Generalizando o Encaminhamento Ad Hoc Sem-fios para Futuras Aplicações na Berma

André Rosa, Pedro Ákos Costa, and João Leitão

NOVA LINCS & DI/Nova School of Science and Technology, UNL, Portugal
{af.rosa,pah.costa}@campus.fct.unl, jc.leitao@fct.unl.pt

Resumo As redes *ad hoc* sem-fios estão a tornar-se cada vez mais relevantes devido à sua adequação para aplicações no domínio da Internet das Coisas (IoT). Estas redes são compostas por dispositivos que comunicam diretamente entre si através do meio sem-fios. Em aplicações que abrangem uma grande área, cada dispositivo é incapaz de contactar diretamente todos os outros e, portanto, têm de cooperar para suportar comunicação a vários saltos. O serviço essencial para isto é o *Encaminhamento*, que é crucial para a maioria das aplicações e serviços nestas redes. Embora muitas soluções de encaminhamento tenham sido propostas, nenhuma solução singular é considerada a mais adequada em todos os cenários. Portanto, é crucial identificar as diferenças e semelhanças chave entre as soluções para melhor comparar, combinar ou eleger dinamicamente qual usar em diferentes ambientes e condições. No entanto, identificar essas características é desafiante devido a especificações e suposições altamente heterogéneas. Neste artigo, propomos uma *framework* conceptual para especificar soluções de encaminhamento para redes *ad hoc* sem-fios, que abstrai os seus elementos comuns e que pode ser parametrizada para capturar o comportamento de soluções particulares existentes. Para além disso, uma vez que muitos protocolos de encaminhamento carecem de avaliação experimental sistemática em redes reais, utilizando uma implementação da nossa *framework*, realizámos uma avaliação experimental de várias soluções representativas usando dispositivos comuns.

Keywords: Encaminhamento · Ad Hoc Sem-fios · *Framework* · IoT

1 Introdução

Recentemente, temos assistido ao aparecimento de cada vez mais aplicações no domínio da *Internet-das-Coisas (IoT)*: redes ubíquas de objetos quotidianos interligados (e.g., veículos, edifícios, eletrodomésticos) capazes de efetuar computações e trocar dados com outros dispositivos [1, 25]. Um vasto número de aplicações IoT depende de serviços na Nuvem, e as suas implementações dependem de redes sem-fios tipicamente com infraestrutura [6]. Esta arquitetura, todavia, está a tornar-se inadequada para vários cenários de IoT devido às suas limitações inerentes. Por um lado, a quantidade cada vez maior de dados produzidos e consumidos por dispositivos IoT está a tornar a Nuvem incapaz de receber, processar e responder prontamente, além de aumentar os custos operacionais [6]. Por outro lado, embora as redes sem-fios com infraestrutura forneçam ligações relativamente fiáveis e de alta velocidade e largura de banda, também inibem a flexibilidade das aplicações, uma vez que restringem a mobilidade dos dispositivos e exigem atenção à sua implantação, configuração e re-alocação.

A necessidade de aliviar as computações na Nuvem motiva uma mudança de paradigma para a Computação na Berma [19], que explora os recursos computacionais de dispositivos na periferia da rede, situados próximo dos utilizadores finais. Dentro da Berma, encontramos *redes ad hoc sem-fios*: conjuntos de dispositivos que comunicam diretamente entre si através do meio sem-fios. Estas características tornam-nas mais flexíveis e robustas do que as suas contrapartes convencionais, sendo adequadas para situações onde a infraestrutura de rede é inadequada, inexistente, indisponível ou debilitada [1,25]. Aplicações que podem beneficiar destas redes incluem: resgate/apoio em desastres naturais; monitorização ambiental; veículos autónomos; e cidades ou casas inteligentes. Assim, a IoT tem vindo a induzir o ressurgimento contemporâneo das redes *ad hoc* sem-fios.

Nestas redes, os dispositivos, também chamados de nós, encontram-se geralmente espalhados numa vasta área, sendo incapazes de comunicar diretamente com todos os outros. Consequentemente, eles têm de cooperar, retransmitindo mensagens em nome de outros nós, de modo a que a comunicação possa ser alcançada entre todos. Este serviço essencial é chamado *Encaminhamento*.

Inúmeras soluções de encaminhamento foram propostas ao longo dos anos. No entanto, devido à natureza altamente dinâmica e heterogénea destas redes, nenhuma solução é considerada a mais adequada em todos os cenários. Assim, é crucial identificar como as diferentes soluções se relacionam entre si para as melhor comparar, combinar ou selecionar dinamicamente. Contudo, identificar as relações entre elas é uma tarefa desafiante devido a especificações e suposições heterogéneas que as diferentes soluções apresentam. Esta observação motivou-nos a conceber uma *framework* para especificar soluções de encaminhamento para estas redes, que abstrai os seus elementos comuns enquanto oferece parâmetros para materializar instâncias particulares das mesmas.

Adicionalmente, a maioria das soluções de encaminhamento só foram avaliadas através de simulações [2, 7, 8], redes com topologia em grelha com nós equidistantes e sem interferência externa ou redes com poucos nós (menos de 10) [15, 17]. Contudo, estas avaliações são incapazes de capturar as características particulares das redes *ad hoc* sem-fios reais. Assim, recorrendo a um prototipo que materializa a nossa *framework*, realizámos uma avaliação experimental de cinco soluções representativas numa rede *ad hoc* sem-fios formada por dispositivos comuns.

O resto deste artigo está estruturado da seguinte forma: A Secção 2 analisa o encaminhamento em redes *ad hoc* sem-fios; A Secção 3 detalha a nossa *framework*; A Secção 4 apresenta a metodologia e os resultados da avaliação experimental; A Secção 5 discute brevemente o trabalho relacionado; e a Secção 6 conclui o artigo com algumas observações finais.

2 Encaminhamento em Ad Hoc Sem-fios

Na literatura, os protocolos de encaminhamento são classificados principalmente de acordo com a sua estratégia de provisionamento de rotas, em proativos [4, 5, 21], reativos [13, 23, 32] ou híbridos [11, 22]. No entanto, neste artigo, procuramos caracterizar estes protocolos até ao seu funcionamento fundamental, para além da estratégia de provisionamento de rotas que utilizam. Neste sentido, o funcionamento dos protocolos de encaminhamento pode ser dividido em

duas partes complementares, a *Computação de Rotas* e o *Encaminhamento de Mensagens*.

2.1 Computação de Rotas

A computação de rotas é a parte principal dos protocolos de encaminhamento e abrange uma variedade de elementos essenciais, tais como *descobrir* a *vizinhança* de um nó, i.e., nós com os quais se consegue comunicar diretamente; *identificar* o *custo* da comunicação direta; aplicar *estratégias distribuídas* para *computar rotas*; e *disseminar informação* para informar outros nós de rotas existentes.

Na base de qualquer protocolo de encaminhamento está a *descoberta de vizinhos*, que é essencial para calcular rotas pois fornece a cada nó informação a cerca dos nós que podem ser diretamente alcançados por si. No entanto, um protocolo de encaminhamento tem de garantir alguma Qualidade de Serviço (QoS), e portanto, esta descoberta é também responsável por obter propriedades das ligações entre nós vizinhos. Uma destas propriedades é a bidirecionalidade da comunicação [4, 5], pois é geralmente crucial garantir comunicações nos dois sentidos.

Adicionalmente, os protocolos requerem métricas de *custo* para selecionar as melhores rotas, pois em geral podem haver várias rotas disponíveis de cada nó para cada destino. O custo de uma rota é uma função do custo das suas ligações constituintes, em geral a soma [4], embora outras funções possam ser utilizadas [21,32]. Estas métricas podem ser o número de saltos até ao destino, o número esperado de transmissões para entregar uma mensagem (ETX) [14, 18] ou a estabilidade das ligações [21, 32]. Neste sentido, os protocolos de encaminhamento recorrem a uma *função de custo* que avalia as ligações locais e é usada para qualificar as rotas.

O processo de computar rotas requer a cooperação distribuída dos nós e alavanca a informação local das suas vizinhanças. Neste sentido, existem três *estratégias de computação* de rotas: *i*) *vector-de-distâncias* [4, 12, 13, 23], onde cada nó anuncia o custo das suas melhores rotas para os destinos conhecidos, permitindo aos outro nós atribuir como próximo salto o vizinho que fornece a melhor rota; *ii*) *estado-das-ligações* [5, 10, 31], onde os nós reúnem, através de disseminação colaborativa, a topologia completa, ou um subconjunto conexo, da rede e computam localmente as melhores rotas para todos os destinos; e *iii*) *reversão-de-ligações* [9, 22, 24], onde os nós constroem distributivamente um grafo acíclico dirigido (DAG) sobre a topologia da rede para cada destino, com cada caminho dirigido no DAG correspondendo a uma rota para o destino. Note-se que estas estratégias podem ser mais especializadas para se adequar melhor a protocolos de encaminhamento, que discutimos em mais detalhe na próxima secção.

Finalmente, os protocolos necessitam de propagar mensagens de controlo para permitir a computação distribuída de rotas. As diferentes *estratégias de disseminação* empregues podem ser agrupadas de acordo com a sua natureza e destinos pretendidos em: *difusão por toda a rede* ou *com horizonte limitado*, para informar todos ou um subconjunto dos nós; *bordercast*, para informar um subconjunto específico de nós; ou *encaminhamento*, para informar um único nó.

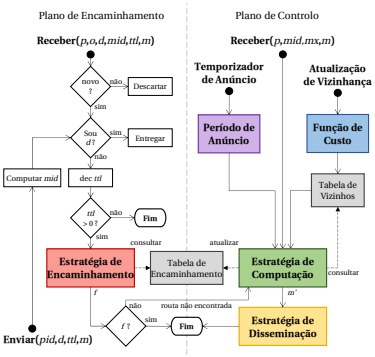


Figura 1: Fluxo de Execução da *framework* de Encaminhamento.

2.2 Encaminhamento de Mensagens

Para além de calcular rotas, os protocolos de encaminhamento são responsáveis por usar as rotas computadas para encaminhar mensagens aplicacionais. Para este fim, os protocolos usam diferentes *estratégias de encaminhamento* que oferecem diferentes contrapartidas entre fiabilidade e carga de comunicação.

A estratégia mais simples é encaminhar para o próximo salto contido numa tabela de encaminhamento local. No entanto, outras estratégias podem ser encontradas na literatura. Por exemplo, protocolos *multicaminho* [20] alavancam em múltiplas rotas para o mesmo destino para aumentar a fiabilidade das mensagens enviadas. Em alternativa, protocolos *oportunistas* [3] elegem dinamicamente, um conjunto de nós candidatos como próximo salto, no qual prossegue com o encaminhamento da mensagem. Como outra opção, os protocolos *geográficos* [26] usam as coordenadas dos nós para tomar as suas decisões. Adicionalmente, protocolos de *encaminhamento-fonte* [13] não exigem que os nós intermédios de cada rota mantenham informações sobre esta. Em vez disso, os nós que originam mensagens mantêm as rotas completas, que são carregadas com a mensagem para permitir que os nós intermédios obtenham o seu próximo salto. Finalmente, alguns protocolos [4, 23, 32] recorrem à *confirmação* de mensagens para aumentar a fiabilidade das suas estratégias de encaminhamento.

3 *Framework* de Encaminhamento para Redes *ad hoc*

Nesta secção, apresentamos a nossa *framework* de encaminhamento conceptual, que captura um amplo espectro de soluções existentes. A seguir, apresentamos a visão geral do fluxo de execução de eventos de um protocolo de encaminhamento genérico, que está no centro da nossa *framework*. Além disto, apresentamos a notação para expressar protocolos de encaminhamento que está entrelaçada com os componentes principais da *framework*, bem como como especificar um conjunto representativo de protocolos de encaminhamento existentes.

3.1 Visão Geral

A Figura 1 ilustra o fluxo de execução capturado pela *framework*, que está dividido em duas partes: o *plano de controlo*, responsável pela computação de rotas, e o *plano de encaminhamento*, que controla o fluxo de mensagens aplicacionais.

Plano de Controlo O plano de controlo é responsável pelo processamento de eventos internos para gerir a estratégia de computação. Existem três pontos de entrada principais neste plano: uma *atualização da vizinhança*, a ser processada pela ***função de custo***, a expiração do *temporizador de anúncio*, a ser processado pelo ***período de anúncio***, e a receção de uma *mensagem de controlo* que é diretamente processada pela ***estratégia de computação***.

As atualizações de vizinhança supõe-se que são fornecidas por um protocolo de descoberta externo que está fora do âmbito deste artigo. No entanto, estas devem codificar a descoberta, suspeita de falha ou atualização do estado da ligação com um nó vizinho. Cada atualização é processada pela função de custo, que atribui um custo ao vizinho, antes de se atualizar a tabela de vizinhos, que contem informações relativas a cada vizinho, como o custo e bidirecionalidade da ligação.

O temporizador de anúncio é um evento periódico que informa a estratégia de computação para disseminar uma nova mensagem de controlo, e é empregue por protocolos com uma estratégia de computação proativa ou híbrida. Quando este temporizador expira, ele é primeiro processado pelo componente *período de anúncio* que é responsável por o reiniciar (permitindo assim períodos com intervalos de tempo dinâmicos).

A receção de uma mensagem de controlo é imediatamente processada pela estratégia de computação. Cada mensagem de controlo é composta por quatro partes: (p, mid, mx, m), respetivamente, o identificador do nó que gerou a mensagem; o identificador da mensagem; metadados obtidos durante a propagação que está associada a uma estratégia de disseminação específica; e o conteúdo útil da mensagem que codifica dados específicos para a estratégia de computação.

Estes três eventos convergem para o componente principal do plano de controlo: a *estratégia de computação*. Esta estratégia avalia estes eventos e efetua uma atualização da tabela de encaminhamento e/ou solicita a disseminação de uma nova mensagem de controlo. Em ambos os casos, a estratégia de encaminhamento pode consultar a tabela de vizinhos para obter métricas de custo a fim de computar rotas. Finalmente, no caso de ser solicitada a dessiminação de uma mensagem de controlo, esta é entregue à ***estratégia de disseminação*** que é responsável por enviar a mensagem a todos os destinos pretendidos.

Plano de Encaminhamento O plano de encaminhamento é responsável por lidar com o fluxo de mensagens aplicacionais e aplicar uma ***estratégia de encaminhamento***, que pode ser desencadeada por dois eventos: a *solicitação de envio* de uma mensagem para um destino arbitrário; ou a *recepção* de uma mensagem encaminhada por um nó vizinho.

Para enviar uma mensagem, a aplicação deve fornecer os seguintes parâmetros: (p, d, ttl, m), respetivamente, o identificador do nó local, o identificador do nó de destino, um tempo de vida da mensagem e o conteúdo útil da mensagem. Após receber um pedido, a *framework* primeiro gera um identificador único para a mensagem (mid) e de seguida entra no fluxo de mensagens recebidas.

Após a receção de uma mensagem, o plano de encaminhamento primeiro verifica se já a processou, descartando-a se assim for. De seguida, se o destino da mensagem for o nó local, a mensagem é entregue à aplicação, continuando para a próxima etapa de processamento caso contrário, onde o ttl é decremen-

Etiqueta	Ref	FS	AP	CF	RS	DS
OLSR	[5]	Simples	t	ETX	EstadoLigação	Dif(∞)
FSR	[10]	Simples	t	ETX	EstadoLigação	Dif(1)
BABEL	[4]	Simples	t	ETX	MultiVecDist	Dif(1) $\cup$ Dif(∞)
BATMAN	[21]	Simples	t	MCX	SingularVecDist(pro)	Dif(∞)
JOKER	[28]	Oportunista	t	MCX	SingularVecDist(pro)	Dif(∞)
AODV	[23]	Simples	$\perp$	ETX	SingularVecDist(re)	Dif(∞) $\cup$ Enc
DSR	[13]	Fonte	$\perp$	ETX	SingularVecDist(re)	Dif(∞) $\cup$ Enc
ABR	[32]	Simples	$\perp$	AGE	SingularVecDist(re)	Dif(∞) $\cup$ Enc
ZRP	[11]	Simples	t	ETX	Zona(i, o, r)	Dif(r) $\cup$ Bordercast $\cup$ Enc
TORA	[22]	Simples	$\perp$	$\perp$	ReversãoLigações(m)	Dif(∞) $\cup$ Dif(1)
GPSR	[16]	Geográfico	$\perp$	DIST	$\perp$	$\perp$

Tabela 1: Especificação de Protocolos de Encaminhamento.

tado e verificado. Se o *ttl* expirou, o encaminhamento da mensagem termina, caso contrário, a mensagem é delegada à estratégia de encaminhamento. Esta estratégia, tenta obter o próximo salto e enviar a mensagem para o destino correto, potencialmente consultando a tabela de encaminhamento e por fim o fluxo de execução termina. Se nenhum próximo salto for encontrado, a estratégia de computação no plano de controlo é notificada, levando potencialmente à disseminação de uma nova mensagem de controlo, como um pedido de rota em protocolos reativos [13, 23].

3.2 Parâmetros da *Framework*

A nossa *framework* representa um meta-protocolo de encaminhamento que pode ser parametrizado para expressar uma multitude de protocolos concretos diferentes. Para definir um protocolo na nossa *framework* basta especificar cinco parâmetros: (FS, AP, CF, RS, DS), onde FS é uma estratégia de encaminhamento, AP um período de anúncio, CF uma função de custo, RS uma estratégia de computação, e DS uma estratégia de disseminação. De seguida, discutimos algumas alternativas de valores possíveis para estes parâmetros, sendo que estes podem também assumir o valor de $\perp$ para representar o facto de uma solução concreta não utilizar um dado parâmetro.

Estratégias de Encaminhamento são responsáveis por selecionar e encaminhar para o próximo salto mensagens aplicacionais. Estas podem ser Simples, que obtém simplesmente o primeiro próximo salto da tabela de encaminhamento; MultiCaminho, que retira vários candidatos a nós próximo salto da tabela e seleciona um de acordo com algum critério; Fonte, que recupera a rota completa da tabela e anexa-a à mensagem, permitindo que nós intermédios a encaminhem; Confirma(s), que estende uma estratégia s com confirmações explícitas e retransmissões de mensagens; Oportunista, onde o próximo salto é selecionado dinamicamente; e Geográfico onde o próximo salto é escolhido como o vizinho geograficamente mais próximo do destino.

Período de Anúncio é um número natural t que representa o intervalo entre anúncios periódicos de mensagens de controlo. Este valor pode ser o resultado de uma função no caso de serem empregues períodos dinâmicos.

Funções de Custo atribuem custos às ligações para qualificar as rotas, e incluem: SALTOS, que é sempre 1 para o custo das rotas ser o seu número de saltos; ETX, que estima o número esperado de transmissões para um encaminhamento bem-sucedido; IDADE, que usa o tempo desde que o vizinho foi detetado como custo, com vizinhos mais antigos representado melhores ligações (i.e., mais estáveis); DIST, que usa a distância entre o nó local e o vizinho, com custos maiores representando melhores ligações (i.e., mais perto do destino); e MCX, onde o número de mensagens de controlo recebidas de origens e vizinhos distintos é considerado, com valores mais altos representando melhores ligações.

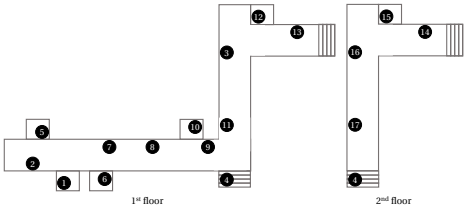


Figura 2: Planta da Rede.

Estratégias de Computação são responsáveis por computar rotas e podem ser: EstadoLigação, que dissemina periodicamente um subconjunto da topologia conhecida, permitindo todos os nós reunir a topologia global que é usada para calcular localmente rotas para todos os destinos. MultiVecDist, que propaga uma parte da tabela de encaminhamento local contendo todos os destinos conhecidos e custos associados, permitindo a cada nó calcular as melhores rotas para cada destino. SingularVecDist(m), que dissemina o identificador do nó local pela rede, com o parâmetro m a ditar se a disseminação é proativa ou reativa, e utiliza informação sobre o caminho percorrido por estas mensagens para calcular rotas para o nó de origem. ReversãoLigações(m), que constrói distributivamente um DAG dirigido para o nó local, com o parâmetro m a ditar se a construção é acionada proativa ou reativamente. E Zona(i,o,r), que emprega uma estratégia proativa i em zonas com um escopo limitado de r saltos, e uma estratégia reativa o para calcular rotas para nós fora dessas zonas.

Estratégias de Disseminação propagam mensagens de controlo para os seus destinos pretendidos, e podem ser: Dif(h), onde as mensagens de controlo são difundidas por toda a rede se h for ∞, ou até um número limitado de saltos h, caso contrário. Bordercast, onde as mensagens são disseminadas para os nós nas fronteiras das zonas de encaminhamento. E Enc, onde as mensagens são enviadas para um único destino, utilizando rotas previamente descobertas.

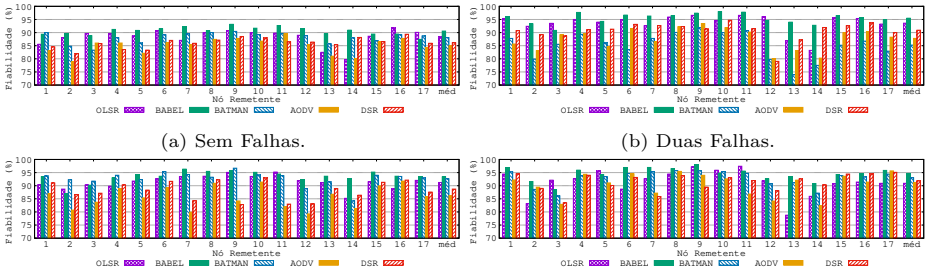
Com estes parâmetros, podemos definir um grande número de protocolos de encaminhamento encontrados na literatura. A Tabela 1 contém um conjunto ilustrativo de exemplos. Na próxima secção, apresentamos a nossa avaliação experimental recorrendo a alguns destes protocolos de encaminhamento.

4 Avaliação Experimental

Nesta secção, apresentamos a avaliação experimental, começando com a configuração das experiências, seguida pela discussão dos resultados.

4.1 Configuração Experimental

A *framework* e os protocolos complementares de descoberta e difusão foram implementados na linguagem C recorrendo à *framework* Yggdrasil [6]. Avaliámos cinco soluções representativas: OLSR, BABEL, BATMAN, AODV e DSR. Os três primeiros são proativos, e empregam diferentes estratégias de computação, e os outros dois são reativos, e empregam diferentes estratégias de encaminhamento. Devido à falta de espaço, omitimos a descrição destes protocolos. Cada protocolo foi configurado conforme indicado na Tabela 1, com o período de anúncio (parâmetro t) e o intervalo dos anuncios do protocolo de descoberta tendo ambos um valor de 5 segundos.



(a) Sem Falhas.

(b) Duas Falhas.

(c) Cinco Falhas.

(d) Nove Falhas.

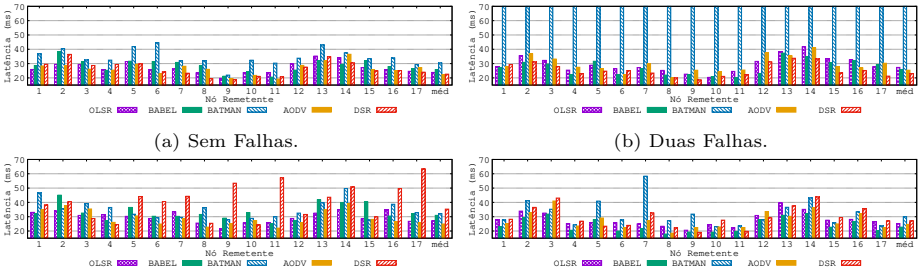
Figura 3: Fiabilidade dos Protocolos de Encaminhamento.

A avaliação foi realizada numa rede *ad hoc* sem-fios formada por 17 Raspberry Pi 3B, que foram dispersos pelo edifício do nosso departamento em dois andares, conforme ilustrado na Figura 2. Cada nó executou um protocolo de encaminhamento e uma aplicação de ping simples por um período de 10 minutos após um período de estabilização inicial de 2 minutos. A aplicação, a cada segundo, solicita ao protocolo de encaminhamento para enviar uma mensagem a um destino selecionado aleatoriamente (diferente do nó local), que após a recepção envia uma resposta para o nó de origem. Este comportamento permite avaliar as rotas selecionadas em ambas as direções, bem como medir com precisão o tempo médio de ida e volta (RTT) das rotas calculadas e empregues por cada protocolo.

Avaliámos cada solução em quatro cenários: sem falhas e com falhas determinísticas dos primeiros dois nós, cinco nós e nove nós da sequência 13, 14, 8, 10, 12, 3, 6, 11, 16. As falhas foram introduzidas simultaneamente no ponto médio de cada experiência (5 minutos). Além disto, os nós que falham nunca são selecionados como destinos das mensagens para não afetar as medições de fiabilidade. Cada experiência foi executada três vezes, numa ordem aleatória, e os resultados mostram a média de todas as execuções. Para cada protocolo, medimos a fiabilidade, como o número de mensagens de ping que foram recebidas de volta com sucesso; a latência, como o RTT médio de cada mensagem e a carga de comunicação, como o número total de transmissões incorridas pela disseminação de mensagens de controlo, incluindo também os protocolos complementares.

4.2 Resultados Experimentais

A Figura 3 apresenta os resultados da fiabilidade, no eixo y, para cada nó e em média (último conjunto de colunas), no eixo x. No geral, todos os protocolos alcançaram uma fiabilidade superior a 80% em todos os cenários, com os protocolos proativos alcançando maior fiabilidade que os reativos. Isto é explicado pelos nós descartarem as mensagens solicitadas enquanto a computação de rotas reativa está em execução e as rotas sendo constantemente quebradas e recalculadas devido a vizinhanças instáveis. O impacto deste último caso é mitigado em soluções proativas uma vez que as rotas são continuamente atualizadas. O BABEL foi o protocolo que alcançou maior fiabilidade em média, independentemente do número de falhas. Suspeitamos que a razão principal para isto se deve a que, de entre os protocolos proativos, o BABEL foi o que apresentou a menor carga (discutido mais adiante) e, como tal, levou a menor interferência



(a) Sem Falhas.

(b) Duas Falhas.

(c) Cinco Falhas.

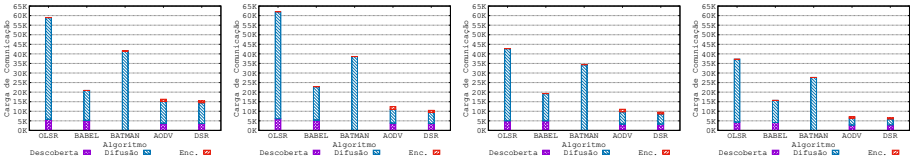
(d) Nove Falhas.

Figura 4: Latência Média dos Protocolos de Encaminhamento.

no meio causando menos perdas de mensagens. Entre os protocolos reativos, o DSR alcançou um valor ligeiramente superior de fiabilidade que o AODV em todos os cenários. Este comportamento parece ter sido causado por relações de vizinhança instáveis que levaram as rotas a quebrar, levando os nós intermédios no AODV a remover as rotas das suas tabelas de encaminhamento. Esta instabilidade afetou menos o DSR, uma vez que as rotas completas são transportadas nas próprias mensagens encaminhadas. Observamos que, conforme o número de falhas aumenta, aumenta também a fiabilidade dos protocolos. Isto deve-se ao facto que a rede resultante após as falhas apresente mais caminhos estáveis, menos caminhos instáveis redundantes e também menos interferência entre os nós. Para além disto, no cenário com duas falhas (Figura 3b), notamos que o BATMAN teve fiabilidade significativamente menor quando comparado aos outros cenários. Isto foi causado pelo surgimento de um grande número de ciclos de encaminhamento de curta duração. Estes ciclos surgem porque a estratégia de computação do BATMAN não possui mecanismos de prevenção de ciclos e a combinação da função de custo e da estratégia de disseminação do BATMAN, aliada a vizinhanças instáveis, faz com que os nós modifiquem frequentemente os seus próximos saltos selecionados.

A Figura 4 apresenta a latência, no eixo y em milissegundos (ms), para todos os nós e em média, no eixo x. No geral, todos os protocolos alcançaram uma latência menor que 35 ms em todos os cenários, com aproximadamente a mesma latência média por cenário. A razão principal para estes resultados é que todos os protocolos convergiram para aproximadamente as mesmas rotas sendo selecionadas (à volta de 2 saltos em média), uma vez que quase todos os protocolos usaram a mesma métrica de custo e as rotas disponíveis para calcular na nossa rede tinha um número relativamente baixo de saltos. A exceção foi o BATMAN, sendo consistentemente o protocolo com a maior latência, devido à formação de ciclos de curta duração que foram observados em todos os cenários.

A Figura 5 apresenta a carga de comunicação, no eixo y, para cada protocolo, no eixo x. A carga é discriminada em três tipos: *descoberta*, como o número de transmissões do protocolo de descoberta, *difusão* como o número de transmissões do protocolo de difusão usado pela estratégia de disseminação e *encaminhamento* como o número de transmissões para disseminar mensagens de controlo com encaminhamento. Começamos por notar que, conforme o número de falhas aumenta, a carga total diminui uma vez que menos nós disseminam



(a) Sem Falhas. (b) Duas Falhas. (c) Cinco Falhas. (d) Nove Falhas.

Figura 5: Carga de Comunicação Total por Protocolo de Encaminhamento.

mensagens de controlo. No geral, o OLSR apresentou a maior carga porque a sua estratégia de computação desencadeou a disseminação de mensagens de controlo não agendadas sempre que o subconjunto da topologia a disseminar é alterado, o que aconteceu frequentemente devido a relações de vizinhança instáveis. Os protocolos reativos, AODV e DSR, apresentaram a menor carga em todos os cenários. Isto resultou do armazenamento de rotas descobertas destinadas a outros nós, o que permitiu que menos pedidos de rota fossem disseminados. O BATMAN apresentou a segunda maior carga, causada pela disseminação periódica da identidade de cada nó. Para além disto, o processo de descoberta de vizinhos do BATMAN é combinado com a disseminação de tais mensagens de controlo, permitindo evitar carga de descoberta adicional.

5 Trabalho Relacionado

Poucos autores tentaram sistematizar protocolos de encaminhamento para redes *ad hoc* sem-fios, porém, alguns exemplos podem ser encontrados na literatura.

A *framework* Independent Zone Routing (IZR) [27] permite a hibridação de protocolos proativos e reativos ao mesmo tempo que permite adaptar dinamicamente a quantidade de comportamento proativo e reativo. Embora a IZR permita combinar praticamente quaisquer soluções proativas e reativas, ela considera-as como "caixas pretas" e não tenta decompo-las nos seus constituintes fundamentais para analisar adequadamente cada solução como fazemos neste artigo.

A Relay Node Set (RNS) [30] em contraste, é uma *framework* analítica para comparar a sobrecarga de comunicação de protocolos de encaminhamento. A RNS vê cada protocolo como um gestor de *conjuntos de nós de retransmissão (RNSs)*: conjuntos de nós que retransmitem mensagens de controlo, sendo que cada protocolo pode gerir mais do que um RNS de cada vez. Neste sentido, a RNS disseca as soluções de encaminhamento do ponto de vista da avaliação, mas não as especifica de acordo com sua arquitetura interna.

Finalmente, a Multi-mode Routing Protocol (MMRP) [29] é uma *framework* que seleciona o protocolo mais adequado para cada região da rede de acordo com suas características locais, permitindo a coexistência de vários protocolos de encaminhamento dentro da mesma rede. No entanto, a MMRP não é flexível o suficiente para especificar a maioria dos protocolos existentes, uma vez que depende fortemente de um padrão de arquitetura único e específico.

6 Considerações Finais

Neste artigo, apresentámos uma *framework* conceptual para especificar, analisar e comparar protocolos de encaminhamento para redes *ad hoc* sem-fios. A nossa *framework* abstrai os aspetos comuns dos protocolos, uma tarefa que não é trivial

devido à sua natureza, e expõe parâmetros que capturam o comportamento de soluções específicas. Através de um protótipo de nossa *framework*, implementámos um conjunto representativo de protocolos existentes e realizámos uma avaliação experimental destes protocolos numa rede *ad hoc* sem-fios real formada por dispositivos comuns. Os resultados mostraram observações interessantes que não foram ainda exploradas e discutidas no contexto de protocolos de encaminhamento em redes *ad hoc* sem-fios. O BATMAN, considerado na literatura como um dos melhores protocolos, não só nunca foi o melhor em relação à fiabilidade em nenhum cenário, como também foi o pior em relação à latência em todos os cenários. Além disto, os protocolos reativos apresentam fiabilidade semelhante aos proativos, apesar das vizinhanças instáveis e das falhas de nós, devido ao uso de armazenamento de rotas, embora apresentem uma carga muito menor.

Referências

1. Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., Ayyash, M.: Internet of things: A survey on enabling technologies, protocols, and applications. IEEE communications surveys & tutorials **17**(4) (2015)
2. Baraković, S., Baraković, J.: Comparative performance evaluation of mobile ad hoc routing protocols. In: The 33rd International Convention MIPRO (2010)
3. Boukerche, A., Darehshoorzadeh, A.: Opportunistic routing in wireless networks: Models, algorithms, and classifications. ACM Comput. Surv. **47**(2) (Nov 2014)
4. Chroboczek, J., Schinazi, D.: The Babel Routing Protocol. RFC 8966 (1 2021)
5. Clausen, T.H., Dearlove, C., Jacquet, P., Herberg, U.: The Optimized Link State Routing Protocol Version 2. RFC 7181 (4 2014)
6. Costa, P.A., Rosa, A., Leitão, J.a.: Enabling wireless ad hoc edge systems with yggdrasil. In: Proceedings of the 35th Annual ACM Symposium on Applied Computing. SAC '20, Association for Computing Machinery, New York, NY, USA (2020)
7. Das, S.R., Castaneda, R., Jiangtao Yan, Sengupta, R.: Comparative performance evaluation of routing protocols for mobile, ad hoc networks. In: Proceedings 7th International Conference on Computer Communications and Networks (Cat. No.98EX226) (1998)
8. Das, S.R., Castañeda, R., Yan, J.: Simulation-based performance evaluation of routing protocols for mobile ad hoc networks. Mobile Networks and Applications **5**(3) (Sep 2000)
9. Gafni, E., Bertsekas, D.: Distributed algorithms for generating loop-free routes in networks with frequently changing topology. IEEE Transactions on Communications **29**(1) (1981)
10. Gerla, P.M.: Fisheye State Routing Protocol (FSR) for Ad Hoc Networks. Internet-draft, Internet Engineering Task Force (Jun 2002)
11. Haas, Z.J.: A new routing protocol for the reconfigurable wireless networks. In: Proceedings of ICUPC 97 - 6th International Conference on Universal Personal Communications. vol. 2 (Oct 1997)
12. He, G.: Destination-sequenced distance vector (dsdv) protocol. Networking Laboratory, Helsinki University of Technology (2002)
13. Hu, Y.C., Maltz, D.A., Johnson, D.B.: The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4. RFC 4728 (Feb 2007)
14. Javaid, N., Javaid, A., Khan, I.A., Djouani, K.: Performance study of etx based wireless routing metrics. In: 2009 2nd International Conference on Computer, Control and Communication (2009)
15. Johnson, D., Hancke, G.: Comparison of two routing metrics in olsr on a grid based mesh network. Ad Hoc Networks **7**(2) (2009)

16. Karp, B., Kung, H.T.: Gpsr: Greedy perimeter stateless routing for wireless networks. In: Proceedings of the 6th Annual International Conference on Mobile Computing and Networking. MobiCom '00, Association for Computing Machinery (2000)
17. Kiess, W., Mauve, M.: A survey on real-world implementations of mobile ad-hoc networks. Ad Hoc Networks **5**(3) (2007)
18. Kim, K.H., Shin, K.G.: On accurate measurement of link quality in multi-hop wireless mesh networks. In: Proceedings of the 12th Annual International Conference on Mobile Computing and Networking. MobiCom '06, Association for Computing Machinery (2006)
19. Leitão, J., Costa, P.Á., Gomes, M.C., Preguiça, N.M.: Towards enabling novel edge-enabled applications. Tech. rep., Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa (2018), https://dblp.org/rec/bib/journals/corr/abs-1805-06989
20. Mueller, S., Tsang, R.P., Ghosal, D.: Multipath routing in mobile ad hoc networks: Issues and challenges. In: Calzarossa, M.C., Gelenbe, E. (eds.) Performance Tools and Applications to Networked Systems. Springer Berlin Heidelberg, Berlin, Heidelberg (2004)
21. Neumann, A., Aichele, C., Lindner, M., Wunderlich, S.: Better Approach To Mobile Ad-hoc Networking (B.A.T.M.A.N.). Internet-draft, Internet Engineering Task Force (Apr 2008)
22. Park, V.D., Corson, D.S.M.: Temporally-Ordered Routing Algorithm (TORA) Version 1 Functional Specification. Internet-draft, Internet Engineering Task Force (Jul 2001)
23. Perkins, C.E., Ratliff, S., Dowdell, J., Steenbrink, L., Pritchard, V.: Ad Hoc On-demand Distance Vector Version 2 (AODVv2) Routing. Internet-draft, Internet Engineering Task Force (Feb 2019)
24. Ramasubramanian, V., Haas, Z.J., Sirer, E.G.: Sharp: A hybrid adaptive routing protocol for mobile ad hoc networks. In: Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking & Computing. MobiHoc '03, Association for Computing Machinery (2003)
25. Reina, D.G., Toral, S.L., Barrero, F., Bessis, N., Asimakopoulou, E.: The Role of Ad Hoc Networks in the Internet of Things: A Case Scenario for Smart Environments. Springer Berlin Heidelberg, Berlin, Heidelberg (2013)
26. Ruehrup, S.: Theory and practice of geographic routing. In: Hai Liu, Yiu-Wing Leung, X.C. (ed.) Ad hoc and sensor wireless networks: architectures, algorithms and protocols, vol. 69, chap. 5. Bentham Science (2009)
27. Samar, P., Pearlman, M.R., Haas, Z.J.: Independent zone routing: an adaptive hybrid routing framework for ad hoc wireless networks. IEEE/ACM Transactions on Networking **12**(4) (2004)
28. Sanchez-Iborra, R., Cano, M.: Joker: A novel opportunistic routing protocol. IEEE Journal on Selected Areas in Communications **34**(5) (May 2016)
29. Santivanez, C.A., Stavrakakis, I.: Towards adaptable ad hoc networks: The routing experience. In: Smirnov, M. (ed.) Autonomic Communication. Springer Berlin Heidelberg, Berlin, Heidelberg (2005)
30. Tao Lin, Midkiff, S.F., Park, J.S.: A framework for wireless ad hoc routing protocols. In: 2003 IEEE Wireless Communications and Networking, 2003. WCNC 2003. vol. 2 (2003)
31. Templin, F.L., Ogier, R., Lewis, M.S.: Topology Dissemination Based on Reverse-Path Forwarding (TBRPF). RFC 3684 (Feb 2004)
32. Toh, C.K.: Long-lived Ad Hoc Routing based on the Concept of Associativity. Internet-draft, Internet Engineering Task Force (Mar 1999)