Monitorização Adaptativa baseada em Clusters Semi-circulares para Redes em Malha sem Fios

Ricardo Pinto
INESC-ID Lisboa/IST
Email: ricardopinto@gsd.inesc-id.pt

José Mocito
INESC-ID Lisboa/FCUL
Email: jmocito@gsd.inesc-id.pt

Luís Rodrigues
INESC-ID Lisboa/IST
Email: ler@ist.utl.pt

***Resumo*—Neste artigo propomos um novo algoritmo de monitorização para redes em malha sem fios, baseado em *clusters* adaptativos. Os nós organizam-se automaticamente numa malha de *clusters* semi-circulares e a informação de monitorização de cada *cluster* é agregada pelos líderes dos *clusters* e posteriormente reencaminhada para a estação de monitorização. O algoritmo tem em conta a existência de fluxos de dados na rede, e tenta minimizar a interferência do tráfego de monitorização nesses fluxos.**

I. Introdução

Na última década os dispositivos que suportam ligações sem fios revolucionaram o nosso quotidiano, oferecendo um acesso à rede ubíquo. As redes em malha sem fios (RMSF) emergiram como uma tecnologia-chave para aumentar a cobertura deste tipo de redes e reduzir os custos inerentes à sua infra-estrutura. Uma RMSF é uma rede multi-salto, auto-organizada, auto-configurada, tolerante a falhas dos seus nós e escalável: onde todos os nós cooperam para assegurar a ligação entre si e para o exterior [1], [2].

Como em todas as redes, a monitorização de uma RMSF é de extrema importância pois permite aos administradores detectar anomalias e prever potenciais pontos de degradação. Para tal, é necessário que cada nó reporte informação sobre os seus indicadores de desempenho, introduzindo tráfego de sinalização que pode contribuir para o congestionamento da rede. Assim, é importante usar soluções que minimizem este potencial efeito negativo da monitorização.

Este artigo propõe uma nova estratégia para monitorizar RMSF, que tem como objectivo reduzir o impacto que o tráfego de monitorização tem nos fluxos de dados existentes na rede. A solução é baseada na auto-organização dos nós em *clusters* que facilitem a agregação da informação de monitorização, através da eleição de um líder em cada *cluster* (denominado *cluster-head*) que agrega a informação dos vários membros do seu *cluster* e envia o resultado para a estação de monitorização. O uso de *clusters* em redes sem fios não é original, sendo frequente em várias áreas, por exemplo nas redes de sensores [3], [4]. No entanto, a nossa solução distingue-se pelas seguintes características: cria *clusters* semi-circulares, permitindo que o fluxo de informação seja feito sempre em direcção à estação de monitorização; a transferência de informação entre nós adapta-se aos fluxos de dados existentes, por forma a minimizar a interferência.

O resto do artigo está dividido em cinco secções. A Secção II descreve o trabalho relacionado. A arquitectura da solução é apresentada na Secção III. A Secção IV reporta a avaliação experimental e a Secção V apresenta as conclusões.

II. Trabalho Relacionado

O sistema mais utilizado para gerir e monitorizar redes é o SNMP (Simple Network Management Protocol) [5] e consiste numa consulta periódica a cada agente SNMP, que reporta a informação de volta. Por ser centralizado, tem limitações ao nível da escalabilidade.

O Mesh-Mon [6] é um sistema que apenas monitoriza um pequeno sub-conjunto de métricas enquanto o desempenho da rede é satisfatório. Quando o limiar dessas métricas é ultrapassado, o sistema passa a coleccionar informação com maior grau de pormenor. Desta forma, a redução de tráfego de monitorização é feita à custa da redução da informação disponibilizada, limitando assim a possibilidade de ser efectuado um diagnóstico adequado.

O MMAN [7] usa nós que passivamente monitorizam a rede e reportam essa informação por uma interface sem fios secundária. Apesar de não injectar tráfego adicional na rede, esta solução aumenta os custos pelo facto de recorrer a nós monitores especializados, que possuem uma interface sem fios adicional. Por sua vez, o DAMON [8] envia o tráfego de monitorização pela mesma interface que o tráfego das aplicações, mas obriga à utilização de estações monitoras cujo número é proporcional ao tamanho da rede e cuja localização deve ser optimizada para promover uma distribuição equilibrada de nós por cada estação de monitorização.

Os sistemas de monitorização devem ser desenhados de forma a lidar com distribuições não homogéneas de nós pela rede e com alterações à topologia. Os algoritmos de *clustering* permitem organizar logicamente a rede em grupos que são automaticamente configurados à medida que existem mudanças de topologia na rede. Todos os nós do mesmo grupo estão a uma distância k (em número de saltos) dos restante membros do grupo, e um destes nós é eleito líder. Esta estrutura pode ser utilizada para optimizar as actividades de monitorização da rede.

Uma solução baseada em *clusters* é proposta em [9], sendo que a formação de *clusters* não tem em conta o tráfego das aplicações que passa na rede. No Mesh-Mon [10] a topologia global é conhecida em cada nó, e é usado um protocolo de eleição para classificar os nós pela sua importância e eleger

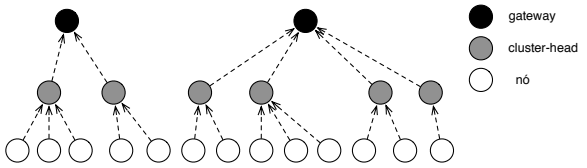


Figura 1. Arquitectura

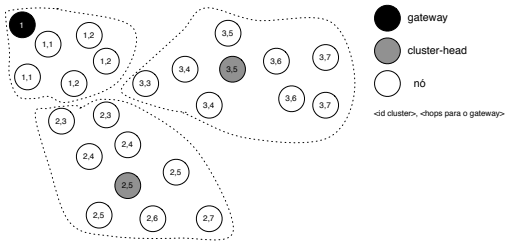


Figura 2. Clustering Circular.

eficientemente os *clusters*. No entanto, esta solução obriga a que os nós meçam constantemente a largura de banda e latência entre si, o que adiciona tráfego de sinalização.

III. ARQUITECTURA

A solução proposta recorre à auto-organização dos nós da RMSF em *clusters* de k-saltos. Em cada *cluster* é eleito um líder, para o qual é encaminhada a informação de monitorização dos restantes nós do *cluster*, que por sua vez a agrega e reencaminha para a estação de monitorização através de um ou vários nós *gateway* como se ilustra na Figura 1.

A. Encaminhamento

O encaminhamento de informação de monitorização para os nós *gateway* deve ser feito de forma independente dos protocolos utilizados para os fluxos das aplicações, porque a utilização de rotas diferentes promove a não interferência entre ambos os tráfegos. Um processo simples e pragmático de descobrir rotas é utilizar um mecanismo similar ao proposto pelo B.A.T.M.A.N. [11]. Para tal, cada gateway envia periodicamente BEACONS, que são reencaminhados pela rede toda utilizando o protocolo seguinte.

Um BEACON tem três campos: o endereço da *gateway*; um contador de época, que é incrementado cada vez que a *gateway* envia um novo BEACON; e um contador de saltos, que inicialmente é colocado a zero e é incrementado uma unidade, cada vez que um nó reencaminha o BEACON.

Quando um nó p recebe um BEACON_q de um nó q, guarda-o num histórico e inicia um temporizador de *quarentena*, de forma a esperar por outras possíveis retransmissões do BEACON. O objectivo desta *quarentena* é certificar que o nó reencaminha o BEACON com menor número de saltos enviados por rotas estáveis. O histórico em cada nó guarda os registos de todos os BEACON_q das últimas e épocas (sendo e um parâmetro configurável do protocolo). No final da *quarentena*, o nó procura no seu histórico pelo BEACON com o número de saltos mais baixo. Suponhamos que o número de BEACONS de uma fonte q é bc_q e representa o número de épocas para os quais o BEACON_q foi registado no histórico ($bc_q \leq e$). A fonte q é estável se para cada outra fonte r no histórico, temos $bc_q \geq bc_r$. De todas as fontes estáveis, o nó p selecciona a fonte t que enviou o BEACON com o número de saltos mais baixo. Finalmente, o nó p atribui t como o seu próximo salto para o nó *gateway*, aumenta o contador de saltos e reencaminha o BEACON para todos os seus vizinhos.

Como resultado deste procedimento todos os nós retêm a seguinte informação: i) distância em número de saltos ao nó *gateway* e; ii) o vizinho a ser usado na rota para o nó *gateway*.

Estabilidade das Rotas: O algoritmo acima descrito tem a desvantagem da omissão de um único BEACON causar uma mudança na rota para o nó *gateway*. Assim, considerando t o actual próximo salto para a *gateway* de um nó p, este só vai substituir t por outro nó t', se a diferença entre o número de BEACONS recebidos $bc_{t'} - bc_t$ for maior que um *limiar de estabilidade*. Em todas as nossas experiências limitamos o tamanho do histórico e a 10 épocas e fixamos o valor do *limiar de estabilidade* em 2.

B. Clustering

O objectivo do algoritmo de *clustering* é garantir que os nós se auto-organizam em grupos com as seguintes propriedades: todos os nós pertencem a um único grupo; em cada *cluster* há um único líder; o caminho mais curto entre dois líderes tem pelo menos $k+1$-saltos; os nós *gateway* são líderes do grupo de nós na sua vizinhança (isto minimiza o custo de encaminhar para estes nós). Todos os nós executam o algoritmo descrito abaixo.

Neste algoritmo, os nós podem encontrar-se num de quatro possíveis estados:QUARENTENA; SEM CLUSTER; COM CLUSTER; LÍDER. Os nós iniciam a execução do algoritmo no estado QUARENTENA. Neste estado, os nós esperam até saberem a sua distância ao nó *gateway*, de acordo com o algoritmo descrito na Secção III-A. Quando isso acontece, os nós iniciam um temporizador, com o valor definido na equação Eq. 1 ou na equação Eq. 2, e mudam o seu estado para SEM CLUSTER

$$Election - c(dist_{gw}) = \begin{cases} dist_{gw} + \lambda(s), \\ se \quad dist_{gw}\%(2k+1) = 0 \\ \\ \alpha \times dist_{gw} + \lambda(s), \\ c.c. \end{cases} \quad (1)$$

Em ambas as equações Eq. 1 e Eq. 2, $\alpha \times dist_{gw}$ é um factor que assegura que a eleição é feita gradualmente, do nó *gateway* para as extremidades da rede, o que optimiza a topologia dos grupos. O factor λ é um número aleatório entre 0 e 1 que evita que múltiplos nós se auto-elejam ao mesmo tempo, reduzindo o tempo de convergência.

Quando o temporizador expira e o nó ainda está no estado SEM CLUSTER, este auto-elege-se como líder, passando o seu

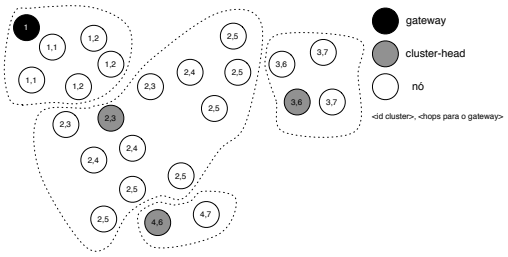


Figura 3. Clustering Semi-circular.

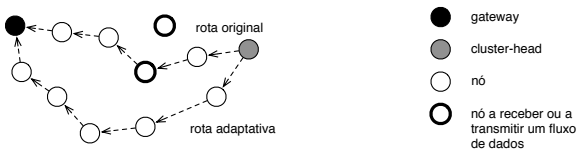


Figura 4. Atraso na propagação dos BEACONS.

estado para CLUSTER-HEAD, e começa a transmitir HELLOS periódicos, com um TTL k, que contêm o seu endereço e a distância ao nó *gateway*.

Se durante o estado SEM CLUSTER o nó receber um HELLO de um líder c, aborta o temporizador, passa o seu estado para COM CLUSTER e o seu líder passa a ser c. Posteriormente, o nó decrementa o TTL do HELLO recebido e, se este valor ainda for superior a 0, retransmite o HELLO. A retransmissão dos HELLOS e a selecção do próximo salto para o líder é efectuada seguindo o algoritmo apresentado na Secção III-A, para processamento de mensagens de BEACON.

Os líderes são responsáveis por agregar a informação de monitorização enviada periodicamente pelos nós do seu grupo e por enviar o resultado para o nó *gateway* mais próximo. O período de agregação é um múltiplo do período que os nós usam para enviar informação para o líder (nas nossas experiências, usamos o dobro). Introduzindo uma camada de agregação entre os nós e a *gateway* confere ao sistema mais flexibilidade e adaptação às condições da rede. Os líderes podem executar diversas operações de agregação sobre os dados coleccionados: médias, máximos ou mínimos, compressão, entre outras.

Clustering Semi-circular: A maioria dos algoritmos de *clustering*, incluindo o algoritmo acima descrito, criam topologias onde o líder se localiza no centro do *cluster*. Um dos problemas desta configuração é o facto da informação de monitorização ser encaminhada para o líder e a direcção deste envio, em alguns casos, ser oposta à direcção do *gateway*, resultando assim num encaminhamento sub-óptimo.

Assim, propomos que se utilize uma variante do algoritmo acima descrito, que favorece a construção de *clusters* semi-circulares (ilustrados na Figura 3). Nesta configuração, o líder de um dado nó nunca está mais afastado do nó *gateway* do que o próprio nó.

Para criar um *clustering* semi-circular os nós iniciam o seu temporizador do estado SEM CLUSTER de acordo com a Eq. 2. Adicionalmente, os nós só retransmitem HELLOS se a distância ao *gateway* do líder que o transmitiu for inferior à distância do nó ao *gateway*.

$$Election - sc(dist_{gw}) = \begin{cases} dist_{gw} + \lambda(s), \\ se \quad dist_{gw}\%(k+1) = 0 \\ \\ \alpha \times dist_{gw} + \lambda(s), \\ c.c. \end{cases} \quad (2)$$

Gateways: Os nós *gateway*, que são o destino final da informação de monitorização, executam um algoritmo ligeiramente diferente dos restantes nós. Em particular, começam sempre no estado CLUSTER-HEAD.

Encaminhamento Optimizado: Os nós que são o próximo salto de outros em direcção ao líder, esperam pela recepção da informação de monitorização desses mesmos nós antes de enviar a sua própria, agregando duas mensagens em apenas um pacote, reduzindo o tráfego de monitorização.

C. Transferência de Informação Adaptativa

A nossa arquitectura inclui um módulo de Transferência de Informação Adaptativa (TIA) que monitoriza as condições da rede e reage a alterações. Este módulo estima a carga de cada nó e tenta minimizar a interferência que o tráfego de monitorização pode causar no tráfego das aplicações. Para este efeito, a propagação dos BEACONS é atrasada proporcionalmente à quantidade do tráfego, e à sensibilidade do mesmo a variações de latência. Indirectamente, isto causa uma queda na qualidade dos BEACONS por parte dos nós que estão a encaminhar fluxos com requisitos de qualidade de serviço que, consequentemente, não serão seleccionados como próximo salto em direcção ao nó *gateway*.

A Figura 4 ilustra este comportamento adaptativo. O líder e nós subsequentes vão escolher o próximo salto em direcção ao *gateway* com uma maior qualidade de BEACONS. No caso do nó estar a reencaminhar tráfego multimédia, o tráfego de monitorização vai interferir e provocar uma queda de qualidade nesse fluxo. Para evitar esta situação, os nós que reencaminham tráfego multimédia vão atrasar a propagação dos BEACONS, o que causa uma queda passiva na qualidade desses mesmos nós como membros da rota para a *gateway*. O líder e nós subsequentes vão escolher outros nós como próximos saltos, fazendo com que o tráfego de monitorização e o tráfego multimédia percorram caminhos distintos.

D. Gateways *Múltiplos*

A presença de múltiplos nós *gateway* causa a propagação desnecessária de BEACONS que não vão ser utilizados pelos nós mais longínquos. Para limitar a inundação de tais mensagens, cada nó que recebe mais que um BEACON só vai

retransmiti-lo se este tiver maior qualidade que os outros. Em caso de igualdade, o BEACON pertencente ao nó *gateway* mais próximo é preferido.

IV. AVALIAÇÃO

Para avaliar o desempenho do sistema recorremos a simulações e a uma bancada experimental que desenvolvemos para esse fim. Comparámos o desempenho da recolha de informação usando a arquitectura descrita anteriormente com um sistema simples, em que os nós enviam a informação de monitorização directamente para o nó *gateway*, usando SNMP sobre OLSR.

A. Ambiente de Simulação

O simulador *ns-2* foi utilizado para avaliar o desempenho do sistema. A rede RMSF consiste de 100 nós estáticos colocados aleatoriamente num espaço de 500m x 800m, configurados com um alcance de transmissão de 100m. O modelo de propagação usado foi o *Two Ray Ground* com MAC 802.11. As simulações têm a duração de 5 minutos e foram executadas em 10 cenários diferentes (gerados com a ferramenta *Bonn-Motion*). Em cada cenário, o nó *gateway* foi colocado no canto inferior esquerdo do espaço, de forma a simular cenários onde os caminhos são longos.

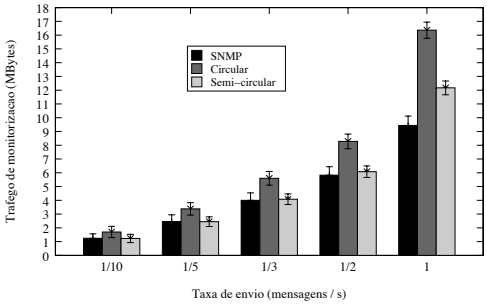
B. Clustering

Medimos número médio de grupos, média de nós por grupo e tempo de criação dos grupos obtidos com os algoritmos de *clustering* circular e semi-circular. Como era expectável, o agrupamento circular, que inclui todos os nós numa vizinhança de 2-saltos do líder, gera menos grupos (7.7) com mais membros (12.97) que o semi-circular (9.94), que gera mais grupos (9.5) devido ao facto dos nós só se juntarem a um grupo se o seu líder estiver mais próximo do nó *gateway*. O tempo de criação dos agrupamentos, medido como o intervalo entre a eleição do líder e o último nó a juntar-se ao grupo, é semelhante em ambos os casos: 9.11 ms para o circular e 8.15 ms para o semi-circular.

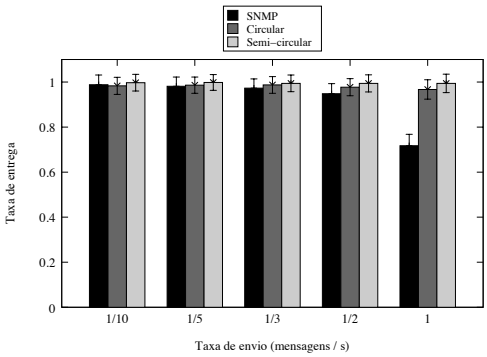
C. Tráfego de Monitorização e Rácio de Entrega

De forma a avaliar o desempenho do sistema no pior caso, foram realizados testes de *stress* com taxas de envio de mensagens de monitorização perto da saturação da rede, sem explorar a capacidade dos líderes de aplicarem funções de agregação para reduzir quantidade de informação de monitorização. Nestes testes e nos seguintes que envolvem tráfego de monitorização, cada nó gera mensagens de monitorização periódicas com 100 bytes. As Figuras 5(a) e 5(b) mostram respectivamente, a quantidade de tráfego gerado e a taxa de entrega desse mesmo tráfego, com diferentes taxas de envio das mensagens de monitorização.

Dado que neste teste os líderes se limitam a fazer *piggyback* das mensagens que recebem dos nós do seu grupo sem aplicar qualquer função de agregação, e que as rotas para o nó *gateway* através do líder não são necessariamente as mais curtas, o tráfego total de monitorização aumenta. Este fenómeno é



(a) Tráfego de Monitorização.



(b) Taxa de entrega.

Figura 5. Comparação entre o tráfego de monitorização e a taxa de entrega.

atenuado no agrupamento semi-circular. Por sua vez, a taxa de entrega permanece elevada com qualquer dos métodos de *clustering*. Por outro lado, com o SNMP, esta taxa desce abruptamente quando a taxa de envio é de 1 mensagem por segundo. Essa queda deve-se ao facto de não haver pontos de agregação (tal como acontece com o nosso sistema) e porque o protocolo de encaminhamento OLSR gera significativamente mais mensagens de controlo que os HELLOS e os BEACONS requeridos pela nossa solução, ocupando assim mais tempo o canal de transmissão.

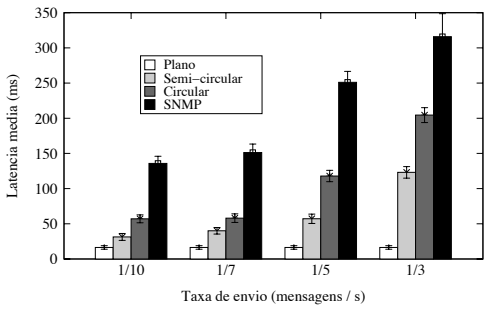
D. Impacto em Fluxos Multimédia

Para testar o impacto da monitorização em fluxos multimédia, simulámos uma chamada VoIP usando o codificador G.729, que usa tramas de 20 ms de 20 bytes cada. A chamada usa uma rota com 3 saltos, por nós colocados perto do nó *gateway*. Para medir a qualidade da chamada VoIP, foi utilizada a seguinte métrica [12]:

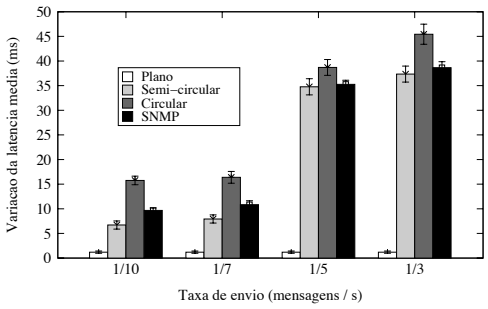
$$R = (94.2 - 0.024d) - (0.11\,(d - 177.3)\,H\,(d - 177.3) - 30ln\,(1 + 15e)$$

onde:

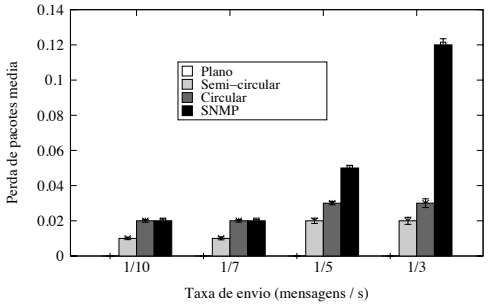
- d = 25 + d_{buffer} + d_{rede}, sendo $25ms$ o atraso total entre o ouvido e a boca, d_{buffer} o atraso no *buffer* e o atraso d_{rede} na rede.
- e = e_{rede} + (1 - e_{rede}) e_{buffer} é a perda de pacotes total na rede multiplicada por um factor de variação de latência no *buffer*;
- H(x) = 1 if x $>$ 0; 0 c.c., é a função de Heaviside.



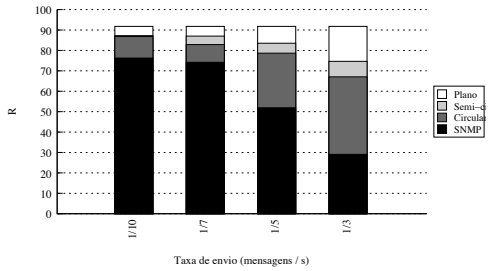
(a) Latência da chamada VoIP.



(b) Variação da latência da chamada VoIP.



(c) Perda de pacotes da chamada VoIP.



(d) Valor *R* da chamada VoIP.

Figura 6. Métricas da chamada VoIP.

A qualidade é definida por *R* (em que 70 é o valor *R* de um chamada VoIP com qualidade média). Medimos todas as métricas essenciais ao cálculo de *R* em cenários sem tráfego de monitorização (cenário *plano*), e com as variantes de monitorização introduzidas anteriormente.

A Figura 6(a) apresenta a degradação da latência para as várias taxas de envio de tráfego de monitorização. Como o OLSR troca mais mensagens de controlo por segundo que a nossa solução, a latência aumenta quando as taxas de monitorização aumentam, pois os pacotes da chamada VoIP

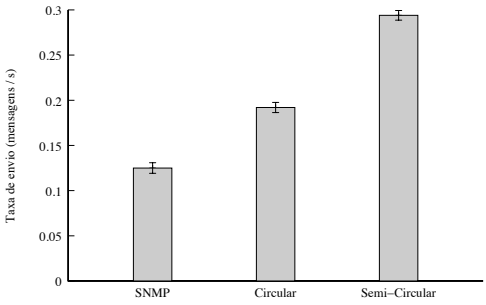


Figura 7. Comparação da taxa de envio.

competem com os pacotes da monitorização e do protocolo de encaminhamento. O *clustering* semi-circular alcança uma latência mais baixa. Ao nível da variação da latência (Figura 6(b)), o *clustering* circular apresenta o pior desempenho, já que a informação de monitorização acumulada e enviada pelos líderes é maior, devido ao maior número de membros médio de cada agrupamento. Este fenómeno poderia ser atenuado caso se estivessem a aplicar funções de agregação. Finalmente, a diferença entre os diversos algoritmos na perda de pacotes só é significativa para as taxas de transmissão de informação de controlo mais elevadas (Figura 6(c)). Em particular, no SNMP, a perda de pacotes é maior devido ao facto de todos nós enviarem os pacotes de monitorização directamente para o nó *gateway*, aumentando a interferência na sua vizinhança.

A Figura 6(d) ilustra o efeito combinado destes factores, usando a métrica *R* anteriormente descrita.

E. Taxa de Monitorização Máxima

Neste teste, a chamada VoIP foi colocada horizontalmente no meio da rede e avaliámos a taxa máxima a que se conseguia transmitir informação de monitorização sem que a qualidade da chamada descesse abaixo do valor $R = 70$. Analisando a Figura 7, pode verificar-se que a nossa arquitectura com *clustering* semi-circular consegue uma mensagem de monitorização a cada 3.4 segundos sem comprometer a qualidade da chamada, enquanto que o SNMP apenas consegue enviar uma por cada 8 segundos (o que corresponde a um aumento de desempenho de 42.5%).

F. Transferência de Informação Adaptativa

Para testar os mecanismos adaptativos da nossa solução uma chamada VoIP foi criada de modo a interferir com os pacotes enviados pelo líder de um grupo em direcção ao nó *gateway*. Os mecanismos adaptativos, baseados no atraso da propagação dos BEACONS, alteram as rotas usadas pelo líder para atingir o nó *gateway*, aumentando a qualidade da chamada (medida, nomeadamente, através da métrica R), como se pode verificar pelos dados apresentados na Figura 8.

G. Bancada Experimental

A bancada experimental foi instalada no *campus* do IST-Taguspark, e usa 8 encaminhadores La Fonera+ [13] equipados com uma interface IEEE 802.11b/802.11g sem fios, uma

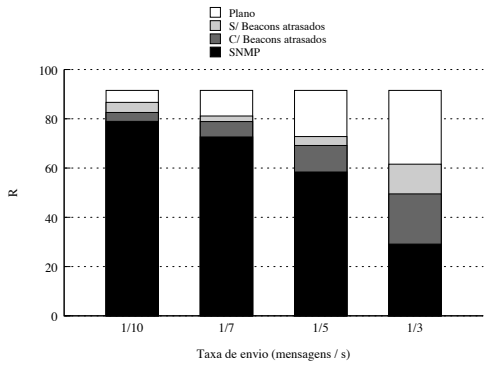


Figura 8. Valor *R* da chamada VoIP.

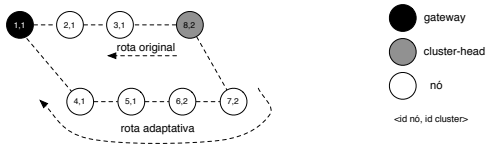


Figura 9. Topologia na bancada La Fonera.

interface LAN e outra WAN, com o firmware OpenWrt [14] 8.09. O nosso sistema foi desenvolvido em *python* e o tráfego foi gerado recorrendo ao *iperf* (ambos os pacotes incluídos no repositório do OpenWrt).

Os aparelhos foram colocados de forma a maximizar o comprimento dos caminhos. O espaço físico limitado para fazer a instalação, e a ocupação do espectro pela rede do IST-Taguspark, diminuiu o número de testes possíveis de realizar, bem como aumentou as interferências observadas.

Para testar mecanismos adaptativos foi usada a topologia da Figura 9. Deixámos os *clusters* emergir, e a transferência de informação de monitorização foi iniciada. Posteriormente uma chamada de 2 minutos (64kbit/s) foi criada entre o nó 8 e o nó *gateway*. A qualidade da chamada foi medida durante esses dois minutos com o mecanismo activado e desactivado e com diferentes taxas de envio. Os resultados são apresentados na Tabela I, podendo verificar-se claramente os efeitos benéficos do mecanismo adaptativo. Foi feita também a simulação de um cenário equivalente, apresentado-se os resultados entre parêntesis. Apesar dos resultados experimentais diferirem muito dos simulados, reflectindo as limitações bem conhecidas deste tipo de simulações, pode observar-se que o comportamento relativo é semelhante.

V. Conclusão

Neste artigo propusemos uma arquitectura para monitorizar os nós de uma RMSF. A nossa solução combina diferentes funcionalidades: é baseada em *clusters* semi-circulares que optimizam o encaminhamento da informação de monitorização e utiliza mecanismos adaptativos que minimizam o impacto da monitorização nos fluxos que decorrem na rede. Avaliámos a nossa arquitectura e protocolos associados recorrendo a simulações extensivas e usando também uma bancada experimental com oito dispositivos La Fonera+. Os resultados

Qualidade da chamada							
msg/s	TIA	Latência (ms)		Jitter (ms)		Perdas	
	TIA	real	simul	real	simul	real	simul
1/10	off	12.10	(11.21)	5.42	(1.04)	0.015	(0)
	on	7.79	(10.96)	3.12	(0.95)	0.15	(0)
1/5	off	13.51	(11.12)	6.58	(1.04)	0.15	(0)
	on	11.99	(11.20)	3.58	(1.07)	0.15	(0)
1/3	off	26.66	(11.43)	7.334	(1.18)	1.1	(0)
	on	14.90	(11.25)	4.21	(1.09)	0.46	(0)
1	off	43.89	(11.68)	29.02	(1.4)	4.4	(0)
	on	20.86	(11.44)	6.50	(1.27)	1.1	(0)

Tabela I
Mecanismos Adaptativos na Bancada La Fonera.

mostram que a solução proposta, nas nossas experiências, consegue um aumento de 42,5% da quantidade de informação de monitorização recolhida sem afectar a qualidade de serviço de uma chamada VoIP em curso. Como trabalho futuro seria interessante testar o efeito prático de usar diferentes funções de agregação nos líderes de grupo para reduzir a quantidade de informação que necessita ser transferida na rede.

Agradecimentos: Este trabalho foi parcialmente suportado pela FCT (financiamento plurianual do INESC-ID) através do PIDDAC e através do projecto "Redico" (PTDC/EIA/71752/2006).

Referências

[1] I. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: a survey," *Computer Networks ISDN Systems*, vol. 47, no. 4, pp. 445–487, 2005.

[2] A. Hamidian, C. Palazzi, T. Chong, J. Navarro, U. Korner, and M. Gerla, "Deployment and evaluation of a wireless mesh network," *Advances in Mesh Networks*, 2009.

[3] T. Anker, D. Bickson, D. Dolev, and B. Hod, "Efficient clustering for improving network performance in wireless sensor networks," in *EWSN'08*, Mar. 2008.

[4] A. Abbasi and M. Younis, "A survey on clustering algorithms for wireless sensor networks," *Computer Communications*, vol. 30, pp. 2826–2841, 2007.

[5] M. Schoffstall, M. Fedor, J. Davin, and J. Case, "Simple network management protocol (SNMP)," United States, 1990.

[6] R. Raghavendra, P. Acharya, E. Belding, and K. Almeroth, "Meshmon: a multi-tiered framework for wireless mesh networkmonitoring," in *MobiHoc S3 '09: Proceedings of the 2009 MobiHoc S3 workshop on MobiHoc S3*. New York, NY, USA: ACM, 2009, pp. 45–48.

[7] H. Kazemi, G. Hadjichristofi, and L. A. DaSilva, "Mman - a monitor for mobile ad hoc networks: design, implementation, and experimental evaluation," in *WiNTECH '08: Proceedings of the third ACM international workshop on Wireless network testbeds, experimental evaluation and characterization*. New York, NY, USA: ACM, 2008.

[8] K. N. Ramach, E. M. Belding-royer, and K. C. Almeroth, "Damon: A distributed architecture for monitoring multi-hop mobile networks," in *In Proceedings of IEEE SECON*, 2004.

[9] F. Sailhan, L. Fallon, K. Quinn, P. Farrell, S. Collins, D. Parker, S. Ghamri-Doudane, and Y. Huang, "Wireless mesh network monitoring: Design, implementation and experiments," in *Globecom Workshops, 2007 IEEE*, Nov. 2007, pp. 1–6.

[10] S. Nanda and D. Kotz, "Mesh-mon: A multi-radio mesh monitoring and management system," *Computer Communications*, vol. 31, no. 8, pp. 1588–1601, 2008.

[11] D. Johnson, N. Ntlatlapa, and C. Aichele, "A simple pragmatic approach to mesh routing using BATMAN," in *2nd IFIP International Symposium on Wireless Communications and Information Technology in Developing Countries, CSIR, Pretoria, South Africa*, 2008, p. 10.

[12] R. G. Cole and J. H. Rosenbluth, "Voice over ip performance monitoring," *SIGCOMM Comput. Commun. Rev.*, vol. 31, no. 2, pp. 9–24, 2001.

[13] "Fon," http://www.fon.com.

[14] "Openwrt," http://www.openwrt.org.