo, que o resultado desse teorema indicam que os limitantes de $Q_{ij}(t)$ e $D_{ij}(t)$ apresentados em [Stiliadis and Varma 1998] podem ser utilizados para caracterizar o máximo tamanho de fila e o máximo atraso de cada fluxo na rede de acesso EPON sob política PSLR.

Fluxo	ρ_i	w_i	ONU 1...8		ONU 9...16		Descrição
	[Mb/s]		t_{on}[s]	t_{off}[s]	t_{on}[s]	t_{off}[s]	
1...6	1	0	0	20	0	30	CBR Básico
7...12	1	1	0	20	0	30	VBR Básico de baixa prioridade
13...18	1	2	10	20	10	30	VBR Básico de alta prioridade
19...24	2	0	0	20	0	30	CBR Premium
25...30	2	1	10	30	10	30	VBR Premium de baixa prioridade
31...36	2	2	0	30	0	30	VBR Premium de alta prioridade
37...42	0	1	0	30	0	30	Melhor esforço de baixa prioridade
43...48	0	2	0	30	0	30	Melhor esforço de alta prioridade

Tabela 1. Parâmetro de serviço dos fluxos.

4.4. Exemplos ilustrativos

Nesta seção, ilustra-se a operação da política PSLR por meio de um exemplo numérico. Considera-se uma rede de acesso EPON com 16 ONUs, cada uma contendo 48 filas independentes com capacidade de 500kb. A distância entre a OLT e as ONUs é uniformemente distribuída entre 1km e 20km, de modo que o tempo de propagação entre os elementos de rede varia de 5μs a 100μs. Para o algoritmo PSLR , supõe-se que a duração desejada para os ciclos de polling (B) seja de 2Mb e que o intervalo mínimo entre as transmissões de ONUs consecutivas (G) seja de 1μs. Estes valores são típicos em simulação de redes EPON. Além disso, supõe-se que o passo do algoritmo PSLR (η) seja igual a 0,1.

Cada fila é utilizada por uma fonte de taxa de bit variável com taxa média de 5Mb/s. Os enlaces que conectam as filas e as fontes têm capacidade de 100Mb/s. Os parâmetros de serviço requeridos por cada fonte e uma descrição sucinta do correspondente perfil de serviço são apresentados na Tabela 1. De modo a verificar o efeito de variações de carga das ONUs na operação do algoritmo PSLR, supõe-se que os fluxos estejam ativos em diferentes intervalos de tempo $[t_{on}; t_{off}]$ que também são apresentados na Tabela 1.

Para cada fonte, o tráfego foi obtido por meio do gerador dado em [Kramer], que agrega 256 subfluxos on/off independentes com períodos de silêncio e de atividade paretianos com $\gamma = 1,3$. O tamanho dos quadros segue a distribuição obtida em uma rede de acesso real por Sala e Gummalla [Sala and Gummalla 2001]. Assim, o tráfego obtido corresponde ao que seria produzido por uma aplicação multimídia real que tenha tráfego com dependência de longa duração e $H = 0,85$.

Os resultados de simulação ora apresentados foram obtidos por meio do simulador Omnet++, em que foram implementadas classes C++ para as ONUs, a OLT e o enlace óptico [Varga]. A taxa destinada pela política PSLR a alguns dos fluxos da ONU 1 e da ONU 9 são apresentados nas Figuras 3 e 4. Estes valores correspondem a taxa média de bits transmitidos em intervalos de tempo não-sobrepostos de 0.1s. Os resultados obtidos para outros fluxos são omitidos por restrições de espaço.

Note que, para cada intervalo $[0; 10\text{s}]$, $[10\text{s}; 20\text{s}]$ e $[20\text{s}; 30\text{s}]$ em que não há variação na carga das ONUs, os valores de taxa obtidos por simulação são próximos aos que seriam obtidos aplicando (5) aos fluxos ativos. Além disso, os mesmos valores são obtidos para fluxos correspondentes tanto na ONU 1 como na ONU 9, mesmo quando o conjunto de

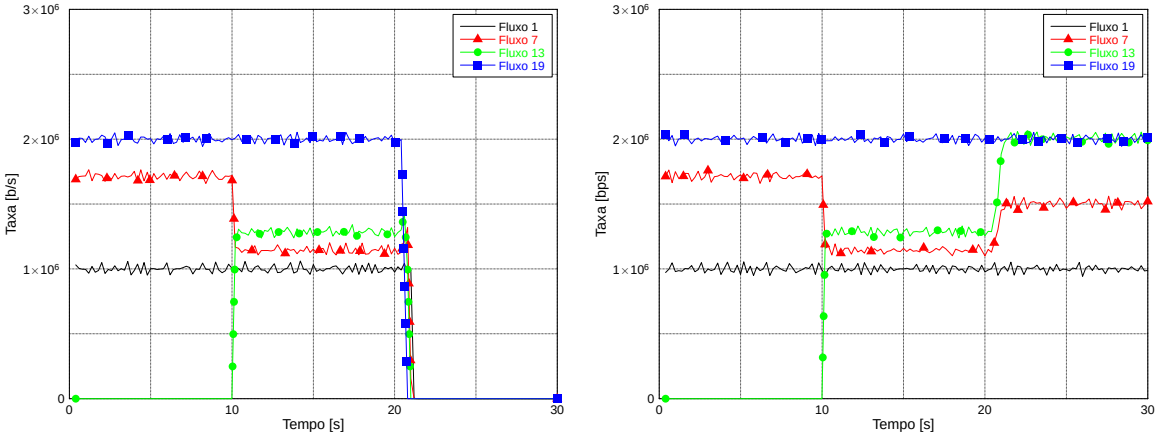


Figura 3. Operação da política PSLR. Resultados para os fluxos 1, 7, 13 e 19 das ONU 1 e 9.

fluxos ativos em cada uma dessas ONUs é diferente. Isso indica que o algoritmo satisfaz o critério de justiça dado por (5) mesmo quando se consideram fluxos de ONUs diferentes.

É possível ainda ver nas Figuras 3 e 4 que o algoritmo converge rapidamente para a nova alocação justa de taxas quando a carga da ONU varia em $t = 10$s. Para $t = 20$s, a resposta é mais lenta pois a rede precisa primeiro transmitir todo o tráfego em espera dos fluxos que se tornaram inativos antes de atingir a nova alocação justa.

A Figura 5(a) apresenta a duração dos ciclos de *polling* medidos durante a simulação. Note que o algoritmo rapidamente converge para a duração desejada, mesmo quando há variação na carga das ONUs. A vazão da rede é apresentada na Figura 5(b), que mostra que cerca de 97% da capacidade da rede é utilizada na transmissão dos quadros. Este ótimo resultado se deve principalmente à antecipação do envio das mensagens de permissão às ONUs, o que reduz o tempo ocioso entre as janelas de transmissão das ONUs. Colaboram também para isso o baixo custo computacional do algoritmo, que reduz o intervalo ocioso entre os ciclos de *polling*,e o reduzido tamanho das mensagens de controle.

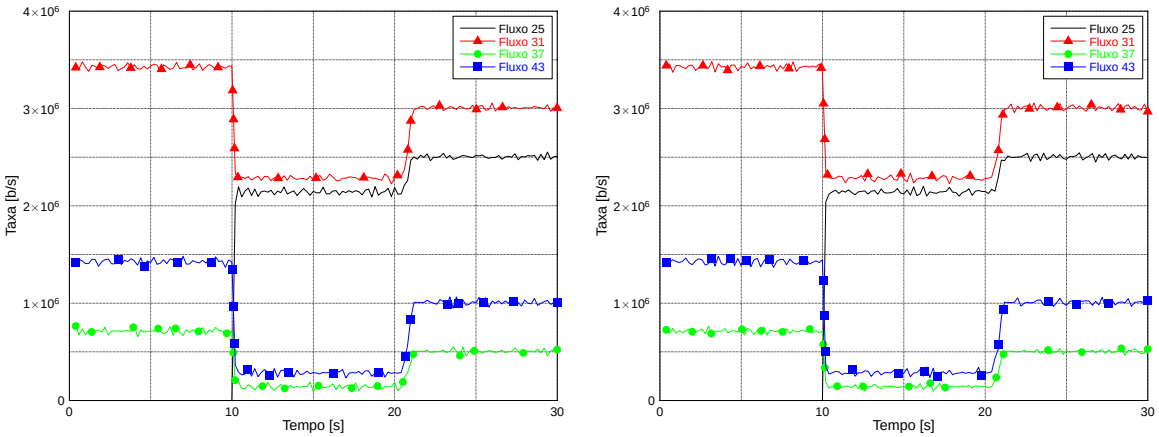


Figura 4. Operação da política PSLR.Resultados para os fluxos 25, 31, 37 e 43 das ONU 1 e 9.

Os resultados de simulação indicam que a política PSLR é efetivamente satisfaz o critério de justiça definido na Seção 4.2. Além disso, ela é capaz de garantir taxa

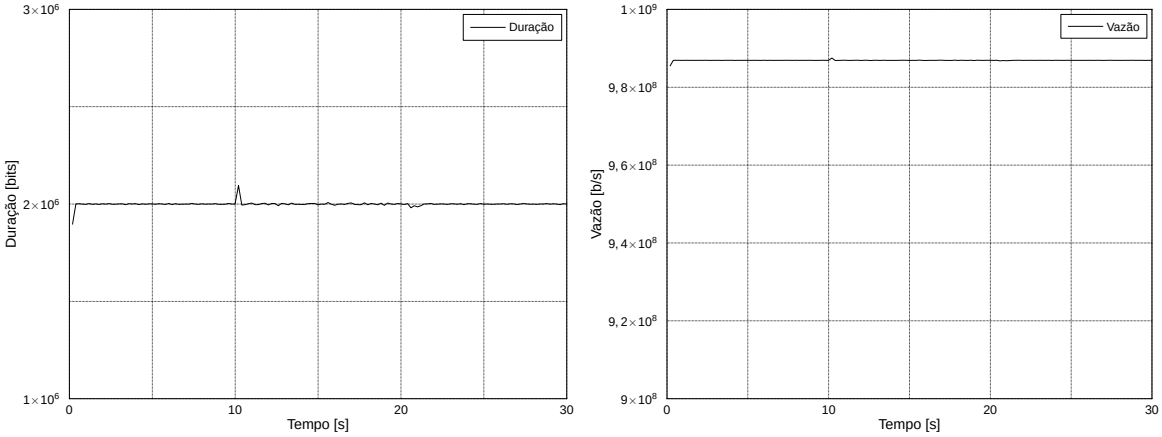


Figura 5. Operação da política PSLR (a) Duração dos ciclos de *polling*. (b) Utilização da rede.

mínima aos fluxos e de distribuir a capacidade ociosa entre eles de maneira eficiente, independentemente da ONU a que pertençam.

4.5. Resumo Conclusivo da Proposta de Escalonamento

Neste relatório, foi proposta uma nova política de escalonamento para redes EPON denominada Compartilhamento Proporcional com Reserva de Carga (PSLR). Esta política apresenta significativas vantagens em relação a outras disciplinas já apresentadas na literatura. As mais importantes são o oferecimento de garantias de desempenho fluxo-a-fluxo, a capacidade de prover uma distribuição justa dos recursos da rede entre os fluxos e o baixo custo computacional. Na política PSLR, cada fluxo pode estabelecer seu próprio contrato de serviço com a rede e receber a banda mínima contratada e uma parcela justa dos recursos ociosos independentemente da ONU em que se origina. A política FQSE proposta em [Kramer et al. 2004] atinge resultados similares, mas a custa de uma maior complexidade computacional. Ademais, a política FQSE não dispõe de expressões analíticas para limitantes de justiça e de retardo dos fluxos. Cabe ressaltar ainda que, assim como a política FQSE, a política PSLR pode também ser empregada em outras redes ponto-multiponto, dado que ela não requer qualquer característica particular das redes EPON para a sua operação.

5. Uso de Grafos Evolutivos em Redes com Comportamento Previsível

Esta seção descreve as tarefas realizadas na “Utilização de Grafos Evolutivos no Desenvolvimento de Protocolos de Roteamento para Redes com Comportamento Previsível” do projeto e realizadas até maio de 2007. Especificamente, as seguintes tarefas foram executadas:

- Implementação de um protocolo de roteamento para redes ad-hoc;
- Extensivas Simulações usando o NS2;
- Análise dos resultados e propostas de melhorias.

Os resultados destas atividades são descritos a seguir.

5.1. Motivação

Uma área de pesquisa muito interessante é a das redes móveis ad hoc, conhecidas como MANETs (Mobile Ad hoc NETworks). Estas são compostas por uma coleção de dispositivos móveis que são conectados dinamicamente entre si e de forma arbitrária, sem a necessidade de infra-estrutura fixa ou administração centralizada [Corson and Macker 1999]. Estes dispositivos móveis com capacidade de comunicação sem fio são chamados *nós*. Quando dois nós precisam trocar informações e não estão ao alcance um do outro, os vizinhos que estão fisicamente entre eles irão cooperar na comunicação, agindo como roteadores de dados, passando adiante a informação recebida até que esta chegue ao seu destinatário. Esta rede é denominada rede de comunicação sem fio ad hoc multi-saltos (*multi-hop*).

Em diversas aplicações estes nós estão livres para se movimentar e podem ter características não uniformes, o que torna estas redes complexas [Estrin et al. 1999]. Este comportamento altamente dinâmico e as suas características típicas [Corson and Macker 1999] como, por exemplo, limitações no consumo de energia, baixa capacidade de processamento, pouca capacidade de comunicação e sujeita a altas taxas de erro, motivaram a pesquisa e o desenvolvimento de inúmeros algoritmos de roteamento [Lang 2003, Royer and Toh 1999, Akyildiz et al. 2002].

O comportamento dinâmico das redes sem fio as torna muito peculiares e de difícil análise. No entanto, algumas destas, como as redes de sensores sem fio (RSSFs) [Akyildiz et al. 2002, Loureiro et al. 2003] e as do sistema de satélites de órbita baixa (LEO) [Ferreira et al. 2002, Werner and Maral 1997] têm um comportamento dinâmico relativamente previsível, pois as variações da topologia no tempo são de alguma forma determinísticas.

Recentemente, um modelo teórico – *Grafos Evolutivos* [Ferreira 2003, Bui-Xuan et al. 2003a, Bui-Xuan et al. 2003b, Bhadra and Ferreira 2002, Bhadra and Ferreira 2003, Ferreira and Jarry 2004] – foi proposto com o intuito de capturar o comportamento dinâmico destas redes e formalizar algoritmos de roteamento de custo mínimo, além de outros. Os algoritmos e idéias obtidos com este modelo são teoricamente muito eficientes, mas, no entanto, não existem estudos do uso destes modelos em situações práticas. Assim, o objetivo desta parte do projeto é analisar a aplicabilidade da teoria de grafos evolutivos na construção de protocolos de roteamento eficientes em cenários realistas.

Um dos objetivos deste projeto de pesquisa é analisar a capacidade de um algoritmo de roteamento baseado no recente modelo combinatório de EG e sua aplicabilidade em redes de sensores sem fio e redes previsíveis em geral.

Até o presente momento já foi realizada a implementação e extensivos testes de um protocolo de roteamento baseado em EG no simulador de redes *NS-2* [VINT Project 2005].

Os resultados preliminares [Monteiro et al. 2006], mostraram que este recente modelo tem muito potencial para ser uma ferramenta poderosa no desenvolvimento e análise de algoritmos para redes dinâmicas com comportamento previsível.

O estudo completo deste modelo, compreende também a utilizacão de teorias de comportamento estocástico para a criacão de protocolos adaptativos.

5.2. Grafos Evolutivos

A grande maioria dos estudos sobre topologia de rede encontrados na literatura hoje estão relacionados a redes estáticas, com isso conceitos já estabelecidos nestas redes, tais como, conectividade, árvores geradoras e caminhos de menor custo, foram revistos para redes dinâmicas e o resultado é o modelo teórico de *grafos evolutivos*.

5.2.1. Redes com Comportamento Previsível

Como visto nas seções anteriores, as redes móveis ad hoc (MANETs) tem um comportamento altamente dinâmico. A comunicação direta entre os nós da rede, a liberdade de movimentação e a não uniformidade de seus participantes tornam o desenvolvimento de algoritmos de roteamento eficientes uma tarefa complexa. No entanto podemos destacar alguns sub-tipos de MANET em que de certa forma torna-se possível capturar a dinâmica das alterações na topologia da rede, este é o caso das redes de satélites (LEO), rede de sensores e outras em que a topologia da rede no decorrer do tempo pode ser pré-determinada. Redes com esta característica foram denominadas: Redes com Comportamento Previsível (FSDNs - *Fixed Schedule Dynamic Networks*) [Ferreira 2003] e são o objeto de estudo neste trabalho.

No caso dos sistemas de satélites LEO [Ferreira et al. 2002, Bui-Xuan et al. 2003a, Werner and Maral 1997], os nós se comunicam através de conexões entre os satélites que estão ao alcance um do outro. Enquanto as conexões estão no mesmo plano orbital (*intraplanar*) a topologia não se altera, pois os satélites tem velocidade angular relativa zero entre si, mas as conexões em diferentes planos orbitais (*interplanar*) fazem com que a topologia se modifique enquanto os satélites se aproximam ou se afastam um dos outros. O resultado disso é uma rede com uma topologia dinâmica, contudo, as trajetórias dos satélites são conhecidas de antemão e portanto a topologia da rede no tempo também o é. Esse determinismo pode ser explorado para otimizar estratégias de roteamento.

Certas redes de sensores sem fio também possuem características similires às redes FSDN, devido às suas limitações de energia, os nós destas redes podem ser programados para se desligar por certos períodos de tempo, com o intuito de economizar recursos. Definidos os horários de funcionamento dos sensores, é possível pré determinar as alterações da topologia da rede no tempo.

5.3. Conceitos

A noção de *grafos evolutivos*, introduzida recentemente [Ferreira 2003], consiste basicamente em formalizar um domínio no tempo em grafos. Surpreendentemente, isto nos leva a inúmeras questões interessantes na teoria dos grafos, com aplicações imediatas no controle da topologia de MANETs, no qual incluem-se as RSSF, relacionando às propriedades de roteamento ótimo e conectividade no tempo. *Grafos evolutivos* representam uma abstração formal de redes dinâmicas, mais precisamente, um *grafo evolutivo* é uma seqüência indexada de τ sub-grafos de um dado grafo, onde um sub-grafo de um certo índice corresponde à rede de conectividade no intervalo de tempo indicado pelo número do índice, como mostrado na figura 7.

O domínio no tempo é posteriormente incorporado ao modelo restringindo as *Jornadas* (i.e, equivalente a *caminhos no tempo*) para aquelas que não contenham arestas que

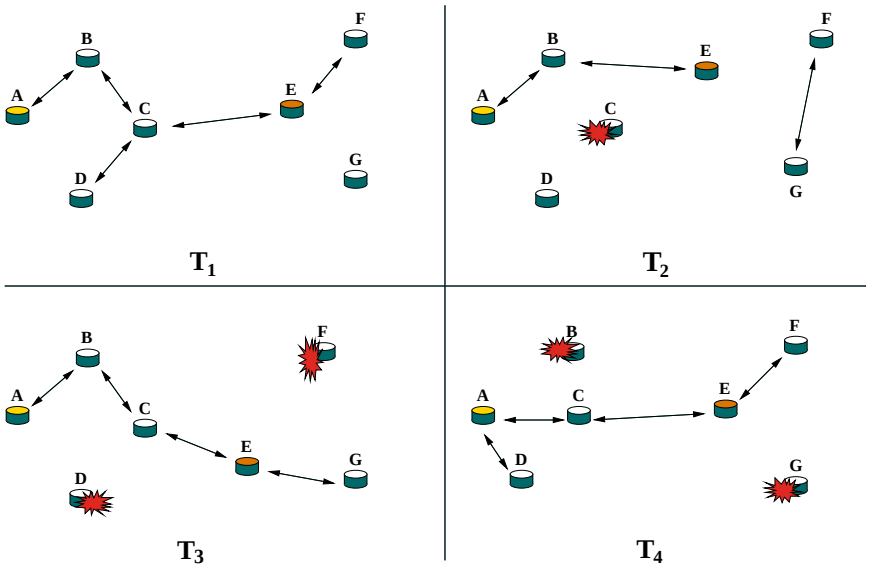


Figura 6. Evolução de uma rede MANET no tempo. Os índices de T correspondem à sucessivos momentos da rede

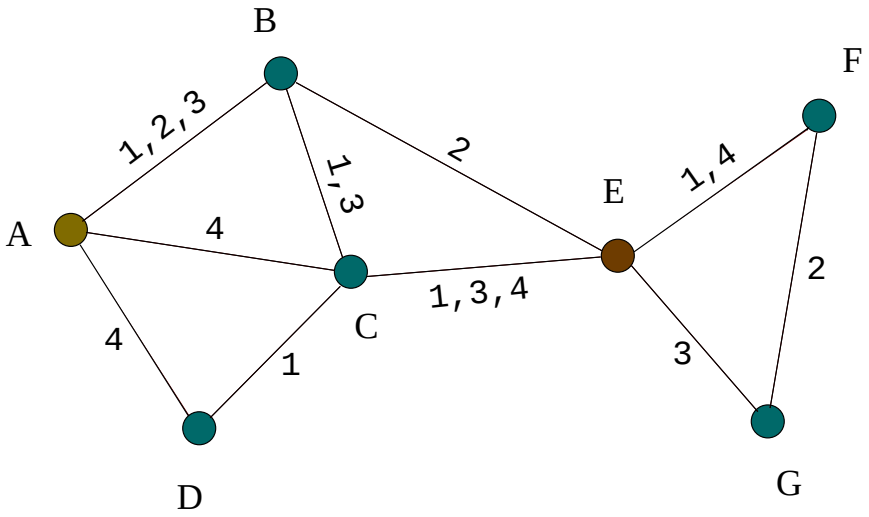


Figura 7. Grafo evolutivo correspondente à MANET da figura 6. As arestas estão nomeadas com os intervalos correspondentes à existência da aresta. Observe que E,G,F não é uma jornada válida, pois a aresta {G,F} existe somente no passado em relação à aresta {E,G}

só existiam em grafos no passado. Com isso uma *Jornada* em um *grafo evolutivo* é um caminho nos grafos subjacentes nos quais o horário de existência das arestas estão em ordem não-decrescente.

Pode-se ver facilmente na figura 7 que **D,C,E,F,G** é uma *Jornada*, assim como **D,C,E,G** também o é, e realiza um número menor de saltos, mas neste último a informação só chega em **G** em um horário posterior (intervalo 3 ao invés de 2). Note que neste exemplo o tempo para percorrer uma aresta é zero (instantâneo), portanto, dados enviados por **A** no instante 1 podem chegar ao nó **F** no mesmo instante 1.

Minimizar a quantidade de saltos ou o horário de chegada são exemplos das

otimizações que podem ser realizadas utilizando este modelo.

Assim como toda teoria em redes de comunicação sem fio, o conceito de *grafos evolutivos* é muito recente, então vemos a necessidade de desenvolvimento de modelos de referência. Os primeiros estudos sobre *grafos evolutivos* [Bui-Xuan et al. 2003a, Ferreira 2003, Ferreira and Jarry 2004], mostraram ser uma relevante ferramenta de modelagem para redes dinâmicas.

5.4. Métricas de Otimização

Com o intuito de desenvolver algoritmos de roteamento eficientes, três métricas de otimização já foram abordadas até então pelos estudos de *grafos evolutivos*. As *jornadas* de custo mínimo (*least cost journeys*) explicitadas em [Bui-Xuan et al. 2003a, Ferreira and Jarry 2004] compreendem a elaboração de algoritmos ótimos em relação às métricas: jornada que chega mais cedo – menor tempo de chegada (*foremost journeys*); jornada mais curta – menor número de saltos (*min-hop count*); jornada mais rápida — menor tempo gasto no percurso (*fastest journeys*). Cada uma delas pode ser calculada em tempo polinomial [Bui-Xuan et al. 2003a].

5.4.1. Foremost Journey *(Jornada que chega mais cedo)*

Jornada que chega mais cedo também pode ser entendida como 'menor horário de chegada' e é utilizada para calcular os trajetos em que um pacote partindo de s chega ao nó u no horário mais cedo possível, independente do número de nós percorridos ou do tempo gasto no percurso. O algoritmo utilizado para o cálculo da *Foremost Journey* é detalhado na seção 5.5.

5.4.2. Min-hop Count Journey *(Jornada mais curta)*

A métrica de 'número mínimo de saltos' pode ser entendida como o trajeto mais curto e compreende a elaboração de algoritmos em que a quantidade de nós percorridos no roteamento de um pacote da origem s ao destino v é o menor possível. A dificuldade em calcular este trajeto se deve ao fato do algoritmo levar em conta o tempo do percurso de uma aresta, e também que prefixos de jornadas mais curtas não são necessariamente jornadas mais curtas (como visto na figura 8).

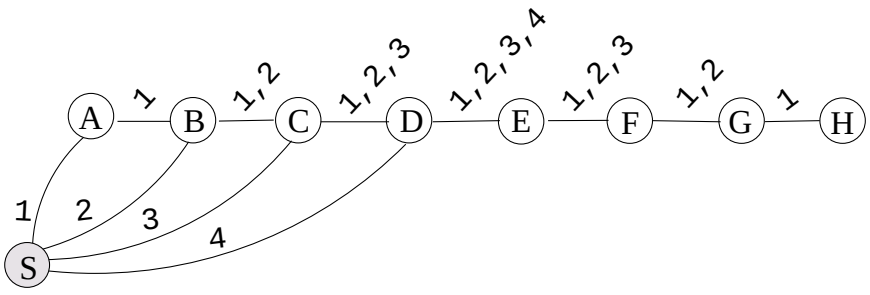


Figura 8. Jornada mais curta de S a H tem 8 saltos e chega no tempo 1, enquanto a jornada mais curta de S a D tem 1 salto mas chega no tempo 4.

5.4.3. Fastest Journey *(Trajeto mais rápido)*

O objetivo nesta métrica é otimizar o tempo gasto no percurso entre a origem s e o nó u. Esta é a métrica mais complexa a ser calculada dentre as três estudadas. O trajeto mais rápido pode estar bem a frente no tempo (chegar mais tarde do que *foremost journey*) e, talvez, tenha um número maior de saltos se comparado com a métrica *jornada mais curta*.

5.5. Algoritmo *Foremost Journey*

O algoritmo *Foremost Journey*, que será utilizado na seção 5.9, calcula a partir da origem s os trajetos que chegam o mais cedo possível em todos os outros nós alcançáveis. Este algoritmo é uma adaptação do conhecido algoritmo de Dijkstra para caminhos de custo mínimo [Cormen et al. 1990].

Lembrando que, para calcular os caminhos de custo mínimo, o algoritmo de Dijkstra procede construindo um conjunto C de vértices *fechados*, para os quais o caminho mais curto já foi calculado, dessa maneira, escolhendo um vértice $u \notin C$ com o custo $d(u)$ mínimo e adicionando u em C, i.e., *fechando* u.

Neste ponto, todos as arestas de u a $V - C$ estão *abertas*, i.e., elas já foram examinadas e seus respectivos custos mínimos, d, já foram atualizados para todos os destinos. Para se ter um acesso rápido ao caminho de menor custo estimado, o algoritmo mantém uma fila de prioridades (*heap*) Q com todos os vértices em $V - C$, com a chave no custo d. Note que d é inicializado com ∞ para todos os vértices, com exceção da origem s que tem custo $d = 0$.

Uma das características do método de Dijkstra é que o custo de caminhos que são prefixos de um caminho de custo mínimo, também são caminho de custo mínimo. Infelizmente, o prefixo de uma *foremost journey* não é necessariamente uma *foremost journey* (Exemplo, considerando o EG na figura 7, uma mensagem enviada de A para G no momento de índice 1 pode utilizar o trajeto **A,B,E,G**. O pacote irá chegar em **G** no momento de índice 3. Note que o trajeto **A,B,E** que é prefixo do anterior irá chegar em **E** no momento 2, mas **A,B,E** não é um *foremost journey* entre **A** e **E**, que seria **A,B,C,E** e chegaria ao destino no momento 1, mas percorrendo um número maior de nós.

No entanto, foi provado em [Bui-Xuan et al. 2003a, Ferreira 2003] que existe pelo menos uma *foremost journey* no EG que satisfaz a propriedade anterior.

Para se calcular o *foremost journey* partindo de s a todos outros nós utiliza-se uma adaptação direta do algoritmo de Dijkstra, a seguir temos um resumo do funcionamento do algoritmo, que é detalhado em [Bui-Xuan et al. 2003a, Ferreira 2003].

1. Inicialize $d(s) = 0$, e $d(u) = \infty$ para todos outros nós.
2. Inicialize *min-heap* Q, ordenado por d, com somente s no topo da heap.
3. Enquanto Q $\neq \emptyset$ faça:
   (a) $x \leftarrow$ raiz do heap Q.
   (b) Remova a raiz do heap Q.
   (c) Para cada vizinho aberto v de x faça:
      i. Calcule o primeira aresta válida com horário de existência maior ou igual ao horário atual

ii. Insira v no heap Q se ele não estiver lá ainda.
iii. Se necessário, atualize $d(v)$ e suas chaves.

(d) Atualize o heap Q.

(e) Feche x. Insira-o na árvore de *foremost journeys*.

Ao final da execução do algoritmo tem-se uma árvore geradora com os trajetos a serem percorridos pelos *foremost journeys* partindo de s a todos os outros nós.

O protocolo de roteamento originado pelo algoritmo acima, que a partir deste ponto será referenciado por $\text{EG}_{Foremost}$ está detalhado na seção 5.9

5.6. Trabalhos Relacionados

Outros trabalhos relacionados a redes dinâmicas (redes que variam no tempo) podem ser encontrados em [Ford and Fulkerson 1958, Köhler et al. 2002], sendo que estes estão, em sua maioria, relacionados ao fluxo em redes quando as suas arestas tem custos que variam no tempo. Kotnyek faz em [Kotnyek 2003] uma revisão geral sobre este tema.

O conceito de grafos expandidos no tempo (*time expanded graph*) definido por Ford e Fulkerson em [Ford and Fulkerson 1958] cria um grafo estático equivalente à partir de um grafo dinâmico (com arestas variando no tempo). Para isso é necessária uma discretização do tempo, e para cada instante t, uma nova *camada* contendo os mesmos vértices do original é adicionado ao grafo, mas as arestas são atualizadas para aquele instante. Feito isso, os algoritmos combinatórios para redes estáticas podem ser aplicados diretamente no grafo expandido. Se as variações nas arestas no tempo não forem discretas, a solução do grafo expandido pode até ser utilizada, mas a complexidade para o cálculo dos caminhos mínimos poderá se tornar inviável. Note portanto que esta solução é viável apenas quando estamos utilizando tempos discretos.

Infelizmente, a abordagem de [Ford and Fulkerson 1958] não se aplica à FSDN, pois o fator de expansão será enorme devido à quantidade de camadas que serão necessárias para representar um grafo com variações não-discretas no tempo. O conceito de grafos evolutivos foi desenvolvido com o objetivo de representar redes dinâmicas e mais especificamente FSDN's.

Em [Köhler et al. 2002] são estudados os grafos expandidos em que o custo das arestas (e.g. tempo de percurso) variam com o tempo e a carga do sistema. Os trabalhos realizados até este momento na teoria de grafos evolutivos está restrito à formalização de uma FSDN e a exploração do modelo com o intuito de diminuir a complexidade de algoritmos conhecidos e criar novos modelos de referência. O conceito de *EG* e algumas métricas de maximização já foram detalhados em alguns artigos, mas, até o presente momento em nenhum destes trabalhos foram realizados testes experimentais ou simulações realistas.

5.7. Experimentação

5.7.1. Simulador de Redes NS2

A parte experimental está sendo realizada com o auxílio do simulador de redes NS2 (*Network Simulator 2*) [VINT Project 2005] desenvolvido em Berkeley. O NS2 [VINT Project 2005] é um simulador baseado em eventos discretos, e foi desenvolvido

para auxiliar a pesquisa na área de redes de comunicação. É um projeto muito abrangente e genérico, que permite a simulação de várias camadas de rede, incluindo protocolos de roteamento e transporte (TCP, UDP, ICMP, etc...) em redes com ou sem fio.

Esta ferramenta vem sendo desenvolvida desde 1989, e em 1995 teve o apoio da DARPA através do projeto VINT (Virtual InterNetwork Testbed) e hoje está sob a licença de software livre (GPL). O seu desenvolvimento continua pleno e é realizado por diversos pesquisadores ao redor do mundo.

Inicialmente o projeto não contemplava a simulação de redes sem fio ou redes móveis. Esta extensão foi desenvolvida pela CMU (Carnegie Mellon University) no projeto Monarch [Rice University Monarch Project 2005] e agregou as características do padrão IEEE 802.11, protocolos de acesso ao meio (MAC) [IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications 2005] e modelos de transmissão via rádio.

5.7.2. Arquitetura e Funcionamento

O simulador é implementado utilizando duas linguagens: C++ e a linguagem de *script* OTcl [1]. A parte de simulação de eventos e protocolos, que precisa ser eficiente, foi desenvolvida em C++, já a parte 'configurável', ou parte 'do usuário', é feita em OTcl, que é uma linguagem simples e por ser interpretada deixa o processo de criação e execução de testes mais ágil e menos suscetível a erros.

Uma simulação no NS2 segue o fluxo representado na figura 9, sendo que o usuário precisa criar um *script* na linguagem OTcl definindo as características da simulação, que são:

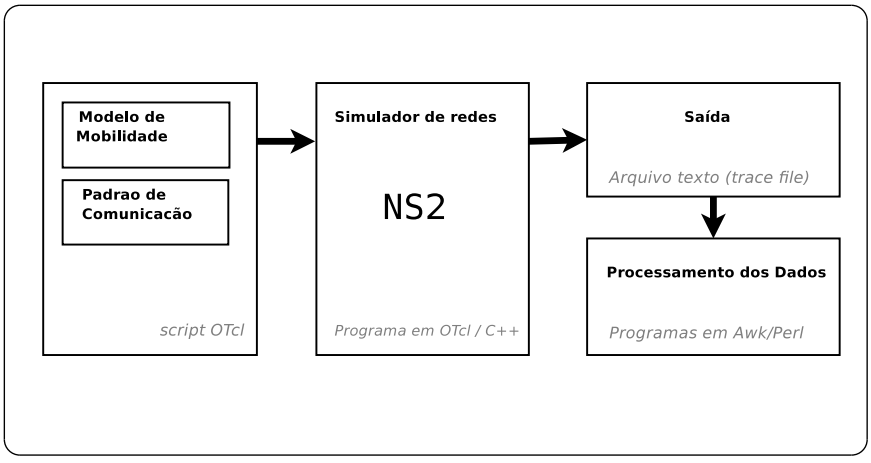


Figura 9. Fluxo de uma simulação no NS2

1. **Modelo de mobilidade** : Seqüência de movimentos bem definidos para os nós da rede (algo como: no momento t o nó n deve se deslocar para a posição x, y com velocidade v), esses movimentos geralmente são gerados por um programa auxiliar, que recebe as características da mobilidade e gera uma seqüência de comandos OTcl representando aquele modelo.

[1] Object Tool Command Language

2. **Padrão de comunicação** : Seqüência de comandos que definem os enlaces de comunicação entre os nós da rede, definem o fluxo e o tipo da comunicação que será realizada (algo do tipo: a partir no instante t, o nó s irá transmitir dados para r, utilizando o protocolo FTP com uma taxa de transmissão de 2 $pacotes/seg$).
3. **Características da rede** : Tipo de rede (com fio, sem fio), número de nós, protocolo de roteamento, modelo de mobilidade, padrão de comunicação, alcance da antena, modelo de energia, modelo de falhas, etc.

O resultado de uma simulação no NS2 é um arquivo texto (*trace file*) contendo detalhes de toda a comunicação realizada entre os nós da rede, desde as camadas mais altas (aplicação) até as trocas de mensagens para acesso ao meio (MAC). Este arquivo, que potencialmente tem um tamanho grande (centenas de megabytes), deve ser posteriormente analisado através de programas auxiliares (geralmente *scripts* Awk ou Perl) para obtenção de dados do tipo: quantidade de pacotes perdidos, atraso no recebimento, vazão, etc.

É importante notar que uma simulação no NS2 é totalmente reprodutível, isto é, o simulador apenas interpreta os *scripts* de entrada que contêm toda a seqüência da simulação e comunicação entre os nós. Para se simular, por exemplo, diferentes padrões de mobilidade, é necessário criar diversos *scripts*, sendo que cada um deles contém a instância de uma simulação.

5.8. Ambiente de Simulação

Para analisar o comportamento deste recente protocolo estamos utilizando a versão 2.29 do NS2 , com extensões para redes móveis desenvolvidas pelo grupo CMU Monarch e características de uma rede IEEE 802.11 para transmissão de rádio, camada física e controle de acesso ao meio (MAC). O modelo de comunicação utiliza características similares ao produto comercial *Lucent WaveLAN*, modelado como meio compartilhado (*shared-media*), com uma banda nominal de 2 MB/s e raio de propagação nominal médio de 250m. Em todas as simulações, 50 nós foram espalhados aleatoriamente e de forma uniforme em um campo com dimensões de 1500m x 500m. A idéia de definir um campo retangular é com o intuito de aumentar o número de saltos necessários no roteamento de informações de uma ponta a outra. A duração das simulações é de 600 segundos.

Por enquanto, não está sendo utilizado TCP para comunicação de dados, pois não estamos interessados em analisar o protocolo TCP e suas características como: controle de fluxo, retransmissão, etc., mas sim uma visão geral de como os protocolos se comportam utilizando comunicação simples dados, como o UDP.

Para simular a transmissão de dados foram criados 10 fluxos de comunicação do tipo UDP entre pares de nós, com taxa de transmissão constante (CBR - Constant Bit Rate) de 2 pacotes/segundo e tamanho de 256 bytes cada pacote. Estes valores são similares aos usados em artigos conhecidos de análise de desempenho neste tipo de rede: [Corson and Macker 1999, Broch et al. 1998, Das et al. 2000, Perkins et al. 2001, Johansson et al. 1999].

Não esperamos que este modelo seja uma precisa reprodução de uma FSDN ou RSSF real, mas desejamos neste primeiro passo comparar o protocolo de roteamento EG$_{Foremost}$ em relação aos protocolos bem difundidos para redes móveis em geral (DSDV, DSR, AODV).

O modelo de mobilidade da simulação foi dividido em dois tipos de cenários distintos: o simples e bem conhecido *Random Waypoint* (RWP) [Johnson and Maltz 1996] apresentado por Johnson e Maltz, e o modelo *Intermitente*, mais apropriado a redes FSDN que é definido mais adiante.

5.8.1. Modelo de Mobilidade Random Waypoint

No cenário de mobilidade *Random Waypoint* (RWP) os nós estão alternadamente se movimentando e fazendo breves pausas. A duração destas pausas é definida pela variável PAUSETIME. Após a pausa o nó irá continuar sua movimentação para uma nova posição escolhida aleatoriamente dentro do campo. A velocidade de locomoção é escolhida de forma aleatória entre os valores 1 e 19 *m/s*. As simulações neste cenário foram realizadas utilizando valores de PAUSETIME variando de 0 segundo (sem pausa, alta mobilidade) a 500 segundos (longas pausas, baixa mobilidade). Para evitar problemas discutidos em [Yoon et al. 2003] sobre o modelo RWP, está sendo utilizado somente valores de velocidade não nulos.

5.8.2. Modelo de Mobilidade Intermitente

O modelo *Intermitente*, definido neste projeto, baseia-se em nós com posição fixa e que ininterruptamente se desligam/ligam por certos períodos. Aqui, os parâmetros que são variados nas diferentes simulações são: SLEEPPROBABILITY, que varia de 0 a 50%, e HOLDTIME, que varia de 6 a 180 segundos. No início da simulação todos os nós estão ligados (acordados), e por toda a simulação estes terão a probabilidade SLEEPPROBABILITY de se desligar (dormir), ao dormir, eles ficarão neste estado por um tempo uniformemente aleatório de HOLDTIME. Assim que este tempo expira, o nó acorda e após um determinado tempo acordado começa o processo novamente. Este modelo foi chamado de *Intermitente* e tem a intenção de reproduzir mais fielmente o modelo de redes de sensores, no qual os nós podem ficar indisponíveis por determinado tempo (por economia de energia, horários de funcionamento, etc.) e depois voltar à operação normal. Este modelo também representa de forma mais clara uma rede com comportamento previsível (FSDN).

5.8.3. Simulações

Para cada combinação de parâmetros foram realizadas 5 simulações com diferentes sementes aleatórias. Assim, para o primeiro modelo, RWP, foram executadas 40 simulações, i.e., 5 vezes para cada valor de PAUSETIME: 0, 5, 25, 50, 125, 250, 375, 500 segundos. Para o segundo modelo, *Intermitente*, foram realizadas 180 simulações, sendo 5 vezes para cada combinação de SLEEPPROBABILITY: 0, 10, 20, 30, 40, 50% e HOLDTIME: 6, 15, 30, 60, 120, 180 segundos. Note que no segundo modelo de mobilidade estamos variando 2 parâmetros diferentes nas simulações.

Todos os 4 protocolos de roteamento (DSDV, DSR, AODV, $\text{EG}_{Foremost}$) foram executados nos mesmos 220 cenários. O mesmo exato cenário e fluxo de comunicação foram utilizados nas simulações dos diferente protocolos.

A implementação utilizada para simular os protocolos de roteamento, com exceção do EG$_{Foremost}$, foi a padrão fornecida no pacote do NS2 . Todos eles foram implementados pelo grupo CMU Monarch [Rice University Monarch Project 2005]. As constantes e os parâmetros não foram modificados. A implementação do protocolo AODV, na versão 2.29 do NS2 , já inclui algumas otimizações implementadas por *Marina e Das* [AODV code for CMU Wireless and Mobility Extensions to NS-2 2005, Perkins et al. 2001].

5.9. Protocolo de Roteamento Baseado em Grafos Evolutivos – EG$_{Foremost}$

Para testarmos o algoritmo de roteamento *Foremost Journey* mostrado na seção 5.5, foi implementado um protocolo de roteamento (EG$_{Foremost}$) como lá descrito e alguns programas auxiliares.

Como já mencionado, o modelo de mobilidade no NS2 é representado por uma seqüência de comandos e armazenada em um arquivo separado que é usado durante a simulação. O algoritmo de roteamento da seção 5.5 precisa receber como entrada um grafo evolutivo (representação das modificações da topologia da rede no tempo), para isso foi criado um programa – **mobility2eg** – que captura a movimentação dos nós durante uma simulação prévia e gera um *EG* correspondente.

Antes da simulação começar, o *EG* é distribuído entre os nós da rede. Com isso, cada nó sabe de antemão todo o comportamento da rede durante o período da simulação, i.e., cada nó sabe o exato momento em que um vizinho estará disponível para comunicação ou quando deixará de estar.

Em um primeiro momento, esta importante suposição não parece ser realista, no entanto, existem diversas situações ([Akyildiz et al. 2002, Ferreira et al. 2002, Estrin et al. 1999]) em que o *EG* pode ser construído antes da fase de roteamento.

5.9.1. Implementação

Dado que todos os nós da rede possuem uma cópia do *EG*, a implementação do protocolo no NS2 é trivial, como mostrada a seguir.

Seja ARESTA-HORÁRIOS um conjunto de intervalos de tempo representando o horário de existência de um canal de comunicação entre dois nós. Uma aresta existe quando dois nós estão ao alcance um do outro, isto é, um enlace de comunicação pode ser estabelecido pois os nós estão fisicamente próximos. O alcance da transmissão foi definido em 250 metros para todos os nós. O grafo evolutivo *(EG)* de uma rede dinâmica pode ser representado por uma lista de ARESTA-HORÁRIOS para cada par de nós, em outras palavras, cada nó tem uma lista de todos os vizinhos alcançáveis em um determinado instante.

Quando um pacote chega ao nó u, este calcula todo o trajeto a ser percorrido pelo pacote desde a origem (s) a todos os outros nós da rede, utilizando o algoritmo *Foremost Journey* (como mostrado na seção 5.5). Seja v o próximo nó trajeto depois de u. Se v estiver disponível (i.e. existe uma aresta $u - v$ em ARESTA-HORÁRIOS naquele instante) então o pacote é enviado a v. Caso v não seja alcançável naquele momento, então o nó u irá armazenar o pacote e agendar a transmissão de v para o próximo instante possível em ARESTA-HORÁRIOS(U,V).

Aresta	Horários
A – B	1, 2, 3
A – C	4
A – D	4
B – C	1, 3
B – E	2
C – D	1
C – E	1, 3, 4
E – F	1, 4
E – G	3
F – G	2

Tabela 2. ARESTA-HORÁRIOS PARA O EG DA FIGURA 7

A tabela 2 mostra os horários de existência das arestas do exemplo da figura 7 (página 17) . Note que os horários de existência são os intervalos mostrados nas arestas do grafo, e compreendem os instantes em que a comunição pode ser estabelecida entre os nós. Como mencionado anteriormente, neste exemplo não estamos levando em conta o tempo gasto na transmissão, portanto, um pacote enviado em um determinado instante pode chegar ao destino no mesmo exato instante, caso o trajeto exista. Na implementação do $EG_{Foremost}$ no NS2 cada intervalo em ARESTA-HORÁRIOS possui um horário exato de início e término, e o tempo gasto na transmissão é levado em conta.

Como exemplo, o *Foremost Journey* para uma mensagem enviada no momento 1 de **D** a **G** será **D,C,E,F,G**. O pacote irá chegar em **F** no mesmo momento 1, este irá agendar o envio do pacote para **G** no próximo momento em ARESTA-HORÁRIOS(F,G), que será o instante 2. Se existe mais de uma rota possível naquele momento, aquela com menor número de nós (mais curta) será escolhida.

Por simplicidade na nossa implementação, cada nó do percurso irá recalcular *todo* o trajeto da origem ao destino, para depois escolher para qual vizinho (e quando) irá retransmitir os dados. Feito desta maneira, o protocolo implementado possui uma sobrecarga de processamento, pois todos os nós do percurso irão calcular todo o trajeto do pacote. Este abordagem foi utilizada para minimizar possíveis erros de implementação de um protocolo mais complexo, no entanto, para a próxima fase pretendemos otimizar esta parte.

É possível, portanto, com poucas modificações alterar o protocolo para que somente o nó que originou o pacote precise calcular a rota de envio, feito isto ele poderia utilizar a técnica de *Source-Routing* e adicionar no cabeçalho do pacote o trajeto completo a ser percorrido. Cabendo então aos nós no meio do percurso apenas retransmitir o pacote para o próximo da lista no momento apropriado.

5.10. Resultados Preliminares

Cada simulação executada gera uma arquivo para posterior análise, contendo toda a comunicação realizada entre os nós, incluindo a camada MAC. Este arquivo é então analisado e os resultados são consolidados.

Assim como os modelos de mobilidade, os resultados mostrados aqui estão dividi-

dos em duas partes, a primeira utilizando o modelo RWP e a segunda utilizando o recente modelo *Intermitente*. Em ambos, as seguintes métricas quantitativas e qualitativas serão observadas:

- **Vazão média**: Número médio de pacotes que chegam ao destino por unidade de tempo;
- **Atraso médio origem-ao-destino**: O tempo médio do percurso de todos os nós entre o envio até o recebimento dos pacotes;
- **Razão de pacotes perdidos por falta de rota (NRTE)**: Razão de pacotes perdidos por *não existir uma rota* (NRTE) em relação ao número total de pacotes enviados;
- **Razão de pacotes perdidos por fila cheia no enlace (IFQ)**: Razão de pacotes perdidos por IFQ em relação ao número total de pacotes enviados.

O protocolo EG$_{Foremost}$ utilizado nestes primeiros experimentos não possui nenhum sobrecusto relativo a pacotes de controle, pois da maneira como foi implementado, cada nó da rede contém uma cópia do EG, não sendo necessária nenhuma comunicação extra entre os nós. Portanto, não estamos interessados neste primeiro momento em analisar o sobrecusto do protocolo.

5.10.1. Resultados - Random Waypoint

Como mencionado anteriormente, no cenário *Random Waypoint* (RWP) parâmetro que será alterado é o PAUSETIME. Sendo que valores pequenos de PAUSETIME significam *alta mobilidade*, e em contrapartida, valores grandes de PAUSETIME significam *pouca mobilidade*. Como pode-se ver na figura 10, o protocolo EG$_{Foremost}$ tem melhor desempenho na métrica de vazão para todos os valores de PAUSETIME, apesar do fato que todos os protocolos obtiveram valores bem próximos nesta métrica (o DSDV foi omitido do gráfico pois seu desempenho foi muito baixo, em torno de 13 kb/segundo). Na verdade, como esperado, o EG$_{Foremost}$ obteve o melhor resultado nesta métrica em todas as simulações em ambos cenários.

Na figura 11, o número de pacotes perdidos pelo EG$_{Foremost}$ foi praticamente zero para todos os valores de PAUSETIME, isto é, menos de 0,2% de perda de pacotes. A razão de pacotes perdidos no DSR foi muito boa também, com uma média de 2,5% de perdas. Foi uma surpresa o fato do AODV não desempenhar tão bem em valores de alta mobilidade, se opondo aos valores mostrados por Perkins, Royer, Das, and Marina em [Perkins et al. 2001]. Nos atribuímos este comportamento às características dos nossos testes, que possuem cargas relativamente baixas de comunicação (10 canais de tráfego com 2 pacotes/segundo cada), ou também se deva por algum ajuste por nós desconhecido nos parâmetros do algoritmo.

O objetivo da métrica de jornada foremost é calcular as jornadas que chegam ao destino no menor tempo possível. Mas, ao calcularmos o atraso médio fim-a-fim levando em conta todos os pacotes entregues, os valores obtidos pelo EG$_{Foremost}$ serão muito altos, devido ao comportamento do algoritmo baseado em EG de ”segurar”pacotes por um longo período até que uma aresta exista. Este tempo despendido, que é potencialmente longo faz com que não sejamos justos com o EG$_{Foremost}$, pois outros protocolos estariam jogam pacotes fora ao invés de aguardar por uma aresta.

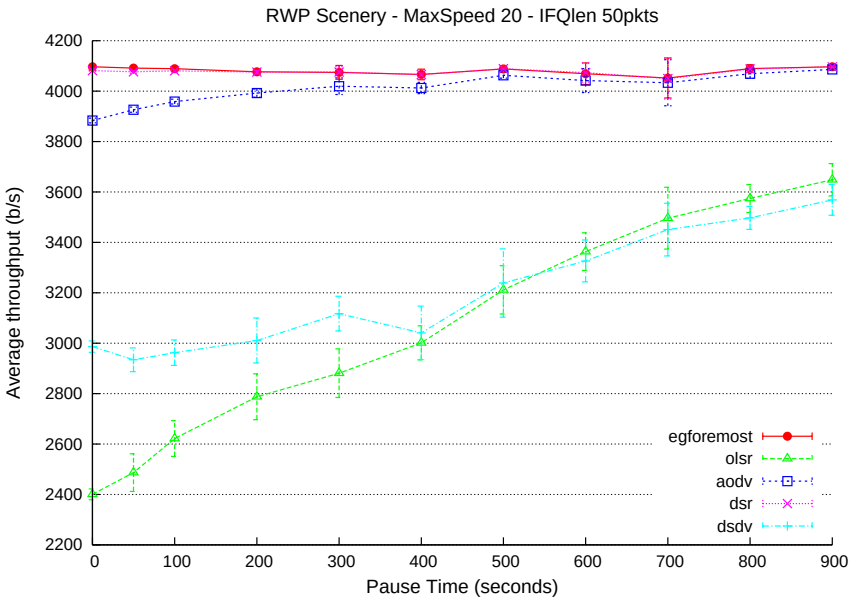


Figura 10. Vazão média em função do PAUSETIME (mobilidade).

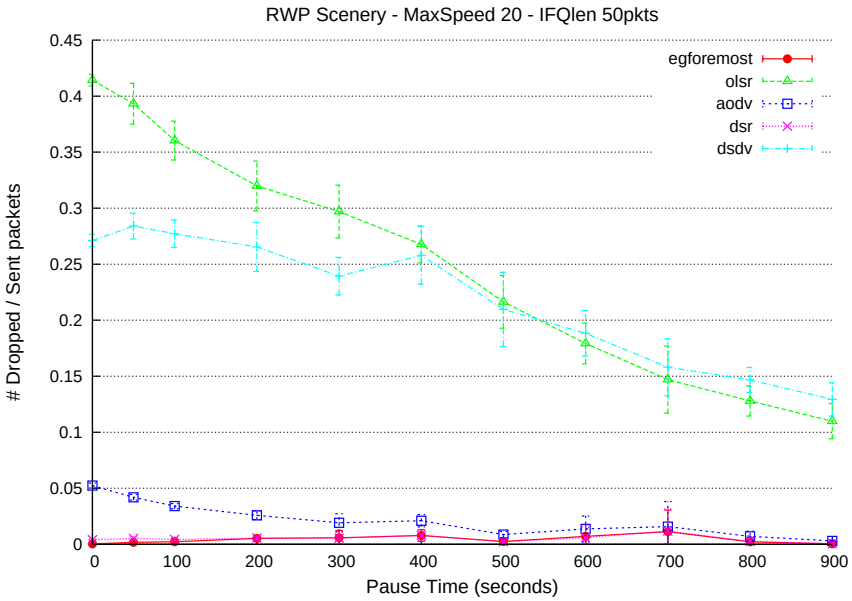


Figura 11. Número de pacotes perdidos em função do PAUSETIME (mobilidade).

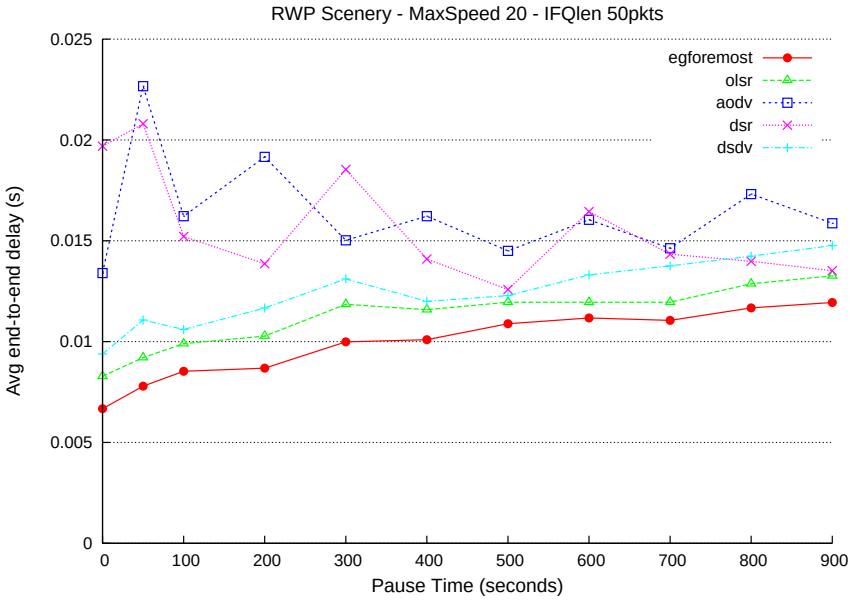


Figura 12. Atraso médio fim-a-fim para pacotes entregues (intersecção dos que foram entregues com sucesso em todos os protocolos)

Em outras palavras, quando usamos os algoritmos baseados em EG, os pacotes podem demorar muito para chegar ao destino. Apesar de estar provado [Bui-Xuan et al. 2003a] que o $\text{EG}_{Foremost}$ entrega os pacotes o mais rápido possível.

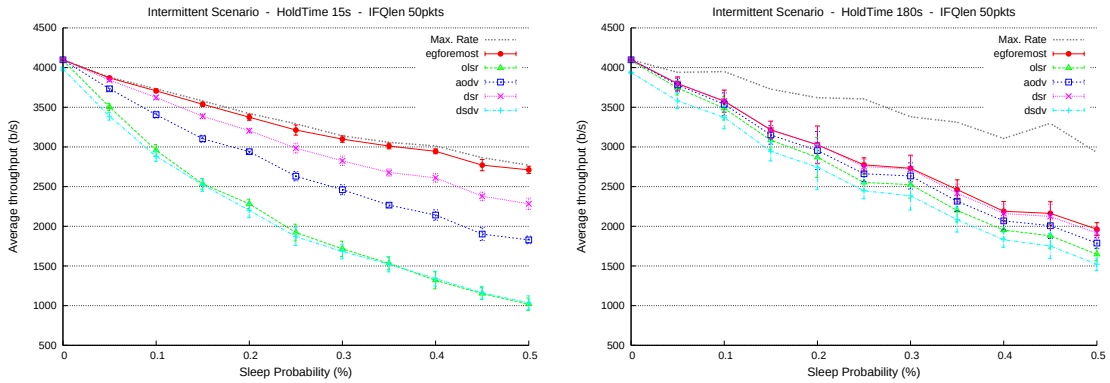
Logo, para ser justos com o $\text{EG}_{Foremost}$ e conseguir medir o desempenho deste em relação aos outros protocolos, calculamos o atraso médio fim-a-fim levando em conta somente os pacotes que são entregues com sucesso em todos os protocolos (i.e. intersecção dos pacotes recebidos). Os resultados na Fig. 12 mostram que o $\text{EG}_{Foremost}$ é um limitante inferior nesta métrica. Além disso,

5.10.2. Resultados - Modelo Intermitente

O cenário intermitente se mostra mais realista para o caso de redes FSDN, no sentido que os nós tem posições fixas e os momentos em que estes se ligam/desligam podem ser previsíveis.

Nos próximos testes, estamos variando os valores de SLEEPPROBABILITY de 0 a 50%. Alta probabilidade para dormir significa que a rede tem baixa conectividade, i.e., as conexões entre os nós são esparsas, vários nós estão desconectados uns dos outros pois vários estão dormindo. O outro parâmetro, HOLDTIME significa dinamicidade, i.e., quão freqüente as conexões entre os nós são alteradas, valores baixos de HOLDTIME significam alta dinamicidade, e vice versa.

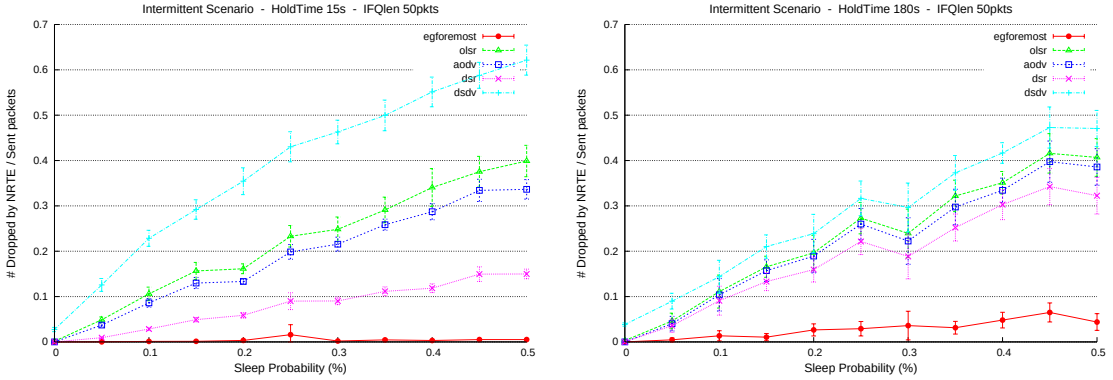
Na figura 13, novamente está sendo mostrado que o $\text{EG}_{Foremost}$ obteve melhor desempenho na métrica de vazão média. Mas, note que, no caso de valores altos de dinamicidade (HOLDTIME 15 seg) a vazão do $\text{EG}_{Foremost}$ é praticamente constante para todos os cenários de conectividade. Por outro lado, no caso de baixa dinamicidade, os valores da vazão decaem juntamente com os outros protocolos à medida que a conectividade cai



(a) HOLDTIME 15seg (b) SLEEP TIME 180seg

Figura 13. Vazão média em função de SLEEPPROBABILITY (conectividade).

(de 0 para 50% de SLEEPPROBABILITY).



(a) HOLDTIME 15seg (b) HOLDTIME 180seg

Figura 14. Razão de pacotes perdidos por NRTE em função de SLEEPPROBABILITY (conectividade).

Esta diminuição acentuada do $\text{EG}_{Foremost}$ deve-se primeiramente à real inexistência de rotas, como mostra o gráfico da figura 14(b). Os valores altos para o número de pacotes perdidos por falta de rota (NRTE) significam que os nós estão realmente desconectados (é o caso dos pequenos valores de conectividade), e a vazão média naturalmente será menor.

Os valores do $\text{EG}_{Foremost}$ na figura 14 mostram que o número de pacotes perdidos por não existir rota (NRTE) do $\text{EG}_{Foremost}$ é o limite inferior para essa métrica para todos os outros protocolos, i.e., se o $\text{EG}_{Foremost}$ joga fora um pacote por NRTE, significa que este caminho não existe e não existirá em nenhum instante. Contudo, o protocolo $\text{EG}_{Foremost}$ pode ser usado como referência para estabelecer quão eficientes estão sendo os outros protocolos.

O comportamento intrínseco do $\text{EG}_{Foremost}$ de agendar pacotes para serem enviados somente quando o nó vizinho acorda, trás a tona o problema dos *gargalos*, em que vários nós agendam o envio de pacotes para mesmo exato momento, que é o primeiro instante posterior ao que o nó em questão é ativado, assim a fila de pacotes do canal (IFQ)

não suporta esta rajada de dados e vários pacotes são jogados fora. Para minimizar este problema de uma forma simplista, aumentamos o tamanho desta fila no NS2 de 50 para 500 pacotes. A simulação realizada com o modelo RWP são as que menos sofreram deste efeito (*gargalos*), atribuímos isso à alta mobilidade do sistema. Este problema se acentuou nos cenários de baixa mobilidade e baixa conectividade do modelo *intermitente*, no qual os nós permanecem desconectados por longos períodos e, portanto, uma quantidade maior de dados é agendada para posterior envio.

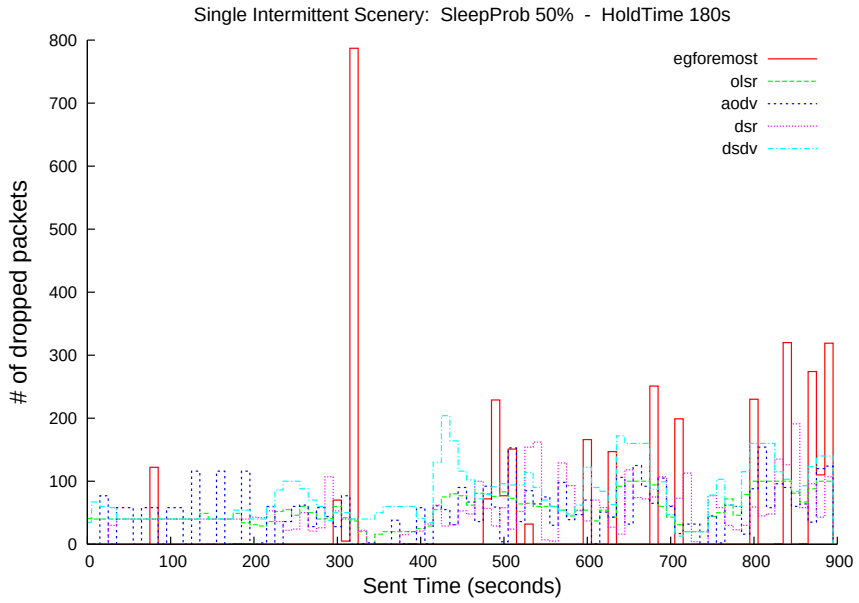


Figura 15. Number of dropped packets over time for one single Intermittent Scenario (SLEEPPROBABILITY 50% and HOLDTIME 180s).

Para ilustrar o problema dos gargaldos, a Fig. 15 contém um histograma com o númeor de pacotes perdidos no tempo em um cenário com SLEEPPROBABILITY of 50% e SLEEPPROBABILITY of 180s. Nele podemos observar que os pacotes são perdidos em forma de rajada (e.g. 800 pacotes são perdidos entre os instantantes 310 e 320s), e novamente, devido a algum ou alguns nós que dormem por longos períodos de tempo, e muitos pacotes são agendados para serem enviados no instante em que estes nós acordam. Logo, estes pacotes são jogados fora pelo enchimento da fila na interface de rede (IFQ).

O problema da fila cheia (IFQ) pode ser minimizado usando um artifício conhecido em simulação de redes, o *jitter*, que é um simples atraso induzido no envio do pacote, com valores escolhidos aleatoriamente dentro de um pequeno intervalo pré definido. Mas, nos nossos testes para que algum resultado fosse observado, o valor máximo do intervalo foi da ordem de 0,5 segundo, valor este que pode não ser plausível em aplicações tradicionais (estaríamos adicionando um atraso médio de 0,25 segundo em cada pacote enviado).

Na Fig. 16 vemos que os valores de perda de pacotes por IFQ são bem altos em um cenário de baixa conectividade (35% dos pacotes). Esses valores afetam negativamente a métrica de vazão do $EG_{Foremost}$ neste tipo de cenário, como já mostrado na Fig. 13.

Os resultados obtidos induzindo um jitter entre 0 e 0,5s podem ser observados na Fig. 17. *EG original* é a curva de referência, (assim como observado na Fig. 16). Obtivemos uma melhora de 30% em relação aos pacotes perdidos, porém, por induzir

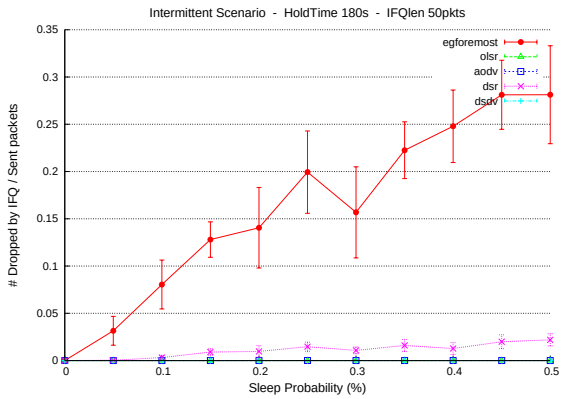


Figura 16. Número de pacotes perdidos por IFQ com HOLDTIME 180s

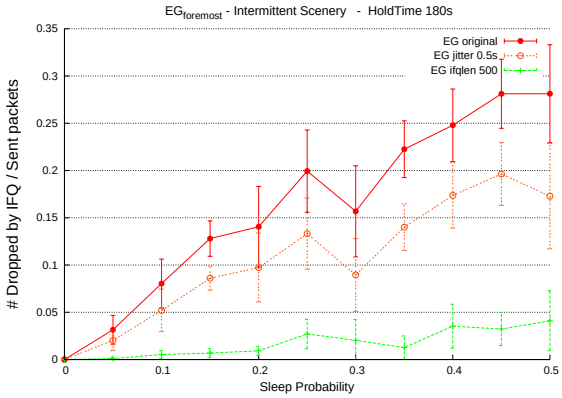


Figura 17. Diferentes alternativas para diminuir as perdas por IFQ (jitter induzido e aumentar a fila para 500 pacotes).

um atraso no envio de cada pacote, a métrica de atraso fim-a-fim é prejudicada. Na mesma figura, podemos ver os resultados obtidos usando-se uma fila IFQ com tamanho aumentado de 50 para 500 pacotes, nesta curva os dados de perda de pacote obtiveram uma melhora bem relevante.

5.11. Considerações finais e trabalhos futuros

Os resultados preliminares deste trabalho [Monteiro et al. 2006, Monteiro et al. 2007] já mostraram que um protocolo baseado nos algoritmos de EG se encaixa muito bem em redes com comportamento previsível no tempo, e que esta teoria como um todo é uma eficiente ferramenta para o desenvolvimento de protocolos de rede ad hoc.

Um protocolo de rede baseado em EG foi formalizado para prover o roteamento ótimo em relação à métrica para que foi construído. Foi implementada a métrica *Foremost journey* e realizamos extensivos testes de simulação usando o NS2 . O desempenho do protocolo EG$_{Foremost}$ foi comparado com os de três protocolos clássicos: DSDV, DSR e AODV. Os resultados mostraram que a vazão na rede usando EG$_{Foremost}$ foi a maior dentre todos os outros em qualquer cenário. Para o número de pacotes perdidos por não existir rota (NRTE), o EG$_{Foremost}$ também obteve os melhores valores como esperado. Apontamos que os valores obtidos com EG$_{Foremost}$ podem ser usados como um limite inferior nesta métrica.

Esta primeira implementação de um protocolo baseado em EG destacou alguns pontos a serem estudados com mais profundidade. Um deles, o problema dos *gargalos*, que acontece quando um nó importante fica indisponível por muito tempo, causando uma sobrecarga na rede no seu retorno, e pacotes são jogados fora devido a colisões e filas cheias. O uso de atrasos no envio (*jitters*) podem minimizar o problema das filas cheias, mas o uso deste artifício não é muito bem visto em aplicações práticas. O desenvolvimento de um bom protocolo adaptativo utilizando EG poderia solucionar este problema, detectando as rotas congestionadas e sugerindo trajetos alternativos.

É importante realçar que os valores altos de atraso médio origem-ao-destino é uma característica intrínseca das redes de comunicação dinâmicas. No caso dos protocolos baseados em EG utilizando a métrica *foremost journey*, o atraso médio alcançado é o

menor possível para cada pacote. Se longos atrasos não são desejáveis, então uma política de jogar fora pacotes por estarem esperando a muito tempo pode ser uma alternativa, ou, uma solução ainda melhor é utilizar outra métrica já estudada em EG, que é a jornada mais curta *(fastest delay)*.

As próximas fases deste projeto incluem a implementação de outros protocolos baseados em EG que utilizem outras métricas de otimização, como a jornada mais curta *(min-hop count)* e jornada mais rápida *(fastest delay)*, bem como o estudo de protocolos adaptativos para lidar com o problema dos gargalos já mencionado. Em relação à análise dos resultados, pretendemos estudar o consumo de energia dos nós ao utilizar os protocolos baseados em EG.

Uma extensão natural para este trabalho está relacionada aos desvios (diferenças) na rede dinâmica prevista, é o caso em que o EG não representa mais a realidade da topologia. Este problema motiva a utilização de teorias de comportamento estocástico para representar uma rede dinâmica de forma mais realista.

Referências

Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., and Cayirci, E. (2002). **Wireless sensor networks: a survey**. *Computer Networks (Elsevier) Journal*, 38(4):393–422.

AODV code for CMU Wireless and Mobility Extensions to NS-2 (Page accessed on Dec 2005). **AODV code for CMU Wireless and Mobility Extensions to NS2**. http://www.cs.sunysb.edu/ mahesh/aodv/.

Bhadra, S. and Ferreira, A. (2002). **Computing multicast trees in dynamic networks using evolving graphs**. Research Report 4531, INRIA.

Bhadra, S. and Ferreira, A. (2003). **Complexity of Connected Components in Evolving Graphs and the Computation of Multicast Trees in Dynamic Networks**. In *Proceedings of Adhoc-Now'03*, volume 2865 of *Lecture Notes in Computer Science*, pages 259–270. Springer Verlag.

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 ACM annual international Conference on Mobile Computing and Networking (MobiCom'98)*, pages 85–97, Dallas, TX, USA. ACM Press.

Bui-Xuan, B., Ferreira, A., and Jarry, A. (2003a). **Computing Shortest, Fastest, and Foremost Journeys in Dynamic Networks**. *International Journal of Foundations of Computer Science*, 14(2):267–285.

Bui-Xuan, B., Ferreira, A., and Jarry, A. (2003b). **Evolving graphs and least cost journeys in dynamic networks**. In *Proceedings of Modeling and Optimization in Mobile, Ad-Hoc and Wireless Networks (WiOpt'03)*, pages 141–150. INRIA Press.

Cormen, T. H., Leiserson, C. E., and Rivest, R. L. (1990). **Introduction to Algorithms**. The MIT press, Cambridge, MA, USA.

Corson, S. and Macker, J. (1999). **Mobile Ad Hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations**. RFC 2501, IETF.

Das, S., Castañeda, R., and Yan, J. (2000). **Simulation based performance evaluation of mobile, ad hoc network routing protocols**. In *ACM/Baltzer Mobile Networks and Applications (MONET) Journal*, pages 179–189. Kluwer Academic Publishers.

Estrin, D., Govindan, R., Heidemann, J., and Kumar, S. (1999). **Next Century Challenges: Scalable Coordination in Sensor Networks**. In *Proceedings of the 5th ACM annual international Conference on Mobile Computing and Networking (MobiCom'99)*, pages 263–270, Seattle, WA, USA. ACM Press.

Ferreira, A. (2003). **Building a Reference Combinatorial Model for Dynamic Networks: Initial Results in Evolving Graphs**. Research Report 5041, INRIA.

Ferreira, A., Galtier, J., and Penna, P. (2002). **Topological design, routing and handover in satellite networks**. In Stojmenovic, I., editor, *Handbook of Wireless Networks and Mobile Computing*, pages 473–493. John Wiley and Sons, New York, NY, USA.

Ferreira, A. and Jarry, A. (2004). **Complexity of Minimum Spanning Tree in Evolving Graphs and the Minimum-Energy Broadcast Routing Problem**. In *Procedings of Modeling and Optimization in Mobile, Ad-Hoc and Wireless Networks (WiOpt'04)*, pages 55–61, Cambridge, UK.

Ford, L. R. and Fulkerson, D. R. (1958). **Constructing Maximal Dynamic Flow from Static Flows**. *The Journal of the Operations Research Society of America*, 6:419–433.

IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications (Page accessed on Dec 2005). **IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications**. http://grouper.ieee.org/groups/802/11/main.html.

Johansson, P., Larsson, T., Hedman, N., Mielczarek, B., and Degermark, M. (1999). Scenario-based performance analysis of routing protocols for mobile ad-hoc networks. In **Proceedings of the 5th annual ACM international conference on Mobile computing and networking (MobiCom'99)**, pages 195–206, Seattle, WA, USA. ACM Press.

Johnson, D. B. and Maltz, D. A. (1996). **Dynamic Source Routing in Ad Hoc Wireless Networks**. In Imielinski and Korth, editors, *Mobile Computing*, volume 353, chapter 5, pages 153–181. Kluwer Academic Publishers.

Kotnyek, B. (2003). **An annotated overview of dynamic network flows**. Research Report 4936, INRIA.

Kramer, G. Generator of self-similar traffic (version 3).

Kramer, G. (2005). *Ethernet Passive Optical Networks*. McGraw-Hill.

Kramer, G., Banerjee, A., Singhal, N., Mukherjee, B., Dixit, S., and Ye, Y. (2004). Fair Queuing with Service Envelopes (FQSE): A cousin-fair hierarchical scheduler for subscriber access networks. *IEEE J. Select. Areas Commun.*, 22(8):1497–1513.

Kramer, G., Mukherjee, B., and Pesavento, G. (2001). Ethernet PON (ePON): Design and analysis of an optical access network. *Photonic Network Communications*, 3(3):307–319.

Kramer, G., Mukherjee, B., and Pesavento, G. (2002). Interleaved Polling with Adaptive Cycle Time (IPACT): A dynamic bandwidth distribution scheme in an optical access network. *Photonic Network Communications*, 4(1):89–107.

Köhler, E., Langkau, K., and Skutella, M. (2002). **Time-Expanded Graphs for Flow-Dependent Transit Times**. In *Proceedings of the 10th annual European Symposium on Algorithms (ESA'02)*, pages 599–611, London, UK. Springer-Verlag.

Lang, D. (2003). **A comprehensive overview about selected Ad Hoc Networking Routing Protocols**. Technical report, Technische Universitaet Munchen, Department of Computer Science.

Loureiro, A. A., Nogueira, J. M. S., Ruiz, L. B., Nakamura, E., Seródio, C. M., and Mini, R. (2003). **Redes sensores sem fio**. Capítulo 4 Livro texto de mini-cursos do XXI Simpósio Brasileiro de Redes de Computadores (SBRC).

Low, S. and Lapsley, D. (1999). Optimization flow control — I: basic algorithm and convergence. *IEEE/ACM Transactions on Networking*, 7(6):861–874.

McGarry, M. P., Maier, M., and Reisslen, M. (2004). Ethernet PONs : a survey of dynamic bandwidth allocation DBA algorithms. *IEEE Commun. Mag.*, 42(8):s8–s15.

Monteiro, J., Goldman, A., and Ferreira, A. (2006). **Performance Evaluation of Dynamic Networks using an Evolving Graph Combinatorial Model**. In *Proceedings of the 2nd IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob'06)*, pages 173–180, Montreal, CA. Best Student Paper Award.

Monteiro, J., Goldman, A., and Ferreira, A. (2007). **Using Evolving Graphs Foremost Journey to Evaluate Ad-Hoc Routing Protocols**. In *In Procedings of 25th Brazilian Symposium on Computer Networks (SBRC'07)*, Belem, Brazil.

Naser, H. and Mouftah, H. T. (2006). A joint-ONU interval-based dynamic scheduling algorithm for ethernet passive optical networks. *IEEE/ACM Trans. Networking*, 14(4):889–899.

Pereira, F. M. (2006). *Modelagem, policiamento e escalonamento de tráfego em redes Ethernet PON*. Tese de doutorado, Universidade Estadual de Campinas.

Perkins, C. E., Royer, E. M., Das, S. R., and Marina, M. K. (2001). **Performance Comparison of two on-demand routing protocols for ad hoc networks**. In *IEEE Personal Communications*, volume 8, pages 16–28. IEEE Communications Society.

Rice University Monarch Project (Page accessed on Dec 2005). **The CMU Monarch Wireless and Mobility Extensions to NS**. http://www.monarch.cs.rice.edu/.

Royer, E. M. and Toh, C.-K. (1999). **A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks**. *IEEE Personal Communications Magazine*, pages 46–55.

Sala, D. and Gummalla, A. (2001). PON functional requirements: Services and performance.

Shreedhar, M. and Varghese, G. (1995). Efficient fair queueing using Deficit Round Robin. In *Proc. SIGCOMM*, pages 231–242.

Stiliadis, D. and Varma, A. (1998). Latency-Rate servers: a general model for analysis of traffic scheduling algorithms. *IEEE/ACM Transactions on Networking*, 6(5):611–624.

Takeuti, P. (2005). Projeto e dimensionamento de redes ópticas passivas (PONs). Dissertação de mestrado, Universidade de São Paulo.

Varga, A. The Omnet++ simulator (version 3.0).

VINT Project (Page accessed on Dec 2005). **Network Simulator – NS2**. http://www.isi.edu/nsnam/ns/.

Werner, M. and Maral, G. (1997). **Traffic flows and dynamic routing in LEO intersatellite link networks**. In *International Mobile Satellite Conference (IMSC'97)*, pages 283–288.

Yoon, J., Liu, M., and Noble, B. (2003). **Random Waypoint Considered Harmful**. In *The 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'03)*, volume 2, pages 1312–1321.

Zhou, Y. (2003). *Resource allocation in computer networks: fundamental principles and practical strategies*. PhD thesis, Drexel University.