Um Algoritmo de Roteamento Ciente de Agregação de Dados para Redes de Sensores sem Fio

Leandro A. Villas[1], **Heitor S. R. Filho**[1], **Regina B. Araujo**[2], **Antonio A. F. Loureiro**[1]

[1]Departamento de Ciência da Computação – Universidade Federal de Minas Gerais
31270-010 – Belo Horizonte– MG – Brasil

[2]Departamento de Ciência da Computação – Universidade Federal de São Carlos
13565-905 – São Carlos, SP – Brazil

{leandro,heitor,loureiro}dcc.ufmg.br, regina@dc.ufscar.br

Abstract. *In this work, we propose a lightweight algorithm to perform in-network aggregation in wireless sensor networks. The algorithm, called DAARP (Data-Aggregation Aware Routing Protocol), differentiates itself from existing solutions in at least three aspects: reduces the number of messages necessary to set up a routing tree; maximizes the number of overlapping routes, and deals with the selection of routes with the highest aggregation rate. Simulations results show that DAARP is more efficient than the compared solutions in all scenarios analyzed in this study.*

Resumo. *Este artigo apresenta um algoritmo de roteamento de baixo custo para realizar agregação de dados em redes de sensores sem fio, DAARP (Data-Aggregation Aware Routing Protocol), que diferencia-se das soluções existentes em pelo menos três aspectos: reduz o número de mensagens necessárias para a construção da infra-estrutura de roteamento; maximiza o número de rotas sobrepostas; e trata da seleção de rotas com maior taxa de agregação. Resultados de simulações mostram que o DAARP é mais eficiente do que as soluções comparadas para todos os cenários avaliados neste trabalho.*

1. Introdução

Os recentes desenvolvimentos nas áreas de comunicação sem fio e sensores multifuncionais com capacidade de comunicação e processamento impulsionaram o crescimento no campo das redes de sensores sem fio (RSSFs). Dessa forma, as RSSFs estão cada vez mais presentes em aplicações como monitoramento ambiental, vigilância de campos militares e muitas outras onde a presença humana não é possível ou não desejada [Younis et al. 2006, Boukerche et al. 2007b]. Nós sensores são dispositivos com energia limitada e o seu consumo é geralmente associado à quantidade de troca de mensagens realizadas, uma vez que a comunicação é a atividade que mais despende energia. Por esse motivo, algoritmos e protocolos concebidos para RSSFs devem considerar o consumo de energia em seu projeto.

A capacidade de processamento dos nós sensores pode ser utilizada para prover melhorias na tarefa de roteamento. Quando essas melhorias são no sentido de reduzir o tráfego de dados, isso é conhecido como *in-network aggregation*[1]. Neste contexto,

[1]Neste trabalho, fusão de informação e agregação de dados são usados como sinônimos.

a fusão de informação tem sido utilizada para: (i) tirar proveito da redundância de dados e aumentar a precisão dos dados, e (ii) reduzir a comunicação e economizar energia [Nakamura et al. 2007, Boukerche et al. 2007a].

A principal contribuição deste trabalho é a proposição de um novo algoritmo de roteamento ciente de agregação de dados para RSSFs. O algoritmo procura rotas que maximizam a agregação de dados. Este algoritmo, denominado DAARP, estabelece uma organização hierárquica da rede de maneira que os nós fonte (nós que transmitem dados de um evento de interesse) são organizados em *clusters* e a comunicação de um *cluster* com o *sink* é realizada por múltiplos saltos. A topologia resultante é uma solução aproximada para o problema da árvore de Steiner conectando nós fonte ao nó sorvedouro (*sink*). O DAARP diferencia-se de outras soluções existentes por encontrar uma estrutura de roteamento com uma taxa maior de agregação de dados que as soluções avaliadas e com menor custo em termos de mensagens de controle transmitidas para criar e manter a estrutura de roteamento.

Os resultados de simulação mostram que o DAARP obtém melhores resultados que outras soluções, apresenta maior eficiência e suas rotas possuem maiores taxas de agregação de dados.

O artigo está organizado da seguinte forma: a seção 2 apresenta os trabalhos relacionados. A seção 3 descreve o algoritmo DAARP, proposto neste trabalho. A avaliação do algoritmo é discutida na seção 4, seguida de conclusões na seção 5.

2. Trabalhos Relacionados

O problema tratado neste trabalho envolve encontrar uma estrutura de roteamento que conecte todos os nós que possuem informações para enviar ao *sink*, incluindo o próprio *sink*. Encontrar a estrutura de roteamento mínima com essas características é um problema conhecido por ser NP-difícil [Krishnamachari et al. 2002], também é conhecida por árvore mínima de Steiner. Segundo [Krishnamachari et al. 2002] a agregação ótima é alcançada quando uma árvore mínima de Steiner é utilizada.

O problema da árvore mínima de Steiner pode ser enunciado da seguinte maneira: dado um grafo $G(\boldsymbol{N}, \boldsymbol{E})$, onde $\boldsymbol{N}$ é o conjunto de vértices, $\boldsymbol{E}$ é um conjunto de arestas. Existe um custo associado que é definido como uma função não negativa aplicada às arestas do grafo, e, seja $\boldsymbol{T} \subseteq \boldsymbol{N}$ um subconjunto de vértices de interesse, a árvore mínima de Steiner é qualquer árvore que conecta todos os vértices $\boldsymbol{T}$ e apresenta custo mínimo, sendo o custo da árvore definido como a soma dos pesos atribuídos às suas arestas. Neste trabalho, G é o grafo que representa uma RSSFs, $\boldsymbol{N}$ é o conjunto de sensores, $\boldsymbol{E}$ é o conjunto de interconexões que ligam os nós sensores e $\boldsymbol{T}$ é o conjunto formado pelos nós sensores que desejam transmitir informações para o *sink*.

Neste problema, os vértices são divididos em dois subconjuntos, o subconjunto $\boldsymbol{T}$ que é chamado de vértices terminais e o subconjunto dos vértices não-terminais. Os vértices terminais são necessariamente incluídos na árvore mínima de Steiner e os vértices não-terminais podem ser incluídos a fim de reduzir o custo da árvore.

Existem diversos trabalhos que propõem heurísticas para determinação de árvores aproximadas para o problema da árvore de Steiner. Algumas propostas como [Robins and Zelikovsky 2000, Hougardy and Prömel 1999] necessitam de uma grande

quantidade de troca de mensagens para estabelecer uma árvore de roteamento e, consequentemente, apresentam altos custos de energia. Essas soluções não são apropriadadas para aplicações como as redes de sensores sem fio devido às restrições de energia apresentadas por esse tipo de rede. Existem algumas soluções mais apropriadas para redes de sensores sem fio como *Shortest Path First* (SPT) [Krishnamachari et al. 2002], *Center-at-Nearest Source* (CNS) [Krishnamachari et al. 2002] e o *Information Fusion-based Role Assignment* (InFRA) [Nakamura et al. 2006].

A utilização da heurística SPT é uma das formas mais simples de construir uma solução aproximada para a árvore de Steiner distribuída. Nessa heurística, cada nó N_i que detecta um evento envia seu pacote contendo as informações coletadas para o *sink* utilizando o menor caminho entre ele e o *sink*. Quando os caminhos dos nós fonte ao *sink* se sobrepõe, os pacotes são agregados. Na heurística CNS, cada nó sensor N_i que detecta um evento na rede transmite os pacotes contendo as informações sensoriadas para algum nó N_j mais próximo do *sink*. O nó N_j agrega todas os pacotes recebidos e os envia para o *sink*.

O algoritmo InFRA constrói a estrutura de roteamento considerando que os nós se agrupam ao detectarem os mesmos eventos e um dos nós do agrupamento o nó coordenador fica responsável por agregar os dados fornecidos pelos nós do agrupamento e enviar os dados agregados para o *sink*. Após os agrupamentos serem formados, as rotas são criadas escolhendos os nós que leva ao menor caminho (em saltos) ao *sink* e que tenha a menor distância-agregada para os coordenadores [Nakamura et al. 2006]. Uma das desvantagens do InFRA é que a cada novo evento que surge na rede é necessário inundar toda a rede para informar os outros nós da rede sobre a ocorrência do novo evento.

O algoritmo proposto nesse trabalho, o DAARP realiza o agrupamento dos nós que detectarem os mesmos eventos como no InFRA, mas usa outra política na eleição do nó líder que é descrita no algoritmo 2. Os coordenadores enviam os dados agregados para o *sink*. Após os agrupamentos serem formados, as rotas são criadas escolhendo os nós que leva ao menor caminho (em saltos) para um nó mais próximo que pertence a estrutura de roteamento já existente, onde esse nó será um ponto de agregação. A estrutura de roteamento do DAARP tende a maximizar os pontos de agregações, utiliza menos pacotes de controle para a construção da estrutura de roteamento, pois diferentemente do InFRA, o DAARP não necessita inundar toda a rede na ocorrência de um novo evento. As próximas seções descrevem o DAARP e o compara com outras abordagens como o InFRA, o CNS e o SPT.

3. O Algoritmo DAARP

O DAARP é um protocolo para redes de sensores sem fio, que visa construir uma árvore de roteamento com as menores rotas (em saltos), que conecte todos os nós fontes ao *sink* e que maximize a agregação de dados. O DAARP é apropriado para redes estáticas (sem mobilidade) e considera os seguintes papéis para a criação da infra-estrutura de roteamento:

sink: nó interessado em receber dados de um conjunto de eventos;
colaborador: nó que detecta evento, ex. membro do cluster;
coordenador: nó que detecta e notifica eventos, i.e. nó líder do cluster;
retransmissor: nó que retransmite as informações sobre os eventos detectados em direção ao *sink*.

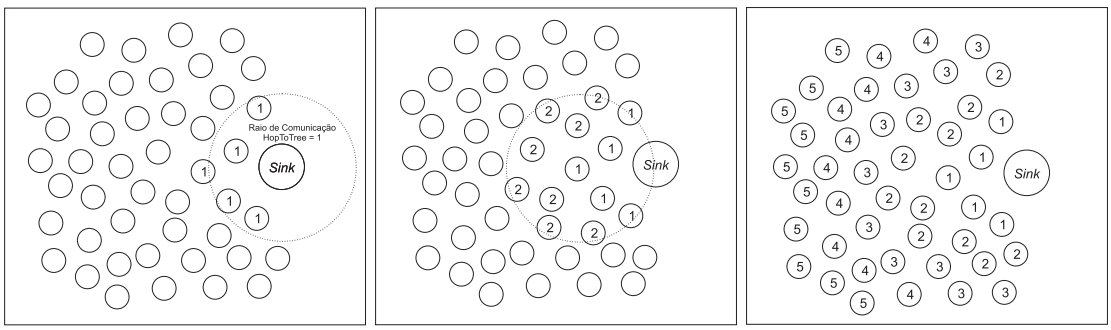
O algoritmo de roteamento DAARP é realizado em três fases. A primeira fase envolve a construção da árvore de saltos (hops) dos nós sensores para o nó *sink*. O nó *sink* inicia o processo de construção da árvore de saltos, que será usada pelo nó coordenador para propagação de mensagens de dados. A segunda fase consiste na formação do cluster e eleição de um líder entre os nós que detectaram a ocorrência do novo evento na rede. A terceira fase é responsável pelo estabelecimento da nova rota para a propagação de mensagens de dados e pela atualização da árvore de saltos.

3.1. Construção da Árvore de Saltos

Na fase de construção da árvore de saltos é configurada a distância, em saltos, que cada nó sensor encontra-se do nó *sink*. Isso é feito a partir do nó *sink* que envia, através de inundação, uma mensagem MCH (Mensagem de Configuração de Hops) para configuração de saltos. A mensagem MCH é mostrada na figura 3.1 onde ID é o identificador do nó que iniciou/retransmitiu a mensagem MCH e HopToTree é a distância, em saltos, pelos quais a MCH passou. O parâmetro HopToTree é iniciado com valor 1 no *sink*, ou seja, o *sink* é a raiz da árvore de roteamento.

Figura 1. Formato da Mensagem MCH

Nessa fase, os nós sensores armazenam os parâmetros ID e HopToTree em sua tabela de roteamento. algoritmo 1 descreve a construção da árvore de saltos. O nó *sink* inunda a rede com uma mensagem MCH e configura no campo HopToTree o valor 1 e o transmite aos seus vizinhos (algoritmo 1, linha 1). Esse passo é ilustrado na figura 2(a). Os nós vizinhos, por sua vez, armazenam, incrementam o campo HopToTree e retransmitem o pacote MCH para seus nós vizinhos como mostra a figura 2(b). Os passos acima ocorrem várias vezes, até que toda a rede esteja configurada (algoritmo 1, linhas 3 à 9), como mostra a figura 2(c).



(a) Sink inicia a configuração (b) Nós atualizam e retransmitem a mensagem MCH (c) Níveis de saltos configurados

Figura 2. Fase de configuração dos níveis de salto

Algoritmo 1: Construção da Árvore de Saltos

1 Nó *sink* faz um *broadcast* da mensagem MCH com o valor de HopToTree = 1
2 **para cada** $u \in \boldsymbol{R}_u$ **faça**
    // R_u é o conjunto de nós que recebeu a mensagem MCH
3 **se** $\text{HopToTree}(u) > \text{HopToTree}(MCH)$ **então**
4 $NextHop_u \leftarrow ID_{MCH}$ // Nó u atualiza o campo $NextHop_u$
5 $HopToTree_u \leftarrow HopToTree_{MCH} + 1$ // Atualiza o valor de $HopToTree_u$
6 $ID_{MCH} \leftarrow ID_u$ // Nó u atualiza o valor do campo ID na mensagem MCH
7 $HopToTree_{MCH} \leftarrow HopToTree_u$ // Nó u atualiza o valor do campo $HopToTree$ na mensagem MCH
8 Nó u faz um *broadcast* da mensagem MCH com os novos valores
9 **fim se**
10 **senão**
11 Nó u descarta a mensagem MCH recebida
12 **fim se**
13 **fim para**

Como a comunicação entre os nós da rede é realizada através da difusão de sinais de rádio (RF), todos os nós vizinhos recebem a transmissão. Dessa forma, um nó que acabou de transmitir uma mensagem, pode receber a mesma mensagem de seu vizinho, gerando um caminho fechado (loop). Para evitar essas transmissões inúteis e economizar energia, cada nó, ao receber uma transmissão da mensagem MCH, compara o HopToTree recebido com o seu HopToTree local. Se o valor do HopToTree local for maior que o valor recebido, o nó atualiza seu HopToTree, incrementa seu valor e o retransmite para seus vizinhos (algoritmo 1, linhas 3 à 9). Caso do valor HopToTree armazenado na tabela de roteamento do nó seja menor ou igual ao valor HopToTree recebido, o nó descarta a mensagem MCH (algoritmo 1, linhas 10 a 13).

3.2. Formação dos Clusters e Eleição do Líder

Quando os nós detectam um evento no ambiente monitorado iniciam o algoritmo de eleição de líder e se candidatam a nó líder (coordenador do grupo). O algoritmo 2 descreve esse processo. Na eleição do coordenador do grupo, todos os nós são elegíveis (algoritmo 2 linha 2). Porém, é eleito apenas o nó que tenha a menor distância da árvore de roteamento (linhas 5 e 6 do algoritmo 2). Quando há empate, isto é, dois ou mais nós concorrentes têm a mesma distância da árvore de roteamento, apenas o nó de maior ID continua elegível (linhas 9 e 10 do algoritmo 2).

Ao final do algoritmo de eleição de líder só existe um nó líder (coordenador) no grupo, se for o primeiro evento o nó líder do grupo é o nó que se encontra mais próximo do *sink*, caso contrário o nó líder do grupo é o nó que se encontra mais próximo da árvore de roteamento já existente, e em caso de empate, o critério de desempate é o ID. Todos os outros nós que detectaram o mesmo evento são nós com papel de colaborador. Os nós colaboradores transmistem as informações coletadas do ambiente para o *sink* através do nó coordenador do grupo.

Algoritmo 2: Formação dos Clusters e Eleição do Líder

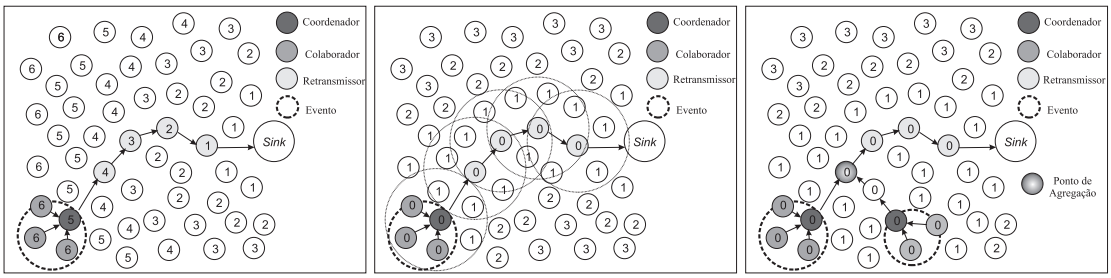
Entrada: S // Conjunto de nós que detectou o evento
Saída: u // Um nó do conjunto S é eleito líder do grupo
1 **para cada** $u \in S$ **faça**
2 $role_u \leftarrow coordenador$ // Nó u se candidata a nó líder
3 Anuncio da detecção de evento // Nó u faz um broadcast da mensagem MCC
4 **para cada** $w \in \mathcal{N}_u$ **faça**
  // N_u é o conjunto de vizinhos do nó u
5 **se** $\mathrm{HopToTree}(u) > \mathrm{HopToTree}(w)$ **então**
6 $role_u \leftarrow colaborador$ // Nó u muda seu papél para *Collaborator*
7 Nó u retransmite a mensagem MCC recebida do nó w
8 **fim se**
9 **senão se** $\mathrm{HopToTree}(u) = \mathrm{HopToTree}(w)$ *e* $\mathrm{ID}(u) > \mathrm{ID}(w)$ **então**
10 $role_u \leftarrow colaborador$ // Nó u muda seu papél para *Collaborator*
11 Nó u retransmite a mensagem MCC recebida do nó w
12 **fim se**
13 **senão**
14 Nó u descarta a mensagem MCC recebida do nó w
15 **fim se**
16 **fim para**
17 **fim para**

3.3. Estabelecimento da Nova Rota e Atualização da Árvore de Saltos

O nó eleito líder do grupo no algoritmo 2 inicia o estabelecimento da nova rota para a propagação de informações sobre o evento. Esse processo está descrito no algoritmo 3. Para isto o coordenador envia uma mensagem MER (Mensagem de Estabelecimento de Rota) para o seu NextHop (algoritmo 3, linha 1). O nó destino ao receber a mensagem MER muda seu papel para retransmissor, retransmite a mensagem MER para o seu NextHop e inicia a atualização da árvore de saltos (algoritmo 3, linhas 4 à 7). Esses passos são realizados até encontrar o nó *sink* ou um nó pertencente à estrutura de roteamento já estabelecida.

As rotas são formadas escolhendo o melhor vizinho a cada salto. Neste trabalho, considera-se as seguintes escolhas de melhor vizinho: (i) quando ocorre apenas um evento, é escolhido o nó que leva a um caminho mais curto (em saltos) até o nó *sink* (figura 3(a)), (ii) após a ocorrência do segundo ou dos eventos subsequentes, o melhor vizinho é o nó que leva a um caminho mais curto até algum nó que pertence a estrutura de roteamento já estabelecida (figura 3(c)). Esse procedimento tende a maximizar os pontos de agregação garantindo que eles ocorram o mais próximo possível dos eventos. A rota resultante é uma árvore que conecta os nós coordenadores ao nó *sink*.

No momento em que é realizado o estabelecimento da rota é também iniciada a fase de atualização da árvore de saltos, isso é feito a partir dos novos nós retransmissores que fazem parte da nova rota estabelecida. Esses nós enviam, através de inundação, a mensagem MCH (figura 3(b)) para a atualização de saltos. O algoritmo para a atualização de saltos segue o mesmo princípio do algoritmo para construção da árvore de saltos descrito na seção 3.1.



(a) Exemplo de árvore de roteamento para 1 evento (b) Atualização da árvore de saltos (c) Exemplo de árvore de roteamento para 2 eventos

Figura 3. Exemplo de estabelecimento de novas rotas e atualização da árvore de saltos

Algoritmo 3: Estabelecimento de Rota e Atualização da Árvore de Saltos

1 Nó líder v do novo evento envia uma mensagem MER para o seu $NextHop_v$
2 **repeat**
3 **se** $u = Nextop_v$ **então**
// u é o nó que recebeu a mensagem MER que foi enviada pelo nó v
4 $HopToTree_u \leftarrow 0$ // Nó u muda seu $HopToTree$ para 0
5 $Role_u \leftarrow Retransmissor$ // Nó u faz parte da nova rota construída
6 Nó u envia a mensagem MER para o seu $NextHop_u$
7 Nó u faz um *broadcast* da mensagem MCH com o valor de HopToTree = 1
// Os nós que receberem a mensagem MCH enviada pelo nó u, vão executar os comandos da linha 2 até a linha 13 do algoritmo 1
8 **fim se**
9 **until** *encontrar o nó sink ou um nó pertencente a estrutura de rotemaento já estabelecida.*
10 **repeat**
// $filhos_u$ é o número de descendentes de u
11 **se** $filhos_u > 1$ **então**
12 Agrega todos os dados e envia para o $nexthop_u$
13 **fim se**
14 **senão**
15 Envia os dados para $nexthop_u$;
16 **fim se**
17 **until** *O nó tem dados para transmitir/retransmitir*

O envio de dados no DAARP (algoritmo 3, linhas 10 à 17) faz uso das técnicas de agregação aplicando-as em dois contextos diferentes, dentro e fora do cluster. No interior do *cluster* a agregação é realizada pelos nós colaboradores quando as rotas se sobrepõem dentro do *cluster*. Além disso, o nó líder realiza a agregação dos dados e envia os resultados para o nó *sink*. Externamente ao cluster a agregação de dados é realizada pelos nós retransmissores quando as rotas de 2 ou mais eventos se sobrepõem durante o roteamento dos dados.

4. Avaliação de Desempenho

A solução proposta neste trabalho é comparada a três outros protocolos de roteamento. O objetivo principal dessa comparação é avaliar o desempenho do DAARP em relação ao InFRA, CNS e SPT usando quatro métricas importantes: (i) quantidade de pacotes de dados entregues, (ii) quantidade de mensagens de controle, (iii) eficiência (pacotes por dados processados), e (iv) custo da árvore de roteamento.

4.1. Cenário de Simulação e Métricas Utilizadas

A simulação realizada neste trabalho avalia a escalabilidade da nossa proposta em termos de número de nós ($n \in \{121, 256, 529, 1024\}$), número de vizinhos (raio de comunicação) ($r_c \in \{60, 70, 80, 90, 100\}$ metros), número de eventos ($ne \in \{1, 2, 3, 4, 5, 6\}$) e duração dos eventos ($dt_i \in \{1, 2, 3, 4, 5\}$ horas). O cenário padrão usado nas simulações é apresentado na tabela 1, quando nada for mencionado esses valores são usados. Para algumas simulações, algum parâmetro apresentado na tabela 1 será variado e isso é descrito no cenário avaliado. Cada simulação foi replicada 33 vezes. Em todos os resultados, as curvas representam os valores médios, enquanto que as barras de erros representam o intervalo de confiança de 95%. O primeiro evento acontece no tempo 1000 segundos e todos os outros eventos começam aleatoriamente no intervalo $[1000, 3000]$ segundos, onde esses eventos ocorrem em posições aleatórias. A. O simulador usado para executar as simulações foi o Sinalgo versão $v.0.75.3$ [Sinalgo 2008]. Para cada simulação que foi variado o número de nós, a dimensão do campo sensoriado é ajustada para manter a densidade de aproximadamente 21.7 nós vizinhos. Foi considerado também que a densidade da rede é $n\pi r_c^2/A$, sendo A a dimensão do campo sensoriado.

Frery et al. [Frery et al. 2008] descrevem um modelo de processo pontual espacial repulsivo que utilizamos para gerar a topologia de rede das simulações descritas nesse trabalho. Processos pontuais estocásticos são modelos que descrevem a localização de pontos no espaço e são úteis em diversas áreas em aplicações científicas (ver mais detalhes em [Baddeley 2006]). Processos repulsivos são aptos a descrever a deposição dos sensores numa área de interesse quando a localização dos sensores não é completamente controlada. Por exemplo, quando os sensores são lançados por um avião voando a baixa altitude. Esse modelo de deposição torna nossas simulações mais realistas.

Tabela 1. Parâmetros de simulação

Parâmetro	Valor
Nó sink	1 (canto superior esquerdo)
Número de nós sensores	1024
Alcance do rádio de comunicação ($metros$)	80
Número de eventos	3
Raio do evento ($metros$)	50
Duração do evento ($horas$)	3
simulation duration ($horas$)	7
Taxa de envio de dados ($pacotes/s$)	60
Área monitorada (m^2)	700×700

Para fornecer um limite inferior para transmissões de pacotes, foi utilizada uma função que recebe p pacotes de dados e envia apenas um pacote fundido (agregado) de tamanho fixo. Essa função é executada pelos pontos de agregação cada vez que esses nós enviarem um pacote. Qualquer outra função de agregação pode ser utilizada e tirar proveito das características do DAARP.

Os algoritmos avaliados usaram a estratégia de agregação *periodic simple aggregation* [Solis and Obraczka 2006]; nessa estratégia os nós agregadores transmitem periodicamente as informações recebidas e agregadas.

Foram utilizadas as seguintes métricas na avaliação dos algoritmos:

- **Quantidade de pacotes de dados entregues:** é o número de pacotes que atinge o nó *sink*. Essa métrica indica a qualidade da árvore de roteamento construída pelos algoritmos, quanto menor o número de pacotes entregue maior é a taxa de agregação da árvore construída pelo algoritmo;
- **Quantidade de mensagens de controle:** define o número de mensagens de controle transmitidas para criar e manter a estrutura de roteamento;
- **Eficiência (pacotes por dados processados):** é a relação do total de pacotes transmitidos (pacotes de dados e de controle) e a quantidade de pacote de dados recebidos pelo *sink*;
- **Custo da árvore de roteamento:** é o número de arestas que compõem a árvore de roteamento construída pelo algoritmo.

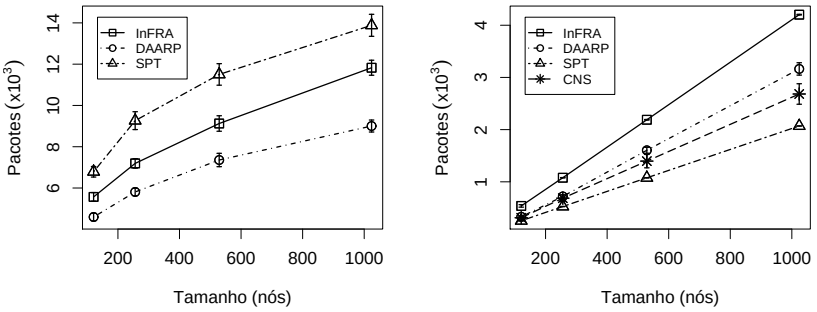
4.2. Resultados de Simulação

Em algumas avaliações o CNS não foi mostrado no gráfico devido seus valores estarem em uma, duas ou três ordens de grandeza maior que o DAARP, mas em todas as avaliações são mostradas comparações quantitativas do DAARP em relação ao CNS.

4.2.1. Variando o Tamanho da Rede (Número de nós sensores)

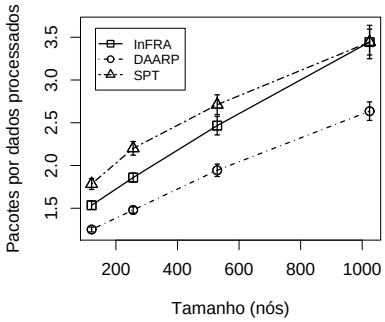
Nesse cenário variamos o parâmetro número de nós sensores da tabela 1, para avaliar o comportamento dos algoritmos para redes com $121, 256, 529$ e 1024 nós sensores. A figura 4 apresenta resultados quando o tamanho da rede é avaliado. A figura 4(a) mostra que o DAARP envia cerca de 77% de pacotes de dados enviados pelo INFRA e cerca de 65% dos pacotes de dados enviados pelo SPT, enquanto CNS envia muito mais pacotes (cerca de 300% a mais que o DAARP). Isso indica que o DAARP mantém a qualidade da árvore de roteamento quando o número de nós aumenta. A figura 4(b) mostra que o DAARP é mais escalável que o InFRA uma vez que necessita de menos mensagens de controle para construir a estrutura de roteamento (em média 30% a menos de mensagens de controle). O DAARP necessita em média 25% a mais de mensagens de controle em relação ao SPT, entretanto o SPT constrói árvores de roteamento (30% em média) pior que as árvores construída pelo DAARP (figura 4(d)). Note que o CNS necessita quase a mesma quantidade de mensagens de controle que o DAARP, mas o CNS apresenta árvores de roteamento de pior qualidade para a agregação de dados. A figura 4(c) mostra que o DAARP é (20% e 28%) mais eficiente do que o InFRA e SPT, respectivamente. Em relação ao CNS o DAARP é 50% em média mais eficiente. Isto ocorre, devido ao fato do DAARP necessitar de menos mensagens de controle para construir a árvore de

roteamento em relação ao InFRA e a árvore de roteamento construída pelo DAARP tem a melhor qualidade de agregação de dados em relação ao InFRA, CNS e SPT, como mostra a figura 4(d).

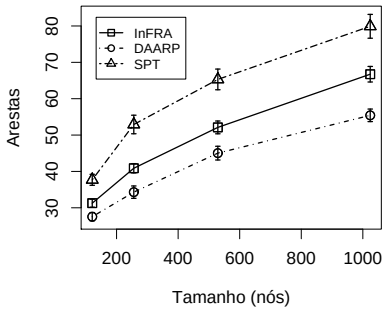


(a) Pacotes de dados

(b) Pacotes de controle



(c) Eficiência

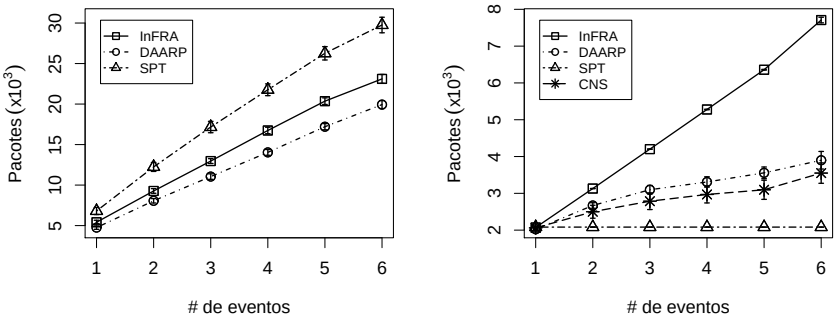


(d) Custo da árvore

Figura 4. Tamanho da Rede

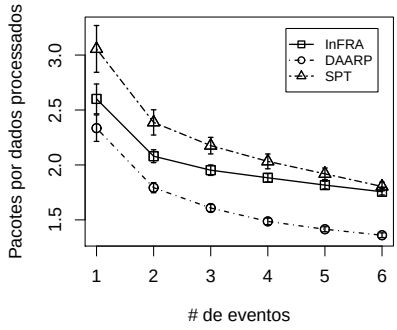
4.2.2. Variando o Número de Eventos

Nesse cenário variamos o parâmetro número de eventos da tabela 1, para avaliar o comportamento dos algoritmos para redes com $1, 2, 3, 4, 5$ e 6 eventos ocorrendo simultaneamente. Os resultados estão apresentados na figura 5. Como pode ser observado na figura 5(a), o DAARP envia menos pacote de dados que o InFRA e SPT. Por exemplo, o DAARP envia aproximadamente 81%, 67% e 57% do número de pacotes de dados enviados pelo InFRA, SPT e CNS, respectivamente. Isso indica que variando o número de eventos, o DAARP consegue construir árvores de roteamento com maior taxa de agregação de dados em relação ao InFRA, SPT e CNS. A figura 5(b) mostra que o DAARP necessita de apenas 50% das mensagens de controle usadas pelo InFRA na ocorrência de 6 eventos e, em média, apenas 29% das mensagens de controle usadas pelo InFRA para a construção da estrutura de roteamento. Consequentemente, para mais de um evento o DAARP é mais eficiente do que o InFRA e SPT como mostra a figura 5(c). O custo da árvore de roteamento construída pelo DAARP é 10% menor do que o InFRA e 30% menor do que o SPT como pode ser observado na figura 5(d).

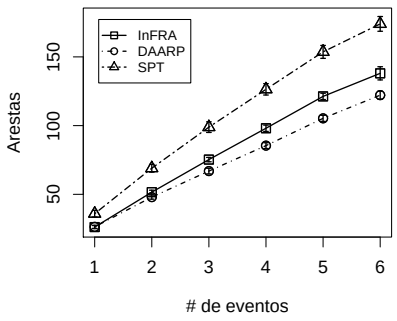


(a) Pacotes de dados

(b) Pacotes de controle



(c) Eficiência

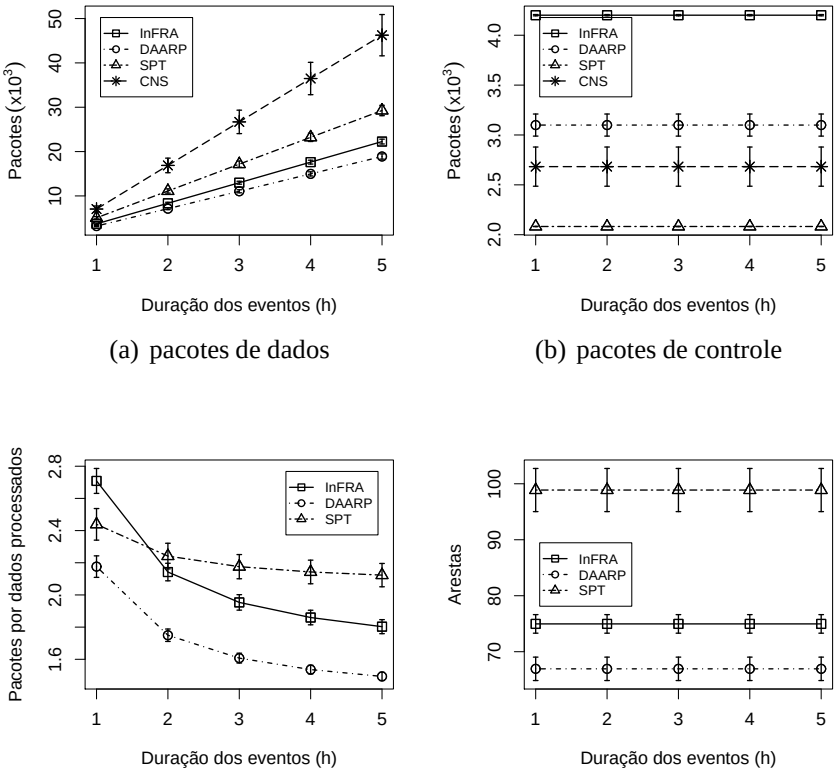


(d) Custo da árvore

Figura 5. Número de Eventos

4.2.3. Duração do Evento

Nesse cenário variamos o parâmetro duração do evento da tabela 1, para avaliar o comportamento dos algoritmos para redes com eventos de duração de $1, 2, 3, 4$ e 5 horas. Os resultados estão apresentados na figura 6. A figura 6(a) mostra que o DARRP envia menos pacote de dados do que os outros algoritmos. Em particular, o DAARP envia aproximadamente 84%, 64% e 42% de pacotes de dados enviados pelo InFRA, SPT e CNS, respectivamente. Isso indica que variando o tempo da duração dos eventos, o DAARP consegue uma taxa de agregação de dados maior que o InFRA, SPT e CNS. A figura 6(b) mostra que o DAARP necessita de menos mensagens de controle para criar a estrutura de roteamento em relação ao InFRA e mais mensagens de controle do que o SPT e CNS. O DAARP necessita apenas de 72% das mensagens de controle exigida pelo InFRA para construir a infra-estrutura de roteamento. Note que o DAARP necessita de 33% a mais de mensagens de controle do que o SPT, entretanto o SPT não tem como objetivo construir uma infra-estrutura de roteamento ciente da agregação de dados. A Figura 6(c) mostra que o DAARP é mais eficiente do que o InFRA e SPT. Note que o DAARP supera todos os outros algoritmos mesmo em cenários de eventos de curto prazo, enquanto o InFRA supera o SPT somente em cenários onde a duração dos eventos é de prazo mais longo (mais que 2 horas).



(a) pacotes de dados (b) pacotes de controle

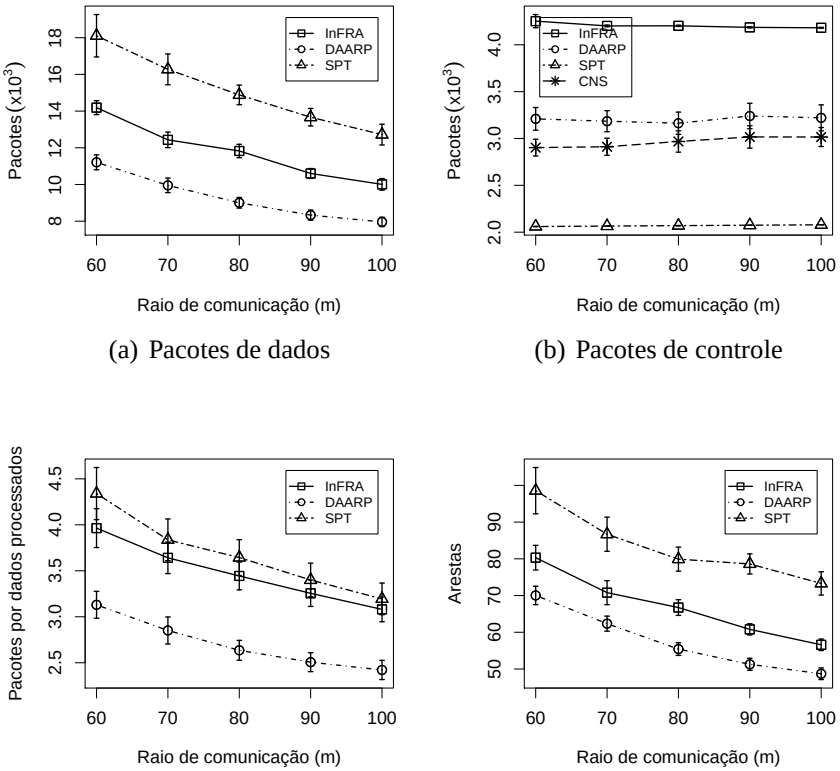
(c) Eficiência (d) Custo da árvore

Figura 6. Duração do Evento

4.2.4. Variando o Alcance do Rádio de Comunicação

Nesse cenário variamos o parâmetro o alcance do rádio de comunicação da tabela 1, para avaliar o comportamento dos algoritmos para cenários onde o raio de comunicação varia entr $60, 70, 80, 90$ e 100 metros. Os resultados estão apresentados na figura 7. Esta simulação tem por objetivo avaliar a escalabilidade de nossa proposta em relação ao número de vizinhos. O DAARP necessita de um pouco mais de mensagens de controle para construir a árvore de roteamento em relação ao SPT e CNS, e menos mensagens de controle em relação ao InFRA (figura 7(b)). Mais uma vez o DAARP consegue construir a árvore de roteamento de menor custo em relação ao SPT e InFRA (figura 7(d)). Além disso, o DAARP possui um baixo custo de mensagens de controle, alcançando a melhor eficiência em relação ao CNS, SPT e InFRA, como mostra a figura 7(c).

Os resultados apresentados nesta seção mostram que o DAARP é mais escalável que o InFRA, o SPT e o CNS em todos os cenários que nós avaliamos nesse trabalho em relação ao tamanho da rede, ao número de eventos, à duração dos eventos e ao raio de comunicação.



(a) Pacotes de dados (b) Pacotes de controle

(c) Eficiência (d) Custo da árvore

Figura 7. Alcance do rádio de comunicação

5. Conclusões e Trabalhos Futuros

Este artigo apresenta o DAARP, um novo algoritmo de roteamento ciente da agregação de dados para redes de sensores sem fio. O DAARP foi extensivamente comparado aos algoritmos InFRA, SPT e CNS com relação à escalabilidade, custo de comunicação, eficiência de entrega e taxa de agregação. O DAARP mostrou ser mais eficiente em todas as avaliações realizadas, o que o coloca como potencial solução para aplicações em que a qualidade da entrega de dados com redução de gasto de energia sejam primordiais.

Como trabalhos futuros serão considerados a correlação espacial e temporal dos dados na agregação e também a construção de árvore de roteamento a partir de restrições da aplicação. Serão realizados também serviços de balanceamento de energia e tolerância a falhas no protocolo, adição e avaliação de nova estratégia de controle do tempo de espera dos nós agregadores baseada em dois critérios (distância média dos coordenadores dos eventos e a correlação espacial e semântica dos eventos), além de avaliação da latência, pois apesar do processo de agregação reduzir o número de mensagens na rede, ele gera atrasos adicionais. Finalmente, serão analisados o quão distante o DAARP se encontra do caso ótimo, no pior caso e no caso médio, uma vez que a agregação ótima é atingida quando as informações são encaminhadas por meio da árvore mínima de Steiner.

Agradecimentos: Os autores agradecem ao CNPq e à CAPES pelo financiamento dos bolsistas de doutorado, Sr. Leandro A. Villas e Sr. Heitor Soares Ramos Filho.

Referências

Baddeley, A. (2006). Spatial point processes and their application. In Weil, W., editor, *Stochastic Geometry*, volume 1892 of *Lecture Notes in Mathematics*, pages 1–75. Springer, Berlin.

Boukerche, A., Araujo, R. B., Silva, F. H. S., and Villas, L. (2007a). Wireless sensor and actor networks context interpretation for the emergency preparedness class of applications. *Computer Communication*, 30(13):2593–2602.

Boukerche, A., Araujo, R. B., and Villas, L. (2007b). Optimal route selection for highly dynamic wireless sensor and actor networks environment. In *MSWiM '07: Proceedings of the 10th ACM Symposium on Modeling, analysis, and simulation of wireless and mobile systems*, pages 21–27, New York, NY, USA. ACM.

Frery, A. C., Ramos, H., Alencar-Neto, J., and Nakamura, E. (2008). Error estimation in wireless sensor networks. In *SAC'08: Proceedings of ACM Symposium on Applied Computing*, volume 3, pages 1927–1932.

Hougardy, S. and Prömel, H. J. (1999). A 1.598 approximation algorithm for the steiner problem in graphs. In *SODA '99: Proceedings of the 10th annual ACM-SIAM symposium on Discrete algorithms*, pages 448–453, Philadelphia, PA, USA. Society for Industrial and Applied Mathematics.

Krishnamachari, B., Estrin, D., and Wicker, S. B. (2002). The impact of data aggregation in wireless sensor networks. In *ICDCSW '02: Proceedings of the 22nd International Conference on Distributed Computing Systems*, pages 575–578, Washington, DC, USA. IEEE Computer Society.

Nakamura, E. F., de Oliveira, H. A. B. F., Pontello, L. F., and Loureiro, A. A. F. (2006). On demand role assignment for event-detection in sensor networks. In *ISCC '06: Proceedings of the 11th IEEE Symposium on Computers and Communications*, pages 941–947, Washington, DC, USA. IEEE Computer Society.

Nakamura, E. F., Loureiro, A. A. F., and Frery, A. C. (2007). Information fusion for wireless sensor networks: Methods, models, and classifications. *ACM Computing Surveys*, 39(3):9–1/9–55.

Robins, G. and Zelikovsky, A. (2000). Improved steiner tree approximation in graphs. In *SODA '00: Proceedings of the 11th annual ACM-SIAM symposium on Discrete algorithms*, pages 770–779, Philadelphia, PA, USA. Society for Industrial and Applied Mathematics.

Sinalgo (2008). Simulator for network algorithms. Distributed Computing Group - ETH-Zurich, last visited in October, 2008.

Solis, I. and Obraczka, K. (2006). In-network aggregation trade-offs for data collection in wireless sensor networks. *International Journal of Sensor Networks (IJSNET)*, 1(3/4):200–212.

Younis, O., Krunz, M., and Ramasubramanina, S. (2006). Node clustering in wireless sensor networks: Recent developments and deployment challenges. *IEEE Network*, 20(3):20–25.