

**SOLUÇÕES CIENTES DE AGREGAÇÃO DE  
DADOS, DA CORRELAÇÃO ESPAÇO-TEMPORAL  
E CONSUMO DE ENERGIA PARA REALIZAR  
COLETA DE DADOS EM REDES DE SENSORES  
SEM FIO**



LEANDRO APARECIDO VILLAS

SOLUÇÕES CIENTES DE AGREGAÇÃO DE  
DADOS, DA CORRELAÇÃO ESPAÇO-TEMPORAL  
E CONSUMO DE ENERGIA PARA REALIZAR  
COLETA DE DADOS EM REDES DE SENSORES  
SEM FIO

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Doutor em Ciência da Computação.

ORIENTADOR: ANTONIO ALFREDO FERREIRA LOUREIRO  
CO-ORIENTADORA: REGINA BORGES DE ARAUJO

Belo Horizonte

Março de 2012



LEANDRO APARECIDO VILLAS

**DATA AGGREGATION, SPATIO-TEMPORAL  
CORRELATION AND ENERGY-AWARE  
SOLUTIONS TO PERFORM DATA COLLECTION  
IN WIRELESS SENSOR NETWORKS**

Thesis presented to the Graduate Program  
in Computer Science of the Federal Univer-  
sity of Minas Gerais in partial fulfillment of  
the requirements for the degree of Doctor  
in Computer Science.

ADVISOR: ANTONIO ALFREDO FERREIRA LOUREIRO  
CO-ADVISOR: REGINA BORGES DE ARAUJO

Belo Horizonte

March 2012





UNIVERSIDADE FEDERAL DE MINAS GERAIS  
INSTITUTO DE CIÊNCIAS EXATAS  
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

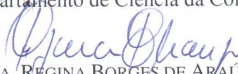
## FOLHA DE APROVAÇÃO

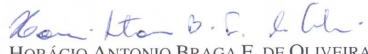
Soluções cientes de agregação de dados, da correlação espaço-temporal e consumo de energia para realizar coleta de dados em redes de sensores sem fio

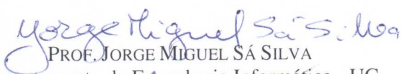
### LEANDRO APARECIDO VILLAS

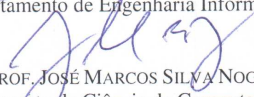
Tese defendida e aprovada pela banca examinadora constituída pelos Senhores:

  
PROF. ANTONIO ALFREDO FERREIRA LOUREIRO - Orientador  
Departamento de Ciência da Computação - UFMG

  
PROFA. REGINA BORGES DE ARAÚJO - Co-orientadora  
Departamento de Computação - UFSCAR

  
PROF. HORÁCIO ANTONIO BRAGA F. DE OLIVEIRA  
Departamento de Ciência da Computação - UFAM

  
PROF. JORGE MIGUEL SÁ SILVA  
Departamento de Engenharia Informática - UC

  
PROF. JOSÉ MARCOS SILVA NOGUEIRA  
Departamento de Ciência da Computação - UFMG

  
PROF. LUIZ FILIPE MENEZES VIEIRA  
Departamento de Ciência da Computação - UFMG

Belo Horizonte, 14 de março de 2012.





*Dedico este trabalho:*

*Primeiramente aos meus pais, Antônio Villas Martins e Terezinha Joana Gonçalves Villas, pelo amor, incentivo e dedicação, sempre acreditando no meu sucesso. Certamente orgulhosos por mais um importante passo na minha vida;*

*À minha irmã, Daiane Villas, que sempre torceu para que tudo desse certo;*

*À Verônica, minha namorada e possivelmente futura esposa, pela paciência nos momentos mais difíceis;*

*Ao meu querido sobrinho Geovane Villas, ao qual tenho um amor imenso;*

*Ao Aparecido Gonzales Castilho, meu primo, mesmo não estando presente entre nós, foi um exemplo de vida para mim e me ensinou que nos estudos eu conseguiria minhas realizações.*



# Agradecimentos

A Deus, acima de tudo e todos, pela oportunidade de aperfeiçoamento tanto intelectual quanto moral. Pela presença constante em cada momento de minha vida, pela família e pelos amigos;

A minha família (Antonio, Terezinha, Daiane e Geovane), pessoas sem as quais eu não seria o que sou hoje. Mesmo distantes fisicamente, permaneceram próximos em pensamento através de suas orações, conversas e conselhos;

Ao meu orientador, Prof. Antonio Alfredo Ferreira Loureiro, pela ajuda, incentivo, esclarecimento e, acima de tudo, exemplo. A minha co-orientadora Regina Borges de Araújo pela motivação, encorajamento, amizade e direcionamento. Ao meu supervisor, Prof. Azzedine Boukerche pela oportunidade e direcionamento durante o estágio de sanduíche realizado na University of Ottawa. Ao Prof. Horácio A. B. Fernandes de Oliveira pelas discussões e colaborações e ao meu amigo Daniel L. Guidoni pelas discussões e colaborações em vários trabalhos;

Aos amigos do DCC-UFMG, Alyson, Celso, César Soares, Daniel Galinkin, Daniel Guidoni, Eduardo Mucelli, Felipe, Fernanda, Fernando, Flávio, Guilherme, Heitor, Izabela, John Holiver, Laura, Letícia, Marcelo, Max, Pedro, Rafael Colares, Rafael Santin, Thiago Pedpano, Tiago Cunha e Zilton pela amizade, companhia e ajuda diária;

Ao pessoal do laboratório *Ubiquitous and Wireless Sensor Networks Lab* pelo ambiente descontraído e, ao mesmo tempo, profissional que encontrei no laboratório;

Aos meus amigos do *Paradise Laboratory*, Aissa, Cristiano, Daniel, Heitor, Richard e Robson pela companhia, amizade e apoio desde minha chegada em Ottawa;

A todos os meus familiares pela confiança irrestrita;

Aos meus amigos de infância, que sempre estiveram ao meu lado, mesmo quando distantes.



# Resumo

Este trabalho oferece uma discussão geral sobre o tema de agregação de dados e exploração da correlação espaço-temporal dos dados em redes de sensores sem fio (RSSFs) que permite: (i) a identificação de problemas em aberto e (ii) o entendimento dos requisitos e implicações do uso de agregação de dados em RSSFs, além da exploração da correlação espaço-temporal dos dados.

Esta discussão é feita através de um levantamento bibliográfico do estado-da-arte envolvendo agregação e correlação espaço-temporal de dados em RSSFs. Como resultado da análise de arquiteturas, modelos e métodos de agregação e correlação espaço-temporal de dados identificados neste levantamento bibliográfico, propomos quatro soluções diferentes para o problema de agregação e exploração da correlação espaço-temporal de dados considerando diferentes cenários em RSSFs: os algoritmos DAARP, DDAARP, DST e EAST. Os algoritmos propostos reduzem o número de mensagens necessárias para criar uma árvore de roteamento, maximizam o número de rotas sobrepostas, selecionam as rotas com maior taxa de agregação e realizam transmissões confiáveis de dados agregados.

As soluções propostas foram amplamente comparadas com outras soluções da literatura em relação aos custos de comunicação, eficiência de entrega, taxa de agregação e taxa de entrega de dados agregados. Os resultados mostram que as soluções propostas podem ser uma boa alternativa para agregar dados e explorar a correlação espaço-temporal dos dados durante o roteamento. Diversos experimentos são mostrados para avaliar o desempenho dos algoritmos propostos.



# Resumo Estendido

O documento desta tese está redigido em inglês com o título “*Data Aggregation, Spatio-Temporal Correlation and Energy-Aware Solutions to perform Data Collection in Wireless Sensor Networks*”. Para atender às normas da Universidade Federal de Minas Gerais, este resumo em português faz um resumo estendido de cada capítulo desta tese.

## Capítulo 1 – Introdução

Os recentes desenvolvimentos nas áreas de comunicação sem fio e sensores multifuncionais com capacidade de comunicação e processamento impulsionaram o crescimento das redes de sensores sem fio (RSSFs). 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. Um nó sensor, por si só, apresenta uma capacidade limitada de detecção de sensoriamento de uma dada grandeza, mas a capacidade global de detecção pode ser aumentada drasticamente quando os nós são combinados formando uma rede de sensores sem fio. Logo, nós sensores em uma RSSF podem monitorar cooperativamente uma determinada área de interesse. Por exemplo, se ocorrer um vazamento de gás em uma sala repleta de botijões de gás e existir apenas um nó sensor nessa sala, será possível apenas dizer se há ou não vazamento de gás. Por outro lado, se for utilizada uma RSSF adequadamente projetada, será possível não só detectar o vazamento, mas indicar onde o vazamento iniciou e como ele evoluiu. Um monitoramento dessa forma pode salvar vidas e patrimônio, além de diminuir custos de seguros.

Os nós sensores são dispositivos tipicamente com restrições de energia. O consumo de energia é geralmente associado à quantidade de dados transmitidos na rede, pois a comunicação é a atividade que tende a demandar uma maior quantidade de energia. Uma solução simples para esse problema seria a reposição da bateria dos nós sensores. Entretanto, essa técnica é inviável devido à grande quantidade de nós na rede ou porque

os nós sensores podem ser inacessíveis em algumas aplicações, como monitoramento de vulcões ou do espaço. Dessa forma, algoritmos e protocolos projetados para uma RSSF devem considerar o consumo de energia em sua concepção. Além disso, os nós sensores podem coletar uma grande quantidade de dados que precisam ser processados e encaminhadas, muitas vezes usando comunicação *multihop*, em direção a um nó *sink*, o qual funciona como um *gateway* para uma estação de monitoramento. Nesse cenário, o roteamento desempenha um papel muito importante no processo de coleta de dados.

Para realizar a coleta de dados de forma mais eficiente e eficaz com um uso mínimo de recursos limitados, nós sensores devem ser configurados para reportar dados de forma inteligente tomando decisões locais. Para isso, a agregação de dados e a exploração da correlação espaço-temporal de dados são técnicas eficazes de economia de energia em RSSFs. Devido à redundância dos dados brutos recolhidos pelos nós sensores, a agregação de dados e a correlação espaço-temporal de dados muitas vezes podem ser usadas para diminuir o custo de comunicação, eliminando a redundância de dados e reportando apenas informações agregadas.

A agregação de dados tem sido utilizada em RSSFs com dois propósitos: (i) tirar proveito da redundância e melhorar a precisão dos dados; e (ii) reduzir o tráfego de dados e economizar energia. No entanto, as propostas atuais têm um custo alto para criar estruturas de roteamento cientes de agregação de dados e muitas delas não consideram a correlação espaço-temporal de dados. Além disso, a maioria das propostas não lida com falhas nos nós e interrupções nas comunicações, o que provoca perda de dados e não garante a entrega dos dados coletados.

A principal contribuição desta tese é o desenvolvimento de quatro diferentes soluções cientes de agregação de dados, da correlação espaço-temporal e consumo de energia para coleta de dados em RSSFs, que nos referimos como DAARP, DDAARP, DST e EAST, quais serão apresentados, respectivamente, nos capítulos 3, 4, 5 e 6.

## Capítulo 2 – Fundamentação Teórica

### Redes de Sensores sem Fio

Uma Rede de Sensores sem fio (RSSF) pode ser definida como uma rede cooperativa de nós sensores sem fio, operados tipicamente por bateria, cujo principal objetivo é duplo: monitorar o ambiente e transmitir os dados recolhidos para um nó sorvedouro (*sink*) usando normalmente comunicação *multihop*. Este nó sorvedouro será responsável por processar todos os dados recebidos dos nós fontes e reportá-los para uma estação de monitoramento (veja figura 2.1). Tipicamente, uma RSSF é composta por um grande



número de nós sensores que são colocados dentro ou muito próximos ao fenômeno a ser analisado. Geralmente, cada nó sensor é equipado com vários tipos de sensores como, por exemplo, temperatura, pressão, sísmico, acústico, radiação e infravermelho. Esses nós sensores são construídos para serem baratos e normalmente possuem limitações computacionais, memória, comunicação e energia.

As RSSFs possibilitam a coleta de informações necessárias em ambientes onde o uso de fios ou cabeamento não seja possível ou viável. Elas podem estar inseridas na estrutura de um prédio, ponte, no interior de máquinas, tubulações, dentro de casas, florestas, áreas de desastre, plantações, vulcões, campo de batalha e até mesmo dentro do corpo humano como, por exemplo, a retina. O baixo custo dos nós sensores e o potencial dessa tecnologia justificam a sua utilização em diversas áreas, tais como:

- **Saúde:** controle de doenças contagiosas; interface para deficientes; monitoramento de pacientes; diagnóstico de distúrbios; administração de drogas em hospitais, monitoramento e localização de pacientes e médicos em hospitais.
- **Aplicações Militares:** monitoramento de tropas; reconhecimento de terreno; detecção de alvos e de ataques biológicos, químicos ou nucleares.
- **Meio-ambiente:** rastreamento do movimento dos pássaros, pequenos animais; monitoramento de condições ambientais que afetam colheitas e plantio (por exemplo, combate à geada, detecção de componentes químicos ou biológicos, irrigação); mapeamento da bio-complexidade ambiental, estudo da poluição e muitas outras.
- **Monitoramento de estrutura/equipamentos:** monitoramento e identificação de falhas em estruturas (pontes, prédios, etc); monitoramento da fadiga de máquinas e equipamentos (motores, dutos de gás, etc); diagnósticos de máquinas.
- **Aplicações comerciais:** automação de vendas e processo industriais; manutenção de inventário, monitoramento de qualidade de produtos; detecção e vigilância de veículos e estabelecimentos.

A tendência é a produção dos nós sensores em larga escala, barateando o seu custo e o investimento no desenvolvimento tecnológico levando a novas melhorias, como aumento de processamento e armazenamento e redução do tamanho dos nós sensores. Portanto, novas aplicações podem surgir aumentando a abrangência de uso das RSSFs.

A posição de cada nó sensor não precisa ser necessariamente pré-determinada, o que possibilita uma disposição aleatória em locais de difícil acesso, como em áreas de

desastres e incêndios. Por outro lado, isto significa que os algoritmos e protocolos para redes de sensores devem possuir a característica de auto-organização dos nós.

## Agregação de Dados no Roteamento

Agregação de dados durante o roteamento em redes de sensores sem fio envolve diferentes formas de transmitir pacotes de dados a fim de combinar dados provenientes de fontes diferentes, mas destinado a um nó específico chamado *sink*. Um componente-chave para a agregação de dados em RSSFs é um protocolo de roteamento ciente de agregação de dados bem concebido, que determina onde a agregação deve ser realizada. Agregação de dados requer um paradigma de encaminhamento diferente do roteamento clássico. Protocolos de roteamento clássico tipicamente usam os caminhos mais curtos “em relação a alguma métrica específica” para transmitir dados ao *sink*. Em protocolos de roteamento ciente de agregação de dados, os nós devem encaminhar os pacotes de dados com base no conteúdo do pacote e escolher o próximo *hop* que maximiza a sobreposição de rotas para promover a agregação de dados na rede durante o roteamento.

Para realizar a agregação de dados na rede é necessária alguma forma de sincronização entre os nós. Tipicamente os nós não enviam dados, logo que seja possível. A espera de informações provenientes de nós vizinhos pode levar a melhores oportunidades de agregação de dados e, conseqüentemente, melhor desempenho. As principais estratégias de temporização propostas na literatura são resumidas a seguir:

- *Periodic simple aggregation* exige que cada nó espere por um período de tempo pré-definido, para agregar todos os pacotes de dados recebidos durante esse tempo pré-definido e, em seguida, envia um pacote de dados com o resultado da agregação de todos os pacotes de dados recebidos;
- *Periodic per-hop aggregation* é bastante semelhante à abordagem anterior. A única diferença é que o pacote com os dados agregados é transmitido logo que o nó recebe um pacote de dados de todos os seus filhos. Isto requer que cada nó conheça os seus filhos. Além disso, um tempo limite é usado em caso de um pacote de dados de algum filho ser perdido durante a transmissão.
- *Periodic per-hop adjusted aggregation* ajusta o tempo de espera de um nó, dependendo da posição do nó na estrutura de roteamento.

É importante observar que a escolha da estratégia de tempo afeta fortemente a taxa de agregação em rede e a latência para relatar os dados coletados. Se o tempo de espera no ponto de agregação é alto, a latência e a taxa de agregação é maior. Se

o tempo de espera em pontos de agregação é baixo, a taxa de agregação e a latência é menor.

## Correlação Espaço-Temporal de Dados

Podemos encontrar atualmente na literatura três categorias principais de protocolos cientes de correlação de dados: (i) correlação espacial; (ii) correlação temporal e (iii) correlação espaço-temporal. A seguir, apresentamos os benefícios da exploração da correlação espacial/temporal de dados em RSSFs:

1. *Correlação Espacial*: nós espacialmente próximos tendem a sensoriar valores semelhantes. No entanto, essa proximidade depende dos requisitos da aplicação e características do evento. Algumas aplicações são mais críticas e são menos tolerantes a discrepâncias nos valores sensoriados sobre o fenômeno observado, exigindo que nós próximos reportam os dados sensoriados (região de correlação é menor). Por outro lado, outras aplicações podem ser mais tolerantes a discrepâncias nos valores sensoriados, não exigindo que nós próximos reportam os dados sensoriados (região de correlação é maior).

**Região de correlação**: *em uma região de correlação, os valores sensoriados pelos nós sensores são considerados semelhantes para a aplicação e, portanto, uma única leitura dentro dessa região é o suficiente para representá-la. O tamanho da região de correlação varia de aplicação para aplicação e de evento para evento. Assim, o tamanho da região de correlação está diretamente relacionado à aplicação.*

2. *Correlação Temporal*: tipicamente, a leitura feita pelos sensores no ambiente é periódica. Consequentemente, os dados sensoriados constituem uma série temporal. Devido à natureza do fenômeno físico, há uma correlação temporal significativa entre cada observação consecutiva de um nó sensor e os dados recolhidos são geralmente semelhantes em um curto período de tempo. Assim, nesses casos, os nós sensores não precisam transmitir suas leituras se a leitura atual estiver dentro de um limiar aceitável em relação à última leitura reportada.

**Correlação temporal**: *cada nó fonte mantém a última leitura reportada ( $R_{old}$ ). Quando a leitura atual ( $R_{new}$ ) estiver disponível,  $R_{new}$  é comparada com  $R_{old}$ .  $R_{new}$  de um nó fonte é reportada se um dado limiar é maior que a tolerância na coerência temporal ( $tct$ ), isto é,  $\left(\frac{|R_{new}-R_{old}|}{R_{old}}\right) \times 100 > tct$ , onde  $tct$  é a porcentagem de tolerância na coerência temporal. Caso contrário o valor  $R_{new}$  é suprimido.*

3. *Correlação Espaço-Temporal*: acontece quando a natureza dos dados coletados apresenta tanto a correlação espacial quanto a temporal, ou seja, nós espacialmente próximos tendem a sensoriar valores semelhantes e os dados recolhidos são geralmente semelhantes em um curto período de tempo. Neste caso, as soluções que utilizam ambas as correlações podem tirar proveito da natureza do evento detectado e reduzir o número de dados reportados.

Na literatura existente de algoritmos que exploram a correlação espacial e/ou temporal dos dados, a maioria das abordagens propostas não considera o nível de energia dos nós na seleção dos nós representativos e as características do evento durante a coleta de dados para melhor escolher os nós representantes de cada região de correlação. As abordagens que consideram o nível de energia dos nós apresentam um alto custo de controle e geralmente resultam em altos atrasos e dados desatualizados são encaminhados para o nó *sink*. No entanto, elas não exploram a correlação espaço-temporal eficientemente.

## Capítulo 3 – DAARP: Um Protocolo de Roteamento Ciente de Agregação de Dados para RSSFs

É um novo protocolo de roteamento ciente de agregação de dados para RSSFs. A principal motivação para projetar um novo algoritmo de roteamento ciente de agregação de dados é que as soluções da literatura apresentam um alto custo para criar estruturas de roteamento cientes de agregação de dados. O algoritmo DAARP constrói uma árvore de roteamento com os caminhos mais curtos (em saltos) que conecta todos os nós fontes ao *sink* enquanto maximiza a agregação de dados, cuja principal contribuição é maximizar a agregação de dados ao longo da rota de comunicação, de uma forma mais confiável, através de um mecanismo de encaminhamento tolerante a falhas. Resultados de simulações (apresentados na seção 3.6) revelam que o DAARP tem alguns aspectos chave exigidos pela agregação de dados em RSSFs como um número reduzido de mensagens para a criação de uma estrutura de roteamento, maximiza o número de rotas sobrepostas, alta taxa de agregação e transmissão e agregação de dados confiáveis.

## Capítulo 4 – DDAARP: Um Protocolo de Roteamento Dinâmico e Ciente de Agregação de Dados para RSSFs

É um novo protocolo de roteamento ciente de agregação de dados dinâmico para RSSFs, que usa o nó *sink* para o processamento e configuração das rotas cientes de agregação de dados. A principal motivação para projetar uma abordagem dinâmica, que cria estruturas de roteamento dinâmicas cientes de agregação de dados, é que a qualidade das estruturas de roteamento criadas pelo DAARP e também pela maioria dos algoritmos na literatura depende da ordem de ocorrência dos eventos. Assim, uma vez criada essa estrutura, as rotas são mantidas estáticas durante a ocorrência do evento. A principal contribuição do DDAARP é que as rotas criadas não dependem da ordem de ocorrência dos eventos e não são mantidas fixas durante a ocorrência de eventos. Resultados de simulações (apresentados na seção 4.6) revelam que o DDAARP apresenta baixo custo em termos de pacotes de controle, melhora a qualidade da estrutura de roteamento e maximiza a agregação de dados ao longo da rota de comunicação de uma forma mais confiável, através de um mecanismo de roteamento tolerante a falhas.

## Capítulo 5 – DST: Um Protocolo de Roteamento Escalável, Dinâmico e Ciente de Agregação de Dados para RSSFs

É um novo protocolo de roteamento ciente de agregação de dados que leva a uma solução escalável, dinâmica e apresenta baixo custo para criar estruturas de roteamento. Apesar do DDAARP ter mostrado bons resultados, ele sofre com problemas de escalabilidade e torna-se inviável para redes de larga escala. Além disso, o nó *sink* precisa de conhecimento global da rede. DST é uma solução eficiente de agregação de dados que permite o roteamento escalável e dinâmico em RSSFs, que constrói estruturas de roteamento com as rotas mais curtas (distância Euclidiana) que conecta todos os nós fontes ao nó *sink* maximizando a agregação de dados e reduzindo a distância para conectar cada nó fonte ao *sink*. Resultados de simulações (apresentados na seção 5.6) revelam que o DST apresenta baixo custo em termos de pacotes de controle, maximiza os pontos de agregação e melhora a qualidade da estrutura de roteamento oferecendo rotas dinâmicas.

## Capítulo 6 – EAST: Um Protocolo de Roteamento Eficiente para Coleta de Dados Ciente da Correlação Espaço-Temporal para RSSFs

É um novo algoritmo ciente de energia para o encaminhamento de dados em RSSFs que aproveita os mecanismos de correlação espacial e temporal para economizar energia e manter o encaminhamento de dados precisos em tempo oportuno para o nó *sink*. A principal motivação para a concepção do EAST é que a maioria dos algoritmos cientes de correlação espacial e/ou temporal não considera o consumo de energia durante a coleta de dados para melhor escolher os nós representativos. Além disso, essas soluções possuem um grande número de mensagens de controle e não explora de forma eficiente a correlação espaço-temporal dos dados, nem a sua dinamicidade. A principal contribuição é um mecanismo de correlação espaço-temporal de dados em que os nós que detectaram o mesmo evento são dinamicamente agrupados em regiões correlacionadas e um nó representante é selecionado em cada região de correlação para observar o fenômeno. Resultados de simulações (apresentados na seção 6.4) mostram claramente que, usando ambas as correlações espacial e temporal, as informações sobre o evento podem ser sentidas com alta precisão e ainda economizar a energia residual dos nós.

## Capítulo 7 – Conclusões

Esta tese estudou a importância de realizar agregação de dados e explorar a correlação espaço-temporal dos dados durante o roteamento. Devido à impossibilidade de se ter uma única solução para um determinado problema em RSSFs, nesta tese são propostos quatro algoritmos diferentes para a agregação de dados e exploração da correlação espaço-temporal dos dados durante o roteamento.

# Abstract

This work provides a general discussion for data aggregation that exploits spatio-temporal data correlation in wireless sensor networks (WSNs), allowing us to identify open issues and understand the requirements and the implications regarding data aggregation, and spatio-temporal data correlation in WSNs.

In this discussion, we survey the state-of-the-art of data aggregation and spatio-temporal data correlation in WSNs. By assessing the architectures, models, and methods of data aggregation and spatio-temporal data correlation identified in the survey, we propose four different solutions for the data aggregation and spatio-temporal data correlation that are suitable for different scenarios in WSNs. The proposed solutions are called DAARP, DDAARP, DST and EAST. The proposed algorithms reduce the number of message necessary to set up a routing tree, maximize the number of overlapping routes, select routes with the highest aggregation rate, and perform reliable data aggregation transmission.

The proposed solutions have been extensively compared with other solutions in the literature and the results show that the proposed solutions may be potential alternatives to perform data aggregation and spatio-temporal data correlation during the routing process. We also present an extensive set of experiments to evaluate the performance of our algorithms. Our results indicate that our proposed solutions are suitable for implementation in WSNs.

**Keywords:** WSNs, routing algorithms, data aggregation, spatio-temporal correlation.





# List of Figures

2.1	Data routing in WSNs [Oliveira et al., 2009]. . . . .	8
2.2	Data Aggregation Aware Routing . . . . .	9
2.3	Energy consumption of nodes during the data collection when using (a) a classical approach; and when using (b) a spatial correlation based approach. . . . .	17
2.4	(a) The time series and (b) The piecewise linear presentation . . . . .	19
3.1	Example of establishing new routes and updating the hop tree . . . . .	27
3.2	Example of path repair . . . . .	29
3.3	Number of Steiner nodes in the routing tree built by the DAARP, InFRA, MST, and SPT algorithms . . . . .	34
3.4	Impact of the Network Size . . . . .	36
3.5	Impact of the Number of Events . . . . .	37
3.6	Impact of the Event Duration . . . . .	38
3.7	Impact of Communication Failures . . . . .	39
4.1	Overhead . . . . .	50
4.2	Efficiency . . . . .	52
4.3	Number of Steiner nodes . . . . .	53
5.1	Examples of routing structure establishment for DST variations . . . . .	59
5.2	Impact of Event Scale . . . . .	62
5.3	Impact of Network Scale . . . . .	63
5.4	Impact of Event Duration . . . . .	64
6.1	Examples of routing structure used by the EAST algorithm. . . . .	69
6.2	Spatial Correlation Mechanism applied to the event area . . . . .	72
6.3	Data collected from Amazon rainforest. . . . .	75
6.4	Number of representative nodes . . . . .	76
6.5	Energy consumption of the nodes that are reporting data . . . . .	77

6.6 Accuracy in the readings . . . . . 78  
6.7 Accuracy in the readings of min value . . . . . 80  
6.8 Accuracy in the readings of max value . . . . . 80  
6.9 Accuracy in the readings of mean value . . . . . 80  
6.10 Notifications  $\times$  Readings . . . . . 83  
6.11 Average energy consumption . . . . . 84  
6.12 Data accuracy . . . . . 85  
6.13 Average delay in high reporting rate . . . . . 86  
6.14 Average delay in low reporting rate . . . . . 86

# List of Tables

2.1	Summary of the basic characteristics of some data aggregation aware routing protocols . . . . .	12
2.2	Summary of the basic characteristics of the main proposed spatial and/or temporal data correlations algorithms for WSNs . . . . .	16
3.1	Communication complexity of assessed algorithms . . . . .	30
3.2	Summary of the basic characteristics of assessed algorithms. . . . .	31
3.3	Simulation parameters . . . . .	32
3.4	Scenario with 6 events, 256 nodes and density 20 . . . . .	35
3.5	Scenario with 6 events, 256 nodes and density 30 . . . . .	35
3.6	Scenario with 6 events, 2048 nodes and density 20 . . . . .	35
3.7	Scenario with 6 events, 2048 nodes and density 30 . . . . .	35
4.1	Communication complexity of assessed algorithms . . . . .	48
4.2	Simulation parameters . . . . .	49
5.1	Communication complexity of assessed algorithms . . . . .	60
5.2	Simulation parameters . . . . .	61
6.1	Simulation parameters . . . . .	74
6.2	Simulation parameters . . . . .	81



# List of Algorithms

1	Building the hop tree . . . . .	25
2	Cluster formation and leader election . . . . .	26
3	Route formation, hop tree updates and data transmission . . . . .	27
4	Building the hop tree . . . . .	43
5	Collecting information about nodes' position . . . . .	44
6	Cluster formation and leader election . . . . .	45
7	Routing formation . . . . .	46
8	Data Transmissions . . . . .	47
9	Discovery of neighbors' position . . . . .	56
10	Cluster formation and leader election . . . . .	57
11	EAST Algorithm. . . . .	73



# Contents

<b>Agradecimientos</b>	<b>xi</b>
<b>Resumo</b>	<b>xiii</b>
<b>Resumo Estendido</b>	<b>xv</b>
<b>Abstract</b>	<b>xxiii</b>
<b>List of Figures</b>	<b>xxv</b>
<b>List of Tables</b>	<b>xxvii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	1
1.2 Objectives . . . . .	2
1.3 Main Contributions . . . . .	3
1.4 Organization of the Thesis . . . . .	5
<b>2 Background</b>	<b>7</b>
2.1 Wireless Sensor Networks . . . . .	7
2.2 In-network Data Aggregation . . . . .	9
2.2.1 Routing Scheme Aware Data Aggregation . . . . .	11
2.3 Exploiting Spatio-Temporal Correlation . . . . .	15
2.3.1 Spatial Correlation . . . . .	15
2.3.2 Temporal Correlation . . . . .	18
2.3.3 Spatio-Temporal Correlation . . . . .	20
2.4 Final Remarks . . . . .	21
<b>3 DAARP: Data Aggregation Aware Routing Protocol</b>	<b>23</b>
3.1 Building the Hop Tree . . . . .	24

3.2	Cluster Formation and Leader Election . . . . .	24
3.3	Routing Formation, Hop Tree Updates and Data Transmission . . . . .	25
3.4	Route Repair Mechanism . . . . .	28
3.5	Complexity Analysis . . . . .	29
3.6	Performance Evaluation . . . . .	31
	3.6.1 Methodology . . . . .	31
3.7	Final Remarks on DAARP . . . . .	39
<b>4</b>	<b>DDAARP: Dynamic Data Aggregation Aware Routing Protocol</b>	<b>41</b>
4.1	Building the Hop Tree and Gathering Information About Nodes' Position	42
4.2	Cluster Formation and Leader Election . . . . .	44
4.3	Routing Formation . . . . .	45
4.4	Data Transmission . . . . .	46
4.5	Complexity Analysis . . . . .	47
4.6	Performance Evaluation . . . . .	48
	4.6.1 Simulation Scenario and Metrics Used . . . . .	48
	4.6.2 Simulation Results . . . . .	49
4.7	Final Remarks on DDAARP . . . . .	54
<b>5</b>	<b>DST: Dynamic and Scalable Tree</b>	<b>55</b>
5.1	Discovery of Neighbors' and Sink's Positions . . . . .	56
5.2	Cluster Formation and Leader Election . . . . .	57
5.3	Notification of a New Event . . . . .	57
5.4	Routing Tree Creation and Data Transmissions . . . . .	58
5.5	Complexity Analysis . . . . .	60
5.6	Performance Evaluation . . . . .	60
	5.6.1 Methodology . . . . .	60
5.7	Final Remarks on DST . . . . .	65
<b>6</b>	<b>EAST: Efficient Data Collection Aware of Spatio-Temporal Correlation</b>	<b>67</b>
6.1	Spatial Correlation Model . . . . .	67
6.2	Temporal Correlation Model . . . . .	68
6.3	Overview of the EAST Algorithm . . . . .	68
	6.3.1 Node Localization . . . . .	70
	6.3.2 Cluster Formation, Leader Election, and Division of the Event Area into Cells . . . . .	70
	6.3.3 Data Transmissions . . . . .	72



6.4	Performance Evaluation . . . . .	72
6.4.1	Methodology . . . . .	74
6.4.2	Event Model . . . . .	74
6.4.3	Performance Evaluation of the Spatial Correlation Mechanism . . . . .	75
6.4.4	Performance Evaluation of Temporal Correlation Mechanism . . . . .	81
6.5	Final Remarks on EAST . . . . .	87
<b>7</b>	<b>Final Remarks</b>	<b>89</b>
7.1	Conclusions . . . . .	89
7.2	Limitations . . . . .	91
7.3	Directions for Future Research . . . . .	91
7.4	Comments on Publications . . . . .	92
7.4.1	Journals . . . . .	92
7.4.2	Conferences . . . . .	93
	<b>Bibliography</b>	<b>97</b>



# Chapter 1

## Introduction

### 1.1 Motivation

Recent developments in the areas of wireless communication and multifunctional sensors with communication and processing capability have stimulated the development and use of wireless sensor networks (WSNs) in many different domains such as the environmental, medical, industrial, military fields and many other where human presence is not possible or desired [Boukerche et al., 2007]. A sensor node typically presents a limited sensing capability, but the overall sensing capability can be increased when the nodes are combined with many other nodes forming a WSN. For example, if a gas leak occurs in a room full of gas cylinders and there is only one sensor in this room, it will only be possible to say that there is a leak or not. On the other hand, if a WSN is used, with appropriate protocols, it is possible not only to detect the leak, but to indicate where the leak started and how it evolved. A monitoring in this way can save lives and assets, and reduce cost insurance.

Sensor nodes are energy-constrained devices and the energy consumption is generally associated with the amount of gathered data, since communication is often the most expensive activity in terms of energy. For that reason, algorithms and protocols designed for WSNs should consider the energy consumption in their design [Olariu et al., 2004, AbdelSalam and Olariu, 2009, Villas et al., 2010a, Villas et al., 2011]. Moreover, WSNs are data-driven networks that usually produce a large amount of information that needs to be routed, often in a multihop fashion, toward a sink node, which works as a gateway to a monitoring center. Given this scenario, routing plays an important role in the data gathering process.

For more efficient and effective data gathering with a minimum use of the limited resources, sensor nodes should be configured to smartly report data

by making local decisions [Chatzigiannakis et al., 2005, Chatzigiannakis et al., 2006, Eftymiou et al., 2006, Villas et al., 2010b]. For this, data aggregation and spatio-temporal data correlation are effective techniques for saving energy in WSNs. Due to the inherent redundancy in raw data gathered by sensor nodes, in-networking aggregation and spatio-temporal data correlation can often be used to decrease the communication cost by eliminating data redundancy and forwarding only aggregated information. Since minimal communication leads directly to energy savings, which extends the network lifetime, in-network data aggregation is a key technology to be supported by WSNs.

Data aggregation has been used in WSNs with two purposes: (i) to take advantage of the redundancy and improve data accuracy [Schmid and Schossmaier, 2001, Chakrabarty et al., 2002], and (ii) to reduce the overall data traffic and save energy [Krishnamachari et al., 2002]. Nevertheless, current proposals have a high cost to create routing structures aware of data aggregation and many of them do not consider the spatio-temporal data correlation. In addition, most proposals do not deal with node failures and interruptions during a communication, which cause data loss and do not guarantee delivery of the sensed data.

One of the main challenges in routing algorithms for WSNs is how to guarantee the delivery of the sensed data even in the presence of node failures and interruptions during communication. These failures become even more critical when data aggregation is performed along the routing paths since packets with aggregated data contain information from various sources and, whenever one of these packets is lost a considerable amount of information will also be lost. For this reason, data aggregation aware routing protocols should present a reliable data transmission, through a fault-tolerant routing mechanism.

## 1.2 Objectives

The main goals of this work are twofold. First, we provide a general discussion for data aggregation and spatio-temporal data correlation problems in WSNs. The second goal is to propose, design, and evaluate the performance of different types of algorithms for the problems of *data aggregation* and *spatio-temporal data correlation* for WSNs. To achieve these goals, some secondary objectives should be accomplished:

1. For the first main goal, i.e., in order to allow us to identify open issues and understand the requirements and the implications regarding data aggregation

and spatio-temporal data correlation in WSNs, the following goals need to be achieved:

- 1.1. survey the state-of-the-art about the use of data aggregation and spatio-temporal data correlation in WSNs;
  - 1.2. assess the architectures, models, and methods of data aggregation and spatio-temporal data correlation identified in the survey;
  - 1.3. identify drawbacks of current proposals to propose new solutions that overcome the drawbacks of current proposals; and
2. For the second main goal, to propose different solutions for the problems of *data aggregation* and *spatio-temporal data correlation* that are suitable for different scenarios in a WSN, the following goals need to be achieved:
- 2.1. specify and design algorithms that consider the data structure, data correlations (spatial and temporal), the network topology and application restrictions;
  - 2.2. propose a solution for the data aggregation problem to be used in medium scale WSNs;
  - 2.3. propose a solution for the data aggregation problem to be used in large scale WSNs; and
  - 2.4. analyze the performance of the proposed solutions.

### 1.3 Main Contributions

The main contributions of this thesis are the design and development of four different solutions for data aggregation and spatio-temporal data correlation for WSNs, which we refer to as the DAARP, DDAARP, DST, and EAST algorithms, respectively. In summary, we have:

- *Data Aggregation Aware Routing Protocol for WSNs (DAARP)* is a novel reactive data aggregation aware routing protocol for WSNs. The main motivation to design a new data aggregation aware routing protocol is that the proposed solutions in the literature present high cost to create routing structures aware of data aggregation. The DAARP algorithm builds a routing structure with the shortest paths (in hops) that connect all source nodes to the sink while maximizing data aggregation, whose main contribution is to maximize data aggregation

along the communication route, in a more reliable way, through a routing fault-tolerant mechanism. Simulation results (presented in Section 3.6) reveal that DAARP has some key aspects required by data aggregation in WSNs such as a reduced number of messages for setting up a routing structure, maximized number of overlapping routes, high aggregation rate, and reliable data aggregation and transmission. This algorithm is fully explained in Chapter 3.

- *Dynamic Data-Aggregation Aware Routing Protocol for WSNs (DDAARP)* is a novel dynamic data-aggregation aware routing protocol for WSNs, which uses the sink node for processing and configuration of routes aware of data aggregation. The main motivation to design a dynamic approach to create dynamic routing structures aware of data aggregation is that we have identified that the quality of routing structures created by DAARP and the most algorithms in the literature depend on the order of events occurrence and once created, these routes are held fixed during the occurrence of events. The main contribution is that the routes created by DDAARP do not depend on the order of events occurrence and are not held fixed during the occurrence of events such as the DAARP. Simulation results (presented in Section 4.6) reveal that DDAARP presents low cost in terms of packet control, improves the quality of the routing structure and maximizes data aggregation along the communication route in a more reliable way, through a routing fault-tolerance mechanism. This algorithm is fully explained in Chapter 4.
- *Dynamic and Scalable Tree for WSNs (DST)* is an efficient data aggregation solution that allows scalable and dynamic routing in WSNs, which builds routing structures with the shortest routes (in Euclidean distance) that connects all source nodes to the sink node maximizing data aggregation and reducing the distance to connect each source node to the sink. Also, the routing structure created does not depend on the event order. The main motivation to design the DST was the lack of a solution in the literature scalable, dynamic and presents low cost to create routing structures aware of data aggregation. DDAARP presents good results, but it suffers from scalability problems and becomes impractical for large-scale networks. In addition, the sink node needs to have a global knowledge of the network. Simulation results (presented in Section 5.6) reveal that DST presents a low cost in terms of packet control, maximizes aggregation points and improves the quality of routing structure offering dynamic routes. This algorithm is fully explained in Chapter 5.

- *Efficient Data Collection Aware of Spatio-Temporal Correlation for WSNs (EAST)* is an algorithm for energy-aware data forwarding in WSNs that takes full advantage of both spatial and temporal correlation mechanisms to save energy while still maintaining real-time, accurate data report towards the sink node. The main motivation to design the EAST is that most of the current spatial and/or temporal correlation algorithms do not consider the energy dissipation during data collection to better choose the representative nodes. Also, these solutions present a high number of control messages and do not exploit efficiently the spatio-temporal correlation nor their dynamicity. The main contribution is an energy-aware spatio-temporal correlation mechanism in which nodes that detected the same event are dynamically grouped in correlated regions and a representative node is selected at each correlation region for observing the phenomenon. The entire region of sensors per event is effectively a set of representative nodes performing the task of data collection and temporal correlation. Simulation results (presented in Section 6.4) clearly show that by using both spatial and temporal correlation, the information about the event can be sensed with a high accuracy while still saving the residual energy of nodes. This algorithm is fully explained in Chapter 6.

## 1.4 Organization of the Thesis

This thesis is divided into seven chapters. Chapter 2 provides an overview about wireless sensor networks, in-network aggregation and spatio-temporal data correlation. The chapter introduces wireless sensor networks, discusses in-network data aggregation and presents the main techniques to perform data aggregation. In addition, it discusses spatio-temporal data correlation and provides an overview of the existing approaches that exploit spatio-temporal data correlation.

In the second part of this work, composed of Chapters 3, 4, 5 and 6, we propose and explain the DAARP, DDAARP, DST and EAST algorithms, respectively. In each chapter, the performance of the proposed solution is evaluated through simulations.

Finally, in Chapter 7, we present some final remarks about the studied problems, their solutions, and the obtained results. We also present some possible extensions of our work.





# Chapter 2

## Background

This chapter presents the theoretical foundation for this work. This chapter is organized as follows: Section 2.1 introduces wireless sensor networks, Section 2.2 discusses in-network data aggregation and presents the related work, and Section 2.3 discusses spatio-temporal data correlation and provides an overview of the existing approaches which exploit spatio-temporal data correlation.

### 2.1 Wireless Sensor Networks

Wireless Sensor Networks (WSNs) [Akyildiz et al., 2002, Romer and Mattern, 2004, Boukerche et al., 2007, Anastasi et al., 2009] can be defined as a cooperative network of small, battery-operated, wireless sensor nodes whose main goal is twofold: to monitor their surroundings for local data and to forward the gathered data toward a sink node using typically multihop communication. This sink node will then be responsible for processing all of the received data from several source nodes and reporting them to a monitoring facility (Figure 2.1). This type of network has become popular due to its applicability that includes several areas such as environment, homeland security, industry, domestics, agriculture, meteorology, health, space, military and many other applications that can be critical to save lives and assets [Younis et al., 2006, Anastasi et al., 2009, Villas et al., 2010b]. Several physical properties can be monitored, including temperature, humidity, pressure, ambient light, sound, vibration, and motion.

One of the main limitations of the WSNs is the battery-operated nature of their sensor nodes, which makes this kind of network highly energy-constrained. A simple solution to this problem could be the periodic replacement of the node battery. However, this solution is not feasible due to the large number of nodes in the net-

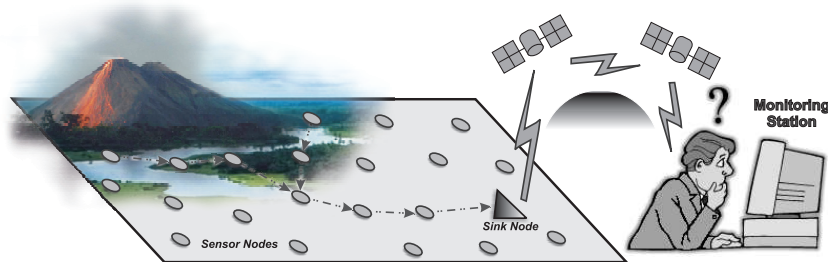


Figure 2.1. Data routing in WSNs [Oliveira et al., 2009].

work or because the sensor nodes may be inaccessible in some applications such as monitoring volcanoes or space exploration. For that reason, algorithms and protocols designed for WSNs should consider the energy consumption in their conception [Olariu et al., 2004, AbdelSalam and Olariu, 2009, Villas et al., 2011].

For more efficient data gathering with a minimum use of limited resources, sensors should be configured to report data more intelligently by making local decisions [Chatzigiannakis et al., 2005, Chatzigiannakis et al., 2006, Efthymiou et al., 2006, Villas et al., 2010b]. Data aggregation<sup>1</sup> and spatio-temporal<sup>2,3</sup> data correlation are possible techniques for local decision-making, which will be presented in the following sections. Such strategies help to maximize energy conservation in an application-specific sensor network.

These techniques have been exploited in the literature such as data aggregation [Krishnamachari et al., 2002, Chandrakasan et al., 2002, Nakamura et al., 2006, Fan et al., 2006, Nakamura et al., 2009], spatial correlation [Akyildiz et al., 2004, Yoon and Shahabi, 2005, Liu et al., 2007b, Le et al., 2008, Yuan and Chen, 2009] and temporal correlation [Min and Chung, 2010, Pham et al., 2010]. Nevertheless, current proposals have a high cost to create routing structures aware of data aggregation and many of them does not deal nodes failures and interruptions in communications, which causes loss of data and does not guarantee delivery of the sensed data. In addition, these solutions not only introduce delays in data transmissions but also lead to the reception of outdated information by the sink node.

---

<sup>1</sup>**Data Aggregation** eliminates inherent redundancy in raw data gathered by the sensor nodes and forwarding only smaller aggregated information.

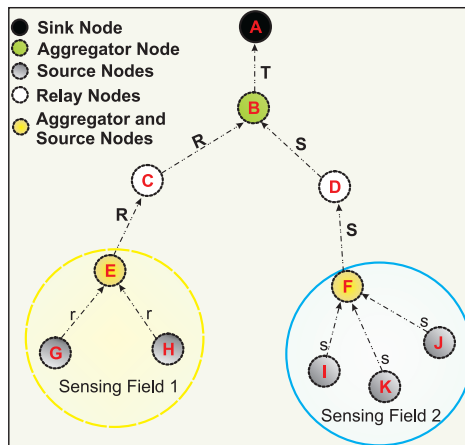
<sup>2</sup>**Spatial correlation:** the change pattern of the data sensed by nearby nodes is expected to be the same or similar. Thus, exploit the spatial data correlation can eliminate the similars data reporting.

<sup>3</sup>**Temporal correlation:** the change pattern in readings of a sensor node and gathered data is usually similar over a short-time period. Due to the nature of the physical phenomenons, there is a significant temporal correlation among each consecutive observation of a sensor node. Thus, exploit the temporal correlation can eliminate the similars data reporting.

## 2.2 In-network Data Aggregation

In the context of WSNs, in-network data aggregation refers to different techniques to forward data packets toward the sink node. During this process, nodes combine data collected by different sources, i.e., by fusing sensor readings related to the same event or physical quantity, or by locally processing raw data before its transmission. A key component of in-network data aggregation is the design of a data aggregation aware routing protocol, which determines where the aggregation shall be performed. Data aggregation requires a forwarding paradigm that is different from the classic routing, which typically involves the shortest path “in relation to some specific metric” to forward data toward the sink node. Differently from the classic approach in data aggregation aware routing protocols, chooses the node as next hop based on their proximity in the topology that maximizes the overlap of routes in order to promote in-network data aggregation.

Before classifying the literature on solutions aware data aggregation, first we illustrate the importance of coupling between routing and data aggregation in WSNs. As depicted in Figure 2.2, assume that the routing structure of data collection in a wsn is a inverted multicast tree rooted at the sink (node A). The concept of in-network data aggregation can be illustrated as follow.



**Figure 2.2.** Data Aggregation Aware Routing

Let each of the nodes in  $\{E, G \text{ and } H\}$  in the sensing field 1 and  $\{F, I, J \text{ and } K\}$  in the sensing field 2 generate one raw sensory packet of size  $r$  and  $s$  bits respectively. The arrows in the figure indicate the data gathering structure. For example, data collected by the sensing nodes  $\{E, G \text{ and } H\}$  in the sensing field 1 will be sent to the sink via nodes  $C$  and  $B$ . As collected data from physically proximate nodes are usually correlated,  $E$  can aggregate the data from  $G$  and  $H$  before forwarding to  $C$ .

The result is that the size of the outgoing packet from  $E$ ,  $R$  bits, will be less than the summation of all the incoming packets (including its own) and often larger than any of the individual incoming packets. The amount of the reduction depends on the data correlation as specified by the application. For example, in the extreme case where there is no data correlation, we have  $R = 3r$  (including the data sensed by node  $E$ ) as no data reduction can be achieved. On the other hand, if the desired result is, say, simply average of the measurements,  $R = r$ . However, most applications which data has a certain degree of correlation will satisfy  $k < R < 3k$ . The node  $B$  aggregates the collected data in the sensing field 1 and 2. The result is that the size of the outgoing packet from  $B$ ,  $T$  bits, will be less than the summation of the incoming packets of  $R$  and  $S$  bits and often larger than the lower individual incoming packets of size  $\min(R, S)$  bits, ie,  $\min(R, S) < T < R + S$ .

Based in the above example, a key factor in the process of data aggregation is the routing scheme, which determines where the aggregation shall be performed. For in-network data aggregation to be realized, some form of synchronization is needed among the nodes. Typically, in these algorithms, a node usually does not send data as soon as it is available since waiting for data from neighboring nodes may lead to better data aggregation opportunities. This in turn, will improve the performance of the algorithm and save energy. Three main timing strategies are found in the literature [Solis and Obraczka, 2004, Hu et al., 2005]:

- *Periodic simple aggregation*: requires each node to wait for a pre-defined period of time while aggregating all received data packet and, then, forwards a single packet with the result of the aggregation.
- *Periodic per-hop aggregation*: quite similar to the previous approach, but the aggregated data packet is transmitted as soon as the node hears from all of its children. This approach requires each node to know the number of its children. In addition, a timeout may be used for the case of some children's packet being lost.
- *Periodic per-hop adjusted aggregation*: adjusts the transmission time of a node according to this node's position in the gathering tree.

Note that the choice of the timing strategy strongly affects the aggregation rate in-network and latency to report data collected. If the waiting time in aggregation point is high, the latency and aggregation rate is higher. If the waiting time in aggregation points is low, the latency and aggregation rate is lower.

### 2.2.1 Routing Scheme Aware Data Aggregation

In-network data aggregation plays an important role in energy constrained WSNs since data correlation is exploited and aggregation is performed at intermediate nodes reducing size and the number of messages exchanged across the network. In data gathering based applications, a considerable number of communication packets can be reduced by in-network aggregation, resulting in a longer network lifetime. The problem of obtaining the optimal aggregation tree is known to be  $\mathcal{NP} - \text{Hard}$  [Al-Karaki et al., 2004], which is equivalent to the Steiner tree problem [Krishnamachari et al., 2002].

**Definition 1 (Steiner Tree)** *given a network represented by a graph  $G = (V, E)$ , where  $V = \{v_1, v_2, \dots, v_n\}$  is the set of sensor nodes,  $E$  is the set of edges representing the connections among the nodes, i.e.,  $\langle i, j \rangle \in E$  iff  $v_i$  reaches  $v_j$ , and  $w(e)$  is the cost of edge  $e$ , a minimal cost tree is to be built that spans all source nodes  $S = \{s_1, s_2, \dots, s_m\}$ ,  $S \subseteq V$ , and the sink node  $s_0$ . The cost of the resulting Steiner tree ( $W$ ) is the sum of the costs of its edges. This problem is a well-known NP-hard problem.*

In the literature, there are different heuristics to the Steiner tree problem, some of them present 1.598 approximation factor [Hougardy and Prömel, 1999, Robins and Zelikovsky, 2000]. However, those solutions are not affordable to resource-constrained networks, such as WSNs, since their distributed implementation requires a large amount of messages to setup the routing tree, which consequently leads to high energy consumption.

Some research efforts have also been made to develop routing algorithms for WSNs. Table 2.1 presents a summary of the basic characteristics of the main proposed routing protocols for WSNs. In this work, we classify these proposals into three categories: *tree-based*, *cluster-based*, and *structure-less* algorithms. In the next sections, we will briefly review these protocols and their structures.

#### 2.2.1.1 Tree-Based Approaches

Protocols in this family [Al-Karaki and Kamal, 2004, Akkaya and Younis, 2005, Fasolo et al., 2007] are usually based on a hierarchical organization of the nodes in the network. In fact, the simplest way to aggregate data flowing from the sources to the sink node is to elect some special nodes that work as aggregation points and define a preferred direction to be followed when forwarding data.

In these protocols, a tree structure is constructed first and then used later to either route the gathered data or respond to queries sent by the sink node. Aggregation is performed during the routing when two or more data packets arrive at the same node

**Table 2.1.** Summary of the basic characteristics of some data aggregation aware routing protocols

Scheme	Route Structure	Objective	Aggregation Nodes	Overhead	Scalability	Drawback
<b>LEACH</b>	Cluster-based	Maximize lifetime	Clusterheads	Medium	Low	Low scalability
<b>SPT</b>	Tree	Shortest-Path	Opportunistic	Low	High	Data redundancy
<b>GIT</b>	Tree with path sharing	Minimize total energy cost	Intermediate nodes	Very High	Very Low	High cost
<b>CNS</b>	Tree	Aggregate closer to the sink	Aggregator node	High	Medium	Only one aggregation point
<b>InFRA</b>	Tree-based cluster	Maximize overlap routes	Clusterheads and intermediate nodes	Very High	Low	Low scalability and high cost
<b>DAARP</b>	Tree-based cluster	Maximize overlap routes	Clusterheads and intermediate nodes	Medium	Medium	Static routes
<b>DDAARP</b>	Tree-based cluster	Maximize overlap routes	Clusterheads and intermediate nodes	Low	Medium	Requires global knowledge
<b>DST</b>	Based on straight line segments and cluster	Maximize overlap routes and minimize overhead	Clusterheads and intermediate nodes	Very Low	Very High	Requires position information
<b>DAA</b>	Anycast	Minimize overhead	Intermediate nodes	Very Low	Very High	Not all packets may be aggregated

of the tree. This node then aggregates all received data with its own data and forwards only one packet to its neighbor that is lower in the tree. However, this approach has some drawbacks. For instance, when a packet is lost at a certain level of the tree (e.g., due to channel impairments), data from the whole sub tree will be lost as well. Thus, tree-based approaches require a mechanism for fault tolerance to reliably forward the aggregated data.

Despite the potentially high cost of maintaining a hierarchical structure in dynamic networks and the scarce robustness of the system in case of link/device failures, these approaches are still particularly suitable for designing optimal aggregation functions and performing efficient energy management. For instance, there are some proposed solutions [III et al., 2007, Villas et al., 2010a] where the sink node organizes routing paths to evenly and optimally distribute the energy consumption while still favoring the aggregation of data at the intermediate nodes.

In most cases, tree-based protocols build a traditional shortest path routing tree. For instance, the Shortest Path Tree (SPT) algorithm [Krishnamachari et al., 2002] uses a very simple strategy to build a routing tree in a distributed fashion. In this approach, every node that detects an event reports its collected information by using a shortest path to the sink node. Data aggregation occurs whenever paths overlap (opportunistic data aggregation). The Directed Diffusion [Intanagonwiwat et al., 2003] algorithm is one of the earliest solutions to also propose attribute-based routing.

In these cases, data can be opportunistically aggregated when they meet at any intermediate node. Based on Directed Diffusion, the Greedy Incremental Tree (GIT) [Intanagonwiwat et al., 2002] approach was proposed. The GIT algorithm establishes an energy-efficient path and greedily attaches other sources onto the established path. In the GIT strategy, when the first event is detected, nodes send their information as in the SPT algorithm and, for every new event, the information is routed using the shortest path to the current tree. There is a new aggregation point every time a new branch is created. Some practical issues make GIT not appropriate in WSNs [Nakamura et al., 2006]. For example, each node needs to know the shortest path to all nodes in the network. The communication cost to create this infrastructure is  $O(n^2)$ , where  $n$  is the number of nodes. Furthermore, the space needed to store this information at each node is  $O(Dn)$ , where  $D$  is the number of hops in the shortest path connecting the farthest node  $v \in V$  to the sink node (network diameter). After the initial phase the algorithm needs  $O(mn)$  messages to build the routing tree, where  $m$  is the number of source nodes.

Another interesting solution is the Center at Nearest Source (CNS) algorithm [Krishnamachari et al., 2002]. In CNS, every node that detects an event sends its information to a specific node, called the aggregator, by using a shortest path. The aggregator is the closest node to the sink (in hops) that detects an event. CNS reduces the amount of data sent to the sink in relation to the classical approaches, but the overhead in CNS is highly dependent on the events' occurrence positions. In scenarios with many events occurring simultaneously, CNS has a high cost to change the aggregator node. In this algorithm, data redundancy is only reduced when it is already close to the sink node.

### 2.2.1.2 Cluster-Based Approaches

Similarly to tree-based approaches, cluster-based schemes [Chandrakasan et al., 2002, Nakamura et al., 2006, Villas et al., 2009, Villas et al., 2010a] also consist of a hierarchical organization of the network. However, in this approach, nodes are subdivided into clusters. Moreover, special nodes, referred to as cluster-heads, are elected to aggregate data locally and transmit the result of such an aggregation to the sink node.

In the Low-Energy Adaptive Clustering Hierarchy (LEACH) algorithm [Chandrakasan et al., 2002], clustered structures are exploited to perform data aggregation. In this algorithm, cluster-heads can act as aggregation points and they communicate directly to the sink node. In order to evenly distribute energy consumption among all nodes, cluster-heads are randomly elected in each round.

LEACH-based algorithms assume that the sink can be reached by any node in only one hop, which limits the size of the network for which such protocols can be used. In addition, in scenarios where the data can not be perfectly aggregated, LEACH-based protocols do not necessarily have significant advantage since the cluster-heads have to send many packets to the sink using a high transmission power.

The Information Fusion-based Role Assignment (InFRA) algorithm [Nakamura et al., 2006] builds a cluster for each event including only those nodes that were able to detect it. Then, cluster-heads merge the data within the cluster and send the result to the sink node. The InFRA algorithm aims to build the shortest path tree that maximizes the information fusion. Once clusters are formed, cluster-heads choose the shortest path (to the sink node) that maximizes the information fusion with already formed paths/clusters [Nakamura et al., 2006]. A disadvantage of the InFRA algorithm is that for each new event that arises in the network, the information about the event must be flooded throughout the network to inform other nodes about its occurrence and to update the paths from the already existing cluster-heads to the sink node. This procedure limits InFRA's scalability.

Another interesting solution is the Data Aggregation Aware Routing Protocol (DAARP) [Villas et al., 2009]. For each event this algorithm performs the clustering of nodes that detected the same event, as well as the election of a cluster-head. Then, cluster-heads merge data within the cluster and send the result to the sink node. After the cluster-head formation, routes are created by selecting nodes in the shortest path (in hops) to the nearest node that is part of an existing routing infrastructure in which this node will be an aggregation point. The DAARP routing infrastructure tends to maximize the aggregation points and uses fewer control packets to build the paths. Different from InFRA, DAARP does not flood a message to the whole network whenever a new event occurs. DAARP is not feasible for scenarios with long duration events because the routes are static, which quickly consumes the energy of the nodes that are part of the routing structure.

The Dynamic Data Aggregation Aware Routing Protocol (DDAARP) [Villas et al., 2010a] adds an improvement over DAARP. In the DDAARP algorithm, the routes are computed at the sink node and do not depend on the order of events. Routes created by DDAARP are not kept fixed throughout the duration of events, i.e., routes may change when necessary. The drawback of this proposal is that packets containing information from nodes tend to increase their size at the information collecting stage and this solution becomes impractical for large-scale networks. In addition, the sink node needs to have a global knowledge of the network, such as node positions, residual energy of nodes, and nodes that detected events.



### 2.2.1.3 Structure-Less Approaches

Few algorithms for routing aware of data aggregation have been proposed that use a structure-less approach. The Data-Aware Anycast (DAA) algorithm [Fan et al., 2006], a structure-less data aggregation algorithm, uses anycast to forward packets to one-hop neighbors that have packets to be aggregated. It involves mechanisms to increase the chance of packets meeting at the same node (spatial aggregation) at the same time (temporal aggregation). Since the approach does not guarantee aggregation of all packets, the cost of transmitting packets with no aggregation increases with the size of the network. In addition, packets that are unable to be aggregated will not benefit from the energy savings achieved by eliminating the control overhead.

The Dynamic and Scalable Tree (DST) algorithm [Villas et al., 2010b] aims to build a routing tree with the shortest routes (in Euclidean distance) that connects all source nodes to the sink node, maximizing data aggregation while reducing the distance connecting each coordinator node to the sink. Routes are based on straight line segments, which are computed by the coordinator nodes. The created paths do not depend on the event order. Similar to all approaches that do not exploit the spatial correlation, DST does not show a good performance in scenarios where many nodes detect the same event, since nodes that report information about the event can consume their energy quickly.

## 2.3 Exploiting Spatio-Temporal Correlation

In this section, we present the benefits of exploiting spatio-temporal data correlation in WSNs. We also discuss some of the existing approaches and algorithms that take advantage of spatio-temporal correlation in WSNs. Table 2.2 presents a summary of the basic characteristics of the main proposed spatial and/or temporal data correlations algorithms for WSNs.

In the current literature, we can find three main categories of data correlation protocols: (i) spatial correlation; (ii) temporal correlation and (iii) spatio-temporal correlation. In the following, we present some of these protocols as well as the benefits of exploiting spatial/temporal data correlation in WSNs.

### 2.3.1 Spatial Correlation

In a WSN, nodes that detect the same event are typically grouped to save energy, and a node is elected as the coordinator of the group [Chandrakasan et al., 2002,

**Table 2.2.** Summary of the basic characteristics of the main proposed spatial and/or temporal data correlations algorithms for WSNs

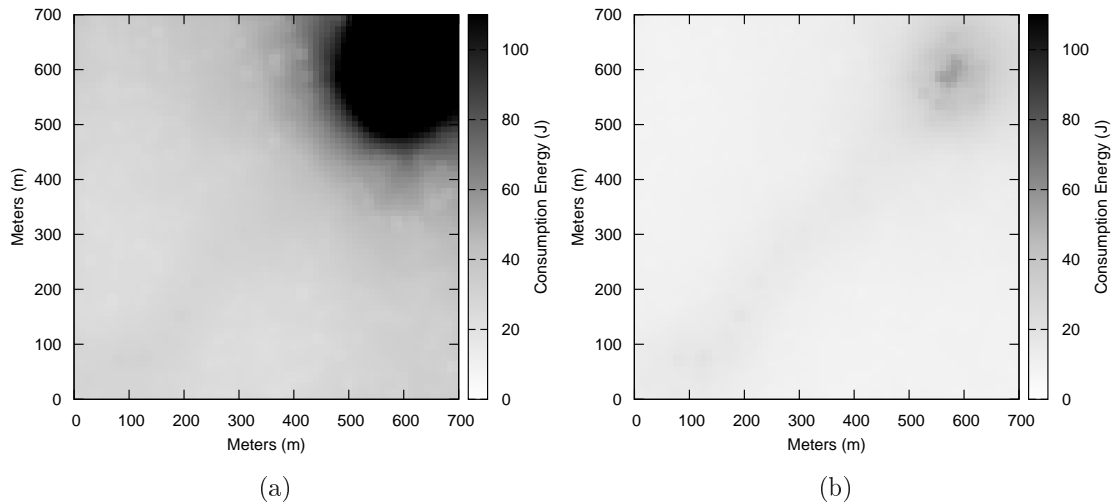
Scheme	Route Structure	Objective	Spatial Correl.	Temporal Correl.	Overhead	Scalability	Drawback
<b>EEDC</b>	Single hop	Eliminate control overhead	Yes	No	Very Low	Very Low	Centralized and single-hop network
<b>CAG</b>	Tree-based cluster	Eliminate data redundancy	Yes	No	Very High	Medium	Maintenance data-centric
<b>GSC</b>	Tree-based cluster	Eliminate data redundancy	Yes	No	High	Low	Is not applied to multi-hop members
<b>SBR</b>	Tree-based	Eliminate data redundancy	No	Yes	Medium	High	Sink node can receive outdated information
<b>SCCS</b>	Tree-based cluster	Eliminate data redundancy	Yes	Yes	Medium	High	Sink node can receive outdated information
<b>EAST</b>	Based on straight line segments and cluster	Maximize overlap routes and minimize control overhead	Yes	Yes	Very Low	Very High	Requires position information

Yoon and Shahabi, 2005, Yuan and Chen, 2009, Nakamura et al., 2009, Villas et al., 2009, Villas et al., 2010a, Villas et al., 2010b, Villas et al., 2011]. The elected node is then responsible for receiving all the event notifications and forwarding them toward the sink node. The energy consumption of the nodes that detect events is greater than the other network nodes (see Figure 2.3(a)). This occurs because nodes within the group (nodes that detect events) consume a great deal of energy receiving and forwarding data packets from their neighbors, besides their own notifications.

As an initial motivation, Figure 2.3 presents the energy consumption in the process of data collection in a WSN of two different routing approaches when the sink node, located at position (0,0), receives data from a detected event that has a radius of 70 m and is located at position (600,600). The first approach (Figure 2.3(a)) is a simple method for data collection where all nodes that detected the event send the sensed data toward the sink node. The second approach (Figure 2.3(b)) is a more sophisticated strategy that uses spatial correlation to save energy. In this case, only a subset of nodes that detected the event sends sensory data to the sink node. In both scenarios, the notification of the detected event was performed at each second, and the event duration was of only 10 m. The first approach (Figure 2.3(a)) sends 32157 notifications, whereas the second approach (Figure 2.3(b)) sends only 5667 notifications.

The difference between the two approaches is notable and, by using spatial correlation, the second approach was able to save a large amount of energy, extending the overall network lifetime.

The spatial correlation of sensory information among the nodes that detect an event exists when those nodes are geographically close, i.e., they have similar infor-



**Figure 2.3.** Energy consumption of nodes during the data collection when using (a) a classical approach; and when using (b) a spatial correlation based approach.

mation. In this case, instead of having all sensor nodes reporting the same data, it is more efficient to choose a few representative nodes to notify the sink node about the detected event (see Figure 2.3(b)). A representative node reports the event information of a given area on behalf of a group of nodes that collects similar information in the same area.

Akyildiz et al. [Akyildiz et al., 2004] studied the relation between reliability of event detection and spatial location of the sensor nodes in the event area. Their solution estimates the number of sensor nodes (representative nodes) required to send the detected event to the sink in order to have reliable event information. Each representative node represents a spatially correlated group of nodes. Although their solution achieves overall energy gain, it fails to consider the remaining energy during the selection of the representative nodes – an assumption that should not be neglected in a WSN because of hardware constraints. Thus, if a representative node works in the correlation region for a long period of time, it will spend more energy due to the number of transmitted messages compared to the other nodes.

Yoon and Shahabi [Yoon and Shahabi, 2005] proposed a new mechanism for spatial correlation in WSNs. The proposed mechanism, called Clustered Aggregation Technique (CAG), creates clusters of nodes with similar sensing values and only a node inside the cluster notifies its reading to the Sink node whereas the other nodes ignore their readings. The CAG algorithm is divided into two phases: query and response. In the query phase, the data-centric clusters are created according to a user-specified error threshold  $\tau$ . Nodes that have sensed values smaller than this threshold

belong to the same cluster. In the second phase (response phase), just one node per cluster (cluster-head) sends its sensed value to the sink node notifying the detected event. The authors showed that the proposed mechanism can reduce significantly the number of transmitted messages during the data collection. However, during the first phase, the CAG algorithm uses a flooding-based protocol to disseminate the query to all sensor nodes, which is not needed in most scenarios. Moreover, the maintenance of the data-centric clusters remains a difficult problem [Boukerche et al., 2003].

Liu et al. [Liu et al., 2007a] proposed another clustering algorithm, named Energy-Efficient Data Collection framework (EEDC), to exploit spatial data correlation. They consider that nodes collect data continuously and are one-hop connected to the sink node or to a center node. The algorithm was designed to be executed at the sink node, since this node has the entire data network information. The algorithm creates clusters of nodes that are spatially correlated. Also, the sink node manages the cluster formation dynamically in order to reflect environmental changes. The primary limitation of that scheme is the assumption of the single-hop communication. This assumption is impractical in a distributed system and difficult to have in large-scale wireless sensor networks. Another disadvantage is the clustering algorithm that is centralized at the sink node. Because of this, all network data needs to be sent to the sink node, which will store and process a great amount of data.

Shah et al. [Shah and Bozyigit, 2007] proposed a new mechanism for spatial correlation in WSNs, named Gridiron Spatial Correlation (GSC). The GSC is adaptive to achieve the required reliability by dynamically changing the correlation region. The correlation regions are formed as squared rectangles and nodes lying in the rectangle are assumed to be spatially correlated. Cluster-head identifies the redundant and close sources in its vicinity and turns off the activity of nodes by considering their energy level and closeness as criterion. The limitation of GSC is the control mechanism which is not applied to multi-hop members, ie, it only works well for scenarios where the event radius is smaller than the communication radius of the cluster-head.

### 2.3.2 Temporal Correlation

Sensor readings about the environment are typically periodic; consequently, the time-ordered sequence of sensed data constitutes a time series (see Figure 2.4(a)). Due to the nature of the physical phenomenon, there is a significant temporal correlation among each consecutive observation of a sensor node and gathered data is usually similar over a short-time period. For example, in a daily sampling of temperature performed at each minute, the temperature may not change significantly. Thus, in these case,

sensor nodes do not need to transmit their readings if the current reading is within an acceptable error threshold regarding the last reported reading (see Figure 2.4(b)). The sink node can just assume that any unreported data is unchanged from the previously received ones. The degree of correlation between consecutive sensor measurements might vary according to the characteristics of the phenomenon.

The temporal correlation can be captured by mathematical models such as wavelet transforms or linear models [Liu et al., 2007a] (see an example in Figure 2.4). Therefore, the time series can be approximated using a suitable mathematical model. The result obtained is the amount of approximating data, and is usually much lower than the volume of the whole data series. Transferring approximation data, instead of raw data, can significantly reduce energy consumption on communication within the network.



**Figure 2.4.** (a) The time series and (b) The piecewise linear presentation

Vuran et al. [Vuran et al., 2004] proposed a new framework to create data centric protocols that explore the nature of the physical phenomenon observed by a WSN. The main goal of the framework is to incorporate temporal correlation among consecutive observations of the phenomenon to reduce communication costs. The authors also explore spatial correlation by showing that nearby nodes tend to have the same observed data. The proposed framework can be used in two ways: (i) to develop efficient protocols, and (ii) to develop reliable sensed information reporting in WSN.

Deligiannakis and Kotidis [Deligiannakis and Kotidis, 2008] proposed a framework based on temporal correlation that uses a Self-Based Regression (SBR) algorithm [Deligiannakis et al., 2004] to decrease the number of transmitted messages required to monitor a physical phenomenon. The goal of the SBR algorithm is to process the observed data before sending it to the sink node. The framework stores the sensed information in a buffer and, when it is full, the SBR algorithm processes the data to find representative information. The authors claim that by sending just the representative information, the sink node can reconstruct the observed event without losing accuracy. However, the main drawback of such an approach is the waiting time until the buffer fills up. In this case, the sink node can receive outdated information about

the sensed event.

### 2.3.3 Spatio-Temporal Correlation

The spatio-temporal correlation happens when the nature of the collected data has both spatial and temporal correlations, i.e., nodes close geographically have the same reading that is similar to the previous one. In this case, solutions that use both correlations can take advantage of the nature of the detected event to decrease the number of reported data.

Pham et al. [Pham et al., 2008, Pham et al., 2010] proposed a spatio-temporal solution, called Spatiotemporal Clustering and Compressing Schemes (SCCS), which uses a buffer to store the monitored data. When the buffer is full, SCCS executes a divide and conquer algorithm (DCA) to find the representative information inside the buffer exploring the temporal correlation. The goal of the DCA is to find the minimum data set to be transmitted to the sink node. Considering all readings inside the buffer, the DCA creates a dividing line between the first element and the last one. For each buffer data, the algorithm calculates the distance between this value and the created line. If the value is smaller than a predefined threshold, the solution indicates that it has already been considered so that it is not necessary to include it again in the packet to be sent to the sink. When the value is greater than the threshold, the algorithm splits the line into two (one line up to this value and another one up to the endpoint of the previous line). When a line is split into two lines, all buffer values are verified again. These steps are repeated until a created line is not split into two anymore. Also, the SCCS solution creates a cluster among nodes that sensed the event in order to perform spatial correlation to reduce the number of transmitted messages. As mentioned before, by using a buffer to perform the temporal correlation, the waiting time to deliver the gathered data can be inappropriate for a number of real-time sensor network applications.

Xu et al. [Yu et al., 2006] proposed a wavelet-based spatio-temporal data compression algorithm for WSNs. Their algorithm employs a ring topology that explores simultaneously the temporal and spatial correlations among the sensed data. The algorithm considers that the sensor network is divided into clusters and each cluster is controlled by a cluster head. The algorithm also considers a virtual grid where the nodes inside each cell have spatial and temporal correlation. The nodes execute a wavelet transform algorithm to compress the sensed data on the ring in such a way it can be energy efficiently transmitted to its cluster head and then, delivered to the sink node. However, the authors did not investigate the processing task to execute the

proposed wavelet transform in sensor nodes with limited capabilities. Also, the created virtual grid is not based on the event characteristics, which can result in inaccurate information.

Most of the current work on spatial and/or temporal correlation algorithms does not consider the energy dissipation and the event characteristics during data collection to better choose the representative nodes. Also, these solutions usually result in high delays and outdated data arriving at the sink node. The proposed algorithm, called EAST, presented in the Chapter 6, exploit both spatial and temporal correlations to perform near real-time data collection in WSNs. In our algorithm nodes that detected the same event are dynamically grouped in correlated regions and a representative node is selected at each correlation region for observing the phenomenon. The entire region of sensors per event is effectively a set of representative nodes performing the task of data collection and spatio-temporal correlation.

## 2.4 Final Remarks

This chapter presents the main aspects of WSNs, including their main features, common requirements and operation. Note that energy is a key factor in designing solutions for WSNs. The lifetime of the network depends on the adoption of measures to save energy. This chapter was also introduced the concept of in-networking data aggregation and spatio-temporal data correlation and some solutions in the literature. The next chapters describe the work done and obtained results.





## Chapter 3

# DAARP: Data Aggregation Aware Routing Protocol

The DAARP is a novel reactive data aggregation aware routing protocol for WSNs. The main motivation to design a new data aggregation aware routing protocol is that the solutions in the literature presents high cost to create routing structures aware of data aggregation. The main goal of our proposed DAARP algorithm is to build a routing tree with the shortest paths that connect all source nodes to the sink while maximizing data aggregation. The proposed algorithm considers the following roles in the routing infrastructure creation:

- *Collaborator*: a node that detects an event and reports the gathered data to a coordinator node;
- *Coordinator*: a node that also detects an event and is responsible for gathering all the gathered data sent by collaborator nodes, aggregating them and sending the result toward the sink node;
- *Sink*: a node interested in receiving data from a set of coordinator and collaborator nodes;
- *Relay*: a node that forwards data toward the sink.

The DAARP algorithm can be divided into three phases. In Phase 1, the hop tree from the sensor nodes to the sink node is built. In this phase, the sink node starts building the hop tree that will be used by Coordinators for data forwarding purposes. Phase 2 consists of cluster formation and cluster-head election among the nodes that detected the occurrence of a new event in the network. Finally, Phase 3 is responsible

for both setting up a new route for the reliable delivering of packets and updating the hop tree.

### 3.1 Building the Hop Tree

In this phase, the distance from the sink to each node is computed in hops. This phase is started by the sink node sending, by means of a flooding, the Hop Configuration Message (HCM) to all network nodes. The HCM message contains two fields: `ID` and `HopToTree`, where `ID` is node identifier that started or retransmitted the HCM message and `HopToTree` is the distance, in hops, by which an HCM message has passed.

The `HopToTree` value is started with value 0 at the sink, which forwards it to its neighbors (at the beginning, all nodes set the `HopToTree` as infinity). Each node, upon receiving the message HCM, verifies if the value of `HopToTree` in the HCM message is less than the value of `HopToTree` that it has stored and if the value of `FirstSending` is true, as shown in Algorithm 1 - Line 3. If that condition is true then the node updates the value of the `NextHop` variable with the value of the field `ID` of message HCM, as well as the value of the `HopToTree` variable, and the values in the fields `ID` and `HopToTree` of the HCM message. The node also relays the HCM message, as shown in Algorithm 1 - Line 8. Otherwise, if that condition is false, which means that the node already received the HCM by a shorted distance, then the node discards the received HCM message, as shown in Algorithm 1 - Line 12. The steps described above occur repeatedly until the whole network is configured.

Before the first event takes place, there is no established route and the `HopToTree` variable stores the smallest distance to the sink. On the first event occurrence, `HopToTree` will still be the smallest distance; however, a new route will be established. After the first event, the `HopToTree` stores the smaller of two values: the distance to the sink or the distance to the closest already established route.

### 3.2 Cluster Formation and Leader Election

When an event is detected by one or more nodes, the leader election algorithm starts and sensing nodes will be running for leadership (group coordinator); this process is described in Algorithm 2. For this election, all sensing nodes are eligible. If this is the first event, the leader node will be the one that is closest to the sink node. Otherwise, the leader will be the node that is closest to an already established route (Algorithm 2, Lines 7 to 9). In the case of a tie, i.e., two or more concurrent nodes have the same

**Algorithm 1:** Building the hop tree

---

```

1 Node sink sends a broadcast of HCM messages with the value of HopToTree = 0;
  //  $R_u$  is the set of nodes that received the message HCM
2 foreach  $u \in R_u$  do
3   if HopToTree( $u$ ) > HopToTree(HCM) and FirstSending( $u$ ) then
4     |  $NextHop_u \leftarrow ID_{HCM}$ ;
5     |  $HopToTree_u \leftarrow HopToTree_{HCM} + 1$ ;
6     | // Node  $u$  updates the value of the ID field in the message HCM
7     |  $ID_{HCM} \leftarrow ID_u$ ;
8     | // Node  $u$  updates the value of the HopToTree field in the message HCM
9     |  $HopToTree_{HCM} \leftarrow HopToTree_u$ ;
10    | Node  $u$  sends a broadcast message of the HCM with the new values;
11    |  $FirstSending_u \leftarrow false$ ;
12  end
13 else
14   | Node  $u$  discards the received message HCM;
15 end

```

---

distance in hops to the sink (or to an established route), the node with the smallest ID maintains eligibility, as shown in Lines 11 to 13 of Algorithm 2. Another possibility is to use the energy level as a tiebreak criterion.

At the end of the election algorithm only one node in the group will be declared as the leader (Coordinator). The remaining nodes that detected the same event will be the Collaborators. The Coordinator gathers the information collected by the Collaborators and sends them to the sink. A key advantage of this algorithm is that all of the information gathered by the nodes sensing the same event will be aggregated at a single node (the Coordinator), which is more efficient than other aggregation mechanisms (e.g., opportunistic aggregation).

### 3.3 Routing Formation, Hop Tree Updates and Data Transmission

The elected group leader, as described in Algorithm 2, starts establishing the new route for the event dissemination. This process is described in Algorithm 3, (Lines 2 to 10). For that, the Coordinator sends a route establishment message to its `NextHop` node. When the `NextHop` node receives a route establishment message, it re-transmits the message to its `NextHop` and starts the hop tree updating process. These steps are repeated until either the sink is reached or a node that is part of an already established

**Algorithm 2:** Cluster formation and leader election

---

```

1 Input:  $S$  // Set of nodes that detected the event
2 Output:  $u$  // A node of the set  $S$  is elected leader of the group
3 foreach  $u \in S$  do
4    $role_u \leftarrow$  coordinator;
   // Node  $u$  sends message MCC in broadcast
5   Announcement of event detection ;
   //  $N_u$  is the set of neighbors of node  $u \in S$ 
6   foreach  $w \in N_u$  do
7     if  $HopToTree(u) > HopToTree(w)$  then
8        $role_u \leftarrow$  collaborator ;
9       Node  $u$  retransmits the MCC message received from node  $w$  ;
10    end
11    else if  $HopToTree(u) = HopToTree(w) \wedge ID(u) > ID(w)$  then
12       $role_u \leftarrow$  collaborator ;
13      Node  $u$  retransmits the MCC message received from node  $w$ ;
14    end
15    else
16      Node  $u$  discards the MCC message received from  $w$ ;
17    end
18  end
19 end

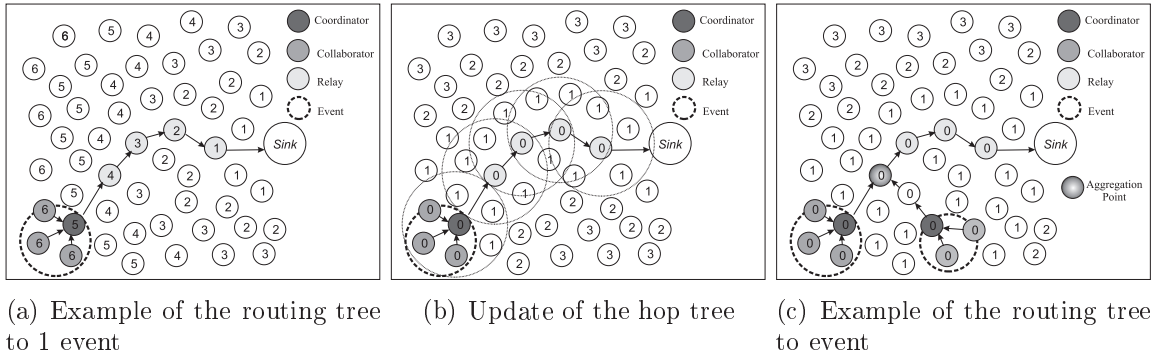
```

---

route is found. The routes are created by choosing the best neighbor at each hop. The choices for the best neighbor are twofold: (i) when the first event occurs, the node that leads to the shortest path to the sink is chosen (Figure 3.1(a)); and (ii) after the occurrence of subsequent events, the best neighbor is the one that leads to the closest node that is already part of an established route (Figure 3.1(c)). This process tends to increase the aggregation points, ensuring that they occur as close as possible to the events.

The resulting route is a tree that connects the Coordinator nodes to the sink. When the route is established, the hop tree updating phase is started. The main goal of this phase is to update the `HopToTree` value of all nodes so they can take into consideration the newly established route. This is done by the new relay nodes that are part of an established route. These nodes send an HCM message (by means of a controlled flooding) for the hop updating (Figure 3.1(b)). The whole cost of this process is less than a flooding, i.e. only the set of nodes inside the scope-limited flooding for the event will send one packet for the hop updating. This algorithm for the hop updating follows the same principles of the hop tree building algorithm, described in Section 3.1.

The data transmission performed by DAARP uses aggregation techniques that



**Figure 3.1.** Example of establishing new routes and updating the hop tree

---

**Algorithm 3:** Route formation, hop tree updates and data transmission

---

```

1 Leader node  $v$  of the new event sends a message REM to its  $NextHop_v$  ;
2 repeat
    //  $u$  is the node that received the REM message, that was sent by node  $v$ 
3   if  $u = Nextop_v$  then
4      $HopToTree_u \leftarrow 0$  ;
        // Node  $u$  is part of the new route built
5      $Role_u \leftarrow Relay$  ;
6     Node  $u$  sends the message REM to its  $NextHop_u$  ;
7     Node  $u$  broadcasts the message HCM with the value of  $HopToTree = 1$ ;
8     Nodes that receive the HCM message sent by node  $u$ , will run the
        command Line 2 until the Line 14 of Algorithm 1;
9   end
10 until Find out the sink node or a node belonging to the routing structure already
    established;
11 while The node has data to transmit/retransmit do
    //  $sons_u$  is the number of descendants of  $u$ 
12   if  $sons_u > 1$  then
13     | Aggregates all data and sends it to the  $nexthop_u$ ;
14   end
15   else
16     | Forwards the data to  $nexthop_u$ ;
17   end
18   Execute the mechanism of Section 3.4
19 end

```

---

are applied in three different contexts: 1) cluster inside opportunistic aggregation; 2) leader inside aggregation; and 3) cluster outside aggregation. When the routes overlap inside the cluster, the aggregation is performed by the collaborator nodes (cluster inside opportunistic aggregation). Furthermore, the leader node performs data aggregation and sends the results to the sink node (leader inside aggregation). Outside the cluster, aggregation is performed by the relay nodes when two or more events overlap along

routing (cluster outside aggregation).

The process of data transmission is described in Algorithm 3 (Lines 11 to 19). While the node has data to transmit, it verifies whether it has more than one descendant that relays its data (Line 12 of Algorithm 3). If it is the case, it waits for a period of time and aggregates all data received and sends the aggregated data to its `NextHop` (Line 13 of Algorithm 3). Otherwise, it forwards the data to its `NextHop`. For every packet transmission with aggregated data, the Route Repair Mechanism is executed as shown in Line 18 of Algorithm 3. A route repair mechanism is used to send information in a reliable way. Sender nodes wait a pre-defined time period to receive a packet delivery confirmation. When the confirmation is not received by the sender node, a new destination node is selected and the message is retransmitted by that node. This route repair mechanism (Line 18 of Algorithm 3) is described in Section 3.4.

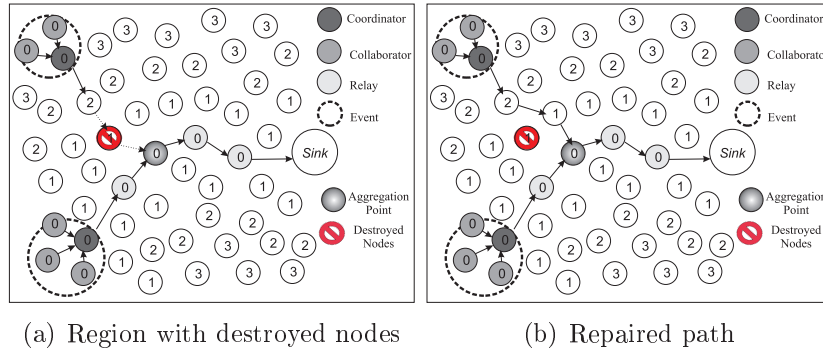
### 3.4 Route Repair Mechanism

The route created to send the data toward the sink node is unique and efficient since it maximizes the points of aggregation and, consequently, the data aggregation. However, because this route is unique, any failure in one of its nodes will cause disruption, preventing the delivery of several gathered event data. Possible causes of failure include low energy, physical destruction, and communication blockage. Some fault tolerant algorithms for WSNs have been proposed in the literature. Some are based on periodic flooding mechanisms [Intanagonwiwat et al., 2000, Hill et al., 2000], and rooted at the sink, to repair broken paths and to discover new routes to forward traffic around faulty nodes. This mechanism is not satisfactory in terms of energy saving because it wastes a lot of energy with repairing messages. Furthermore, during the network flooding period, these algorithms are unable to route data around failed nodes, causing data losses.

Our DAARP algorithm offers a piggybacked, ACK-based, route repair mechanism, which consists of two parts: failure detection at the `NextHop` node, and selection of a new `NextHop`.

When a relay node needs to forward data to its `NextHop` node, it simply sends the data packet, sets a timeout, and waits for the re-transmission of the data packet by its `NextHop`. This re-transmission is also considered an ACK message. If the sender receives its ACK from the `NextHop` node, it can infer that the `NextHop` node is alive and, for now, everything is ok. However, if the sender node does not receive the ACK from the `NextHop` node within the pre-determined timeout, it considers this node as

offline and another one should be selected as the new `NextHop` node. For this, the sender chooses the neighbor with the lowest hop-to-tree level to be its new `NextHop`; in case of a tie, it chooses the neighbor with the highest energy level. After that, the sender updates its routing table to facilitate the forwarding of subsequent packets. As an example, a disrupted route is shown in Figure 3.2(a). After the repairing mechanism is applied, a newly partial reconstructed path is created as depicted in Figure 3.2(b).



**Figure 3.2.** Example of path repair

## 3.5 Complexity Analysis

In this section we derive the communication cost bounds for DAARP, InFRA, and SPT algorithms (briefly described in Section 2.2.1). These two algorithms were chosen for being well known in the literature and have the same goals that the proposed DAARP algorithm. More specifically, we present the limits for the communication cost of these algorithms to create the routing structure. We also present the best and worst cases, while the average case will be shown in the simulation results (see Section 3.6).

In the SPT algorithm, there is linear communication cost to build the routing infrastructure. In a reactive fashion operation, it is necessary one flooding started by the nodes that sensed the first event in order to build the routing tree. One more flooding, initiated by the sink node, is also necessary for the other nodes to set up their ancestors in the tree infrastructure. Hence, the constant communication cost for SPT is  $2n$ , where  $n$  is number of nodes.

The InFRA algorithm also presents constant communication costs since it needs one flooding for every elected cluster head, followed by one flooding initiated by the sink node. Each flooding is necessary to set up and update the aggregated coordinators-distance at each node. Also, it is necessary  $m$  transmissions to create the cluster, where  $m$  is number of transmissions to create the clusters. For this reason, InFRA presents

a constant communication cost of  $(2kn + m)$ , where  $n$  is number of nodes,  $k$  is number of events. The overhead of the InFRA algorithm can be reduced in some situations by forcing a delay before each announcement of the cluster-heads is sent by the sink node. Thus, if successive events take place almost simultaneously, only one flooding starting at the sink will be necessary and the communication cost will be  $2n + (k - 1)n + m$ .

Our proposed DAARP algorithm needs one flooding from the first elected cluster-head and one more flooding initiated by the sink to building the hop tree. Successive cluster-heads will make a scope-limited flooding to update the nodes `HopToTree` parameter. Thus, the best case scenario for the communication cost is when successive events take place near the previously established routing tree. The closer the event takes place, the lower the communication cost is; thus, the best case will be achieved when the events happen on the routing tree. In this case, each CH is already attached to the tree. The number of transmissions to establish the initial routing tree is  $2n$  plus  $m$  transmissions to create the cluster, i.e., the cost is  $(2n + m)$ . The worst case of the DAARP algorithm happens when successive events take place far from the previously created tree. In this case, the number of transmissions to build the initial tree is  $2n$  plus  $(k - 1)n - \sum_{i=2}^k |U_i|$  transmissions for the following events plus  $m$  messages to create the cluster, where  $n$  is number of nodes,  $k$  is number of events,  $m$  is number of transmissions to create the clusters and  $|U_i|$  is the cardinality of the set of nodes outside the scope-limited flooding for the event  $i$ , which will not update their `HopToTree` for this event. Thus, the worst case for the DAARP algorithm is  $(2n + ((k - 1)n - \sum_{i=2}^k |U_i|) + m)$ .

Table 3.1 presents the communication cost of the algorithms assessed in this work. DAARP requires more control messages compared to the SPT. However, SPT builds routing trees that are worse than the trees built by our DAARP algorithm, therefore this cost will be recovered by the higher quality of the created tree as we will show in the next section. Regarding to the InFRA algorithm, note that  $(k - 1)n - \sum_{i=2}^k |U_i|$  is much smaller than  $(k - 1)n$ .

**Table 3.1.** Communication complexity of assessed algorithms

Algorithm	Best Case	Worst Case
SPT	$2n$	$2n$
InFRA	$(2n + (k - 1)n + m)$	$(2n + (k - 1)n + m)$
DAARP	$(2n + m)$	$(2n + ((k - 1)n - \sum_{i=2}^k  U_i ) + m)$



## 3.6 Performance Evaluation

In this section, we evaluate the proposed DAARP algorithm and compare its performance to two other known routing protocols: the InFRA and SPT algorithms. These two algorithms were chosen for being well known in the literature and have the same goals that the proposed DAARP algorithm. Table 3.2 shows the basic characteristics of SPT, InFRA and DAARP algorithms. We evaluate the DAARP performance under the following metrics: (i) packet delivery rate; (ii) control overhead; (iii) efficiency (packets per processed data); (iv) routing tree cost; (v) loss of raw data; (vi) loss of aggregated data; and (vii) transmissions number.

**Table 3.2.** Summary of the basic characteristics of assessed algorithms.

Scheme	Route Structure	Objective	Aggregation Nodes	Overhead	Scalability	Drawback
SPT	Tree	Shortest-Path	Opportunistic	Low	High	Data redundancy and static routes
InFRA	Tree-based cluster	Maximize overlap routes	Clusterheads and intermediate nodes	Very High	Low	Low scalability and high cost
DAARP	Tree-based cluster	Maximize overlap routes	Clusterheads and intermediate nodes	Medium	Medium	Static routes

### 3.6.1 Methodology

The performance evaluation is achieved through simulations using the SinalGo version v.0.75.3 network simulator [Sinalgo, 2008]. In all results, curves represent average values, while error bars represent confidence intervals for 95% of confidence from 33 different instances (seeds). The default simulation parameters are presented in Table 3.3. For each simulation set, a parameter shown in Table 3.3 will be varied as described in the evaluated scenario. The first event starts at time 1000 s and all other events start at a uniformly distributed random time between the interval [1000, 3000] seconds. Also, these events occur at random positions. The area of the sensor field is considered as the relation  $\sqrt{n\pi r_c^2/21.7}$ , where  $n$  is number of nodes,  $r_c$  is the communication radius, and 21.7 is the network density. For each simulation in which the number of nodes is varied, the sensor field dimension is adjusted to maintain the node density to 21.7. Sensor nodes are randomly deployed.

To provide a lower bound to the packet transmissions, we used an aggregation function that receives  $p$  data packets and sends only a fixed size merged packet. However, any other aggregation function can be used to take advantage of DAARP features. This function is performed at the aggregation points whenever these nodes send a packet.

**Table 3.3.** Simulation parameters

Parameter	Value
Sink node	1 (top left)
Network size	1024
Communication radius (m)	80
# of events	3
Event radius (m)	50
Event duration ( <i>hours</i> )	3
Loss probability (%)	0
Simulation duration ( <i>hours</i> )	6
Notification interval ( <i>sec</i> )	60
Sensor field ( $m^2$ )	$974 \times 974$
Node density ( <i>node/m<sup>2</sup></i> )	21.7

The evaluated algorithms used periodic simple aggregation strategy [Younis et al., 2006] in which the aggregator nodes transmit periodically the received and aggregated information.

The following metrics were used for the performance evaluation:

- *Data packet delivery rate*: number of packets that reach the sink node. This metric indicates the quality of the routing tree built by the algorithms – the lower the packet delivery rate, the greater the aggregation rate of the built tree;
- *Control packet overhead*: number of control messages used to build the routing tree including the overhead to both create the clusters and set up all the routing parameters for each algorithm;
- *Efficiency*: packets per processed data. It is the rate between the total packets transmitted (data and control packets) and the number of data received by the sink;
- *Routing tree cost*: total number of edges in the routing tree structure built by the algorithm;
- *Loss of aggregated data*: number of aggregated data packets lost during the routing. In this metric, if a packet contains  $X$  aggregated packets and if this packet is lost, it is accounted the loss of  $X$  packets.
- *Number of Steiner nodes*: number of Steiner nodes in the routing structure, i.e., the number of relay nodes;

### 3.6.1.1 Number of Steiner nodes

Since the ideal aggregation is achieved when the information is routed through the minimal Steiner tree [Krishnamachari et al., 2002], in this section we evaluate the number of Steiner tree nodes (i.e., relay nodes) obtained after the construction of the routing tree structure for each of the evaluated algorithms. In this analysis, the network size is varied from 256 to 2048 sensor nodes; the density varied from 20 to 30; and the number of events were also varied from 1 to 6.

The obtained results are presented in Figure 3.3. We can clearly see that the number of Steiner nodes in the routing tree built by DAARP algorithm is lower than the ones obtained by the SPT and InFRA algorithms in all studied scenarios. When compared to the MST algorithm, DAARP algorithm results in a slightly larger number of Steiner nodes. However, the MST algorithm is not suitable for WSNs due to its high overhead (for building the routing structure) and large amount of required memory (to store the shortest paths to all terminals). We are only comparing our proposed approach to the MST algorithm because the number of Steiner nodes in the routing tree built by the MST algorithm is proved to be at most twice the optimum (i.e., minimum) Steiner tree [Takahashi, 1980].

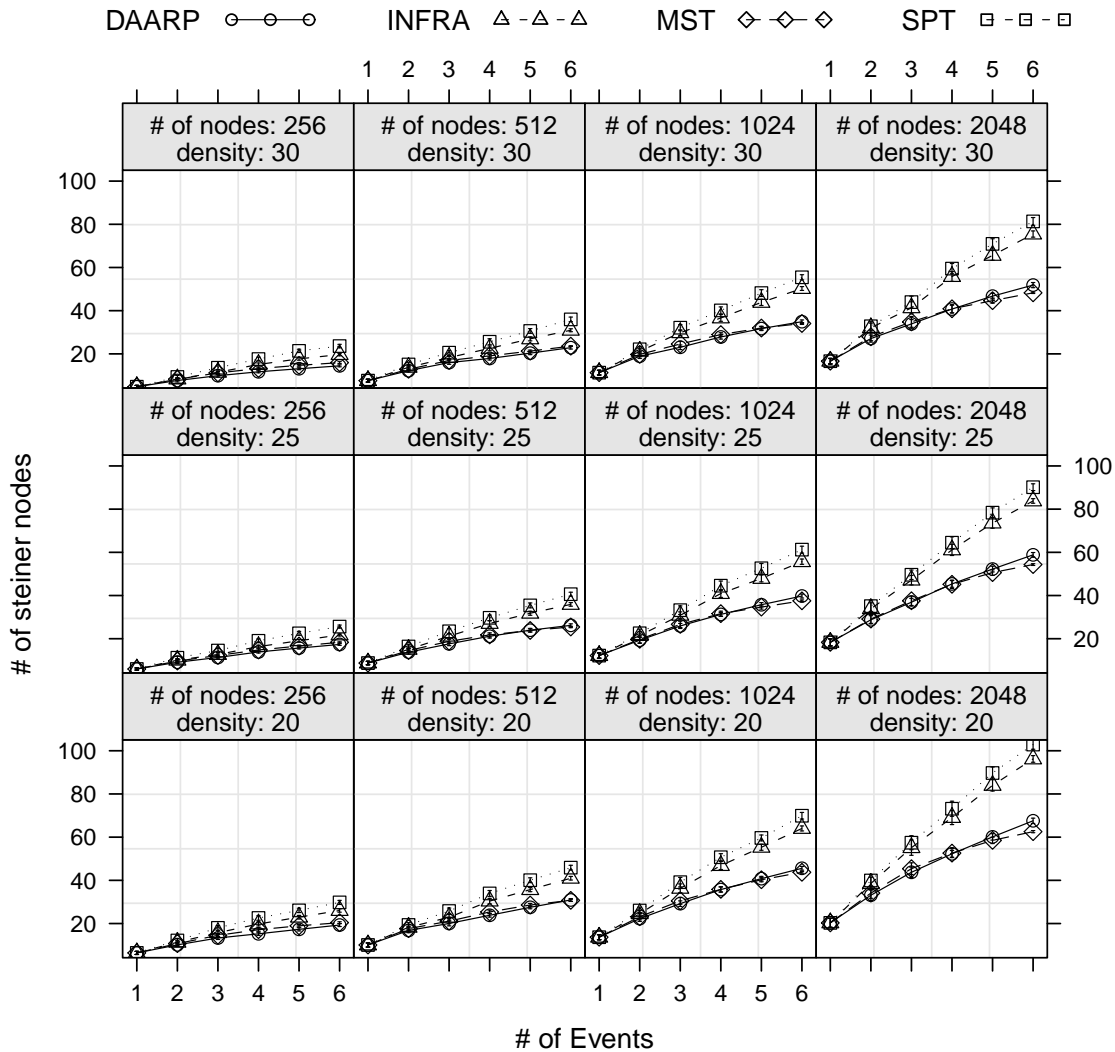
The good results obtained by the DAARP algorithm are due to its characteristic of prioritizing nodes that are closer to already existing routes. The InFRA algorithm, on the other hand, prioritizes the distance to the sink node, resulting in lower and/or later aggregations, which increases the number of Steiner nodes.

The Tables 3.4, 3.5, 3.6, and 3.7 show a different view of the results presented in Figure 3.3. We can see that in the average case, DAARP is very close to MST, of which, as already mentioned, cost is at most twice the optimal Steiner tree. These results also show that the proposed DAARP algorithm is scalable. For instance, Table 3.4 shows that, on average, the routing structure built by the InFRA algorithm has 35% more Steiner nodes than DAARP, while Table 3.6 shows that when increasing the number of nodes to 2048 this difference increases to 42%.

Finally, we can see that in all evaluated scenarios, the minimum routing structures created by the DAARP algorithm have fewer Steiner nodes than the minimum routing structures created by SPT, InFRA, and even the MST algorithm.

### 3.6.1.2 Impact of the Network Size

In this simulation scenario, the network size was varied from 128 to 1024 to evaluate the algorithms' scalability. Figure 3.4 presents the results. Since we are not evaluating the number of Steiner nodes, the MST algorithm is not included in the results. In



**Figure 3.3.** Number of Steiner nodes in the routing tree built by the DAARP, InFRA, MST, and SPT algorithms

**Table 3.4.** Scenario with 6 events, 256 nodes and density 20

Algorithm	Min	1 <sup>st</sup> Quart.	Median	Mean	3 <sup>rd</sup> Quart.	Max
DAARP	14	18	19	19.30	21	24
INFRA	21	25	26	26.03	28	30
MST	19	20	20	20.39	21	24
SPT	24	28	30	29.79	32	35

**Table 3.5.** Scenario with 6 events, 256 nodes and density 30

Algorithm	Min	1 <sup>st</sup> Quart.	Median	Mean	3 <sup>rd</sup> Quart.	Max
DAARP	12	14	14	14.45	15	17
INFRA	16	19	20	19.67	20	23
MST	13	15	16	15.94	17	18
SPT	20	22	23	23.61	25	29

**Table 3.6.** Scenario with 6 events, 2048 nodes and density 20

Algorithm	Min	1 <sup>st</sup> Quart.	Median	Mean	3 <sup>rd</sup> Quart.	Max
DAARP	59	66	68	67.58	70	74
INFRA	87	93	96	96.21	100	104
MST	60	61	62	62.55	64	65
SPT	90	98	104	102.90	106	116

**Table 3.7.** Scenario with 6 events, 2048 nodes and density 30

Algorithm	Min	1 <sup>st</sup> Quart.	Median	Mean	3 <sup>rd</sup> Quart.	Max
DAARP	44	50	52	51.85	54	58
INFRA	67	73	76	75.48	79	83
MST	47	48	48	48.27	49	50
SPT	72	78	82	81.33	85	92

Figure 3.4(a) we can see that our DAARP algorithm sends only 77% of the data packets sent by InFRA and about 65% of the data packets sent by SPT. This result clearly indicates that DAARP maintains the quality of the routing tree even when the number of nodes increases. Furthermore, Figure 3.4(b) shows that DAARP is more scalable than the InFRA algorithm since our algorithm needs 30% less control messages to build the routing structure. On the other hand, the DAARP algorithm requires, on average, 25% more control messages than the SPT algorithm. However, the routing trees built by SPT results in 30% less efficiency than the trees built by DAARP algorithm, as depicted in Figure 3.4(d). At last, Figure 3.4(c) shows that DAARP is 20% and 28%

more efficient than the InFRA and SPT algorithms, respectively. This occurs because DAARP algorithm needs less control messages to build the routing tree when compared to InFRA. Also, the routing tree built by DAARP has a better data aggregation quality than InFRA and SPT, as shown in Figure 3.4(d).

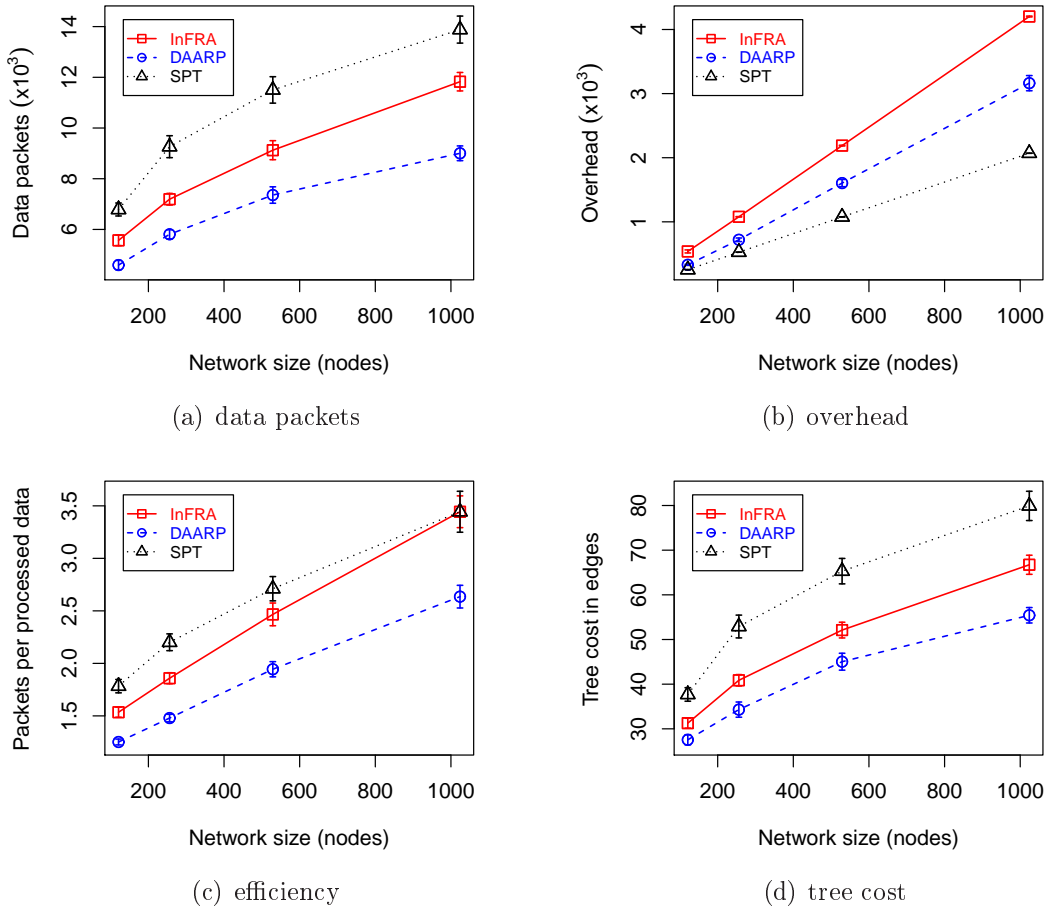
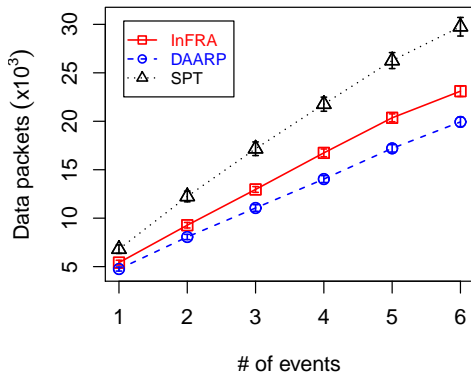


Figure 3.4. Impact of the Network Size

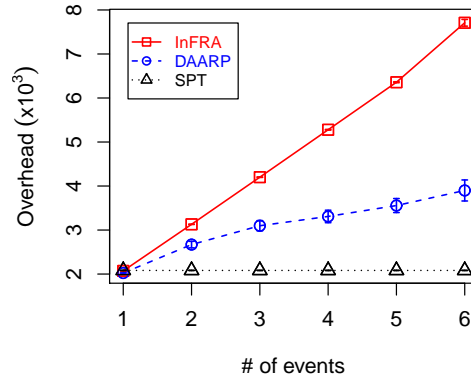
### 3.6.1.3 Impact of the Number of Events

In this simulation scenario, the number of events was varied to evaluate the behavior of the proposed algorithm in networks with 1, 2, 3, 4, 5, and 6 events occurring simultaneously. The results are presented in Figure 3.5. As depicted in Figure 3.5(a), DAARP sends less data packets than the InFRA and SPT algorithms. For instance, DAARP sends approximately 81% and 67% of the data packets sent by InFRA and SPT, respectively. This result indicates one of the main advantages of our DAARP algorithm:

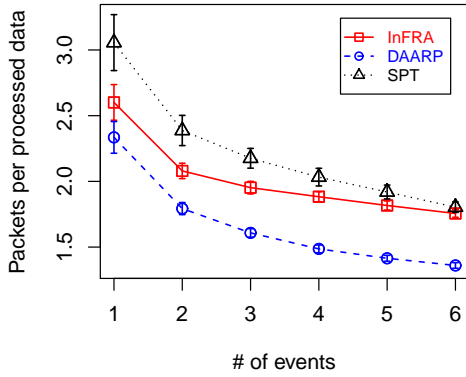
by varying the number of events, DAARP builds routing trees more likely to have higher data aggregation rates. Also, Figure 3.5(b) shows that DAARP needs only 50% of the control messages used by InFRA in the occurrence of 6 events and, on average, only 29% of the control messages used by InFRA to build the routing structure. Thus, for more than one event, DAARP is more efficient than SPT and InFRA, as shown in Figure 3.5(c). Finally, the cost of the routing tree built by DAARP is 10% smaller than in the InFRA algorithm, and 30% smaller than in the SPT, as we can see in Figure 3.5(d).



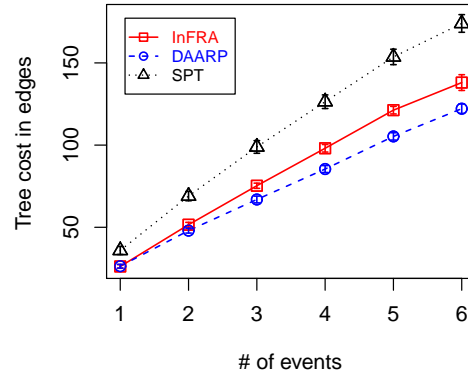
(a) data packets



(b) overhead



(c) efficiency



(d) tree cost

**Figure 3.5.** Impact of the Number of Events

### 3.6.1.4 Impact of the Event Duration

In this simulation scenario, the event duration was varied from 1 to 5 hours. The results are presented in Figure 3.6. As we can see in Figure 3.6(a), our proposed

DAARP algorithm sends less data packets than the other evaluated algorithms. More specifically, DAARP sends approximately 84% and 64% of the data packets sent by InFRA, and SPT respectively. This indicates that by varying the time of an event duration, DAARP obtains a data aggregation rate greater than InFRA and SPT. Also, Figure 3.6(b) shows that DAARP requires less control messages to create the routing structure than InFRA but it requires more control messages than the SPT algorithm. Although DAARP requires 33% more control messages than SPT, SPT does not build a good data aggregation routing infrastructure, as shown in previous results. At last, Figure 3.6(c) shows that DAARP is more efficient than InFRA and SPT. Our proposed algorithm outperforms the other evaluated algorithms even in scenarios of short-term events while InFRA exceeds the SPT only in scenarios where the event duration is longer (typically more than 2 hours).

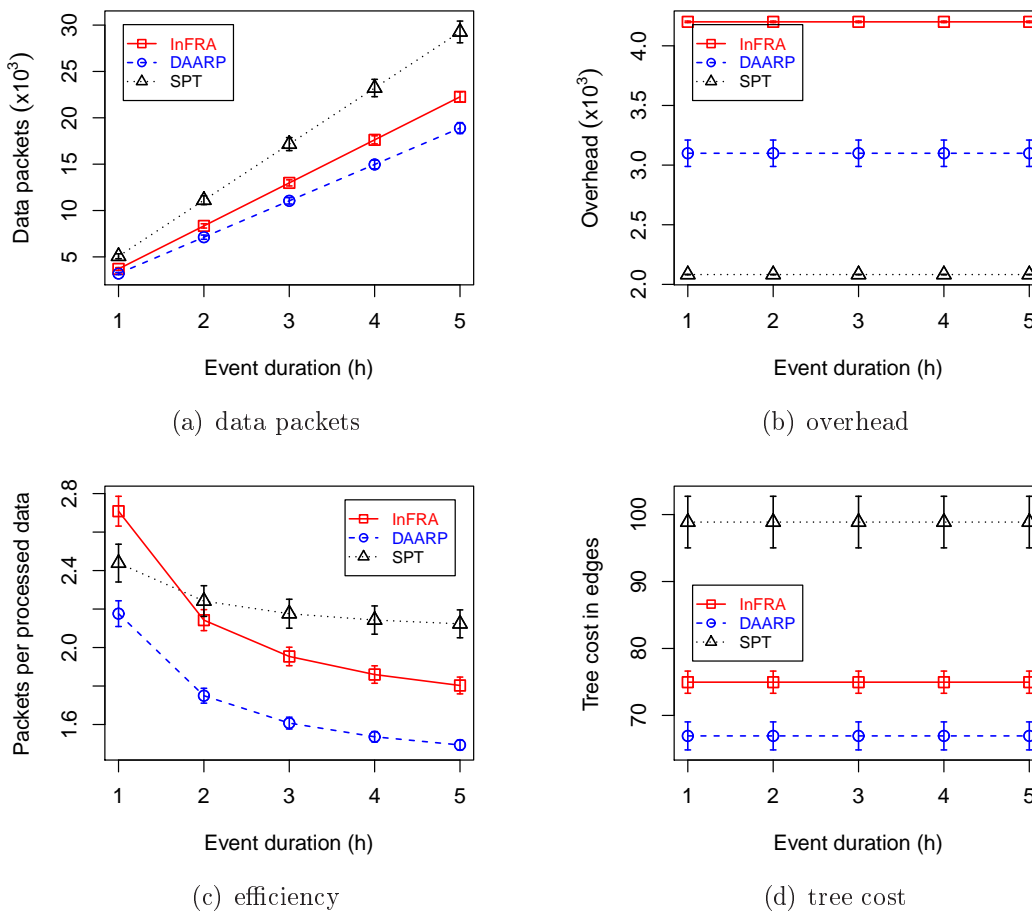


Figure 3.6. Impact of the Event Duration



### 3.6.1.5 Impact of Communication Failures

In this scenario, we evaluate the reliability of our proposed DAARP algorithm. For this, the communication failure probability parameter was varied from 0% to 20%. The results are presented in Figure 3.7. This simulation also aims to evaluate the cost of DAARP path repair mechanism. As we can see in Figure 3.7(a), in the DAARP algorithm, data packet transmission increases when the probability of communication failure increases. This is due to the fact that lost packets with aggregated data are retransmitted. On the other hand, SPT and InFRA protocols send less data packets when the communication failures probability increases. This happens because when a packet is lost due to communication failures the packets are not retransmitted and do not reach the sink, as shown in Figure 3.7(b). In this last figure, we can also see that in a scenario with 20% communication failure, the delivery rate of InFRA is only 30%, while DAARP delivers all aggregated data that have been sent. In summary, DAARP delivers aggregated data reliably with the best performance when compared to SPT and InFRA, as shown in Figure 3.7(c).

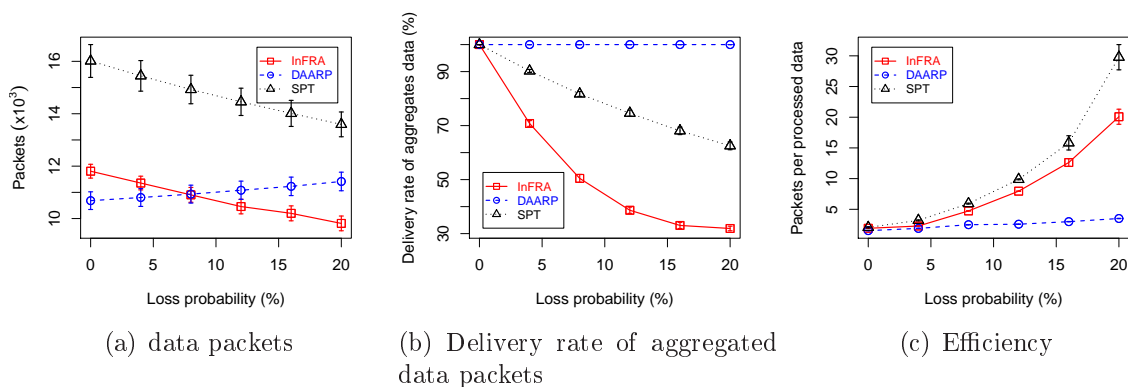


Figure 3.7. Impact of Communication Failures

The results presented in this section clearly show that our proposed DAARP algorithm is more scalable than InFRA and SPT in all considered scenarios in terms of network size, number of events, event duration time and communication failure probability.

## 3.7 Final Remarks on DAARP

Aggregation aware routing algorithms play an important role in event based WSNs. In this section we presented the DAARP algorithm, a novel and reliable Data Aggregation

Aware Routing Protocol for WSNs. Our proposed DAARP algorithm was extensively compared to two other known routing algorithms, the InFRA and SPT, regarding scalability, communication costs, delivery efficiency, aggregation rate and aggregated data delivery rate. By maximizing the aggregation points and offering a fault tolerant mechanism to improve delivery rate, the obtained results clearly show that DAARP outperformed the InFRA and SPT algorithms for all evaluated scenarios. Also, we show that our proposed algorithm has some key aspects required by WSNs aggregation aware routing algorithms such as a reduced number of messages for setting up a routing tree, maximized number of overlapping routes, high aggregation rate, and reliable data aggregation and transmission.

Despite DAARP have shown most efficient that the solutions compared (InFRA and SPT), the routes created by DAARP are kept fixed during the occurrence of the event. In scenarios with long-lasting events, nodes belonging the routes have exhausted their energy faster than other network nodes. Moreover, the quality of the routing structure is dependent on the order of event occurrence. Because of these limitations, there was the motivation for the specification and propose a new Dynamic Data-Aggregation Aware Routing Protocol (DDAARP) to modify routes when necessary, the DDAARP is described in Chapter 4.

## Chapter 4

# DDAARP: Dynamic Data Aggregation Aware Routing Protocol

The construction of routing structures aware of data aggregation has a considerable communication cost and solutions from the literature are not effective in scenarios with short-term events. The proposed DDAARP protocol builds dynamic routes that improves the cost and quality of the routing structure. It also reduces the number of communications necessary to configure the routing structure, maximizes the number of overlapping routes, selects routes with a high rate of aggregation and performs reliable transmission of aggregate data.

The DDAARP differs from the DAARP (presented in the Chapter 3) at least in three aspects:

- The quality of the routing structure created does not depend on the order of occurrence of events;
- The routes created by DDAARP are not kept fixed throughout the duration of events. In addition, routes may change when necessary;
- The DDAARP use the sink node for processing and configuration of the routes.

DDAARP is performed in four phases. Phase 1 builds the hop tree from the sensor nodes to the sink, collects and delivers information about the nodes' positions to the sink node. The sink node starts building the hop tree that will be used in Phase 3 by Coordinators to notify the sink on the occurrence of the event and request information about route for data transmission. Phase 2 consists of the cluster formation and the

election of a cluster-head among the nodes that detected the occurrence of a new event in the network. Phase 3 is responsible for setting up the new route for delivering data packets. Finally, Phase 4 is responsible for sending the collected data to the sink node in a reliable way.

## 4.1 Building the Hop Tree and Gathering Information About Nodes' Position

In Phase 1, the distance from the sink to each node is computed in hops. This is done from the sink, which sends by means of flooding, a Hop Configuration Message (HCM) for the hops configuration. The HCM message contains two fields: `ID` and `HopToSink`, where `ID` is the node identifier that started/retransmitted the HCM message and `HopToSink` is the distance, in hops, by which an HCM message has passed. The `HopToSink` parameter is started with value 0 at the sink. In this phase, sensor nodes compute the hop count to the sink node and stores in `HopToSink` variable the smallest distance (in hops) to the sink and stores in the `NextHop` variable which neighbor will be used to route requests and notifications of events occurrence to the sink node.

This process is described in Algorithm 4. The sink node floods the network with a HCM message, sets the `HopToSink` value to 0, and forwards it to its neighbors (at the beginning, all nodes set the distance to the sink as infinity). Each node, upon receiving the message HCM, verifies if the value of `HopToSink` in the HCM message is less than the value of `HopToSink` that it has stored, as shown in Algorithm 4 - Line 3. If that condition is true then the node updates the value of the `NextHop` variable with the value of the field `ID` of message HCM, as well as the value of the `HopToSink` variable, and the values in the fields `ID` and `HopToSink` of the HCM message. The node also relays the HCM message, as shown in Algorithm 4 - Line 8. Otherwise, the node verifies if the value of `HopToSink` in the HCM message is bigger than the value of `HopToSink` that it has stored, as shown in Algorithm 4 - Line 10. If that condition is true then the node stores the ID of the node that sent the HCM message in its list of neighbors with the higher hop level and discards the received message, as shown in Algorithm 4 - Line 11 and 12 respectively. If it is not the case, then the node discards the received HCM message, as shown in Algorithm 4 - Line 15. The steps described above occur repeatedly until the whole network is configured.

Algorithm 5 shows how information about nodes' position is gathered from the network and then delivered to the sink node. Border nodes are responsible for starting

**Algorithm 4:** Building the hop tree

---

```

1 Node sink sends a broadcast of HCM messages with the value of HopToSink = 0;
  //  $R_u$  is the set of nodes that received the message HCM
2 foreach  $u \in R_u$  do
3   if HopToSink( $u$ ) > HopToSink(HCM) then
4     |  $NextHop_u \leftarrow ID_{HCM}$ ;
5     |  $HopToSink_u \leftarrow HopToSink_{HCM} + 1$ ;
6     | // Node  $u$  updates the value of the ID field in the message HCM
7     |  $ID_{HCM} \leftarrow ID_u$ ;
8     | // Node  $u$  updates the value of the HopToSink field in the message HCM
9     |  $HopToSink_{HCM} \leftarrow HopToSink_u$ ;
10    | Node  $u$  sends a broadcast message of the HCM with the new values;
11  end
12 else if HopToSink(HCM) > HopToSink( $u$ ) then
13   |  $NeighborHopBigger.add(ID_{HCM})$ ;
14   | Node  $u$  discards the received message HCM;
15 end
16 else
17   | Node  $u$  discards the received message HCM;
18 end
19 end

```

---

the process of collecting information about the node's position, i.e., a border node does not have any neighbor with higher hop level than itself. Border nodes transmit to their neighbors of lower hop level a Collecting Information Message (CIM) as shown in Line 1 of Algorithm 5.

When a sensor receives a CIM message from its neighbors of higher hop level, as shown in Line 3 of Algorithm 5, it adds the ID of the node that transmitted the CIM message in the list `ReceivedNeighborHopBigger` and the CIM message is assembled with the received information as shown Lines 4 and 5 of Algorithm 5. When the node receives information from all neighbors of higher hop level and is not the sink as shown in Lines 6 and 10 of Algorithm 5, it broadcasts its CIM message as shown Line 11 of Algorithm 5. If the node is the sink (Line 7 of Algorithm 5) then it creates the adjacency matrix with collected information and puts this adjacency matrix, which represents the network topology as shown in Line 8 of Algorithm 5, to the message CIM.

---

**Algorithm 5:** Collecting information about nodes' position

---

```

1 Border nodes transmit to their neighbors of lower hop level a Collecting Information
  Message (CIM) //  $R_u$  is the set of nodes that received the message CIM
2 foreach  $u \in R_u$  do
3   if NeighborHopBigger.contains(IDCIM) then
4     ReceivedNeighborHopBigger.add(IDCIM) ;
5     UpdatesCIM ;
6     if NeighborHopBigger = ReceivedNeighborHopBigger then
7       if Roleu = Sink then
8         | Create adjacency matrix in the Sink ;
9       end
10      else
11        | Node  $u$  sends message CIM in broadcast;
12      end
13    end
14    else
15      | Node  $u$  discards the received message CIM;
16    end
17  end
18  else
19    | Node  $u$  discards the received message CIM;
20  end
21 end

```

---

## 4.2 Cluster Formation and Leader Election

When an event is detected by one or more nodes, the leader election algorithm starts and sensing nodes will be running for leadership (group coordinator); this process is described in Algorithm 6. For this election, all sensing nodes are eligible. However, the group leader is the node that is closest to the sink. (Algorithm 6, Lines 7 to 9). In the case of a tie, i.e., two or more concurrent nodes have the same distance in hops to the sink, the node with the smallest ID maintains eligibility, as shown in Lines 11 to 13 of Algorithm 6. Another possibility is to use the energy level as a tiebreak criterion.

At the end of the election algorithm only one node in the group will be declared as the leader (Coordinator). The remaining nodes that detected the same event will be the Collaborators. The Coordinator gathers the information collected by the Collaborators and sends them to the sink. A key advantage of this algorithm is that all of the information gathered by the nodes sensing the same event will be aggregated at a single node (the Coordinator), which is more efficient than other aggregation mechanisms (e.g., opportunistic aggregation).

**Algorithm 6:** Cluster formation and leader election

---

```

1 Input:  $S$  // Set of nodes that detected the event
2 Output:  $u$  // A node of the set  $S$  is elected leader of the group
3 foreach  $u \in S$  do
4    $role_u \leftarrow coordinator$ ;
   // Node  $u$  sends message MCC in broadcast
5   Announcement of event detection ;
   //  $\mathcal{N}_u$  is the set of neighbors of node  $u \in S$ 
6   foreach  $w \in \mathcal{N}_u$  do
7     if  $HopToSink(u) > HopToSink(w)$  then
8        $role_u \leftarrow collaborator$  ;
       Node  $u$  retransmits the MCC message received from node  $w$  ;
9     end
10    else if  $HopToSink(u) = HopToSink(w) \wedge ID(u) > ID(w)$  then
11       $role_u \leftarrow collaborator$  ;
       Node  $u$  retransmits the MCC message received from node  $w$ ;
12    end
13    else
14      Node  $u$  discards the MCC message received from  $w$ ;
15    end
16  end
17 end
18 end
19 end

```

---

### 4.3 Routing Formation

The elected group leader described in Algorithm 6 notifies the sink on the occurrence of the event and requests routing information for data transmission. The route used for notification and request is the route created in Phase 1.

The Coordinator starts establishing the new route for the event dissemination. This process is described in Algorithm 7. The Coordinator sends a Route Establishment Message (REM) to its NextHop. When the destination node receives a REM message, it retransmits it to its NextHop. These steps are repeated until the sink is reached as shown in Lines 2 to 4 of Algorithm 7.

When the sink receives the message REM, it adds the Coordinator ID that notified the occurrence of the new event on the network and runs the route selection algorithm as shown in Line 5 of Algorithm 7. This algorithm processes the matrix that represents the network topology, evaluating the cost in terms of relay nodes for each path. The routes selected are routes that insert a smaller number of relay nodes. At the end of the route selection algorithm, a packet is generated with information on the routes that maximize data aggregation. The sink then sends a single message, Route Establishment Message Back (REM-Back), to the sensors that will be included in the routing infrastructure.

When the sensor nodes receive the REM-Back message, they set-up their routing tables updating in the `NextHop` variable and sending REM-Back messages to next ID in the REM-Back as shown in Lines 10 and 11 of Algorithm 7. The resulting route is a tree that connects all source nodes to the sink node.

---

**Algorithm 7:** Routing formation
 

---

```

1 Coordinator sends a Route Establishment Message (REM) to its NextHop ;
2 repeat
   | // u is the node that received the message REM
3   | u forwards the Message (REM) to its NextHop
4 until  $Role_u = Sink$ ;
5 u run the algorithm for route selection ;
6 u creates a message REM-Back ;
7 u sends the message REM-Back ;
8 repeat
   | // w is the node that received the message REM-Back
9   |  $NextHop_w \leftarrow ID_{REM-Back}$ ;
10  | Remove  $ID_{REM-Back}$  ;
11  | w send the message to the next  $ID_{REM-Back}$ ;
12 until  $REM-Back = Empty$ ;

```

---

## 4.4 Data Transmission

The data transmission performed by DDAARP uses aggregation techniques that are applied to two different contexts: inside and outside the cluster. When the routes overlap inside the cluster, the aggregation is performed by the Collaborator nodes. Furthermore, the leader node performs data aggregation and sends the results to the sink node. Outside the cluster, aggregation is performed by the relay nodes when two or more events overlap along routing.

The process of data transmission is described in Algorithm 8 (Lines 1 to 9). While the node has data to transmit, it verifies whether it has more than one descendant that relays its data (Line 2 of Algorithm 8). If it is the case, it waits for a period of time and aggregates all data received and sends the aggregated data to its `NextHop` (Line 3 of Algorithm 8). Otherwise, it forwards the data to its `NextHop`. For every packet transmission with aggregated data, the Route Repair Mechanism is executed as shown in Line 8 of Algorithm 8. A route repair mechanism is used to send information in a reliable way. Sender nodes wait a pre-defined time period to receive a packet delivery confirmation. When the confirmation is not received by the sender node, a



new destination node is selected and the message is retransmitted by that node. This route repair mechanism (Line 18 of Algorithm 3) is described in Section 3.4.

---

**Algorithm 8:** Data Transmissions
 

---

```

1 while The node has data to transmit/retransmit do
  | // sonsu is the number of descendants of u
2   if sonsu > 1 then
3     | Aggregates all data and sends it to the nexthopu;
4   end
5   else
6     | Send data to nexthopu;
7   end
8   Execute the Repair mechanism presented in Section 3.4;
9 end

```

---

## 4.5 Complexity Analysis

In this section, we derive the communication cost bounds of each algorithm assessed in this work. Here, we show the best and the worst case whereas the average case was estimated by simulation (see Section 4.6.2.1).

The InFRA algorithm presents  $(2n + (k - 1)n) + m$  linear communication cost as shown in Section 3.5. The best case for the communication cost of DAARP algorithm is  $(2n + m)$ , and the worst case for the DAARP algorithm is  $(2n + ((k - 1)n - \sum_{i=2}^k u_i) + m)$ , as shown in Section 3.5.

The DDAARP algorithm will demand one flooding initiated by the sink to create the initial tree infrastructure that will be used by the cluster-heads to request routes, and another flooding initiated by the border nodes to collect information about nodes' position, which requires  $2n$  transmissions. It is necessary  $m$  transmissions to create the clusters. For each elected Coordinator, it is necessary to notify the sink on the occurrence of the event and request information about route for data transmission. This requires  $\sum_{i=1}^k 2(h_i)$  transmissions, where  $h_i$  is the distance in hops from the Coordinator  $i$  to the sink node and  $k$  is number of events. Thus, the worst and best cases for the DDAARP algorithm are  $(2n + \sum_{i=1}^k 2(h_i) + m)$ .

Table 5.1 presents the communication cost of the algorithms assessed in this work. Note that  $\sum_{i=1}^k 2(h_i)$  is much smaller than  $(k - 1)n$  and  $(k - 1)n - \sum_{i=2}^k u_i$ .

**Table 4.1.** Communication complexity of assessed algorithms

Algorithm	Best Case	Worst Case
InFRA	$(2n + (k - 1)n + m)$	$(2n + (k - 1)n + m)$
DAARP	$(2n + m)$	$(2n + ((k - 1)n - \sum_{i=2}^k  U_i ) + m)$
DDAARP	$(2n + \sum_{i=1}^k 2(h_i) + m)$	$(2n + \sum_{i=1}^k 2(h_i) + m)$

## 4.6 Performance Evaluation

The proposed solution in this work is compared with two other routing protocols: InFRA and DAARP. The main objective of this comparison is to evaluate the DDAARP performance under the following metrics: (i) complexity analysis, (ii) control overhead, (iii) efficiency (packets per processed data), and (iv) routing tree cost.

### 4.6.1 Simulation Scenario and Metrics Used

The simulation performed in this work evaluates the proposed algorithm in terms of number of nodes ( $n \in \{512, 1024, 2048 \text{ and } 4096\}$ ), number of events ( $ne \in \{1, 2, 3, 4, 5 \text{ and } 6\}$ ), duration of events ( $de \in \{1, 5, 15, 30, 45 \text{ and } 60\}$  minutes), round event ( $re \in \{1, 3, 6 \text{ and } 9\}$ ) and notification rate ( $nr \in \{1, 20, 40 \text{ and } 60\}$  per minutes). In all results, curves represent average values, while error bars represent confidence intervals for 95% of confidence from 33 different instances (seeds). The default scenario used for the simulations is shown in Table 4.2. For some simulations, a parameter shown in Table 4.2 will be varied and this is described in the evaluated scenario. The first event starts at 1000 s and all other events start at a uniformly distributed random time, where these events occur in random positions. SinalGo version *v.0.75.3* [Sinalgo, 2008] was the event simulator used. For each simulation in which the number of nodes is varied, the sensor field dimension is adjusted to maintain the node density to 20 the average degree of neighbors. We consider the area of the sensor field as the relation  $\sqrt{n\pi r_c^2}/20$ , where  $n$  is number of nodes and  $r_c$  is communication radius. Sensor nodes are randomly deployed.

To provide a lower bound to packet transmissions, a function was used that receives  $p$  data packets and sends only a fixed size merged packet. This function is run at the aggregation points whenever these nodes send a packet. Any other aggregation function can be used to take advantage of DDAARP features.

The evaluated algorithms use a periodic simple aggregation strategy [Younis et al., 2006]. In this strategy, aggregator nodes transmit periodically

**Table 4.2.** Simulation parameters

Parameter	Value
Sink node	1 (top left)
Number of Nodes	1024
Communication radius (m)	80
# of events	6
Round Event	1
Event radius (m)	50
Event duration ( <i>minutes</i> )	5
Simulation duration ( <i>hours</i> )	12
Notification rate (per <i>mimute</i> )	1
Density (average degree of neighbors)	20

the received and aggregated information. The following metrics were used for the evaluation:

- **Control overhead:** the overhead is the amount of control messages used to build the routing tree including the overhead to both create the clusters and set up all the routing parameters of each algorithm;
- **Amount of Steiner nodes:** it is the number of Steiner nodes included in the routing structure, i.e., the number of relay nodes that is part of the routing infrastructure;
- **Efficiency (packets per processed data):** it is the rate of total packets transmitted (data and control packets) and the amount of notifications to sink;

## 4.6.2 Simulation Results

### 4.6.2.1 Overhead

In this simulation scenario, the number of nodes, round event and the amount of events presented in Table 4.2 were varied to evaluate the algorithm behavior in networks with 512, 1024, and 2048 nodes; 1, 3, 6, and 9 round events; and 1, 2, 3, 4, 5, and 6 events occurring simultaneously. Figure 4.1 shows that the DDAARP is more scalable than DAARP and InFRA, since it needs fewer control messages to build the routing structure (average 48% less control message over DAARP and 69% in relation the InFRA).

The results in Figure 4.1 show that the DDAARP is scalable and consumes less energy to built and manage routing infrastructure than the other evaluated algorithms, since communication is typically one of the tasks that consumes more energy in a sensor

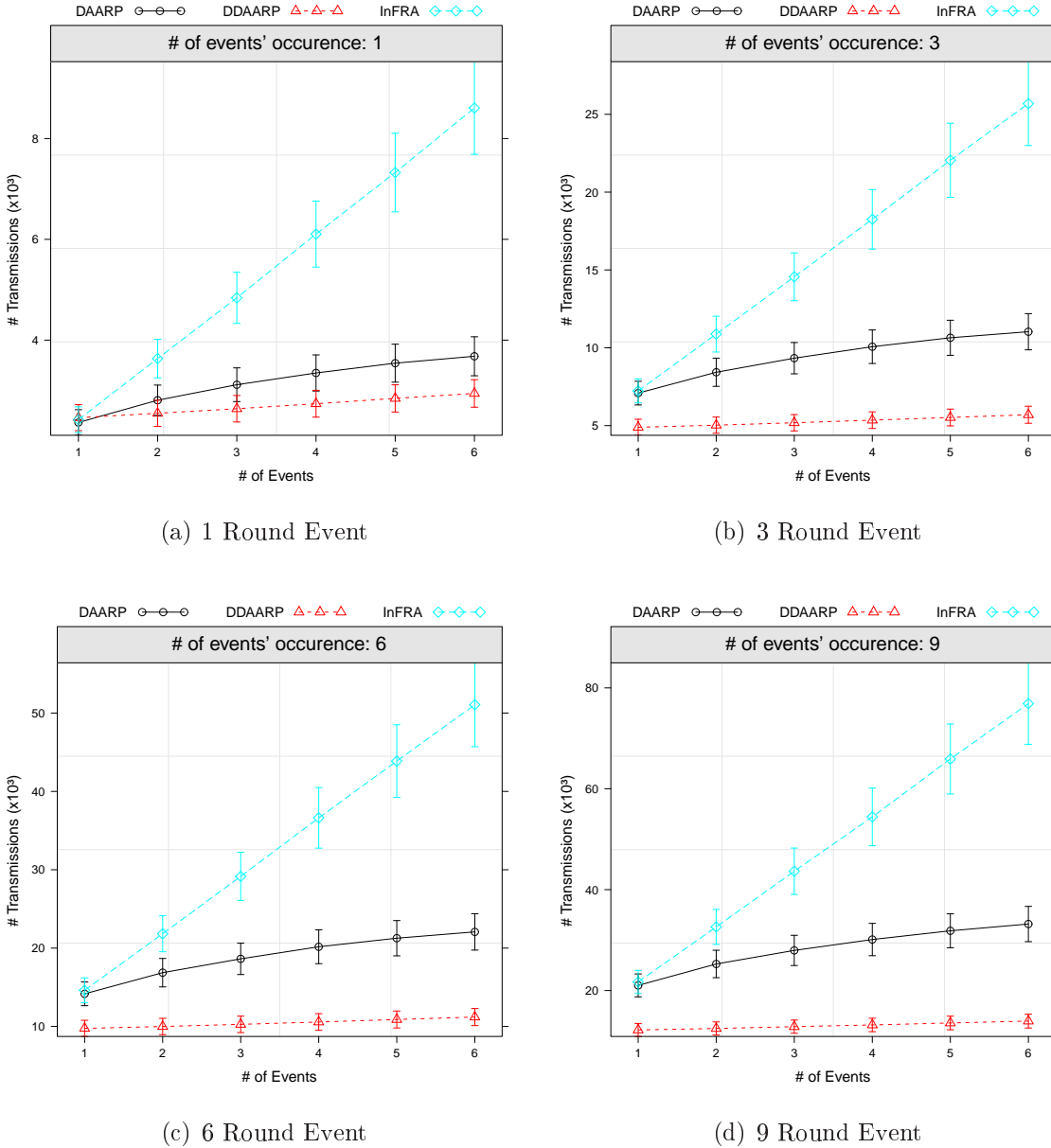


Figure 4.1. Overhead

node. Figure 4.1(d) shows that DDAARP needs only 42% of the control messages used by DAARP in the occurrence of 6 events and, only 18% of the control messages used by InFRA to build the routing structure.

As InFRA and DAARP need to reconfigure the values of `coordinators-distance` and `HopToTree`, respectively, each new round of events is necessary to reconfigure the network to update the values of `coordinators-distance` and `HopToTree`. In DDAARP, each new round of events collects the information of

the nodes' position to keep the most current state of the network topology in the sink.

The results in Figure 4.1 show that the DDAARP has a better performance for environments with dynamic occurrence of events.

#### 4.6.2.2 Efficiency

Figure 4.2 shows that on average DDAARP is 23% and 57% more efficient than DAARP and InFRA, respectively. For the evaluated scenarios DDAARP needs 1.98 packets per processed data whereas DAARP needs 2.58 packets and InFRA needs 4.56 packets. This occurs because DDAARP needs less control messages to build the routing tree compared with DAARP and InFRA as shown in Figure 4.1. Also, the routing tree built by DDAARP has the best quality of data aggregating compared with DAARP and InFRA, as shown in Figure 4.3.

For events of short duration and with low notification rate as shown in Figure 4.2(a), DDAARP needs 6.6 packets per processed data, DAARP needs 9.95, and InFRA needs 20.9. DDAARP presents the best performance, it makes a few data transmissions and has a low overhead.

For events of long duration and with high notification rate as shown in Figure 4.2(d), DDAARP needs 1.39 packets per processed data, DAARP needs 1.61, and InFRA needs 2.35.

For scenarios where the number of notifications is very high, i.e., much higher than the overhead, the tendency is to have a close efficiency of the algorithms. But DDAARP is still better because the amount of data transmitted is smaller, i.e., in all cases DDAARP builds the routing tree with less steiner nodes (see Figure 4.3), which results in fewer retransmissions.

#### 4.6.2.3 Amount of Steiner Nodes

In this analysis, the number of nodes and number of events were varied to evaluate the average case for routing tree cost of DDAARP compared with InFRA and DAARP. The routing tree cost is the total amount of Steiner nodes included in the tree built by the algorithms.

We analyzed scenarios with 512, 1024, 2048, and 4096 sensor nodes, and 1, 2, 3, 4, 5 and 6 number of events. The results are presented in Figure 4.3. The cost of the routing tree built by DDAARP is smaller than DAARP and InFRA for all cases, i.e., DDAARP includes less steiner nodes than the other solutions.

Figure 4.3 shows that in all assessed scenarios, DDAARP presents a performance better than InFRA and DAARP with a lower overhead (see Figure 4.1). In average,

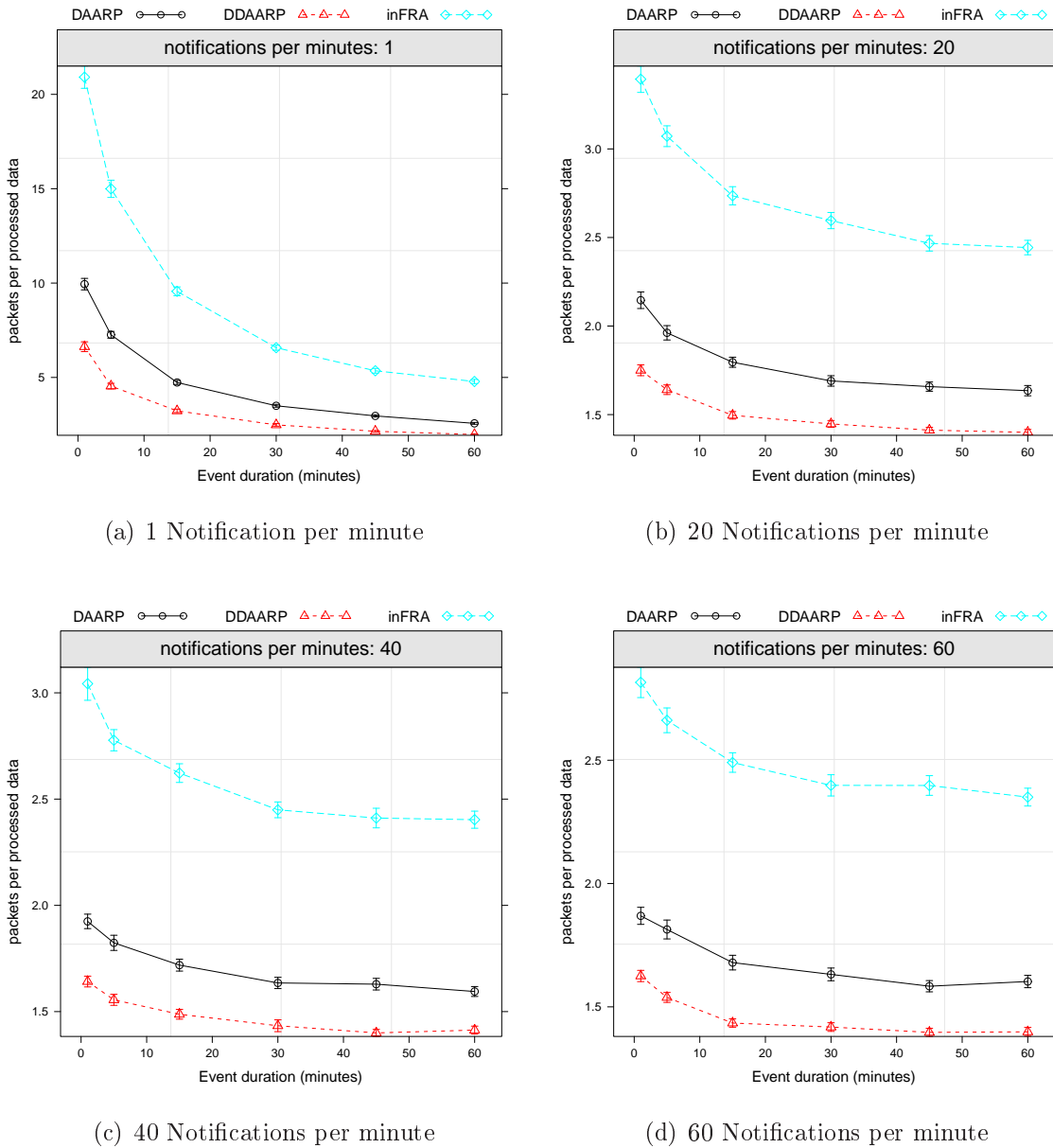


Figure 4.2. Efficiency

DDAARP needs 12% less steiner nodes than DAARP, and 52% than InFRA. Figure 4.3(d) shows that DDAARP includes 14% less steiner nodes than DAARP in the occurrence of 6 events and, only 70% less steiner nodes than InFRA.

DDAARP creates a routing tree with fewer steiner nodes than DAARP due to the fact the routes created by DDAARP do not depend on the order of occurrence of the events and routes already created can be reconstructed to improve the cost of the final routing tree. On the other hand, routes created by DAARP are kept fixed until

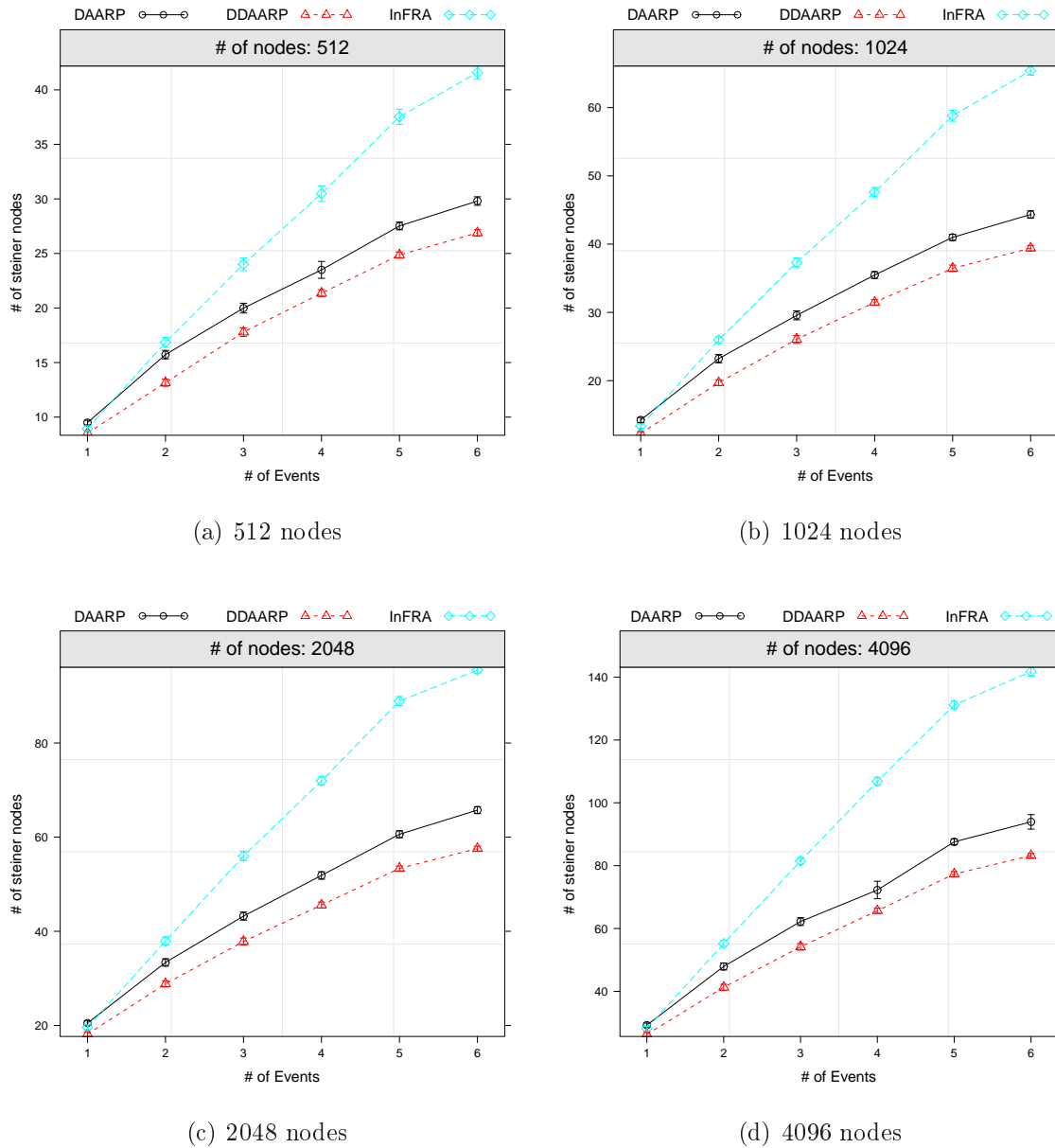


Figure 4.3. Number of Steiner nodes

the end of the event occurrence.

#### 4.6.2.4 Probability of Communication Failures

For this analysis, the results are not shown here because DDAARP use the route repair mechanism presented in Section 3.4 and the performance of DDAARP is identical to the results of DAARP presented in Section 3.6.1.5. In summary, DDAARP delivers aggregated data reliably with best performance compared with InFRA and with a

performance similar to DAARP.

## 4.7 Final Remarks on DDAARP

This chapter presented DDAARP, a Dynamic Data-Aggregation Aware Routing Protocol. DDAARP was extensively compared with two other routing solutions presented in the literature (InFRA and DAARP) regarding scalability, communication cost, delivery efficiency, and aggregation rate. By maximizing the aggregation points and offering a fault tolerant mechanism to improve the delivery rate, the results show that DDAARP outperformed those two protocols for all evaluations.

The DDAARP proved to be a potential solution, but the size of packets containing information about the positions of the nodes, see Section 4.1, tend to increase and this solution may be impractical for large-scale networks. In addition, the sink node must have a global knowledge of network. To overcome these challenges, we proposed the Dynamic Scalable Tree (DST), which will be described in Chapter 5.



# Chapter 5

## DST: Dynamic and Scalable Tree

The main idea of the proposed DST algorithm is to manage the energy consumption of the nodes that detected an event by eliminating redundant notifications. The proposed algorithm builds a routing tree using shortest routes (in Euclidean distance) that connect all coordinator nodes to the sink node while maximizing data aggregation and reducing distances connecting each coordinator node to the sink. The created routing tree does not depend on the order of the events. In the DST algorithm, the nodes can be classified according to their roles in the created routing infrastructure:

- *Collaborator*: A node that detects an event and reports the gathered data to a coordinator node;
- *Coordinator*: A node that is also detecting an event and is responsible for gathering all the data events sent by representative nodes, aggregating them and sending the result toward the sink node.
- *Aggregator*: A node that aggregates data from two or more sources and forwards the aggregated data. It might or might not be detecting an event.
- *Relay*: A node that forwards data toward the sink.
- *Sink*: A node interested in receiving data from a set of coordinator nodes.

The DST algorithm is performed in four phases. In Phase-1, presented in Section 5.1, sensor nodes store the sink's position as well as its neighbor's position. Phase-2, presented in Section 5.2, consists of cluster formation and the election of a Coordinator among the nodes that detects the occurrence of a new event. In Phase-3, presented in Section 5.3, when an event occurs, the Coordinator sends a packet to the sink node informing its position. The sink then notifies all other Coordinators of the

new Coordinator position. The sink also notifies the new Coordinator about the positions of Coordinators that already exist. Finally, Phase-4, presented in Section 5.4, is responsible for creating the routing tree connecting all Coordinators to the sink node and sending the collected data to the sink node.

The tree building (Phase 4) is an important part of the proposed algorithm in terms of energy savings. In this work, we present three different variations for the proposed DST tree creation algorithm. These variations aim at building a routing tree that connects all coordinators to the sink node according to different goals. The first variation, *Dynamic and Scalable Tree – Closest First* (DST-CF), aims at building a routing tree that prioritizes the cost (in Euclidean distance) of the routing tree. The second variation, *Dynamic and Scalable Tree – Farthest First* (DST-FF), aims at building a routing tree that tries to decrease the cost (in Euclidean distance) of high-cost routes (in Euclidean distance) of the tree. The third variation, *Dynamic and Scalable Tree – Best Combination* (DST-BC), aims at building a routing tree that has the lowest cost (in Euclidean distance). The difference between each approach lies on which straight line segments to choose during the creation of the routing structure (as detailed in Section 5.4).

## 5.1 Discovery of Neighbors' and Sink's Positions

This is the first phase of the DST algorithm. It is responsible for discovering and storing the neighbors' position and the sink's position. As shown in Algorithm 9, the sink node starts this phase by flooding a Configuration Message. The message contains three fields: `ID`, `CoordSender` and `CoordSink`, where `ID` is the node identifier that retransmits the message, `CoordSender` is the node's position  $(x_n, y_n)$  that relays the configuration message, and `CoordSink` is the sink's position  $(x_s, y_s)$ .

---

### Algorithm 9: Discovery of neighbors' position

---

```

1 Sink node broadcasts a Configuration Message;
  //  $R_u$  is the set of nodes that received the Configuration Message
2 foreach  $u \in R_u$  do
3    $neighborhood(u).ID \leftarrow ID_{Message}$ ;
4    $neighborhood(u).CoordSender \leftarrow CoordSender_{Message}$ ;
5    $neighborhood(u).CoordSink \leftarrow CoordSink_{Message}$ ;
6   if Node( $u$ ) did not transmit its position then
7     // Node  $u$  updates the value of the ID in Configuration Message
       $ID_{Message} \leftarrow ID_u$ ;
      // Node  $u$  updates the value of the CoordSender in Configuration Message
8      $CoordSender_{Message} \leftarrow Coordinates_u$ ;
9     Node  $u$  broadcasts a configuration message with the new values;
10  end
11 end
```

---

In this phase, sensor nodes store information about their neighbors in the Table neighborhood. For instance, identification, neighbors' position, and sink's position. This information is used in Phases 2, 3 and 4.

## 5.2 Cluster Formation and Leader Election

After the first phase of the DST algorithm, the reactive part is started and is only run in the presence of events. When an event is detected by one or more nodes, the leader election algorithm is started with the nodes running for leadership (group coordinator) – this process is described in Algorithm 10. For this election, all nodes are eligible (Lines 3 and 4 of Algorithm 10), but the group leader will be the node closer to the sink. (Lines 7 and 8 of Algorithm 10). When two or more eligible nodes have the same distance to the sink, the node with the higher ID is elected (Lines 11 and 12 of Algorithm 10). At the end of the election algorithm only one leader node (coordinator) exists in the group. The remaining nodes that detected the same event become collaborator nodes. The coordinator gathers the information collected by the collaborator nodes, aggregates the information, and sends it to the sink.

---

### Algorithm 10: Cluster formation and leader election

---

```

1 Input:  $S$  // Set of nodes that detected the event
2 Output:  $u$  // A node of the set  $S$  is elected leader of the group
3 foreach  $u \in S$  do
4    $role_u \leftarrow coordinator$ ;
   // Node  $u$  broadcasts the cluster configuration message
5   Announcement of event detection ;
   //  $\mathcal{N}_u$  is the set of neighbors of node  $u$ 
6   foreach  $w \in \mathcal{N}_u$  do
7     if  $DistanceToSink(u) > DistanceToSink(w)$  then
8        $role_u \leftarrow collaborator$  ;
9       Node  $u$  retransmits the message received from node  $w$  ;
10    end
11    else if  $DistanceToSink(u) = DistanceToSink(w) \wedge ID(u) > ID(w)$  then
12      // Node  $u$  changes its role to Collaborator
13       $role_u \leftarrow collaborator$  ;
14      Node  $u$  retransmits the MCC message received from node  $w$ ;
15    end
16 end

```

---

## 5.3 Notification of a New Event

The node chosen as the event Coordinator in Phase-2, as discussed in Section 5.2, gathers the information collected by the Collaborators. Based on its position and the sink's position, the Coordinator creates a straight line segment that connects itself to

the sink. The sensor nodes closer to this straight line segment and to the sink are chosen as Relay nodes.

When the sink receives the first data message, it adds `ID` and `CoordSender` to its Table `CoordinatorTable`, where `ID` is the coordinator's identifier and `CoordSender` is the coordinator's position  $(x_n, y_n)$  that notified the occurrence of a new event in the network. The sink node sends to all coordinators the coordinator's positions stored in its Table `CoordinatorTable`. Each coordinator that receives such a message will update its table of coordinators. This information will then be used in Phase 4 to compute the straight line segments starting at each coordinator.

## 5.4 Routing Tree Creation and Data Transmissions

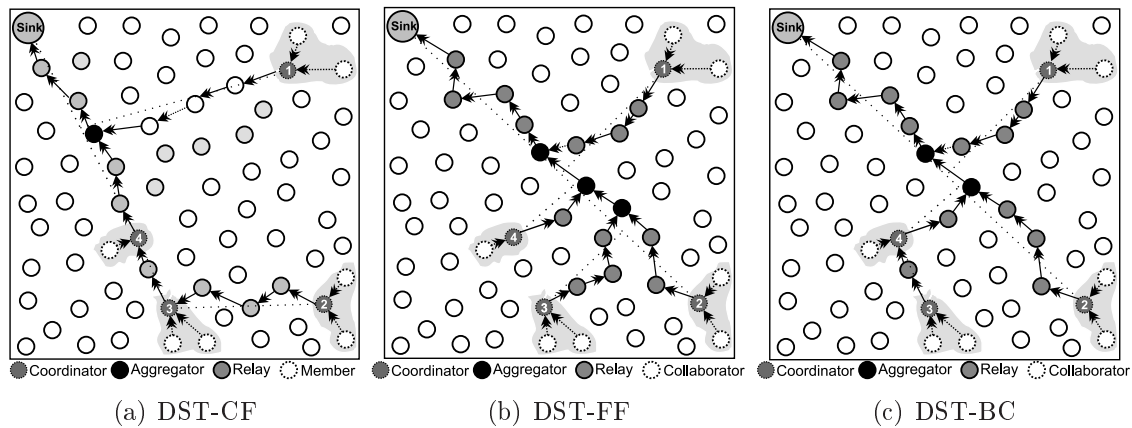
In this work, we present three variations of the DST algorithm. Each variation defines a new approach for a coordinator node to create its straight lines.

1. *DST-CF* (*Closest First*): In this variation, the closest coordinators to the sink node are the first to create their straight line segments to the sink. Then, the second closest coordinators create their straight line segments to the nearest point of the straight line segments that already exist. These steps are repeated until all coordinators create their straight line segments (see Figure 5.1(a)). The computational cost to create the straight lines of closest coordinators has a linear complexity of  $O(e)$ , where  $e$  is the number of events.
2. *DST-FF* (*Farthest First*): In this variation, the farthest coordinators from the sink are the first to create their straight line segments to the sink. Then, the second farthest coordinators create their straight line segments to the nearest point of the straight line segments that already exist. These steps are repeated until all coordinators create their straight line segments (see Figure 5.1(b)). The computational cost to create the straight lines of farthest coordinators has a linear complexity of  $O(e)$ , where  $e$  is the number of events.
3. *DST-BC* (*Best Combination*): This variation of the DST algorithm checks all possible combinations of straight lines and chooses the combination that provides the shortest Euclidean distance to create the routing tree (see Figure 5.1(c)). This approach is optimal, since it finds the routing tree of lowest cost. The computational cost to process all possible combinations of straight lines leads to a factorial complexity of  $O(e!)$ , where  $e$  is the number of events. Due to the resource constraints of a sensor node, this approach is probably unfeasible for

scenarios with a high number of events. Therefore, this approach will be used as a baseline in all evaluations.

When a coordinator node needs to compute its straight line segment, it uses the information obtained in Phases 1–3 (information about the positions of all the other coordinators and the sink). Before computing its straight line segment, the coordinator node computes locally the straight line segments of all other coordinators that have already created their straight lines, depending on the DST variation – DST-CF, DST-FF, or DST-BC. The coordinator node will then create its straight line segment to the nearest point of a straight line segment of other coordinators.

Figure 5.1 shows the routing structure created for each variation of DST. Because the goal here is to illustrate how the line segments are computed, clusterization analysis is presented only in Section 5.2.



**Figure 5.1.** Examples of routing structure establishment for DST variations

This scenario is comprised of four events, whose occurrence order is given by the number inside the coordinator node. The lighter grey nodes are chosen to be part of the routing structure and the darker grey nodes are points of data aggregation.

When the coordinator node performs data transmission, the closest nodes to both its straight line segment and to the endpoint of this straight line will be chosen to forward the data.

The data transmission performed by DST uses aggregation techniques applied to two different contexts: inside and outside the cluster. The coordinator node performs data aggregation inside the cluster and sends the results to the sink node. Outside the cluster, aggregation is performed by aggregator nodes when the occurrence of two or more events overlap along the routing path (dark grey nodes in Figure 5.1).

## 5.5 Complexity Analysis

In this section we derive the communication cost bounds of each algorithm assessed in this work. Here we show the best and the worst case whereas the average case was estimated by simulation (see Sections 5.6.1.1 and 5.6.1.2).

The InFRA algorithm presents  $(2n + (k - 1)n + m)$  linear communication cost as shown in Section 3.5. The best case for the communication cost of DAARP algorithm is  $(2n + m)$ , and the worst case for the DAARP algorithm is  $(2n + ((k - 1)n - \sum_{i=2}^k U_i) + m)$ , as shown in Section 3.5.

The DST algorithm will demand one flooding initiated by the sink to discovery of neighbors' and Sink' positions, which requires  $n$  transmissions. It is necessary  $m$  transmissions to create the clusters. For each elected Coordinator, it is necessary to notify the sink on the occurrence of the event and request information about all the coordinators positions. This requires  $\sum_{i=2}^k 2(h_i)$  transmissions, where  $h_i$  is the distance in hops from the Coordinator  $i$  to the sink node and  $k$  is number of events. Thus, the worst and best cases for the DST algorithm are  $(n + \sum_{i=1}^k 2(h_i) + m)$ .

Table 5.1 presents the communication cost of the algorithms assessed in this work. Note that  $\sum_{i=1}^k 2(h_i)$  is much smaller than  $(k - 1)n$  and  $((k - 1)n - \sum_{i=2}^k |U_i|)$ .

**Table 5.1.** Communication complexity of assessed algorithms

Algorithm	Best Case	Worst Case
InFRA	$(2n + (k - 1)n + m)$	$(2n + (k - 1)n + m)$
DAARP	$(2n + m)$	$(2n + ((k - 1)n - \sum_{i=2}^k  U_i ) + m)$
DST	$(n + \sum_{i=1}^k 2(h_i) + m)$	$(2n + \sum_{i=1}^k 2(h_i) + m)$

## 5.6 Performance Evaluation

In this section, we evaluate the performance of our proposed DST algorithm and its variations. We also compare their performance with two other known routing protocols: InFRA and DAARP.

### 5.6.1 Methodology

The performance evaluation is performed through simulations using the SinalGo v.0.75.3 [Sinalgo, 2008] simulator. In all results, curves represent average values, while

error bars represent confidence intervals for 95% of confidence from 33 different instances (seeds). The default simulation parameters are presented in Table 5.2. For some simulations, a parameter shown in Table 5.2 will be varied and this is described in the scenario evaluated. The first event starts at 1000 s and all other events start at a uniformly distributed random time at uniformly distributed random positions. We consider the area of the sensor field as the relation  $\sqrt{n\pi r_c^2/d}$  to ensure the desired density, when the simulation parameters are varied, where  $n$  is the number of nodes,  $r_c$  is the communication radius, and  $d$  is the average degree of neighbors. In the experimental scenarios, sensor nodes are randomly deployed in the sensor field.

**Table 5.2.** Simulation parameters

Parameters	Values
Sink node	1 (top left)
Event duration (minutes)	60
Number of nodes	1024
Number of events	6
Density	20
Notification rate (per minute)	1
Communication radius (meters)	80
Simulation time (hours)	10
Event radius (meters)	50

To provide a lower bound for packet transmissions, a function was used that receives  $p$  data packets and sends only a fixed-size aggregated packet. This function is run at the aggregation points whenever these nodes send a packet. Any other aggregation function can be used to take advantage of DST features. The evaluated algorithms use a periodic simple aggregation strategy [Younis et al., 2006] employed by the aggregator nodes to periodically transmit the received and aggregated information. The following metrics were used for the performance evaluation:

- **Overhead:** amount of control messages used to build the routing tree including the overhead to create the clusters as well as to set up all the routing parameters of each algorithm;
- **Cost of the routing tree:** number of Steiner nodes included in the routing tree, i.e., the number of relay nodes;
- **Length of the longest route:** number of hops from the farthest coordinator to the sink node in the routing tree;
- **Aggregation rate:** ratio between the number of all data packets sent and the number of data packets received by the sink node; and

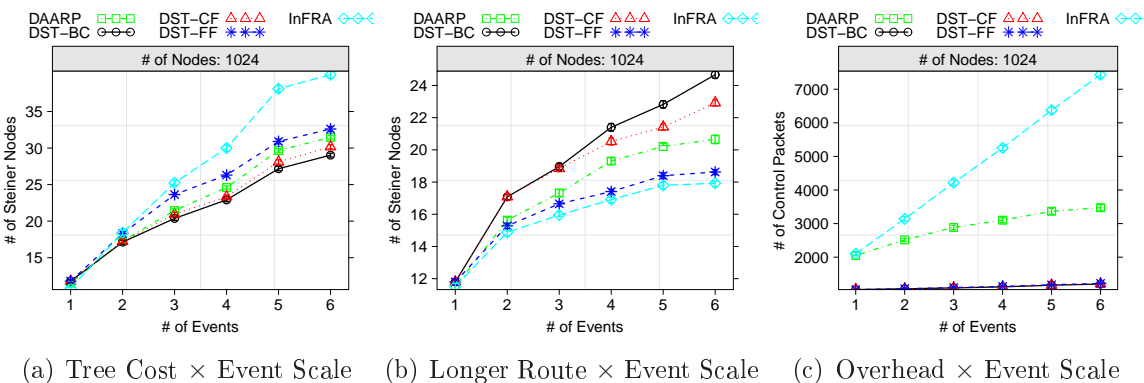
- **Efficiency:** ratio between the total of packets transmitted (including both data and control packets) and the amount of data messages received by the sink node.

### 5.6.1.1 Impact of Event Scale

In this evaluation, the number of events was varied to evaluate the impact of event scale in the routing tree cost, length of the longest route, and overhead. The results are presented in Figures 5.2(a), 5.2(b), and 5.2(c).

Figure 5.2(a) shows that the proposed DST-CF variation is the most efficient in building the lowest cost routing tree and also presents results very close to DST-BC variation. This is a good result since the DST-BC is the optimum result and our baseline protocol. This occurs because the closest coordinators to the sink node are the first to create their straight lines, which consequently prioritizes the low cost routing tree. For six events, DST-CF includes 4% less Steiner nodes than DAARP, 8% less Steiner nodes than DST-FF, and 32% less Steiner nodes than the InFRA algorithm.

Figure 5.2(b) shows that the DST-FF variation is more efficient in building the routing tree with shorter routes. This occurs because the coordinator node that is farthest from the sink creates its straight line segment to the sink, thus a shortest path is created between the coordinator node that is farthest from the sink and sink node. For six events, DST-FF includes 10% less Steiner nodes than DAARP and 23% less Steiner nodes than DST-CF. Regarding the InFRA algorithm, DST-FF includes 3% more Steiner nodes, however, the InFRA algorithm constructs a routing tree 45% worse than DST-FF (see Figure 5.2(a)) and, as we will see, with higher communication overhead (see Figure 5.2(c)).



**Figure 5.2.** Impact of Event Scale

The DST algorithm presents a much lower overhead than DAARP and InFRA. This occurs because the routes are computed locally at coordinator nodes, unlike other



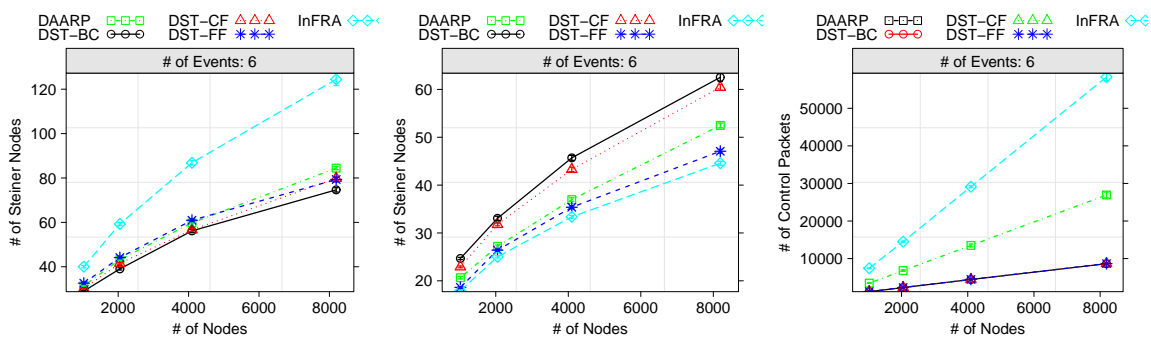
solutions that require many transmissions to configure the routes. Figure 5.2(c) shows that the cost of building the DST routing tree is on average 261% less than DAARP and 428% less than InFRA.

### 5.6.1.2 Impact of Network Scale

In this evaluation, the number of nodes was varied to analyze the impact of the network scale, the routing tree cost, length of the longest route and overhead. Simulation results are presented in Figures 5.3(a), 5.3(b), and 5.3(c).

Figure 5.3(a) shows that DST-CF, on average, is the most efficient algorithm to build a low cost routing tree compared to the other approaches and presents results very close to DST-BC. This occurs because DST-CF builds a routing tree that prioritizes the cost (in Euclidean distance) of the routing tree. Note that when the network scale increases, DST-FF tends to be better than DST-CF. For 8192 nodes, both DST-FF and DST-CF are similar and include 6% less Steiner nodes than DAARP and 37% less Steiner nodes than InFRA.

Figure 5.3(b) shows that DST-FF is more efficient in building the routing tree with shorter routes. This occurs because DST-FF builds a routing tree that tries to decrease the cost (in Euclidean distance) of high-cost routes (in Euclidean distance) of the tree. For 8192 nodes, DST-FF includes 10% less Steiner nodes than DAARP and 22% less Steiner nodes than DST-CF. DST-FF includes 3% more Steiner nodes than InFRA, which constructs a routing tree 56% worse than DST-FF (see Figure 5.3(a)). Finally, the overhead of InFRA is 672% higher than DST-FF (see Figure 5.3(c)).

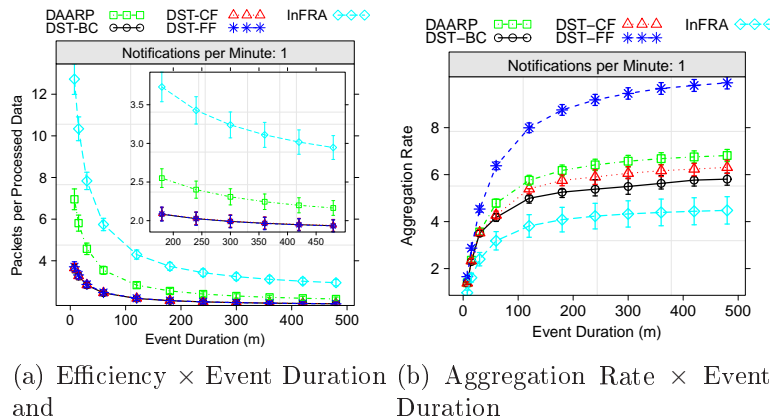


(a) Tree Cost  $\times$  Network Scale (b) Longer Route  $\times$  Network Scale (c) Overhead  $\times$  Network Scale

Figure 5.3. Impact of Network Scale

### 5.6.1.3 Impact of Event Duration

In this evaluation, the event duration was varied to evaluate the aggregation rate and efficiency of the DST algorithm and its proposed variations. Figure 5.4(a) shows that on average DST is 43% more efficient than DAARP and 130% more efficient than InFRA. This occurs because, as mentioned in the previous sections, our proposed DST algorithm needs fewer control messages to build a better low cost routing tree.



**Figure 5.4.** Impact of Event Duration

For short duration events (e.g., 7.5 minutes) as shown in Figure 5.4(a), DST-BC needs 3.63, DST-CF 3.65, and DST-FF 3.71 packets per processed data, whereas DAARP needs 6.95, and InFRA needs 12.9. The DST algorithm presents the best performance from the beginning, since it makes fewer data transmissions and has lower overhead. For long duration events (480 minutes), as shown in Figure 5.4(a), DST-BC needs 1.93, DST-CF needs 1.94, and DST-FF needs 1.93 packets per processed data, whereas DAARP needs 2.16, and InFRA needs 2.94. It is important to note that even in scenarios where the number of notifications is very high, the algorithms present a similar efficiency so the results are not shown. However, the DST algorithm is still better since the amount of transmitted data is smaller.

The internal graph in Figure 5.4(a) is the result for events lasting from 180 to 480 minutes. It is easy to perceive the advantage of the DST algorithm for events of long duration in relation to DAARP and InFRA.

Figure 5.4(b) shows that the DST-FF variation has the highest aggregation rate of all experimented algorithms. The DST-FF algorithm has, on average, 38% higher aggregation rate than DAARP and 111% higher than InFRA. In the DST-FF algorithm, for each aggregated packet that arrives at the sink node, this aggregated packet has information of 7.2 raw packets, while in the DAARP algorithm it has information

of 5.2 and, in the InFRA algorithm, only 3.4. The higher the aggregation rate, the lower the transmission cost for delivering the collected data, and therefore the lower the power consumption for data collection.

## 5.7 Final Remarks on DST

This chapter presented the DST algorithm, an efficient data aggregation solution that allows scalable and dynamic routing in WSNs. Most routing protocols for data aggregation are static, i.e., they cannot change routes dynamically to perform data aggregation efficiently. DST was extensively compared to two other data aggregation aware routing solutions presented in the literature (InFRA and DAARP) regarding scalability, communication cost, routing tree cost, efficiency, and aggregation rate. Simulation results show that DST outperformed those two protocols for all evaluations, maximizing the aggregation points and offering dynamic routes to improve the quality of the routing tree.



# Chapter 6

## EAST: Efficient Data Collection Aware of Spatio-Temporal Correlation

In this chapter, we define our spatio-temporal correlation models and propose the EAST algorithm. One of the key aspects of EAST is that the size of the correlation region and error threshold of readings can be changed dynamically according to the event characteristics in order to achieve the application's accuracy requirements. This results in a better use of the available energy in the nodes that are sensing the event by eliminating redundant notifications as well as by using dynamic routes and low communication overhead. These and other characteristics of our EAST algorithm are discussed in this section.

### 6.1 Spatial Correlation Model

Spatially close nodes tend to detect similar values. However, this closeness ( $\theta$ ), i.e., the Euclidean distance between the nodes that detect similar values, depends on both application requirements and event characteristics. Some applications are more critical and less tolerant to discrepancies in the sensed values of the observed phenomenon, requiring that closer nodes notify the sensed data (the correlation region is smaller). Other applications can be more tolerant to discrepancies in the sensed values, not demanding that closer nodes report the sensed data (the correlation region is greater).

**Definition 6.1.1 (correlation region)** *We define a correlation region as an area where the values sensed by the sensor nodes are considered similar (for the applica-*

tion). Therefore, a single reading within this region is sufficient to represent it. The size of the correlation region ( $c$ ) varies according to both application and event characteristics. For events whose characteristics change significantly at short range, the sink node can decrease the size of the correlated region to keep high accuracy in the collected data, i.e., the event needs to be notified by closer nodes. For events whose characteristics do not change significantly at short range, the sink node can increase the size of the correlated region to save energy of member nodes. The size of the correlation region can be resized by the sink node, which sends the new size of the correlation region to the event's coordinator by using the shortest path. The event's coordinator then disseminates the new size of the correlation region to all of the nodes within the event's area, so the size of the correlation region is reconfigured.

## 6.2 Temporal Correlation Model

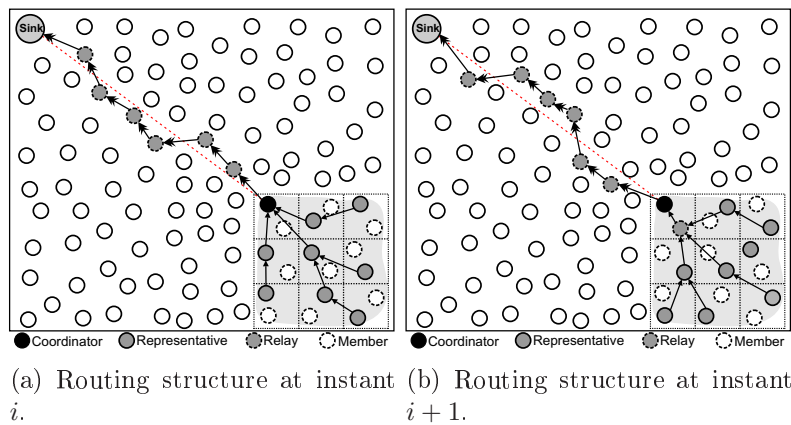
Sensor readings about the environment are typically periodic; consequently, the time-ordered sequence of sensed data constitutes a time series. Due to the nature of the physical phenomenon, there is a significant temporal correlation among each consecutive observation of a sensor node, and gathered data are usually similar over a short-time period. Thus, in these cases, sensor nodes do not need to transmit their readings if the current reading is within an acceptable error threshold regarding the last reported reading. The sink node can just assume that any unreported data is unchanged from the previously received ones. The degree of correlation between consecutive sensor measurements might vary according to the characteristics of the phenomenon.

**Definition 6.2.1 (temporal suppression)** *Each source node keeps the last reported reading. When current reading ( $R_{new}$ ) is available,  $R_{new}$  is compared to the last reported reading ( $R_{old}$ ). The current reading of a source node is reported if the given relative threshold is greater than the temporal coherency tolerance ( $tct$ ), i.e.  $\left(\frac{|R_{new}-R_{old}|}{R_{old}}\right) \times 100 > tct$ , where  $tct$  is the percentage of temporal coherency tolerance. Otherwise the value  $R_{new}$  is suppressed.*

## 6.3 Overview of the EAST Algorithm

The main idea of our proposed EAST algorithm is to manage the energy consumption of nodes that detected an event by eliminating redundant notifications. Our algorithm considers the following roles to perform data routing (see Figure 6.1):

- *Member Node*: A node that is currently detecting one or more events. In the case where its sensed data is redundant, it will not report the gathered data.
- *Representative Node*: A node that detects an event and reports the gathered data to a coordinator representing not only itself but all nearby nodes with similar readings while still applying temporal suppression.
- *Coordinator Node*: A node that detects the event and is responsible for gathering all event data sent by representative nodes. It processes the received data and sends the result towards the sink node.
- *Relay Node*: A node that forwards data towards the sink node.
- *Sink Node*: The gateway between the WSN and the monitoring facility.



**Figure 6.1.** Examples of routing structure used by the EAST algorithm.

The EAST algorithm uses shortest routes (in Euclidean distance) in two different levels for forwarding the gathered data towards the sink node. In the first level, representative nodes use shortest routes to forward data toward the coordinator node. In the second level, the coordinator nodes use shortest routes to forward data toward the sink node. Figure 6.1 shows two examples of the routing structure obtained by the EAST algorithm (the gray field indicates the event area, the cells represent the regions of correlation and the red dotted line shows the shortest route).

The main objective of the EAST algorithm is to reduce energy consumption in data gathering while preserving both data accuracy and real-time reporting. To achieve this goal, EAST dynamically changes the size of the correlation region and the value of the coherency tolerance according to the event characteristics. For this, an event area is divided into cells, as depicted in Figure 6.2. Each cell defines a correlation region and

nodes within each cell are assumed to be spatially correlated. Only one node within a cell notifies the sensed information, if and only if, the given relative error threshold is greater than the temporal coherency tolerance. This last node is the representative node of the cell. Cells are independent from each other, so the change of representative nodes in one cell does not require any reconfiguration. The change of a representative node in each cell is performed to balance the energy consumption of spatially correlated nodes, while temporal suppression is applied to reduce the reporting of redundant data. Since correlation regions are independent, their resizing does not require any additional communication among the nodes within the event areas in order to compute the new cell they belong to. Furthermore, each node performs temporal suppression locally without communicating with its neighbors. The proposed spatio-temporal correlation approach is adaptive and scalable regarding events of different intensities as will be shown during its evaluation.

The EAST algorithm is performed in three phases. In Phase 1, presented in Section 6.3.1, sensor nodes store the sink position as well as their neighbors' positions. In Phase 2, presented in Section 6.3.2, three different actions are performed: cluster formation; coordinator's election; and the division of the event area into cells. Finally, in Phase 3, presented in Section 6.3.3, representative nodes are chosen, the proposed temporal correlation mechanism is applied, and data is then transmitted.

### 6.3.1 Node Localization

After the deployment of the sensor nodes, the sink node starts by flooding a configuration message that contains four fields: `ID`, `CoordSender`, `CoordSink`, and `Phenomenon_of_Interest`, where `ID` is the node identifier that retransmitted the message, `Phenomenon_of_Interest` is the application's interest (e.g., temperature higher than 25 degrees), `CoordSender` is the node's position  $(x_n, y_n)$  that relays the configuration message, and `CoordSink` is the sink's position  $(x_s, y_s)$ . In this phase (Lines 4 to 7 of Algorithm 11), sensor nodes store the received information in a table of neighbors `neighborhood` that will be used in the next two phases.

### 6.3.2 Cluster Formation, Leader Election, and Division of the Event Area into Cells

The second phase of the EAST algorithm starts whenever an event happens. Thus, when an event is detected by one or more nodes, the leader election algorithm is started with the sensing nodes running for leadership (group coordinator) – this process is



described in Algorithm 11. For this election, all detecting nodes are eligible (Lines 8 and 9 of Algorithm 11) and the group leader (Coordinator node) will be the node with the higher residual energy. (Lines 13 and 14 of Algorithm 11). At the end of the election algorithm only one leader node exists in the group. In the case of a tie, the ID parameter is used as a tie breaker. The remaining nodes that detect the same event become member nodes. At each notification, a subset of the member nodes will be representative nodes, as explained later in this section. The coordinator gathers the information collected by the representative nodes, processes the information, and sends it toward the sink node.

After the clustering process, the proposed spatial correlation mechanism is executed. Figure 6.2(a) illustrates the proposed spatial correlation mechanism. For the sake of simplification, the shape of the considered event is a circle, but any shape can work for the proposed solution. The event region is decomposed into  $\left(\frac{2r_e}{c}\right)^2$  cells, where  $r_e$  is the event's maximum radius and  $c$  is the cell's size (correlation region). Figure 6.2(a) shows an example in which the event region is decomposed into 25 cells. Each cell is represented by an ordered pair  $(x_c, y_c)$ . If  $c = 0$ , then there is no spatial correlation between nodes and all nodes in the group are representative nodes. Otherwise, each node computes the coordinates  $x_c$  and  $y_c$  of the cell to which it belongs to. For this computation, the node position  $(x_n, y_n)$ , the central position  $(x_e, y_e)$  of the event, and the cell size ( $c$ ) are required. Lines 18 to 28 of Algorithm 11 show this computation.

If the sink needs to dynamically resize the correlation region or the coherency tolerance value to meet any application requirement, it sends a new value of  $c$  to the nodes of the group so that they can recalculate their cells' size or it sends a new value of  $tct$  to the nodes of the group. Thus, the parameters of our algorithm can be dynamically controlled by the sink node, which has a complete view of the phenomenon.

It is important to point out that the maximum size of a cell  $c$  can be the length of the triangle's leg in a right triangle, since  $r_c$  is the hypotenuse ( $c = r_c \cos 45^\circ$ ) where  $r_c$  is the communication radius of sensor nodes. This consideration is important to ensure that all nodes in the same cell communicate with each other. The cell size can vary to control the tradeoff between precision of the sensed data and energy consumption. In this case, the correlation region may vary between  $0 \leq c \leq r_c \cos 45^\circ$ . When  $c = 0$ , all nodes report the sensed data (an optimal solution in terms of accuracy in the information). For  $c > 0$ , only the representative node at each cell reports the sensed data.

The EAST algorithm selects a single representative node at each cell of dimensions  $c^2$  for each notification. Figures 6.2(b), 6.2(c), and 6.2(d) show representative nodes

at different times in the event region. The representative nodes in the set of member nodes are the nodes that have higher energy residual among nodes belonging to the same cell. This ensures the energy consumption distribution in the network.

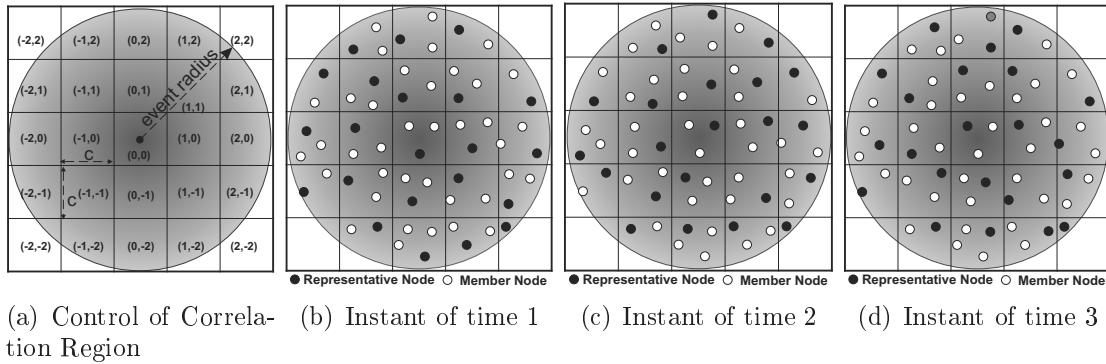


Figure 6.2. Spatial Correlation Mechanism applied to the event area

### 6.3.3 Data Transmissions

After computing the cell that a node belongs to, the node checks whether it is a representative node and also if the relative error threshold is greater than the temporal coherency tolerance (Line 31 of Algorithm 11). If both conditions are satisfied, then the sensed value ( $R_{rew}$ ) is sent towards the group coordinator, which in turn processes and sends the collected information towards the sink using the shortest path. Based on its position and the sink position, the Coordinator creates a straight line segment that connects itself to the sink. When data transmission is performed, the closest nodes to both its straight line segment and the endpoint of this straight line segment will be chosen to forward the data. Figure 6.1 shows the straight line between the coordinator and the sink node as well as the relay nodes. The evaluation of our algorithm is presented in the next Section.

## 6.4 Performance Evaluation

In this section, we evaluate the performance of the spatio-temporal correlation mechanism of our proposed EAST algorithm. We also compare its performance with two other known routing protocols:

- Spatio-temporal Clustering and Compressing Schemes - SCCS (briefly described in Section 2.3.3).

---

**Algorithm 11:** EAST Algorithm.
 

---

```

▷ Variables:
1:  $tct = \{\text{Temporal Coherent Tolerance}\}$ 
2:  $R_{old} = \emptyset \{\text{Last Reported Reading}\}$ 
3: Start Announcement Interest Message

▷ Input:
4: Announcement Interest Message
Action:
5: Stores the Neighbor's and Sink's Positions
6: Stores the Phenomenon of Interest
7: [Re]Start Announcement Interest Message

▷ Input:
8: Event Detected
Action:
9:  $node.Role \leftarrow Coordinator$ 
10: Send Event Announcement Message
11: Start cellComputation

▷ Input:
12:  $msg_i = response(\text{Event Announcement Message})$ 
Action:
13: if ( $EnergyLevel(node) < EnergyLevel(msg_i)$  and  $node.Role == Coordinator$ ) then
14:    $node.Role \leftarrow Member$ ;
15:   Retransmits (Event Announcement Message)
16:   Start DataTransmissions
17: end if

▷ Input:
18: cellComputation timeout
Action:
19:  $x_c \leftarrow 0$ 
20:  $y_c \leftarrow 0$ 
21: if  $\frac{(x_n - x_e)}{(\frac{c}{2})} > 1$  then
22:    $x_c \leftarrow \lfloor \frac{(x_n - x_e) - (\frac{c}{2})}{c} \rfloor + 1$ 
23:    $y_c \leftarrow \lfloor \frac{(y_n - y_e) - (\frac{c}{2})}{c} \rfloor + 1$ 
24: end if
25: if  $\frac{(x_n - x_e)}{(\frac{c}{2})} < -1$  then
26:    $x_c \leftarrow \lfloor \frac{(x_n - x_e) + (\frac{c}{2})}{c} \rfloor - 1$ 
27:    $y_c \leftarrow \lfloor \frac{(y_n - y_e) + (\frac{c}{2})}{c} \rfloor - 1$ 
28: end if

▷ Input:
29: Data Transmissions
Action:
30:  $R_{new} \leftarrow sensed\ value$ 
31: if  $node.Role = Representative$  and  $(\frac{|(R_{new} - R_{old})|}{R_{old}}) \times 100 > tct$  then
32:   Send  $R_{new}$  to Coordinator
33:    $R_{new} \leftarrow R_{new}$ 
34: end if
35: if  $node.Role = Member$  then
36:   Forwards  $R_{new}$  to Coordinator
37: end if
38: if  $node.Role = Coordinator$  then
39:   Processing received  $R_{new}$ 
40:   Forwards the result to Sink
41:   if  $node.Role = Relay$  then
42:     Forwards  $R_{new}$  to Sink
43:   end if
44: end if

```

---

- Accurate data collection strategy, which is the optimal solution in terms of accuracy. In this solution, all nodes send their sensed information to the sink node.

### 6.4.1 Methodology

The evaluation is performed through simulations by using the SinalGo version v.0.75.3 [Sinalgo, 2008] simulator. In all results, curves represent average values, while error bars represent confidence intervals for 95% of confidence from 33 different instances (seeds). The simulation parameters are presented in Table 6.1. The event occurs in random positions. We consider the area of the sensor field as the relation  $\sqrt{n\pi r_c^2/d}$ , where  $n$  is the number of nodes,  $r_c$  is the communication radius, and  $d$  is the average degree of neighbors. Sensor nodes are randomly deployed.

**Table 6.1.** Simulation parameters

Parameters	Values
Sink node	1 (top left)
# of nodes	1024
# of events	1
Density (avg. neigh. number)	(20, 25, 30)
Event diameter (m)	(50, 100, 150, 200)
Correlation Region ( $\phi$ )	(0, 10, 20, 30, 40, 50)
Event duration (hours)	(1 to 10)
Notification rate (per minute)	1
Communication radius (m)	80
Simulation duration (days)	7

### 6.4.2 Event Model

For our event model, we used a set of one-week environmental temperature data (degree in Celsius) from the Amazon rainforest in Brazil collected at intervals of 1 minute. The samples are shown in Figure 6.3.

In our application, the temperature in a region of interest is monitored. All nodes in this region will then send the data according to our proposed algorithm. To use the real-world data in our simulations, we consider the temperature at coordinate  $(x, y)$  in the event area given by Equation 6.1, where  $T_E$  is the temperature at the event center, which was obtained from the real-world data set presented in Figure 6.3,  $D_E$  is the Euclidean distance (meters) to the event center and  $T_D$  is the temperature decrease (degree Celsius per meters).

$$temperature = T_E - (D_E \times T_D) \quad (6.1)$$

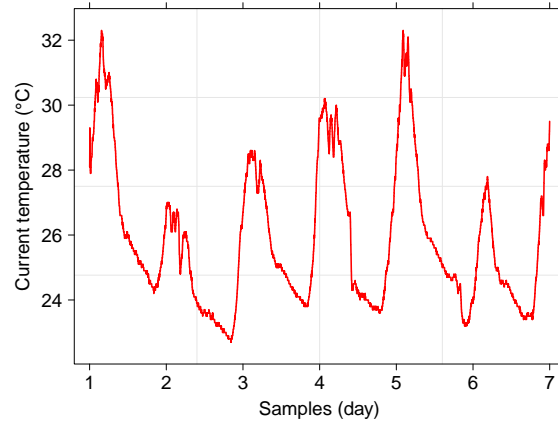


Figure 6.3. Data collected from Amazon rainforest.

### 6.4.3 Performance Evaluation of the Spatial Correlation Mechanism

In this section, the proposed spatial correlation mechanism is evaluated and compared to the accurate data collection strategy, which is the optimal solution in terms of accuracy. In the accurate data collection strategy, every sensor is requested to report its reading to the sink node at each round of data gathering. The main purpose of this subsection is to compare the performance of our proposed spatial correlation mechanism to the accurate data collection strategy considering the following metrics:

- *Number of Representative nodes*: The number of nodes that report data about the phenomenon.
- *Energy consumption in data collection*: The amount of energy consumed by sensors that detected the event. This metric indicates how much of a sensor node energy is possible to save when the spatial correlation technique is exploited.
- *Data accuracy*: The accuracy of the data on the observed phenomenon regarding the original information.

When  $c = 0$  the spatial correlation is not explored, i.e., all nodes report the sensed data, which is the optimal solution in terms of data accuracy. The results for the DST [Villas et al., 2010b], DAARP [Villas et al., 2009], and INFRA [Nakamura et al., 2009] algorithms, as well as the other solutions that do not exploit spatial correlation, are the same as the results obtained with  $c = 0$ .

### 6.4.3.1 Number of Representative Nodes

The number of representative nodes that notifies events depends on three main parameters: event diameter, size of correlation region, and density. In this simulation scenario, the event diameter, the density, and the size of the correlation region (presented in Table 6.1) were all varied to evaluate their impact on the number of representative nodes.

Figure 6.4 presents the number of representative nodes when the correlation region, density, and event diameter were varied. As expected, if the correlation region remains fixed, the number of representative nodes increases when the event diameter increases. It is also easy to see that for larger values of  $c$ , there are less representative nodes and, therefore, a lower reporting rate. The region event is divided into  $(2r_e/c)^2$  correlation regions. In particular, for the event diameter of 200 m, correlation regions of 50 m, and density 30, the number of representative nodes is reduced four times when compared to the accurate data collection strategy ( $c = 0$ ). Consequently, the amount of energy consumed by nodes within the observed phenomena area is also reduced four times (as shown in the next section).

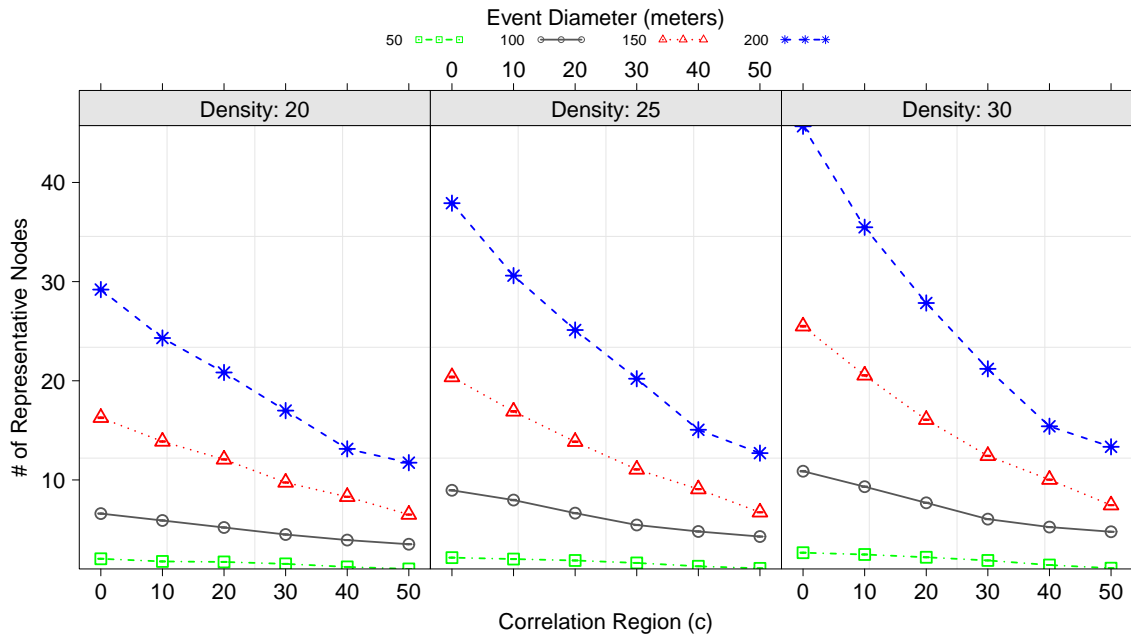


Figure 6.4. Number of representative nodes

Note that for larger correlation regions the number of representative nodes is smaller, hence the energy consumption is lower (as shown in the next section); however, the accuracy will be smaller (see Section 6.4.3.3). In our solution, applications can define the correlation region size by setting the value of  $c$  according to the required

accuracy.

### 6.4.3.2 Energy Consumption

Figure 6.5 shows the energy consumption of nodes within the observed phenomena area. For this analysis, the density, event diameter, event lasting, and correlation region (presented in Table 6.1) were all varied to evaluate their impact on the energy consumption. In the EAST algorithm, the representative nodes are alternated to achieve a more balanced energy consumption of the nodes that are reporting data.

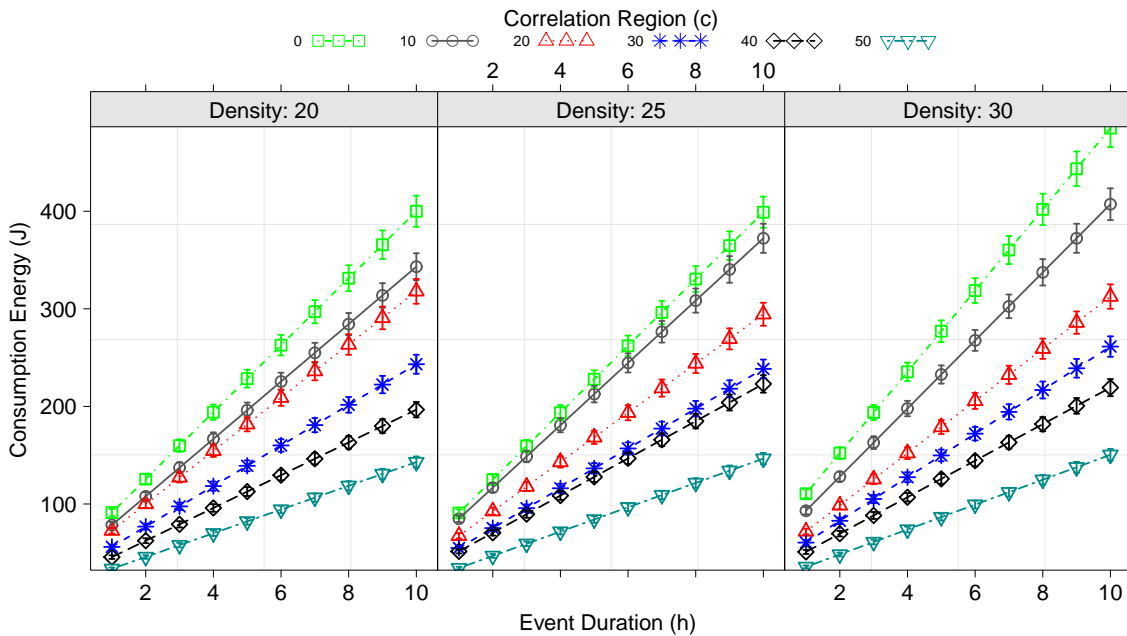


Figure 6.5. Energy consumption of the nodes that are reporting data

As depicted in Figure 6.4, when the size of the correlation region increases, the number of representative nodes decreases. Consequently, the energy consumption also decreases, since fewer sensor nodes report data. In this scenario, it is possible to save up to 75% of the residual energy of the nodes within the observed phenomena area when compared to the classical approach for data collection (accurate data collection strategy) while maintaining an information accuracy greater than 97%, as shown below.

### 6.4.3.3 Data Accuracy

When the spatial correlation is exploited, the level of accuracy in information about the observed phenomenon tends to reduce. In this simulation scenario, the event diameter, density and size of the correlation region were varied to evaluate the data accuracy. As

mentioned before, when  $c = 0$ , the classical approach for data collection is performed (100% accurate data collection strategy). It means that the spatial correlation is not exploited and all nodes that detect an event will report their readings. Consequently, the accuracy is optimum (100%). The phenomenon observed was the temperature, but the proposed mechanism works for any other type of phenomenon with different characteristics. We analyzed the accuracy at the sink node when computing the values for minimum, mean, and maximum temperatures.

As we can see in Figure 6.6, when the node density increases, the data accuracy decreases slightly, which is an unexpected result in most algorithms. It happens because the number of nodes within each cell increases, but the number of representative nodes remains the same. The difference between the combined readings of all nodes and the combined readings of only representative nodes increases.

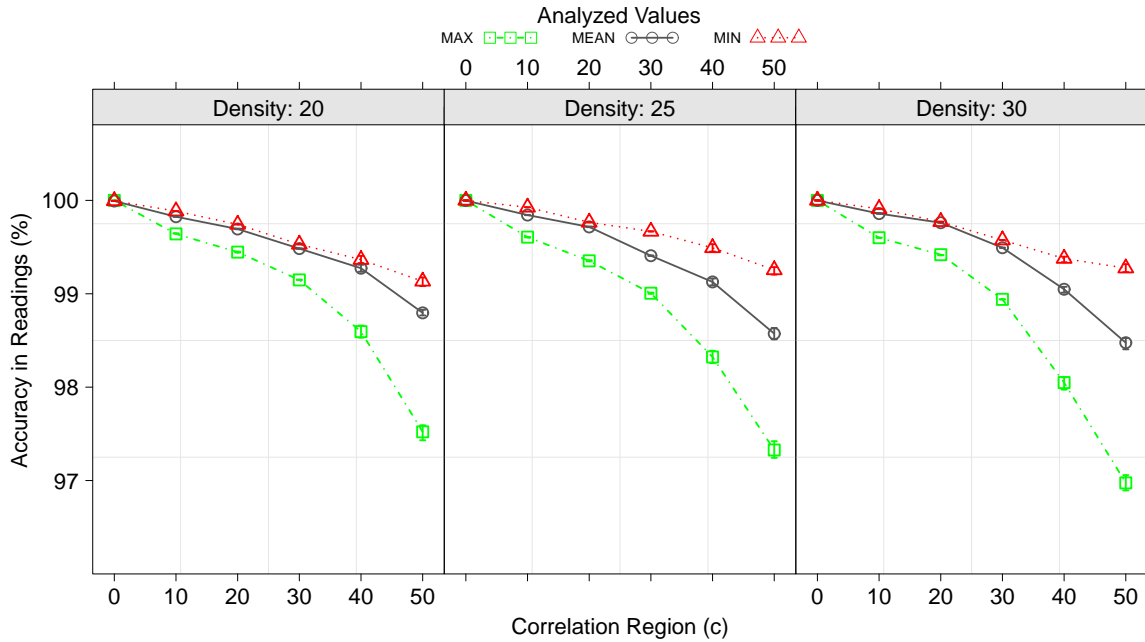


Figure 6.6. Accuracy in the readings

On the one hand, it can be noted that the worst accuracies are obtained by the maximum value readings. This happens because nodes that detect the maximum value are in the central cell (see Figure 6.2). With a greater number of nodes within the central cell, there will be a greater number of nodes that notifies data with values closer to the maximum value. However, for the evaluated scenarios, the smallest observed accuracy was of 97% while the energy consumption was reduced by more than 75%, which indicates the advantages of using the proposed spatial correlation technique of the EAST algorithm.



On the other hand, the reading of the minimum value had the highest accuracy since there are more cells with nodes that detect this value (cells in the border of the event). In the case of this phenomenon, the smallest observed accuracy was greater than 99% while, again, the energy consumption was reduced by 75%.

It is important to note that there is a trade-off between the data accuracy and the energy consumption. For instance, if the application requires the measurement of the maximum value with an accuracy of at least 99.5%, then the value of  $c$  will have to be set to 10 ( $c = 10$ ). In this case, the accuracy in the readings of the maximum value would be more than 99.5% and the reduction in energy consumption would be reduced to 33%.

#### 6.4.3.4 Data Accuracy for Each Round of Data Gathering

In this section, we present the analysis of the data accuracy for each notification, complementing the results presented above that analyzed the average accuracy of the measurements. In this simulation scenario, the size of the correlation region was varied to evaluate the data accuracy at each notification. The objective of this analysis is to show that it is possible to ensure the accuracy of insensitive duplication data (such as maximum and minimum) at different times. Consequently, if time and recent readings are taken into account, it is possible to estimate the exact (minimum and maximum) value.

Figure 6.7 shows that the minimum accuracy for reading the minimum value at a given time is 92% (when the correlation region is 50 m). However, at least for every four reports, the exact minimum value is reported. Because of this, the exact minimum value can be estimated by representative nodes. Consequently, the accuracy of the readings for a minimum value can be increased very close to the exact value. Note that for a correlation region smaller than 30 m, the minimum accuracy is 98% and at least for every two notifications the exact minimum value is reported.

Similarly, Figure 6.8 shows that the minimum accuracy for reading the maximum value at a given time is 93%, but at least for every four reports the exact maximum value is reported. Because of this, the exact maximum value can be estimated by representative nodes and, as a result, increase the accuracy of the reading to a maximum value very close to the exact value.

Different from minimum and maximum values, the exact mean value is not reported at each time interval, as depicted in Figure 6.9. This occurs because mean values are sensitive to duplication data. If taken into account the time and recent readings, it is possible to estimate the mean value to increase the accuracy at each

reading.

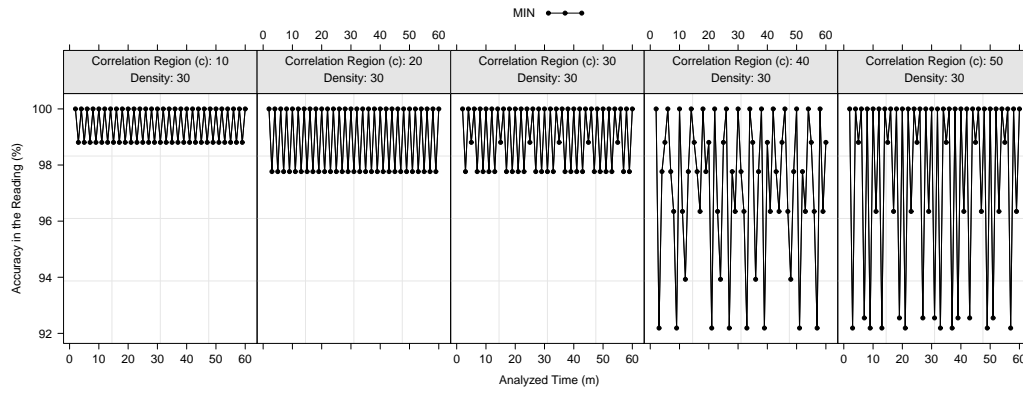


Figure 6.7. Accuracy in the readings of min value

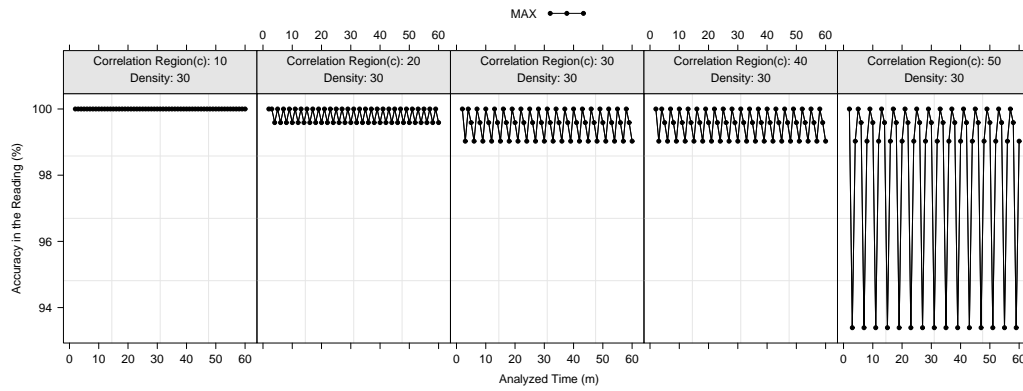


Figure 6.8. Accuracy in the readings of max value

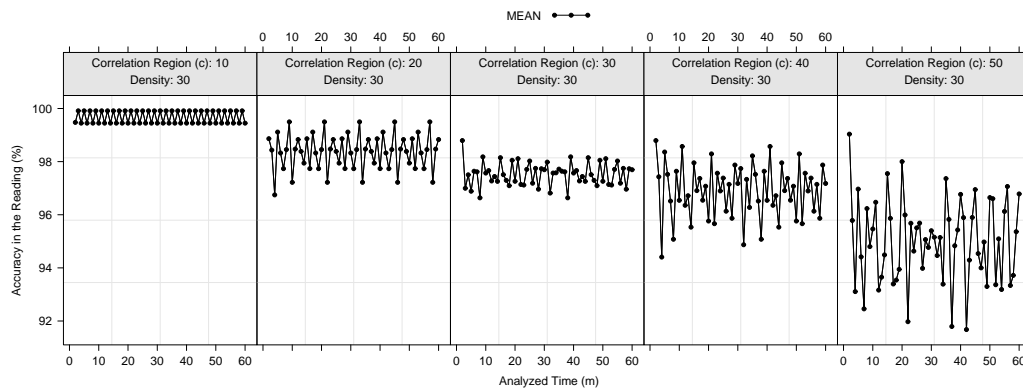


Figure 6.9. Accuracy in the readings of mean value

### 6.4.4 Performance Evaluation of Temporal Correlation Mechanism

In this section, we compared our proposed temporal correlation mechanism for the EAST algorithm to the SCCS algorithm as well as the accurate data collection strategy, which is the optimal solution in terms of accuracy. For our event model, we used a set of one-week environmental temperature data from the Amazon rainforest in Brazil taken at intervals of 1 minute. The samples are shown in Figure 6.3. For the SCCS algorithm, as mentioned before, each node stores its monitored data in a buffer and, when the buffer is full, the node processes the data in its buffer to consider the temporal correlation among the monitored values and report the result to the sink node. For the accurate data collection strategy, every sensor is requested to report its readings to the sink node at each round of data gathering. The main purpose of this comparison is to evaluate the performance of our proposed algorithm considering the following metrics: (i) notifications, (ii) readings reported, (iii) readings per data packet, (iv) energy consumption, (v) data accuracy, and (vi) delay notification.

The simulation parameters used in this performance evaluation are the same of previous experimentations (shown in Table 6.1) with the new values presented in Table 6.2.

**Table 6.2.** Simulation parameters

Parameters	Values
Density (avg. neigh. number)	25
Correlation Region ( $c$ )	(15, 30, 45)
Temporal coherent tolerance	(0.5, 1, 2, 3, 4)
Buffer size (bytes)	(25, 50, 100, 200)
Sensor field (m)	900 × 900

The following metrics were used for the evaluation:

- *Number of notifications*: Number of notifications sent by nodes that detect the event.
- *Number of readings reported*: Number of readings reported.
- *Readings per data packet*: Average number of readings within each packet (it is the ratio of Readings reported and Notifications).
- *Energy consumption*: The amount of energy consumed by sensors that detected the event.

- *Data accuracy*: The accuracy of the information on the observed phenomenon regarding the original information.
- *Delay notification*: Time to deliver the gathered data.

In all evaluated cases, a number of variations of our proposed EAST algorithm was considered. First, we evaluated our EAST algorithm while exploring only the temporal correlation. We also evaluated our algorithm (EAST-15, EAST-30, and EAST-45) exploiting both temporal correlation and spatial correlation (with correlation regions of size 15, 30, and 45). For the SCCS algorithm, we considered buffers with different storage capacities of 25, 50, 100, and 200 readings.

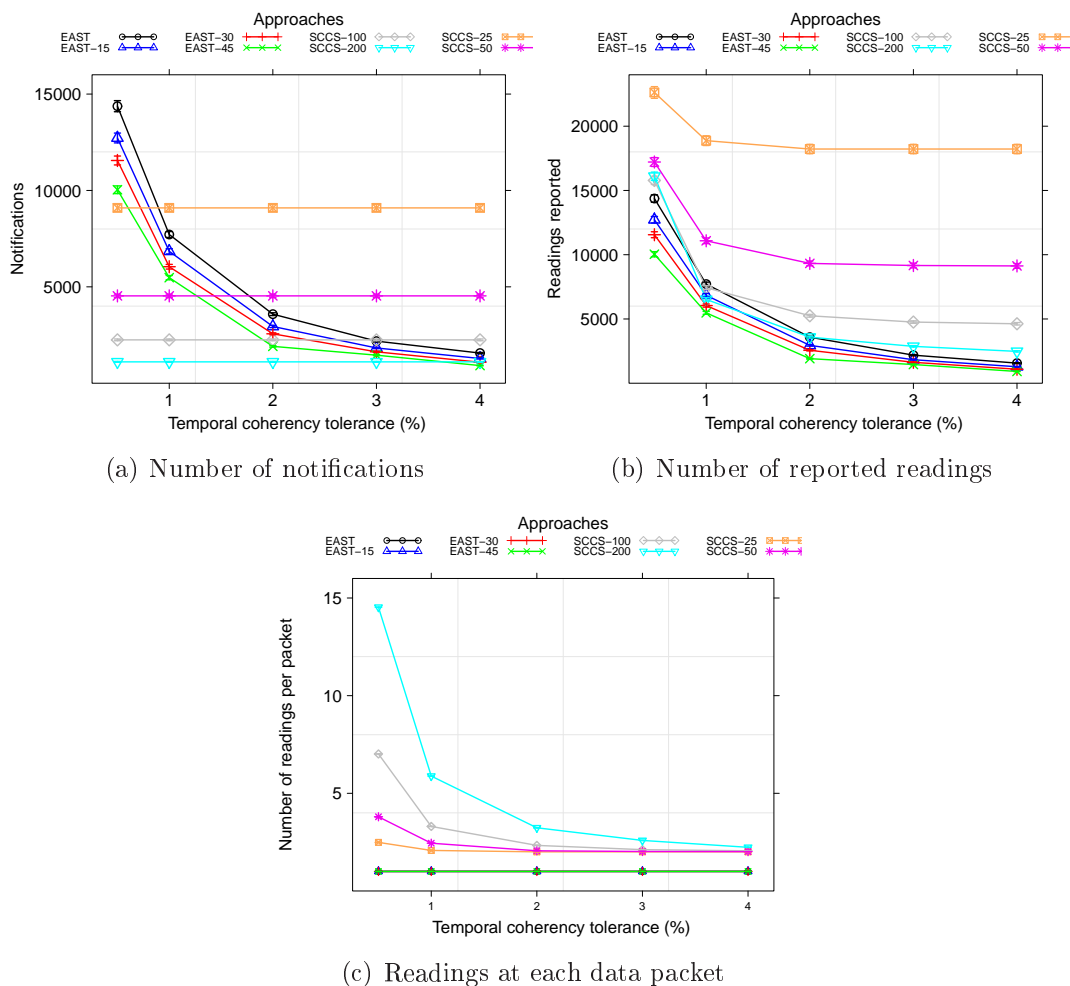
#### 6.4.4.1 Notifications and Readings Reported

For this analysis, the temporal coherency tolerance, buffer size, and correlation region (presented in Table 6.2) were all varied to evaluate their impact on the number of readings that can be eliminated by exploiting the spatio-temporal correlation.

Figure 6.10(a) shows that when the temporal coherency tolerance increases, the number of notifications performed by our EAST algorithm decreases while the number of notifications in the SCCS remains fixed since the readings will only be transmitted when the buffer fills up. Consequently, in the SCCS, the number of notifications depends on the buffer size and not on the temporal coherency tolerance. Moreover, in our EAST algorithm, when the size of the correlation region increases (15, 30 e 45), the number of representative nodes decreases, which also decreases the number of notifications.

In Figure 6.10(b), we can see that the number of readings reported by our EAST algorithm is similar to the ones presented in Figure 6.10(a). This is because whenever the current reading is above the temporal coherency tolerance, the data is notified. Note that in most cases the EAST algorithm reports fewer readings than the SCCS algorithm, but for small values of the temporal coherency tolerance, the SCCS algorithm presents less notifications by exploring the use of a buffer, in which each notification may contain more than one reading. Since the SCCS algorithm creates a line segment between the first and last reading of the buffer, this technique has good results in terms of energy consumption when the buffer size increases (as depicted in Figure 6.11). In this case, it is necessary a few line segments to represent all values inside the buffer.

Figure 6.10(c) shows the average number of readings within each transmitted packet. As we can see, the EAST algorithm, in any situation, sends only one reading within each transmitted packet. But the number of readings per packet in the SCCS



**Figure 6.10.** Notifications  $\times$  Readings

algorithm depends both on the capacity of the buffer and the temporal coherency tolerance. When the temporal coherency tolerance is small, the SCCS algorithm needs to split the original line segment into a high number of other line segments to represent the original values. Because of this, more readings will be necessary to represent the monitored values. It is important to point out that when we increase the buffer size, the SCCS algorithm will split the original line segment into more line segments to represent the original values, since more readings are being considered by the algorithm. When we increase the temporal coherency tolerance, it is not necessary to split the original line segment into a high number of line segments. For instance, when the temporal coherency tolerance is 4%, the SCCS algorithm transmits the same number of readings (compared to our proposal) to represent the sensed event.

### 6.4.4.2 Energy

Figure 6.11 shows the average energy consumption of the nodes within the observed phenomena area. For this analysis, the temporal coherency tolerance, buffer size, and correlation region (presented in Table 6.2) were all varied to evaluate their impact on the energy consumption.

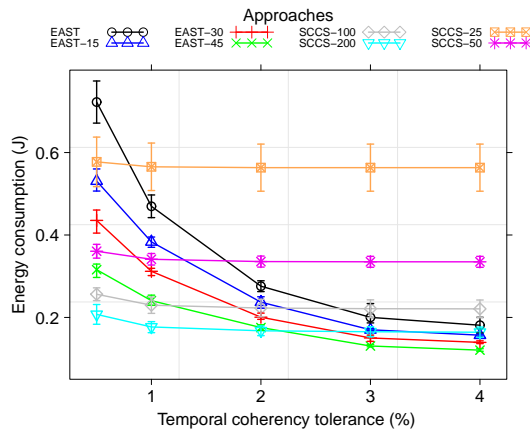


Figure 6.11. Average energy consumption

As depicted in Figure 6.10(a), when the temporal coherency tolerance increases, the number of notifications performed by EAST decreases while the number of notifications in the SCCS remains fixed and depends on the buffer size, which has an impact on the data accuracy (see Figure 6.12). Consequently, the energy consumption also decreases in the EAST algorithm (see Figure 6.11). Moreover, when the size of the correlation region increases (15, 30 e 45), the number of notifications decreases. Consequently, the energy consumption also decreases, since a smaller number of sensor nodes report their readings.

The results for the accurate data collection strategy were not plotted on the graph because of its very high energy consumption, which is due to the fact that all readings are notified to ensure data accuracy of 100%. On average, the accurate data collection strategy consumes  $10J$ , i.e., 14 times more than the SCCS and EAST algorithms.

### 6.4.4.3 Data Accuracy

In this simulation scenario, the temporal coherency tolerance, buffer size, and correlation region were all varied to evaluate the accuracy of the readings. The phenomenon observed was the temperature but, as mentioned before, the proposed mechanism works for any other type of phenomenon with different characteristics. We analyzed the accu-

racy at the sink node when computing the values for minimum, mean, and maximum temperatures.

As depicted in Figure 6.12, when the temporal coherency tolerance increases, the data accuracy in the EAST algorithm slowly decreases. For the SCCS algorithm, when the buffer capacity increases, the data accuracy decreases faster. This happens because when we increase the buffer size, less readings will be necessary to represent the monitored event (see Figure 6.10(b)). Thus, considering less values, the SCCS algorithm will not achieve good values of data accuracy.

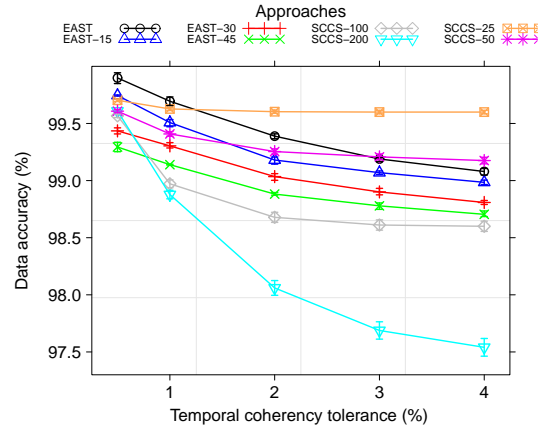


Figure 6.12. Data accuracy

For the evaluated scenarios, the smallest accuracy observed in our proposed algorithm was of 98.7% and the energy consumption was less than  $0.2 J$ , while the accurate data collection strategy consumes  $10 J$ , which indicates the advantages of using our spatial and temporal correlation techniques.

#### 6.4.4.4 Average Delay in Reporting the Readings

In this section, we evaluate the average delay in reporting the readings for both scenarios of low and high notification rate, which is shown in Figures 6.13 and 6.14, respectively. For this analysis, the notification rate was varied to evaluate the average delay in reporting the readings. As mentioned before, the SCCS algorithm explores the use of a buffer to store the readings and, then, it is processed while exploiting temporal correlation. However, the use of a buffer has some disadvantages. The main drawback of this technique is the delay of the notification of each data. Figures 6.13 and 6.14 show that the delay to notify the sensed data is very high in the SCCS. As expected, the larger the buffer size or the range of notification, the greater the delay. For instance, for the scenario of SCCS with a buffer size of 100 readings and an interval

of 1 reading per second (see Figure 6.14), the node will send its values only after 100 readings, which implies an average delay of nearly 100 seconds. Thus, the sink node will be notified about the event considering old readings, which is not acceptable by several WSN applications. One way to overcome this problem is to reduce the buffer size. However, as depicted in Figure 6.11, when the SCCS uses a small buffer size, the average energy consumption is greater than the EAST algorithm. For instance, the use of a buffer of 25 readings consumes on average 55% more energy than the EAST algorithm. Even with a buffer size of 25 readings, the SCCS algorithm reaches, on average, a data accuracy of 99.62% while the EAST algorithm remains on 99.14% (see Figure 6.12).

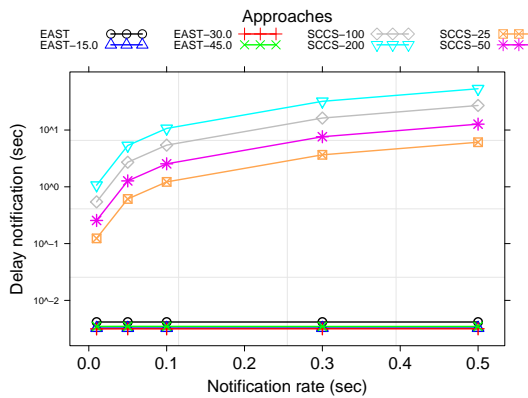


Figure 6.13. Average delay in high reporting rate

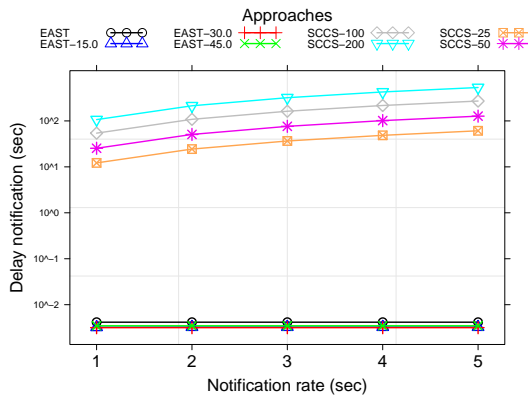


Figure 6.14. Average delay in low reporting rate

Because the EAST algorithm considers the last notified reading to exploit the temporal correlation, as soon as the relative error threshold of the actual reading is greater than the determined tolerance of temporal coherency, the algorithm notifies the



actual reading to the sink node. As a result, every time the sensed value is beyond the error threshold the sink will be notified in real time what is expected in most WSNs applications.

## 6.5 Final Remarks on EAST

This chapter presented the EAST algorithm, an algorithm for energy-aware data forwarding in WSNs that takes full advantage of both spatial and temporal correlation mechanisms to save energy while still maintaining real-time, accurate data report towards the sink node. In the current literature of spatial and/or temporal correlation algorithms, most of the proposed studies do not consider the energy dissipation during data collection to better choose the representative nodes. Also, these solutions present a high number of control messages and do not exploit efficiently the spatio-temporal correlation nor their dynamicity. In this work, we went further and proposed an energy-aware spatio-temporal correlation mechanism in which nodes that detected the same event are dynamically grouped in correlated regions and a representative node is selected at each correlation region for observing the phenomenon. The entire region of sensors per event is effectively a set of representative nodes performing the task of data collection and temporal correlation.

We exhaustively simulated our proposed algorithm considering several scenarios and parameters to allow a better understand of its behavior. Simulation results clearly show that by using both spatial and temporal correlation, the information about the event can be sensed with a high accuracy of more than 99.7% while still saving the residual energy of the nodes in more than 14 times when compared to the accurate data collection strategy. These results are very promising, but some issues still need to be further exploited. As future work we intend to consider not only the last reading, but also the previous readings in the correlation region to improve the accuracy of sensed data about the observed phenomenon. To achieve this goal, representative nodes can estimate the values of their correlation region by taking into account the time and recent readings. In addition, we intend to consider correlation regions of different sizes for the same event to further explore the dynamicity of the event.



# Chapter 7

## Final Remarks

This chapter summarizes this thesis and discusses directions for future research. The objective is to highlight our contributions and point out some possible directions to proceed with the research to address the drawbacks of the proposed solutions. In this context, we first present the thesis conclusions in Section 7.1. Then, in Section 7.2, we point out the limitations of the proposed algorithms. In Section 7.3 we present future directions of this work. Finally, in Section 7.4.1 we present the publications related to this thesis.

### 7.1 Conclusions

In this dissertation, we have provided a survey on the state-of-the-art about the use of data aggregation and spatio-temporal data correlations in WSNs. This survey has allowed us to understand how data aggregation and spatio-temporal data correlations have been used in WSN, and how it can still be used to address open issues on WSNs. In addition, it has allowed us to identify drawbacks of current proposals and to propose new solutions that overcome the drawbacks of current proposals.

The different scenarios in which a WSN can be deployed as well as the broad applicability of WSNs indicate there is no single solution for a problem in WSNs. For this reason, there are several different solutions for the same problem in WSNs. Each solution is designed to work well in a specific scenario such as static or mobile networks; small, medium, or large scale networks; sparse or dense networks; and etc. Due of this, the choice of which protocols and algorithms should be uses to provide routing structure for a WSN depends on both the scenarios and the application.

Based on the survey presented, and on the impossibility of designing a single solution to a problem in WSNs, we have proposed four different solutions for the data

aggregation and exploiting spatio-temporal data correlations for WSNs, which we refer to as DAARP, DDAARP, DST, and EAST algorithms, respectively.

- *DAARP: Data Aggregation Aware Routing Protocol for WSNs (shown in Chapter 3)* builds a routing structure with the shortest paths (in hops) that connect all source nodes to the sink while maximizing data aggregation, whose main contribution is to maximize data aggregation along the communication route, in a more reliable way, through a routing fault tolerant mechanism. Simulations results (presented in Section 3.6) reveal that DAARP has some keys aspects required by data aggregation in WSNs such as a reduced number messages for setting up a routing structure, maximized number of overlapping routes, high aggregation rate, and reliable data aggregation and transmission.
- *DDAARP: Dynamic Data-Aggregation Aware Routing Protocol for WSNs (shown in Chapter 4)* is a novel dynamic data aggregation aware routing protocol for WSNs, which uses the sink node for processing and configuration of the routes aware of data aggregation. The main contribution is that the routes created by DDAARP does not depend on the order of events and are not held fixed during the occurrence of events such as the DAARP and the most algorithms in the literature. Simulations results (presented in Section 4.6) reveal that DDAARP presents low cost in terms of packets control, improves the quality of the routing structure and maximizes data aggregation along the communication route in a more reliable way, through a routing fault tolerance mechanism.
- *DST: Dynamic and Scalable Tree for WSNs (shown in Chapter 5)* is an efficient data aggregation solution that allows scalable and dynamic routing in WSNs, which builds routing structures with the shortest routes (in Euclidean distance) that connects all source nodes to the sink node maximizing data aggregation and reducing the distance to connect each source node to the sink. Also, the routing structure created does not depend on the event order. Simulations results (presented in Section 5.6) reveal that DST presents low cost in terms of packets control, maximizes aggregation points and improve the quality of routing structure offering dynamic routes.
- *EAST: Efficient Data Collection Aware of Spatio-Temporal Correlation for WSNs (shown in Chapter 6)* is an algorithm for energy-aware data forwarding in WSNs that takes full advantage of both spatial and temporal correlation mechanisms to save energy while still maintaining real-time, accurate data report towards the sink node. The main contribution is an energy-aware spatio-temporal correlation

mechanism in which nodes that detected the same event are dynamically grouped in correlated regions and a representative node is selected at each correlation region for observing the phenomenon. The entire region of sensors per event is effectively a set of representative nodes performing the task of data collection and temporal correlation. Simulations results (presented in Section 6.4) clearly show that by using both spatial and temporal data correlations, the information about the event can be sensed with a high accuracy while still saving the residual energy of the nodes.

## 7.2 Limitations

Some limitations have been identified in the current state of the research, and such limitations leads to future directions. First, the proposed algorithms (DAARP, DDAARP, DST and EAST) present improvements on distributed heuristics for the Steiner tree problem when we have resource-constrained networks, such as energy, memory and bandwidth. However, the current version of proposed algorithms considers only static events. In addition, each proposed algorithm presents some drawback as described bellow.

The main drawback of DAARP algorithm is the static route. Since, the routes created by DAARP are held fixed during the occurrence of events, in scenarios where the events are of long duration, the energy of nodes that are part of the routes exhausts quickly to forward the data collected. Despite DDAARP deal with the static routes problem, it suffers from scalability problems and becomes impractical for large-scale networks. In addition, the sink node need a global knowledge network. The DST and EAST algorithms are potential solutions to deal with the scalability and static routes problems. However, some issues still need to be further explored such as correlation regions of different sizes for the same event to further explore the dynamicity of the event to improve the accuracy of sensed data.

## 7.3 Directions for Future Research

The results obtained in this thesis are very promising. The solutions proposed in the current work usually take advantage of both data aggregation and spatio-temporal data correlations to improve the routing performance and reduce energy consumption in data gathering while preserving both data accuracy and real-time reporting. However the current version of these algorithms are impracticable for mobile networks.

As future work we intend extend the solutions proposed to work in mobile wireless sensor networks. Also, we intend to investigate a special kind of Mobile Ad Hoc Network known as Vehicular Ad Hoc Network (VANET), in which vehicles equipped with wireless and processing capabilities can create a spontaneous network while moving along roads. Vehicular Ad Hoc Networks (VANETs) have emerged as an exciting research and application area. The envisioned applications, as well as some inherent VANET characteristics such as highly dynamic topology, frequently disconnected network, and different and dynamic network density, make data dissemination a challenging task in these networks. Several approaches for data dissemination in VANETs have been recently proposed in the literature. However, more work needs to be done since most of the proposed solutions do not effectively address some or all of the main challenges in these scenarios such as the broadcast storm, network partition and temporal network fragmentation.

## 7.4 Comments on Publications

We list all the publications obtained during the doctorate below.

### 7.4.1 Journals

- Villas, Leandro A.; Boukerche, Azzedine; Guidoni, Daniel L.; de Oliveira, Horacio B.F.; de Araujo, Regina Borges; Loureiro, Antonio A.F. "An Energy-aware Spatio-Temporal Correlation Mechanism to Perform Real-Time Data Collection in Wireless Sensor Networks" *Computer Communications*, 2012. To Appear. **[Thesis]**
- Villas, Leandro A.; Boukerche, Azzedine; de Oliveira, Horacio B.F.; de Araujo, Regina Borges; Loureiro, Antonio A.F. "Data Dissemination in Vehicular Networks: Challenges, Solutions, and Future Perspectives" *Wireless Communications Magazine*, 2012. To Appear. **[Thesis]**
- Villas, Leandro A.; Boukerche, Azzedine; de Oliveira, Horacio B.F.; de Araujo, Regina Borges; Loureiro, Antonio A.F. "A Spatial Correlation Aware Algorithm to Perform Efficient Data Collection in Wireless Sensor Networks." *Ad Hoc Networks*, v. 1, p. 10-30, 2011. **[Thesis]**
- Villas, Leandro; Boukerche, Azzedine; Ramos, Heitor; Oliveira, Horacio; de Araujo, Regina; Loureiro, Antonio A.F. "DRINA: A Lightweight and Reliable

Routing Approach for in-Network Aggregation in Wireless Sensor Networks". I.E.E.E. Transactions on Computers (Print), v. 12, p. 1, 2011. [Thesis]

- Nakamura, Eduardo F.; Ramos, Heitor S.; Villas, Leandro A.; de Oliveira, Horacio A.B.F.; de Aquino, Andre L.L.; Loureiro, Antonio A.F. "A reactive role assignment for data routing in event-based wireless sensor networks". Computer Networks, v. 53, p. 1980-1996, 2009.
- Araujo, Regina B.; Villas, Leandro A.; Boukerche, A. "Uma Solução de QoS com Processamento Centrado para Redes de Atuadores e Sensores sem Fio." Revista Brasileira de Redes de Computadores e Sistemas Distribuídos, v. 1, p. 51-60, 2008.

### 7.4.2 Conferences

- Villas, Leandro A. ; Guidoni, Daniel; Boukerche, Azzedine.; Araujo, Regina B.; Loureiro, Antonio A. F. "Um Algoritmo Ciente da Correlação Espaço-Temporal e Consumo de Energia para Coleta de Dados em Redes de Sensores sem Fio." In XXX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos, 2012 (to appear). [Thesis]
- Guidoni, Daniel; Boukerche, Azzedine.; Villas, Leandro A.; Mini, Raquel; Loureiro, Antonio A. F. "A Framework based on Small World Features to Design HSNs Topologies with QoS" In The Seventeenth IEEE Symposium on Computers and Communication (ISCC '12), 2012 (to appear).
- Villas, Leandro A.; Guidoni, Daniel; Boukerche, Azzedine.; Araujo, Regina B.; Loureiro, Antonio A. F. "Dynamic and Scalable Routing to Perform Efficient Data Aggregation in Wireless Sensor Networks." In: IEEE International Conference on Communications ICC 2011, 2011, Kyoto. IEEE International Conference on Communications, 2011. [Thesis]
- Villas, Leandro A.; Boukerche, Azzedine; Guidoni, Daniel L.; de Oliveira, Horacio A.B.F.; Araujo, Regina B.; Loureiro, Antonio A. F., "Time-Space Correlation for Real-Time, Accurate, and Energy-Aware Data Reporting in Wireless Sensor Networks." In: The 14th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, 2011, Miami. [Thesis]
- Villas, Leandro A.; Guidoni, Daniel L.; Boukerche, Azzedine; Araujo, Regina B.; Loureiro, Antonio A.F., "An Energy-Aware Spatial Correlation Mechanism

to Perform Efficient Data Collection in WSNs." In: The 11th IEEE International Workshop on Wireless Local Networks (WLN'11) held in conjunction with The 36th IEEE Conference on Local Computer Networks (LCN'11), 2011, Boon. **[Thesis]**

- Villas, Leandro A.; Guidoni, Daniel L.; Boukerche, Azzedine; Araujo, Regina B.; Loureiro, Antonio A. F., "Explorando a Correlação Espacial na Coleta de Dados em Redes de Sensores sem Fio." In: Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos, 2011, Campo Grande. **[Thesis]**
- Guidoni, Daniel L.; Boukerche, Azzedine; Villas, Leandro A.; MINI, R.; Loureiro, A. A. F. "A Tree-based Approach to Design Heterogeneous Sensor Networks Based on Small World Concepts." In: The 11th IEEE International Workshop on Wireless Local Networks (WLN'11) held in conjunction with The 36th IEEE Conference on Local Computer Networks (LCN'11), 2011, Boon.
- Favarin, G.; Villas, Leandro; Bossonaro, A.; Marcondes, C.; Araujo, R. B. "Exploring Spatial Correlation through Virtual Cells in Wireless Sensor Networks." In: I Brazilian Conference on Critical Embedded Systems 2011, 2011. I Brazilian Conference on Critical Embedded Systems 2011, 2011.
- Villas, Leandro A.; Boukerche, A.; Araujo, Regina B.; Loureiro, Antonio A. F. "Highly Dynamic Routing Protocol for data aggregation in sensor networks." In: IEEE Symposium on Computers and Communications, 2010, Riccione. Computers and Communications (ISCC), 2010 IEEE Symposium on. p. 496-502. **[Thesis]**
- Villas, Leandro A.; Guidoni, Daniel L.; Araujo, Regina B.; Boukerche, Azzedine.; Loureiro, Antonio A. F. "A Scalable and Dynamic Data Aggregation Aware Routing Protocol for Wireless Sensor Networks." In: ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, 2010, Bodrum. The 13-th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, 2010. **[Thesis]**
- Villas, Leandro A.; Araujo, Regina. B.; Ramos, Heitor S.; Loureiro, Antonio A.F. "Um Algoritmo de Roteamento Ciente de Agregação de Dados para Redes de Sensores sem Fio." In: XXVII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos, 2009, Recife. XXVII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos, 2009. v. 1. p. 233-246. **[Thesis]**



- Villas, Leandro A.; Boukerche, Azzedine.; Araujo, Regina B.; Loureiro, Antonio A.F. "A Reliable and Data Aggregation Aware Routing Protocol for Wireless Sensor Networks." In: The 12-th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, 2009, Espanha. The 12-th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, 2009. [Thesis]



# Bibliography

- [AbdelSalam and Olariu, 2009] AbdelSalam, H. S. and Olariu, S. (2009). A lightweight skeleton construction algorithm for self-organizing sensor networks. In *ICC*, pages 1–5. IEEE.
- [Akkaya and Younis, 2005] Akkaya, K. and Younis, M. (2005). A survey on routing protocols for wireless sensor networks. *Ad Hoc Networks*, 3(3):325 – 349.
- [Akyildiz et al., 2002] Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., and Cyirci, E. (2002). Wireless sensor networks: A survey. *Computer Networks*, 38(4):393–422.
- [Akyildiz et al., 2004] Akyildiz, I. F., Vuran, M. C., and Akan, Ö. B. (2004). On exploiting spatial and temporal correlation in wireless sensor networks. In *In Proceedings of WiOpt 2004: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, pages 71–80.
- [Al-Karaki and Kamal, 2004] Al-Karaki, J. and Kamal, A. (2004). Routing techniques in wireless sensor networks: a survey. *Wireless Communications, IEEE*, 11(6):6–28.
- [Al-Karaki et al., 2004] Al-Karaki, J., Ul-Mustafa, R., and Kamal, A. (2004). Data aggregation in wireless sensor networks - exact and approximate algorithms. In *High Performance Switching and Routing, 2004. HPSR. 2004 Workshop on*, pages 241–245.
- [Anastasi et al., 2009] Anastasi, G., Conti, M., Francesco, M., and Passarella, A. (2009). Energy conservation in wireless sensor networks: A survey. *Ad Hoc Networks*, 7(3):537–568.
- [Boukerche et al., 2007] Boukerche, A., Araujo, R. B., and Villas, L. (2007). 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.

- [Boukerche et al., 2003] Boukerche, A., Cheng, X., and Linus, J. (2003). Energy-aware data-centric routing in microsensor networks. In *MSWIM '03: Proceedings of the 6th ACM international workshop on Modeling analysis and simulation of wireless and mobile systems*, pages 42--49, New York, NY, USA. ACM.
- [Chakrabarty et al., 2002] Chakrabarty, K., Member, S., Iyengar, S. S., Qi, H., and Cho, E. (2002). Grid coverage for surveillance and target location in distributed sensor networks. *IEEE Transactions on Computers*, 51:1448--1453.
- [Chandrakasan et al., 2002] Chandrakasan, A. P., Smith, A. C., Heinzelman, W. B., and Heinzelman, W. B. (2002). An application-specific protocol architecture for wireless microsensor networks. *IEEE Transactions on Wireless Communications*, 1:660--670.
- [Chatzigiannakis et al., 2006] Chatzigiannakis, I., Dimitriou, T., Nikolettseas, S. E., and Spirakis, P. G. (2006). A probabilistic algorithm for efficient and robust data propagation in wireless sensor networks. *Ad Hoc Networks*, 4(5):621--635.
- [Chatzigiannakis et al., 2005] Chatzigiannakis, I., Nikolettseas, S., and Spirakis, P. G. (2005). Efficient and robust protocols for local detection and propagation in smart dust networks. *Mob. Netw. Appl.*, 10(1-2):133--149.
- [Deligiannakis and Kotidis, 2008] Deligiannakis, A. and Kotidis, Y. (2008). Geosensor networks. In Nittel, S., Labrinidis, A., and Stefanidis, A., editors, *Book chapter: Exploiting Spatio-temporal Correlations for Data Processing in Sensor Networks*, pages 45--65. Springer-Verlag.
- [Deligiannakis et al., 2004] Deligiannakis, A., Kotidis, Y., and Roussopoulos, N. (2004). Compressing historical information in sensor networks. In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, SIGMOD '04, pages 527--538, New York, NY, USA. ACM.
- [Efthymiou et al., 2006] Efthymiou, C., Nikolettseas, S., and Rolim, J. (2006). Energy balanced data propagation in wireless sensor networks. *Wirel. Netw.*, 12(6):691--707.
- [Fan et al., 2006] Fan, K.-W., Liu, S., and Sinha, P. (2006). On the potential of structure-free data aggregation in sensor networks. In *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1--12.
- [Fasolo et al., 2007] Fasolo, E., Rossi, M., Widmer, J., and Zorzi, M. (2007). In-network aggregation techniques for wireless sensor networks: a survey. *Wireless Communications, IEEE*, 14(2):70--87.

- [Hill et al., 2000] Hill, J., Szewczyk, R., Woo, A., Hollar, S., Culler, D., and Pister, K. (2000). System architecture directions for networked sensors. *SIGPLAN Not.*, 35(11):93--104.
- [Hougardy and Prömel, 1999] 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.
- [Hu et al., 2005] Hu, F., Cao, X., and May, C. (2005). Optimized scheduling for data aggregation in wireless sensor networks. In *ITCC '05: Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05) - Volume II*, pages 557--561, Washington, DC, USA. IEEE Computer Society.
- [III et al., 2007] III, A. F. H., Kravets, R., and Gupta, I. (2007). Building trees based on aggregation efficiency in sensor networks. *Ad Hoc Networks*, 5(8):1317 – 1328. Recent Research Directions in Wireless Ad Hoc Networking.
- [Intanagonwiwat et al., 2002] Intanagonwiwat, C., Estrin, D., Govindan, R., and Heidemann, J. (2002). Impact of network density on data aggregation in wireless sensor networks. In *Distributed Computing Systems, 2002. Proceedings. 22nd International Conference on*, pages 457–458.
- [Intanagonwiwat et al., 2000] Intanagonwiwat, C., Govindan, R., and Estrin, D. (2000). Directed diffusion: a scalable and robust communication paradigm for sensor networks. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 56--67, New York, NY, USA. ACM.
- [Intanagonwiwat et al., 2003] Intanagonwiwat, C., Govindan, R., Estrin, D., Heidemann, J., and Silva, F. (2003). Directed diffusion for wireless sensor networking. *IEEE/ACM Trans. Netw.*, 11(1):2--16.
- [Krishnamachari et al., 2002] 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.
- [Le et al., 2008] Le, T. D., Pham, N. D., and Choo, H. (2008). Towards a distributed clustering scheme based on spatial correlation in wsns. In *Wireless Communications and Mobile Computing Conference, 2008. IWCMC '08. International*, pages 529 – 534.

- [Liu et al., 2007a] Liu, C., Wu, K., and Pei, J. (2007a). An energy-efficient data collection framework for wireless sensor networks by exploiting spatiotemporal correlation. *Parallel and Distributed Systems, IEEE Transactions on*, 18(7):1010–1023.
- [Liu et al., 2007b] Liu, L., Member, S., and Yu, P. S. (2007b). Asap: An adaptive sampling approach to data collection in sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 2007:1766--1783.
- [Min and Chung, 2010] Min, J.-K. and Chung, C.-W. (2010). Edges: Efficient data gathering in sensor networks using temporal and spatial correlations. *J. Syst. Softw.*, 83:271--282.
- [Nakamura et al., 2006] 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 et al., 2009] Nakamura, E. F., Ramos, H. S., Villas, L. A., de Oliveira, H. A. B. F., de Aquino, A. L. L., and Loureiro, A. A. F. (2009). A reactive role assignment for data routing in event-based wireless sensor networks. *Comput. Netw.*, 53:1980--1996.
- [Olariu et al., 2004] Olariu, S., Xu, Q., and Zomaya, A. (2004). An energy-efficient self-organization protocol for wireless sensor networks. In *Intelligent Sensors, Sensor Networks and Information Processing Conference (ISSNIP)*, pages 55–60, Melbourne, Australia. IEEE.
- [Oliveira et al., 2009] Oliveira, H. A., Boukerche, A., Nakamura, E. F., and Loureiro, A. A. (2009). Localization in time and space for wireless sensor networks: An efficient and lightweight algorithm. *Performance Evaluation*, 66(3-5):209 – 222.
- [Pham et al., 2008] Pham, N. D., Le, T. D., Park, K., and Choo, H. (2008). Enhance exploring temporal correlation for data collection in wsns. In *Proceedings of the IEEE International Conference on Research, Innovation and Vision for the Future, RIVF '08*, pages 204--208. IEEE.
- [Pham et al., 2010] Pham, N. D., Le, T. D., Park, K., and Choo, H. (2010). Scs: Spatiotemporal clustering and compressing schemes for efficient data collection applications in wsns. *International Journal of Communication Systems*, 23:1311--1333.

- [Robins and Zelikovsky, 2000] 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.
- [Romer and Mattern, 2004] Romer, K. and Mattern, F. (2004). The design space of wireless sensor networks. *IEEE Wireless Communications*, 11(6):54--61.
- [Schmid and Schossmaier, 2001] Schmid, U. and Schossmaier, K. (2001). How to reconcile fault-tolerant interval intersection with the lipschitz condition. *Distrib. Comput.*, 14:101--111.
- [Shah and Bozyigit, 2007] Shah, G. A. and Bozyigit, M. (2007). Exploiting energy-aware spatial correlation in wireless sensor networks. In *COMSWARE*.
- [Sinalgo, 2008] Sinalgo (2008). Simulator for network algorithms. Distributed Computing Group - ETH-Zurich, last visited in October, 2008.
- [Solis and Obraczka, 2004] Solis, I. and Obraczka, K. (2004). The impact of timing in data aggregation for sensor networks. In *Communications, 2004 IEEE International Conference on*, volume 6, pages 3640--3645 Vol.6.
- [Takahashi, 1980] Takahashi, H., M. A. (1980). An approximate solution for the steiner problem in graphs. *Math. Jap.*, pages 573--577.
- [Villas et al., 2010a] Villas, L., Boukerche, A., de Araujo, R. B., and Loureiro, A. A. F. (2010a). Highly dynamic routing protocol for data aggregation in sensor networks. In *Proceedings of the The IEEE symposium on Computers and Communications, ISCC '10*, pages 496--502, Washington, DC, USA. IEEE Computer Society.
- [Villas et al., 2009] Villas, L. A., Boukerche, A., Araujo, R. B., and Loureiro, A. A. (2009). A reliable and data aggregation aware routing protocol for wireless sensor networks. In *Proceedings of the 12th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems, MSWiM '09*, pages 245--252, New York, NY, USA. ACM.
- [Villas et al., 2011] Villas, L. A., Boukerche, A., de Oliveira, H. A., de Araujo, R. B., and Loureiro, A. A. (2011). A spatial correlation aware algorithm to perform efficient data collection in wireless sensor networks. *Ad Hoc Networks*, (0):--.
- [Villas et al., 2010b] Villas, L. A., Guidoni, D. L., Araújo, R. B., Boukerche, A., and Loureiro, A. A. (2010b). A scalable and dynamic data aggregation aware routing

- protocol for wireless sensor networks. In *Proceedings of the 13th ACM international conference on Modeling, analysis, and simulation of wireless and mobile systems, MSWIM '10*, pages 110--117, New York, NY, USA. ACM.
- [Vuran et al., 2004] Vuran, M. C., Akan, O. B., and Akyildiz, I. F. (2004). Spatio-temporal correlation: theory and applications for wireless sensor networks. *Computer Networks*, 45:245--259.
- [Yoon and Shahabi, 2005] Yoon, S. and Shahabi, C. (2005). Exploiting spatial correlation towards an energy efficient clustered aggregation technique (cag) [wireless sensor network applications]. In *Communications, 2005. ICC 2005. 2005 IEEE International Conference on*, volume 5, pages 3307 – 3313 Vol. 5.
- [Younis et al., 2006] 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.
- [Yu et al., 2006] Yu, J. X., Kitsuregawa, M., and Leong, H. V., editors (2006). *7th International Conference on Advances in Web-Age Information Management*, volume 4016 of *Lecture Notes in Computer Science*. Springer.
- [Yuan and Chen, 2009] Yuan, J. and Chen, H. (2009). The optimized clustering technique based on spatial-correlation in wireless sensor networks. In *Information, Computing and Telecommunication, 2009. YC-ICT '09. IEEE Youth Conference on*, pages 411 –414.