

Um Protocolo de Distribuição de Dados para Redes de Sensores Sem Fio Heterogêneas com Sink Móvel

Guilherme Maia¹, Daniel L. Guidoni¹, Aline C. Viana²
Andre L. L. Aquino³, Raquel A. F. Mini⁴, Antonio A. F. Loureiro¹

¹Departamento de Ciência da Computação
Universidade Federal de Minas Gerais (UFMG)

²INRIA – France

³Instituto de Computação – Universidade Federal de Alagoas (UFAL)

⁴Departamento de Ciência da Computação – PUC Minas

{jgmm, guidoni, loureiro}@dcc.ufmg.br

aline.viana@inria.fr

alla@ic.ufal.br, raquelmini@pucminas.br

Abstract. *This paper presents ProFlex, a distributed data storage protocol for heterogeneous wireless sensor networks with mobile sinks. ProFlex guarantees robustness in data retrieval by intelligently managing data replication among selected storage nodes in the network. Contrarily to related protocols, ProFlex considers the resource constraints of sensor nodes and constructs multiple data replication structures, which are managed by more powerful nodes. Additionally, ProFlex takes profit of the higher communication range of such powerful nodes in the network and use the long range links to improve data distribution to storage nodes. When compared to related protocols, we show by simulation that ProFlex increases the network resilience under failures circumstances, decreases the overhead of transmitted messages, and decreases the occurrence of the energy hole problem.*

Resumo. *Este trabalho apresenta ProFlex, um protocolo de distribuição de dados para redes de sensores sem fio heterogêneas com sinks móveis. ProFlex garante robustez na coleta de dados, pois gerencia de forma inteligente a replicação de dados entre os nós sensores selecionados como armazenadores de dados. Diferente dos protocolos existentes na literatura, ProFlex considera as limitações de recursos dos nós sensores tradicionais e constrói múltiplas estruturas de replicação de dados, as quais são gerenciadas por nós sensores especiais que possuem recursos computacionais mais sofisticados. Além disso, ProFlex tira proveito do poder de comunicação desses nós especiais ao utilizar enlaces de longo alcance para melhorar a distribuição de dados entre os nós armazenadores. Ao compará-lo a outros protocolos existentes, mostra-se por meio de simulações que ProFlex aumenta a resiliência da rede, diminui a sobrecarga de mensagens transmitidas, além de diminuir a ocorrência do problema de energy hole.*

1. Introdução

A deposição de aplicações de Redes de Sensores Sem Fio (RSSFs), as quais operam de forma autônoma e geram uma quantidade de dados considerável, apresenta diversos desafios [Akyildiz et al. 2002]. Dentre eles, um dos principais é *como coletar os dados sensoreados*. A coleta de dados normalmente é realizada por um nó especial chamado *sink*. Diferente dos nós sensores tradicionais, o *sink* possui recursos computacionais mais sofisticados. Além disso, uma RSSF pode empregar um *sink* estático ou móvel. No primeiro caso, os nós sensores precisam rotear os dados sensoreados para o *sink*, portanto a conectividade com pelo menos um nó *sink* deve ser mantida durante o tempo de vida da rede de forma a garantir uma boa coleta de dados. No entanto, essa abordagem sofre do problema conhecido como *energy hole* [Liu et al. 2010] no qual os nós sensores localizados próximo ao *sink* tendem a consumir uma maior quantidade de energia em decorrência do repasse de dados oriundos de outros nós da rede, e portanto acabam morrendo mais cedo, desconectando a rede e comprometendo a coleta de dados. De forma a contornar esse problema, a utilização de *sinks* móveis tem sido estimulada em trabalhos recentes [Basagni et al. 2008, Song and Hatzinakos 2007]. Nessa abordagem, o dado sensoreado não é imediatamente roteado para o *sink*, mas sim armazenado na própria rede de modo a ser coletado posteriormente pelo *sink* móvel. Logo, o *sink* móvel tem a flexibilidade de percorrer a rede para coletar os dados sensoreados diretamente dos nós sensores, e assim nenhum nó sofre a sobrecarga de ter que repassar dados de todos os outros nós da rede.

No entanto, por questões de escalabilidade, pode ser inconcebível para o *sink* visitar todos os nós da rede com o objetivo de coletar os dados sensoreados. Logo, o grande desafio nesse cenário é *como distribuir e armazenar os dados sensoreados entre os nós da rede* de uma maneira que tais dados possam ser coletados pelo *sink* sem a necessidade de visitar todos os nós da rede. Essa distribuição de dados pode ser realizada de forma reativa ou proativa. Nos cenários reativos [Basagni et al. 2008, Yang et al. 2006] os nós sensores reagem à presença do *sink* e repassam os dados para o local em que o *sink* se encontra no momento. Já nos cenários proativos [Chatzigiannakis et al. 2008, Viana et al. 2010], os dados são proativamente distribuídos e armazenados em uma área de interesse para que posteriormente o *sink* possa coletá-los. Ademais, dependendo da forma com que os dados são distribuídos, (1) o *sink* precisa seguir uma trajetória previamente definida, onde ele deve visitar nós ou localizações específicas, (2) ou ele pode ser livre para seguir uma trajetória não especificada previamente. Certamente, evitar restrições com relação a trajetória do *sink* é benéfico tanto para o *sink* quanto para os nós sensores, já que a ausência de tais restrições possibilita que a rede possa adaptar-se mais facilmente à situações especiais. Portanto, nesse trabalho o foco será a distribuição de dados de forma proativa, além da seleção de nós armazenadores em RSSFs com *sinks* móveis cujas trajetórias são desconhecidas para os nós sensores.

Alguns trabalhos existentes na literatura [Sheng et al. 2007, Anastasi et al. 2008] com o mesmo foco propuseram a utilização de nós sensores especiais para realizar a distribuição e armazenamento de dados. Nesse cenário, apenas esses nós especiais são os responsáveis por armazenar todos os dados sensoreados, já que assume-se que tais nós não possuem qualquer restrição quanto a capacidade de armazenamento. As RSSFs que empregam nós especiais em conjunto com nós sensores tradicionais são conhecidas na literatura como RSSFs heterogêneas. Uma característica dessas redes é que elas

não apresentam os mesmos problemas de desempenho e escalabilidade das redes homogêneas [Helmy 2003, Sharma and Mazumdar 2008]. Não obstante, a utilização de nós especiais não contorna o problema de perda de dados, já que esses nós ainda podem falhar. Com o objetivo de aumentar a resiliência da rede com relação a falhas, uma possível abordagem é replicar um pacote de dados e mantê-lo em diferentes nós da rede. Além disso, é importante perceber que na presença de um *sink* móvel, uma boa configuração do número de réplicas e a seleção adequada de nós armazenadores podem permitir ao *sink* coletar uma quantidade representativa dos dados visitando-se apenas uma pequena porcentagem dos nós [Bar-Yossef et al. 2008, Viana et al. 2010, Vecchio et al. 2010]. Portanto, a utilização de nós poderosos ainda requer a existência de mecanismos de replicação com o objetivo de aumentar a resiliência e melhorar a coleta de dados pelo *sink* móvel.

Logo, esse trabalho apresenta *ProFlex*, um protocolo que utiliza nós sensores especiais na distribuição de dados em RSSFs heterogêneas. No entanto, em vez de utilizar os recursos extras de armazenamento desses nós especiais, *ProFlex* tira proveito do poder de comunicação desses nós para melhorar a distribuição de dados entre os nós armazenadores. Além disso, o protocolo provê flexibilidade na escolha dos nós armazenadores, já que a decisão de empregar ou não um determinado nó sensor no armazenamento de dados é baseada em um fator de importância previamente definido. Tal fator de importância dá flexibilidade ao protocolo de poder empregar qualquer estratégia de distribuição de dados, ou seja, é possível escolher os nós armazenadores de acordo com critérios como restrição de energia, localização dos nós ou capacidade de armazenamento.

ProFlex é dividido em três fases (Seção 3). A primeira fase é a construção de árvores de roteamento iniciada por todos os nós especiais. Essas árvores são utilizadas no roteamento dos dados para os nós selecionados como armazenadores. A segunda fase realiza a distribuição dos fatores de importância, os quais são utilizados na última fase para determinar se um nó armazena um dado ou não. A última fase é a distribuição de dados, a qual utiliza as árvores de roteamento e os fatores de importância para determinar se um nó sensor deve armazenar ou não um determinado pacote de dados. Resultados de simulação (Seção 4) mostram que ao utilizar uma rede heterogênea, *ProFlex* aumenta a resiliência da rede em cenários com perda de mensagens e diminui a sobrecarga de mensagens transmitidas, no entanto ao preço de uma diminuição na eficiência da coleta de dados. Além disso, discute-se os trabalhos relacionados e as principais suposições na Seção 2, e finalmente conclui-se o trabalho na Seção 5.

2. Trabalhos Relacionados

2.1. Protocolos de Distribuição de Dados

Na literatura, algumas propostas para a realização da distribuição de dados em RSSFs podem ser encontradas. Aqui são discutidas algumas delas.

Bar-Yossef et al. [Bar-Yossef et al. 2008] propõe um serviço de associação (*membership service*) aleatória para redes *ad hoc* chamado RaWMS. O protocolo provê para cada nó um subconjunto dos dados da rede, os quais são escolhidos de forma aleatória e uniforme. O protocolo é baseado em uma técnica chamada *reverse maximum degree random walk* (RW). Utilizando essa técnica, no RaWMS cada nó sensor produtor de dados cria uma mensagem e inicia vários RWs, onde a mensagem contém a identificação do nó e o dado produzido por ele. Cada RW percorre a rede por um número predeterminado de

passos, portanto cada mensagem possui um tempo de vida associado que define o tamanho do RW. O nó no qual o RW termina comporta-se como um nó escolhido de forma aleatória e uniforme dentre todos os nós da rede para armazenar a mensagem. Os autores provam que ao final do RW, cada nó possuirá um subconjunto dos dados da rede escolhidos de forma aleatória e uniforme. Apesar do RaWMS apresentar resultados encorajadores com relação a distribuição uniforme dos dados, o que resulta em uma excelente eficiência na coleta de dados, conforme será apresentado mais adiante nesse trabalho, RaWMS possui uma sobrecarga de mensagens muito elevada.

Com o objetivo de efetivamente realizar uma distribuição uniforme dos dados e ao mesmo tempo reduzir a sobrecarga de comunicação, em [Vecchio et al. 2010], Vecchio et al. propõem Deep, um protocolo de distribuição de dados proativo baseado na densidade para RSSFs com *sinks* móveis e trajetórias desconhecidas previamente. Deep combina um *flooding* probabilístico com uma decisão de armazenamento também probabilística de forma a possibilitar que o *sink* colete um conjunto significativo dos dados visitando apenas um subconjunto muito pequeno de nós da rede. No Deep, quando um nó recebe uma mensagem pela primeira vez, ele decide retransmitir a mensagem com uma certa probabilidade, a qual depende do número de vizinhos do nó e da quantidade desejada de retransmissões na vizinhança do nó. Além disso, se o nó decidir não retransmitir a mensagem, mas não escutar nenhuma retransmissão da mensagem após um período de tempo, então o nó acaba retransmitindo a mensagem de toda forma. Finalmente, ao receber uma mensagem, um nó decide armazená-la de acordo com uma dada probabilidade, a qual depende da quantidade máxima de dados que o nó pode armazenar. Apesar de apresentar resultados tão satisfatórios quanto o RaWMS, Deep ainda apresenta uma alta sobrecarga de mensagens quando comparado à outros protocolos.

Dentre eles, destaca-se o Supple [Viana et al. 2010], um protocolo flexível para a distribuição de dados em RSSFs composta tanto por *sinks* móveis quanto estáticos. No Supple, inicialmente é construída uma árvore de roteamento iniciada por um nó central. O nó central é responsável por receber todos os dados da rede e replicá-los. Em seguida, ocorre a atribuição de pesos para cada nó, os quais representam a probabilidade de um nó realizar armazenamento de dados para outros nós ou não. Supple utiliza o número de saltos a partir do nó central para calcular essa probabilidade. Finalmente, cada nó sensor envia seus dados para o nó central, e este os replica e os distribui utilizando a árvore de roteamento construída anteriormente. O número de réplicas depende dos pesos dos nós e da quantidade de dados que cada nó pode armazenar. Os autores afirmam que um *sink* móvel ao visitar uma pequena fração dos nós, é capaz de coletar uma grande porcentagem dos dados produzidos pela rede. Além disso, em decorrência da utilização do mecanismo de replicação, a falha de um pequeno número de nós não acarreta na perda de dados. No entanto, a desvantagem do Supple é a sobrecarga de mensagens nos nós próximo ao nó central, semelhante ao problema de *energy hole*.

2.2. Pressuposições Utilizadas

Nesse trabalho, considera-se que existe uma grande quantidade de nós (n) depositada em uma área geográfica com o objetivo de coletar ou monitorar eventos. Todos os nós são unicamente identificados e podem ser de dois tipos. O primeiro, nós *L-sensor*, são nós tradicionais com limitações de recursos. O segundo tipo, nós *H-sensor*, são nós mais poderosos quando comparados aos nós *L-sensor*, ou seja, apresentam maior poder de pro-

cessamento, armazenamento e comunicação. Uma questão é *Por que não implementar uma rede composta apenas de nós H-sensor?* Apesar dos nós *H-sensor* serem mais poderosos, eles também são muito mais custosos que os nós *L-sensor*. Portanto, assume-se que a rede é composta por n_L nós *L-sensor* e n_H nós *H-sensor*, onde $n_L \gg n_H$. Além disso, nós selecionados como armazenadores (*H-sensor* e/ou *L-sensor*), após a distribuição dos dados, são providos de um subconjunto v dos dados sensoreados por outros nós ou por eles mesmos. Esse conjunto de nós armazenadores é identificado como S . Portanto, cada nó pode armazenar dados de alguns nós, mas não de todos os nós da rede.

Ademais, assume-se uma rede conectada. Um nó *L-sensor* i pode comunicar-se com outro nó j (*L-sensor* ou *H-sensor*) que estiver dentro de seu raio de comunicação r_1 , ou seja, a distância entre i e j deve ser menor ou igual a r_1 . Já os nós *H-sensor* são equipados com dois rádios de comunicação, cada um operando em uma frequência diferente e com diferentes raios de comunicação (r_1 e r_2 , $r_2 \gg r_1$). Logo, um nó *H-sensor* pode se comunicar com nós *L-sensor* ou *H-sensor* que estejam dentro de seu raio de comunicação r_1 e r_2 , respectivamente.

Finalmente, cada nó i possui um parâmetro $I(i)$ que define o seu fator de importância na rede ($I: S \rightarrow [0, 1]$, $S \subseteq$ Conjunto de nós da rede). Os fatores de importância são inicialmente atribuídos com base em um critério externo. Tal fator determina quais os nós da rede serão os responsáveis por armazenar dados e o quão importante eles serão no armazenamento. Por exemplo, se o critério utilizado for a localização do nó, então os nós na localização específica terão $I(i) = 1$ e os demais $I(i) = 0$. Já se o critério for utilizar apenas os nós *H-sensor* no armazenamento, então os *H-sensor* terão $I(i) = 1$ e os nós *L-sensor* terão $I(i) = 0$. Por outro lado, se todos os nós da rede podem ser escolhidos de forma uniforme para armazenar dados, então todos os nós terão $I(i) = 1$.

3. Protocolo Proposto

O protocolo ProFlex é composto por três fases, as quais são descritas a seguir.

3.1. Construção das Árvores de Roteamento

A primeira fase é a construção de árvores de roteamento iniciada por cada nó *H-sensor*. Essas árvores agregam o menor caminho de cada nó *L-sensor* para o nó *H-sensor* mais próximo. Nesse trabalho, menor caminho significa o caminho com o menor número de saltos, no entanto qualquer outra métrica pode ser empregada, como atraso, capacidade do enlace, etc. Perceba que apesar de cada nó *H-sensor* construir uma árvore a partir dele mesmo, cada nó *L-sensor* pertence apenas à árvore enraizada no nó *H-sensor* mais próximo. Logo, durante a construção das árvores, quando um nó *L-sensor* recebe várias mensagens de nós *H-sensor*, ele irá atualizar suas informações e repassar adiante apenas as mensagens do nó *H-sensor* mais próximo. Caso contrário, ele apenas descarta as mensagens. Por simplicidade de apresentação, nesse trabalho assume-se que as árvores construídas são árvores binárias.

3.2. Distribuição dos Fatores de Importância

No ProFlex, todos os nós no conjunto de nós armazenadores possuem um fator de importância atribuído de acordo com a função $I: S \rightarrow [0, 1]$. O fator de importância

atribuído a um nó i determina se o nó i irá desempenhar o papel de nó de armazenamento para outros nós ($I(i) > 0$) ou não ($I(i) = 0$), e mais adiante será utilizado no cálculo da probabilidade de armazenamento do nó i .

O fator de importância é definido de acordo com um critério de seleção. Apesar do ProFlex permitir a utilização de qualquer critério, por questões de comparação com os trabalhos descritos na Seção 2.1, aqui optou-se por utilizar o critério de seleção uniforme. Portanto, todos os nós da rede possuem $I(i) = 1$. Tal valor dá a todos os nós do conjunto de nós armazenadores a mesma chance de serem selecionados como armazenadores. É importante perceber que tanto nós *L-sensor* quanto *H-sensor* possuem o mesmo fator de importância, no entanto os nós *H-sensor* poderiam ter um fator de importância maior e assim serem capazes de armazenar mais dados que os nós *L-sensor*. Decidiu-se por não utilizar essa opção, pois nesse trabalho a única intenção é tirar proveito das características topológicas inerentes às RSSFs heterogêneas e não do maior poder de armazenamento dos nós *H-sensor*.

Após definir o conjunto de nós armazenadores, um grande desafio é determinar *qual a quantidade de dados um nó deve armazenar*, ou seja, qual deve ser o tamanho do subconjunto de dados v armazenados em um nó. Nos protocolos existentes [Bar-Yossef et al. 2008, Viana et al. 2010, Vecchio et al. 2010], o tamanho do subconjunto v é um parâmetro configurado de maneira estática. Ele é configurado considerando-se uma distribuição uniforme dos dados e é baseado no tamanho do conjunto de nós armazenadores em uma dada estrutura de replicação. Com isso, garante-se que haverá espaço suficiente para armazenar todos os dados gerados pela rede. Em particular, no Supple, apenas uma estrutura de replicação é utilizada (uma árvore de roteamento), logo se o critério de seleção utilizado for o uniforme, então o conjunto de nós armazenadores será equivalente ao número de nós na árvore de roteamento, ou seja, n nós. Por outro lado, ProFlex emprega várias estruturas de replicação (estruturas T_A , T_B e T_C na Figura 1). Portanto, cada estrutura de replicação define um conjunto diferente de nós armazenadores e isso requer uma configuração dinâmica do tamanho do subconjunto de dados v para cada estrutura de replicação.

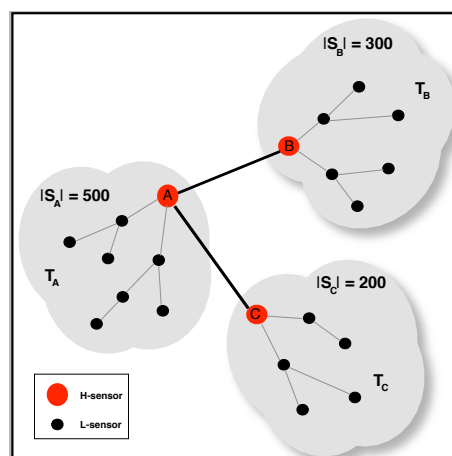


Figura 1. Rede com 1000 nós sensores dos quais 3 são nós *H-sensor*

Viana et al. [Viana et al. 2010] mostram que para um subconjunto de dados de tamanho $|v|$ e para uma rede com n produtores de dados, o conjunto de nós armazenadores

S deve conter no mínimo $\Theta(\frac{n}{v} \ln n)$ nós sensores para garantir com alta probabilidade o armazenamento de todos os n dados coletados (assumindo que cada nó produz apenas um pacote de dados). Logo, $v \geq \frac{n}{|S|} \ln n$. Viana et al. também mostram que um subconjunto $|v| = \sqrt{n}$ provê um bom compromisso entre resiliência e o consumo dos recursos dos nós sensores quando $|S| = n$. Perceba que o tamanho do subconjunto de dados armazenado em cada nó sensor depende do tamanho de S e da quantidade de nós produtores de dados em uma estrutura de replicação. Como o ProFlex emprega várias estruturas de replicação, o tamanho do subconjunto v será diferente para cada árvore. Conforme será discutido na próxima seção, um nó H -sensor h armazena em sua árvore todos os dados produzidos em sua própria árvore e os dados produzidos em árvores vizinhas. Árvores vizinhas são árvores enraizadas em nós H -sensor que possuem conexão com o nó h , denotadas por $N(h)$. Por exemplo, a Figura 1 mostra uma rede com três nós H -sensor e suas respectivas árvores (A , B e C). Nessa figura, a árvore enraizada no H -sensor A tem 500 nós armazenadores ($|S| = 500$), e possui duas árvores vizinhas (B e C).

Algoritmo 1: Distribuição do fator de importância

```

foreach nó  $i$  do
  └ criar  $(I_l(i), I(i), I_r(i))$ 
foreach nó  $i$  em uma pesquisa em profundidade iniciada a partir dos nós raízes do
  └ if  $j$  = filho da esquerda de  $i$  then
    └  $I_l(i) = I_l(j) + I(j) + I_r(j)$ 
    └ if  $k$  = filho da direita de  $i$  then
      └  $I_r(i) = I_l(k) + I(k) + I_r(k)$ 
foreach nó  $H$ -sensor  $h$  do
  └  $|S_h|$  = Tamanho do conjunto de nós armazenadores de  $h$ 
  └ Enviar  $|S_h|$  para os  $H$ -sensor vizinhos
foreach nó  $H$ -sensor  $h$  do
  └  $|S_{agg}^h| = |S_h| + \sum_{j \in N(h)} |S_j|$ 
foreach nó  $H$ -sensor  $h$  do
  └  $|v| = \sqrt{|S_{agg}^h|}$ 
  
```

Para o caso especial em que todos os nós produzem dados, um nó H -sensor precisa saber apenas o número de nós armazenadores em sua árvore e nas árvores vizinhas. Essa informação pode ser encaminhada junto com a distribuição dos fatores de importância. O Algoritmo 1 mostra como o tamanho do conjunto de nós armazenadores e os fatores de importância são distribuídos. A ideia é inicializar cada nó i com a tupla $(I_l(i), I(i), I_r(i))$, onde $I_l(i)$ (semelhante ao terceiro componente $I_r(i)$) é o fator de importância da subárvore da esquerda (semelhante ao da direita) do nó i , e $I(i)$ é o fator de importância do próprio nó i . Perceba que $I_l(i)$ (semelhante ao $I_r(i)$) é a soma de todos os fatores de importância dos nós localizados na subárvore a esquerda (direita) do nó i .

Quando o nó H -sensor h sabe o tamanho do conjunto de nós armazenadores $|S_h|$ em sua árvore, ele encaminha esse valor para as árvores vizinhas (H -sensor vizinhos). Eventualmente, o nó h também irá receber o tamanho do conjunto de nós armazenadores de suas árvores vizinhas. Finalmente, o nó h calcula o tamanho agregado do conjunto de nós armazenadores $|S_{agg}^h|$, ou seja, seu próprio conjunto de nós armazenadores mais

os das árvores vizinhas $|S_{aggr}^h| = |S_h| + \sum_{j \in N(h)} |S_j|$. Com base nessa informação, o nó h calcula o tamanho do subconjunto $|v| = \sqrt{|S_{aggr}^h|}$ e agrega esse valor em todos os pacotes de dados enviados em sua árvore de replicação. Utilizando essa informação, os nós armazenadores sabem quando a sua memória local atingiu sua capacidade máxima. Por exemplo, na Figura 1, para o nó H -sensor A , $|S_A| = 500$ e $\sum_{j \in N(A)} |S_j| = 300 + 200$, logo $|S_{aggr}^A| = 1000$ e $|v| = 31$. Assim, todos os nós armazenadores em A terão $|v| = 31$. Perceba que os nós H -sensor B e C também executarão os mesmos passos para calcular o valor de $|v|$ para suas árvores.

3.3. Distribuição de Dados

A fase de distribuição de dados é a principal fase do ProFlex, pois ela é responsável por distribuir corretamente os dados sensoreados entre os nós armazenadores. No caso de um critério de seleção uniforme (em que todo nó i possui o mesmo valor $I(i)$), ProFlex garante uma distribuição uniforme dos dados entre os nós armazenadores. Ademais, o subconjunto v de cada nó sensor é construído graças a distribuição de $r(v)$ réplicas para cada dado. Mais especificamente, é a transmissão de $r(v)$ réplicas pelo nó raiz i da árvore T_i que irá garantir o armazenamento de $|v| = \sqrt{|S_{aggr}^i|}$ pacotes de dados em cada nó armazenador dessa árvore.

O Algoritmo 2 mostra as principais ações que um nó deve realizar ao produzir ou receber um pacote de dados. Inicialmente, quando um nó L -sensor produz um pacote de dados ou o recebe de um nó filho, ele simplesmente repassa a mensagem para o seu pai, até que a mensagem alcance o nó H -sensor na raiz da árvore. Quando um H -sensor produz um pacote de dados ou o recebe de um de seus filhos, inicialmente ele calcula quantas réplicas $r(v)$ para o pacote devem ser produzidas de modo a enviar para seus filhos e para as árvores vizinhas. Tal cálculo deve garantir que os nós armazenadores recebam com grande probabilidade $|v|$ pacotes de dados distintos. Como no caso do tamanho do subconjunto v , o número de réplicas é calculado com base no tamanho do conjunto de nós armazenadores. Conforme apresentado em [Viana et al. 2010], quando $|S| = n$, então $r(v) = n \ln(\frac{n}{n-v})$; e quando $|S| \neq n$, então $r(v) = n \ln n$. Em especial, $|v| = \sqrt{n}$ provê um bom compromisso entre resiliência e o consumo de recursos e faz com que $r(v) \approx \sqrt{n}$, quando $|S| = n$. Assim, um nó H -sensor h precisa saber apenas o número de nós armazenadores em sua árvore e nas árvores vizinhas. Durante a distribuição de dados, essa informação já está disponível, logo $r(v) = \sqrt{|S_{aggr}^h|}$. Após calcular o número de réplicas $r(v)$ para um dado pacote, o nó h deve determinar quantas réplicas de $r(v)$ serão enviadas em sua própria árvore e quantas vão para as árvores vizinhas. Esse valor é proporcional a porcentagem de nós armazenadores em cada árvore com respeito ao número total de nós armazenadores em S_{aggr}^h . Seja $r_k(v)$ o número de réplicas para uma árvore T_k , onde $k \in N(h)$, logo $r_k(v) = \frac{|S_k|}{|S_{aggr}^h|} \times r(v)$. Por exemplo, na Figura 1 o nó H -sensor A calcula $r(v) = 31$. Portanto, o nó A irá enviar $r_B(v) = \frac{300}{1000} \times 31 = 9$ réplicas para o nó B , $r_C(v) = \frac{200}{1000} \times 31 = 6$ réplicas para o nó C e $r_A(v) = \frac{500}{1000} \times 31 = 16$ para a sua própria árvore. Após determinar o número de réplicas, o nó H -sensor envia $r_k(v)$ réplicas do dado para cada $k \in N(h)$. Finalmente, o nó h chama a função **ForwardData** (Algoritmo 3) $r_h(v)$ vezes. A distribuição na árvore do nó h é feita de acordo com os valores dos fatores de importância das árvores da esquerda e da direita e do seu próprio

valor de importância. Além disso, quando um nó *L-sensor* recebe um pacote de seu pai, ou um nó *H-sensor* recebe um pacote de um outro nó *H-sensor*, ele simplesmente chama **ForwardData** para determinar se o pacote será repassado ou armazenado localmente. O algoritmo naturalmente para quando o pacote é recebido por um nó folha. Ao final da distribuição de dados, todos os nós armazenadores terão com alta probabilidade um subconjunto v de dados uniformemente distribuídos.

Algoritmo 2: Distribuição de dados

```

// Em direção a raiz
if nó L-sensor  $i$  produz dado then
  └ Envia dado para o pai
if nó L-sensor  $i$  recebe dado de um filho then
  └ Repassa dado para o pai
if nó H-sensor  $h$  produz dado ou recebe dado de L-sensor  $i$  then
  └ Calcula  $r_k(v)$  para cada H-sensor  $k \in N(h)$ 
  └ Envia  $r_k(v)$  réplicas do dado para cada H-sensor  $k \in N(h)$ 
  └ Chama ForwardData(dado)  $r_h(v)$  vezes

// Em direção aos filhos
if nó L-sensor  $i$  recebe dado do pai then
  └ Chama ForwardData(data)
if nó H-sensor  $h$  recebe dado de H-sensor  $k$  que pertença a  $N(h)$  then
  └ Chama ForwardData(data)
  
```

Algoritmo 3: Procedimento **ForwardData**(*data*)

```

Input: Pacote de dados
Escolha de maneira aleatória e uniforme  $x \in [0, I_l(i) + I(i) + I_r(i)]$ 
if  $x < I_l(i)$  then
  └ Envia dado para o filho da esquerda
if  $I_l(i) \leq x \leq I(i)$  then
  └ Armazena dado localmente
if  $I(i) + I_l(i) < x$  then
  └ Envia dado para o filho da direita
  
```

4. Avaliação de Desempenho

Com o objetivo de avaliar o desempenho do protocolo proposto, uma série de simulações foi realizada. Todas as simulações foram realizadas utilizando o simulador Signalgo [Group 2008], versão 0.75.3. Daqui em diante, todos os resultados apresentados representam a média de um conjunto de repetições de forma a atingir um intervalo de confiança igual a 95%. Cada simulação é composta por uma etapa de distribuição de dados, onde cada nó produz apenas um pacote de dados, e uma etapa de coleta de dados. Na primeira etapa, cada nó executa um protocolo de distribuição de dados de modo a distribuir os dados entre os nós armazenadores. Então, na segunda etapa, um *sink* móvel visita cada nó na rede aleatoriamente de forma a coletar os dados armazenados na rede, ou seja, n pacotes de dados. Nesse trabalho, foram avaliados tanto cenários confiáveis quanto cenários com perdas de mensagens ou falhas de nós. É importante perceber que

todos os protocolos aqui apresentados foram simulados utilizando uma rede heterogênea, pois assim a comparação com o protocolo proposto torna-se mais justa. Percebe-se que mesmo que um protocolo não tenha sido projetado para operar em redes heterogêneas, ele ainda é capaz de tirar proveito das características favoráveis desse tipo de rede. Os principais parâmetros utilizados nas simulações são apresentados na Tabela 1. Vários desses parâmetros foram variados em nossas análises, no entanto, devido à restrição de espaço, optou-se por utilizar valores para os parâmetros que acarretam em um melhor desempenho para todos os protocolos aqui avaliados. Além disso, os valores de parâmetros como $|v|$, $r(v)$, TTL e β são específicos para os protocolos Supple, Deep e RaWMS, não sendo assim utilizados pelo ProFlex.

Tabela 1. Principais parâmetros de simulação

Parâmetro	Valor
Número de nós <i>H-sensor</i>	20
Número de nós <i>L-sensor</i>	980
Área de sensoriamento	$800 \times 800 \text{ m}^2$
Densidade da rede	20
Raio de comunicação dos <i>H-sensor</i>	480 m
Raio de comunicação dos <i>L-sensor</i>	60 m
Fatores de importância $I(i)$	1
$ v $ and $r(v)$	31
TTL (RaWMS)	125
β (Deep)	5.4

Conforme mencionado anteriormente, um problema recorrente em RSSFs com *sink* estático é o problema de *energy hole*. Nos protocolos ProFlex e Supple, quando um nó possui dados para distribuir para a rede, primeiramente ele envia o dado para a raiz da árvore e então o nó raiz fica responsável por distribuir esse dado para toda a rede (Deep e RaWMS não utilizam árvore na distribuição). Portanto, os nós localizados próximo à raiz da árvore tendem a rotear mais mensagens que os demais nós, assim como no caso do problema de *energy hole*. Logo, avaliou-se o número de mensagens transmitidas por um nó em função da sua profundidade na árvore de roteamento para um cenário sem perda de mensagens ou falhas de nós.

Conforme pode ser observado na Figura 2, os nós localizados próximo a raiz tendem a enviar mais mensagens. No entanto, no ProFlex, cada nó *H-sensor* é também o nó raiz de uma árvore, conseqüentemente mais árvores são criadas para a distribuição de dados na rede, enquanto que o Supple emprega apenas uma árvore de roteamento. Dessa forma, a sobrecarga de mensagens é distribuída entre as várias árvores existentes, aliviando assim o problema de *energy hole*. É importante frisar que apesar do Supple operar sob uma rede heterogênea, ele não foi projetado para tirar proveito das características desse tipo de rede, diferente do ProFlex. Em particular, quando a profundidade do nó na árvore é 1, ProFlex transmite cerca de 91% menos mensagens quando comparado ao Supple.

O resultado anterior mostra que o ProFlex não sofre tanto do problema de *energy hole* quanto o Supple, mas será que isso significa que o ProFlex possui uma sobrecarga de mensagens menor que os demais protocolos? A Tabela 2 mostra o número de mensagens transmitidas para todos os protocolos avaliados. Conforme pode ser observado,

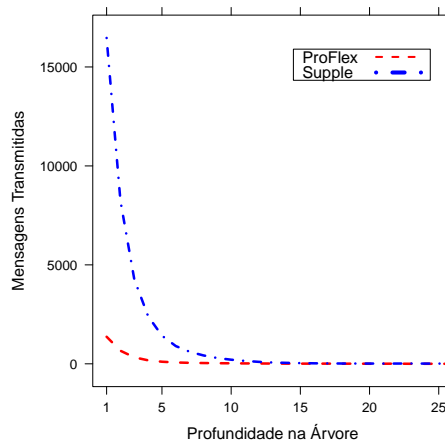


Figura 2. Número de mensagens enviadas por um nó em função da sua profundidade na árvore

Tabela 2. Sobrecarga de mensagens dos protocolos

Protocolo	Número total de mensagens transmitidas
ProFlex	171428
Supple	280274
Deep	344158
RaWMS	3629489

ProFlex é o protocolo que transmite o menor número de mensagens. Em particular, ProFlex transmite cerca de 95% menos mensagens quando comparado ao RaWMS. Já quando comparado ao Supple, a redução é de cerca de 38%. Assumindo que a comunicação é a principal responsável pelo consumo de energia em RSSFs, esse resultado indica também que o ProFlex é o protocolo com o menor consumo de energia, o que acarreta um maior tempo de vida para a rede.

A baixa sobrecarga de mensagens do ProFlex é certamente uma característica desejável devido às restrições de recursos das RSSFs, mas isso certamente apresenta um impacto na eficiência da coleta de dados. A Figura 3(a) mostra a eficiência na coleta de dados para todos os protocolos em um cenário sem perda de mensagens ou falha de nós. Nesse cenário, após a etapa de distribuição de dados para os nós armazenadores, um *sink* móvel localizado inicialmente em uma posição aleatória visita um nó nessa posição e em seguida escolhe aleatoriamente um vizinho desse nó para visitar. Ao visitar um nó, o *sink* coleta todos os dados armazenados nesse nó. Esse processo continua até que todos os nós da rede tenham sido visitados. O protocolo capaz de recuperar 100% dos dados visitando o menor número de nós sensores é o protocolo com a melhor eficiência na coleta de dados. Conforme pode ser observado na figura, ProFlex precisa visitar aproximadamente 10% mais nós que os demais protocolos. Tal resultado deve-se ao fato que o ProFlex transmite menos mensagens que os demais protocolos, conforme mostrou-se no resultado anterior, e portanto há menos réplicas de dados distribuídas na rede. Além disso, já que cada réplica produzida em uma dada árvore é distribuída apenas nessa árvore e nas árvores vizinhas, e nem todas as árvores são vizinhas de todas as outras, então é possível que um dado não seja replicado em algumas partes da rede. Tal fato pode ser observado na Fi-

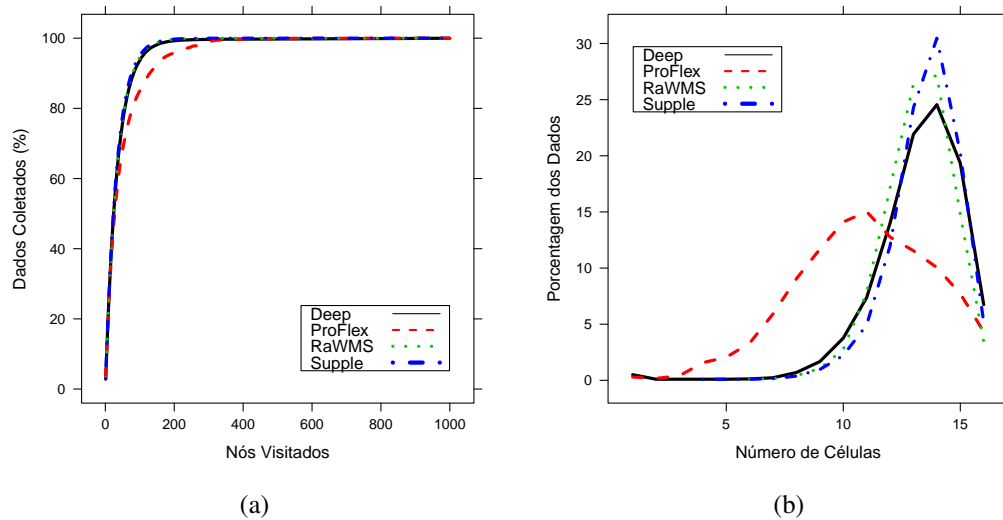


Figura 3. (a) Eficiência na coleta de dados para um cenário confiável. (b) Eficiência na disseminação de dados para um cenário confiável

gura 3(b). Nesse cenário, dividiu-se a área de sensoriamento em 16 células de mesmo tamanho e calculou-se para cada pacote de dados o número de células em que ele foi coletado pelo *sink*. Conforme pode-se observar, os protocolos da literatura distribuem boa parte dos dados em cerca de 14 células da rede, enquanto que o ProFlex distribui os dados em cerca de 11 células. Essa pequena diferença resulta em um impacto na eficiência da coleta de dados. Esse resultado pode ser melhorado aumentando-se o raio de comunicação dos nós *H-sensor*, o que resulta em um aumento no número de conexões entre *H-sensor*, melhorando a distribuição de dados. Além disso, percebe-se que um nó *H-sensor* h poderia enviar réplicas de dados para as árvores que não são suas árvores vizinhas, ou seja, enviar dados para os nós *H-sensor* que estão a mais de dois saltos de distância do nó h (utilizando comunicação de múltiplos saltos), possibilitando assim a distribuição de dados em um número maior de células na rede.

Após avaliar o desempenho do ProFlex em um cenário confiável, verificou-se o comportamento do protocolo em cenários com perda de mensagens e falha de nós. A Figura 4(a) mostra a eficiência na coleta de dados para um cenário com perda de mensagens. Percebe-se que o protocolo com os melhores resultados até então, Supple, não apresenta o mesmo desempenho em cenários com perda de mensagens, apesar de ainda ser capaz de coletar 100% dos dados. Observa-se que o ProFlex precisa visitar mais nós que o Deep e o RaWMS para coletar todos os dados, no entanto quando comparado ao Supple, o número de nós visitados é menor. De fato, o desempenho dos protocolos ProFlex, Deep e RaWMS nesse cenário é praticamente o mesmo que o cenário confiável, o que mostra que tais protocolos são resilientes a perdas de mensagens, o que não ocorre com o Supple.

Já nos cenários com falhas de nós, ProFlex não apresenta a mesma resiliência. Conforme apresentado na Figura 4(b), tanto o ProFlex quanto o Supple são afetados pela falha de nós, enquanto o Deep e o RaWMS praticamente não sofrem qualquer mudança. Tal resultado deve-se ao fato de que tanto o ProFlex quanto o Supple utilizam uma infraestrutura baseada em árvore para realizar a distribuição dos dados, enquanto o Deep e

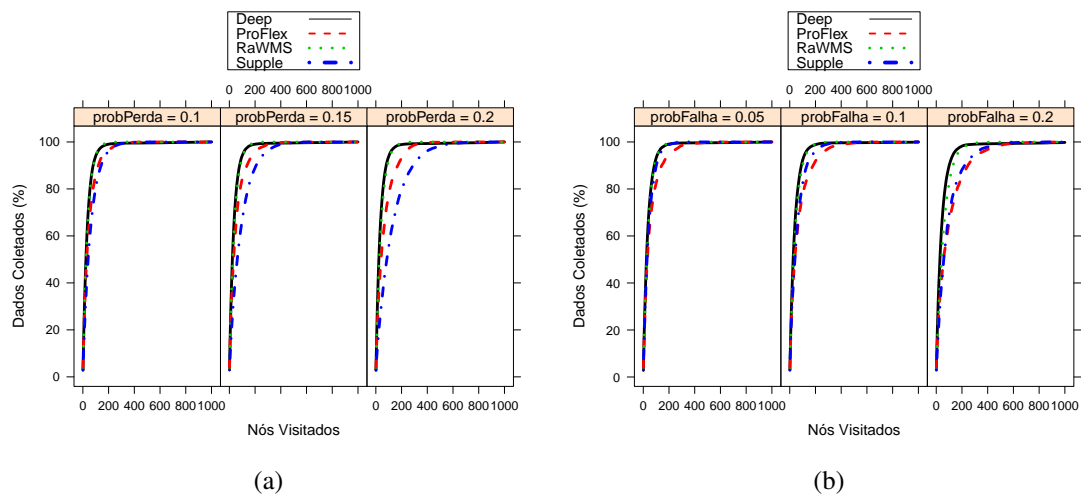


Figura 4. (a) Eficiência na coleta de dados para um cenário com perda de mensagens. (b) Eficiência na coleta de dados para um cenário com falha de nós

o RaWMS não utilizam qualquer tipo de infraestrutura previamente definida. Portanto, quando um nó na árvore falha, todos os seus filhos também são comprometidos, impedindo assim que todos eles possam receber qualquer mensagem. Por outro lado, é importante perceber que tanto o Deep quanto o RaWMS implementam mecanismos de recuperação para cenários com perda de mensagens ou falhas de nós, o que não ocorre no ProFlex e Supple. Logo, o desempenho de ambos os protocolos pode melhorar consideravelmente caso algum mecanismo dessa natureza seja desenvolvido.

5. Conclusões e Trabalhos Futuros

Nesse trabalho apresentou-se ProFlex, um protocolo de distribuição de dados para RS-SFs heterogêneas. Mostrou-se que a utilização de uma rede heterogênea virtualmente diminui a ocorrência do problema de *energy hole* na rede. Por exemplo, nós localizados próximo às raízes das árvores de roteamento enviam cerca de 91% menos mensagens quando comparado ao Supple. Além disso, ProFlex é o protocolo com a menor sobrecarga de mensagens quando comparado aos principais protocolos existentes na literatura, ao custo de uma degradação na eficiência da coleta de dados. Por exemplo, ProFlex transmite aproximadamente 95% menos mensagens que o RaWMS e 38% menos mensagens que o Supple, no entanto necessita visitar aproximadamente 10% mais nós durante a coleta de dados. Além disso, em cenários com perda de mensagens, ProFlex não sofre qualquer alteração com relação ao seu desempenho. Porém em cenários com falha de nós, o desempenho do ProFlex sofre uma degradação semelhante ao protocolo com os melhores resultados, ou seja, Supple. No entanto, tal degradação de desempenho nesse cenário específico pode ser facilmente evitada ao se utilizar um mecanismo para a recuperação de falhas, como por exemplo, o uso de reconhecimento de mensagens. Como trabalhos futuros espera-se avaliar a eficiência da distribuição de dados sob diferentes topologias, propor mecanismos de recuperação de falhas e propor modelos que forneçam o número ótimo de nós *H-sensor* e seus raios de comunicação para uma dada configuração de rede.

Referências

- Akyildiz, I., Su, W., Sankarasubramaniam, Y., and Cayirci, E. (2002). Wireless sensor networks: a survey. *Computer Networks*, 38(4):393 – 422.
- Anastasi, G., Conti, M., and Di Francesco, M. (2008). Data collection in sensor networks with data mules: An integrated simulation analysis. In *IEEE Symposium on Computers and Communications*, pages 1096–1102.
- Bar-Yossef, Z., Friedman, R., and Kliot, G. (2008). Rawms - random walk based lightweight membership service for wireless ad hoc networks. *ACM Trans. Comput. Syst.*, 26:5:1–5:66.
- Basagni, S., Carosi, A., Melachrinoudis, E., Petrioli, C., and Wang, Z. M. (2008). Controlled sink mobility for prolonging wireless sensor networks lifetime. *Wireless Networks*, 14:831–858.
- Chatzigiannakis, I., Kinalis, A., and Nikolettseas, S. (2008). Efficient data propagation strategies in wireless sensor networks using a single mobile sink. *Computer Communications*, 31:896–914.
- Group, E. D. C. (2008). Sinalgo - simulator for network algorithms. <http://dcg.ethz.ch/projects/sinalgo/>.
- Helmy, A. (2003). Small worlds in wireless networks. *Communications Letters, IEEE*, 7(10):490 – 492.
- Liu, A.-F., Wu, X.-Y., Chen, Z.-G., and Gui, W.-H. (2010). Research on the energy hole problem based on unequal cluster-radius for wireless sensor networks. *Computer Communications*, 33(3):302 – 321.
- Sharma, G. and Mazumdar, R. (2008). A case for hybrid sensor networks. *Networking, IEEE/ACM Transactions on*, 16(5):1121 –1132.
- Sheng, B., Tan, C. C., Li, Q., and Mao, W. (2007). An approximation algorithm for data storage placement in sensor networks. In *Proc. of the Inter. Conf. on Wireless Algorithms Systems and Applications*, pages 71–78.
- Song, L. and Hatzinakos, D. (2007). Architecture of wireless sensor networks with mobile sinks: Sparsely deployed sensors. *Vehicular Technology, IEEE Transactions on*, 56(4):1826 –1836.
- Vecchio, M., Viana, A. C., Ziviani, A., and Friedman, R. (2010). Deep: Density-based proactive data dissemination protocol for wireless sensor networks with uncontrolled sink mobility. *Elsevier Computer Communication*, 33(8).
- Viana, A. C., Herault, T., Largillier, T., Peyronnet, S., and Zaïdi, F. (2010). Supple: a flexible probabilistic data dissemination protocol for wireless sensor networks. In *Proc. of the 13th ACM Inter. Conf. on Modeling, analysis, and simulation of wireless and mobile systems*, pages 385–392.
- Yang, H., Ye, F., and Sikdar, B. (2006). Simple: Using swarm intelligence methodology to design data acquisition protocol in sensor networks with mobile sinks. In *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1 –12.