WAGNER MORO AIOFFI

MÉTODOS INTEGRADOS PARA ORGANIZAÇÃO DE REDES DE SENSORES SEM FIO COM SORVEDOURO MÓVEL E CONTROLE DE DENSIDADE

Belo Horizonte

09 de março de 2007

Universidade Federal de Minas Gerais
Instituto de Ciências Exatas
Programa de Pós-Graduação em Ciência da Computação

MÉTODOS INTEGRADOS PARA ORGANIZAÇÃO DE REDES DE SENSORES SEM FIO COM SORVEDOURO MÓVEL E CONTROLE DE DENSIDADE

Dissertação apresentada ao Curso de Pós-Graduação em Ciência da Computação da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

WAGNER MORO AIOFFI

Belo Horizonte

09 de março de 2007

UNIVERSIDADE FEDERAL DE MINAS GERAIS

FOLHA DE APROVAÇÃO

Métodos integrados para organização de Redes de Sensores sem Fio com sorvedouro móvel e controle de densidade

WAGNER MORO AIOFFI

Dissertação defendida e aprovada pela banca examinadora constituída por:

Ph. D. Geraldo Robson Mateus – Orientador
Universidade Federal de Minas Gerais

Dr. Alexandre Salles da Cunha
Universidade Federal de Minas Gerais

Ph. D. Antônio Alfredo Loureiro
Universidade Federal de Minas Gerais

Dr. Luiz Chaimowicz
Universidade Federal de Minas Gerais

Belo Horizonte, 09 de março de 2007

Resumo

Nesse trabalho apresentamos dois novos métodos para organização de **R**edes de **S**ensores **S**em **F**io (RSSF). Esses métodos integram o controle de densidade e mobilidade do sorvedouro para otimizar algumas métricas importantes dessas redes. Em métodos tradicionais, com comunicação multi-saltos, o processo de roteamento de dados é apontado por alguns autores com um grande consumidor de energia. Para reduzir esse consumo, a mobilidade do sorvedouro foi integrada as RSSF possibilitando a coleta de dados utilizando comunicação um-salto. Porém, os métodos propostos apresentam um maior atraso na entegra de dados se comparados aos métodos tradicionais. Para tratar esse problema, propomos o método SHS (*Single-Hop Strategy*) que estende os métodos propostos na literatura com o objetivo de reduzir o atraso na entrega. A imposição da restrição de comunicação um-salto obriga os nós sensores a estarem no raio de comunicação do sorvedouro para reportar seus dados, o que impõe um maior atraso na entrega desses dados. Para aproximar os métodos com sorvedouro móvel ao métodos tradicionais com relaçao ao desempenho no atraso na entrega de mensagens, propomos o MHS (*Multi-Hop Strategy*). O MHS estende o método SHS implementando um esquema de comunicação multi-saltos com o número de saltos limitado. Com isso, reduzimos o tempo de coleta dos dados através de agrupamentos de maior raio. Os dois métodos integram a mobilidade do sorvedouro com controle de densidade, onde o sorvedouro é o responsável pelo controle e implantação das decisões. Os resultados computacionais mostram uma significativa melhora em métricas como tempo de vida das RSSFs, cobertura e conectividade. Entretanto, na métrica de atraso na entrega de dados os métodos com sorvedouro móvel são significativamente menos eficientes que estratégias multi-saltos. Porém, os resultados também mostram que com o ajuste de alguns parâmetros como velocidade do sorvedouro, densidade da rede e o número de sorvedouros, podemos melhorar significativamente o desempenho dessa métrica.

Abstract

In this work we present two new methods for organization of **W**ireless **S**ensor **N**etwork (WSN). These methods integrate the density control and sink mobility to optimize some important WSN' metrics. In traditional methods, with multi-hop communication, the message routing process is pointed, by some authors, as the largest energy consumer. To reduce the routing energy consume, the mobile sink has been introduced in WSN enabling single-hop communication. However, the proposed methods with mobile sink shows a high message delivery delay compared to tradicional methods. To consider this problem, we propose the SHS (Single-hop strategy) method, that extends the existing methods with the objective of reducing message delivery delay. The single-hop communication constraint imposes sensor nodes to be in the sink communication range to report its data, what increases the message delivery delay. To approximate the methods with mobile sink to tradicional methods based on delivery delay performance, we propose the MHS (Multi-Hop Strategy). The MHS method extends the SHS implementing a multi-hop communication with a limited number of hops. With this strategy, the MHS reduces the time needed to collect sensor nodes data and the message delivery delay by increasing the cluster range. Both methods integrate the sink mobility and density control, where the sink is responsible to control and to deploy to the sensor nodes. The computational results show a significant performance increase in metrics such as WSN lifetime, coverage and connectivity. As expected, the proposed methods are not as efficient as the tradicional ones, regarding the delivery delay metric. However, some parameter adjustment such as sink speed, network density and number of mobile sink could increase significantly this metric performance.

Aos meus queridos pais Luiz e Ida.

Agradecimentos

Este trabalho é fruto de dois anos de muito trabalho. Durante esse tempo, muitas pessoas contribuiram direta ou indiretamente para a sua realização. A todas elas gostaria de prestar aqui o meu agradecimento.

Quero começar por algumas pessoas que muito antes de eu imaginar escrever esses agradecimentos já lutavam pelo meu sucesso. Aos meus pais, Luiz e Ida, por empenharem-se tanto na minha formação profissional e, principalmente, pessoal. Vocês são exemplo de luta, perseverança e amor incondicional. Aos meus irmãos Luiz Cláudio, Robson e Wilian, obrigado pelo companheirismo e amizade. Apesar das nossas muitas diferenças, a relação entre nós quatro é eterna, contem sempre comigo.

Aos meus amigos de Aracruz, que, mesmo depois de sete anos de poucos contatos devido à distância física, continuam próximos. A afinidade com vocês foi instântanea, nosso período de convivência juntos nem foi tão grande, mas foi suficiente para ser lembrado para sempre, obrigado, Saurus, Dudu, Tais, Olga, Letícia, Deborah e Larissa.

Aos meus amigos de BH, responsáveis pela minha (quase) adaptação à essa cidade. As pessoas do Synergia, por proporcionarem o melhor ambiente de trabalho que eu já conheci, obrigado por me ajudarem a crescer profissionalmente. A todos do Lapo, que me ajudaram a realizar este trabalho, ou simplesmente estavam lá todos os fins de semana para dividir a sala e tornar o trabalho menos tedioso. Ah, lógico que eu não posso esquecer dos bolos, tortas, salgadinhos, doces e tudo que sempre compartilhamos no laboratório para quebrar a rotina.

Finalmente, gostaria de agredecer a pessoal me levou para o mundo da pesquisa. Prof. Robson, meu orientador de iniciação ciêntifica e projeto orientado na graduação e orientador deste trabalho, agradeço por ter acreditado em mim, por ter sido um orientador presente, e por ter me ajudado a vencer todos os desafios.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Motivação

A associação do avanço em diversas áreas, principalmente microeletrônica, comunicação sem fio e micro sistemas eletromecânicos *(MEMS - Micro Electro-Mechanical Systems)* tem estimulado enormemente o desenvolvimento das **R**edes de **S**ensores **S**em **F**io (RSSF). A crescente redução do tamanho dos dispositivos que compõem o sensor, aumento na capacidade de processamento, aumento na capacidade de armazenamento de dados e desenvolvimentos de baterias capazes de fornecer uma quantidade cada vez maior de energia para os outros dispositivos dos sensores permitem que as RSSFs se apresentem como uma solução para diversas aplicações de monitoração e controle, tais como: monitoração ambiental, monitoração e controle industrial, segurança pública e de ambientes em geral, áreas de desastres e de riscos para vidas humanas, transporte e controle militar.

As RSSFs são um novo tipo de redes sem fio *(MANET - Mobile Ad hoc Network)*. Essas redes são compostas pelos **nós sensores** [1]. Geralmente cada nó sensor é composto por:

- **Processador** com capacidade de processamento limitada;

[1] Nesse texto os termos nó sensor e sensor serão utilizados como sinônimos para designar o elemento computacional com capacidade de processamento, armazenamento e comunicação.

- **Memória** com baixa capacidade de armazenamento de dados;
- **Função de sensoriamento** para aquisição de dados dos eventos monitorados, tais como: acústico, sísmico, infravermelho, vídeo-câmera, temperatura e pressão;
- **Rádio** para comunicação sem fio;
- **Bateria** para prover energia aos outros dispositivos;

As RSSFs diferem de redes de computadores tradicionais em vários aspectos. Normalmente essas redes possuem um grande número de nós (Akyildiz et al., 2002), com autonomia energética limitada. Autonomia é o principal fator limitante do uso das RSSF já que as principais aplicações para elas geralmente projetam um ambiente onde os nós sensores são distribuídos por áreas de difícil acesso impossibilitando a troca ou recarga de sua bateria.

As redes *ad hoc* diferem das redes tradicionais, onde a comunicação entre os seus elementos é feita através de estações rádio base, que constituem uma infra-estrutura de comunicação. Por outro lado, em uma *ad hoc* os seus elementos comunicam entre si formando redes multi-saltos (*multi-hop*) para transmissão dos dados. Organizacionalmente, as RSSFs e MANETs são idênticas, já que possuem elementos computacionais que comunicam diretamente entre si. No entanto, as MANETs têm como função básica prover suporte à comunicação de seus elementos computacionais, que individualmente, podem estar executando tarefas distintas, enquanto as RSSFs tendem a executar uma única função de forma colaborativa. Além de nas RSSFs os elementos computacionais são assumidos serem mais limitados.

Além do nó sensor, as RSSF possuem um ou mais sorvedouro. O sorvedouro é um nó especial, responsável por receber os dados coletados e processados pela RSSF. O sorvedouro é o responsável por transmitir esses dados para os interessados fora da RSSF, funcionado como uma ponte (*gateway*) entre a RSSF e o mundo exterior.

As RSSFs podem ser classificadas em móveis e estáticas dependendo da capacidade de movimentação dos seus elementos. Muitos trabalhos de pesquisa apostam no sorvedouro móvel para o aumento da vida útil dessas redes (Wang et al., 2005a; Jea et al., 2005; Gandham et al., 2003; Kansal et al., 2004; Wang et al., 2005b; Chakrabarti et al., 2003; Jain et al., 2006;

Zhao et al., 2004; Jain et al., 2006). Nesses trabalhos o sorvedouro pode ter sua mobilidade controlada ou não, e no caso de não ser controlável ela pode ser dividida em previsível ou não previsível.

Dadas essas particularidades e recursos limitados, as RSSFs tem estimulado pesquisas em inúmeras áreas do conhecimento como: projeto de hardware de baixo consumo de energia (XBOW, 2006b), desenvolvimento de algoritmos de roteamento (Rajaraman, 2002), otimização de topologia e roteamento (Cardei et al., 2002; Nakamura et al., 2005b; Ye et al., 2002; Zhang e Hou, 2005; Cerpa e Estrin, 2004; Zhou e Krishnamachari, 2003). No geral, o principal objetivo das pesquisas nessa área é o aumento da vida útil da RSSF.

1.2 Objetivos

As pesquisas mais recentes em RSSF têm, geralmente, tratado cada um dos problemas encontrados nessas redes de forma isolada. Recentemente, uma nova área de pesquisa, conhecida como *cross-layer design*, está sendo explorada com o intuito de melhorar o desempenho geral das RSSF em termos de confiança, tempo de vida, entre outros (Siqueira et al., 2006). A idéia básica é romper a divisão de camadas de protocolo de rede promovendo um ganho de desempenho da rede. Por exemplo, em Siqueira et al. (2006) é proposto uma integração entre controle de densidade e roteamento de forma a gerar uma rede mais eficiente tanto em consumo de energia como em confiança na entrega dos dados.

Neste trabalho, utilizamos técnicas de otimização combinatória para resolver diferentes problemas de organização e funcionamento das RSSF de forma integrada, procurando projetar redes com maior longevidade e bom desempenho em métricas como taxa de entrega de mensagens, cobertura da rede entre outras. Para tanto, abordaremos os seguintes problemas:

- **Coleta de dados:** Esse problema consiste em encontrar uma forma do nó sensor enviar os dados coletados por ele ao sorvedouro. Consideramos aqui duas formas de coleta: um-salto (*single-hop*) e multi-saltos (*multi-hop*). Na primeira, o nó sensor estará habilitado a enviar dados para o sorvedouro se ambos estiverem no raio de comunicação

do outro. Na segunda forma, os nós sensores também trabalham como roteadores de dados podendo receber dados de outros nós sensores e entregando-os ao sorvedouro, entretanto o número de enlaces que uma mensagem pode atravessar é limitado.

- **Controle de densidade:** Como o número de nós sensores em uma RSSF pode ser muito elevado, o controle de densidade torna-se um aspecto importante na organização dessas redes (Cerpa e Estrin, 2004; Ye et al., 2002; Zhang e Hou, 2005). Em uma rede muito densa, alguns nós sensores podem trabalhar na mesma área, isso é, mais de um sensor pode está cobrindo a mesma área, ocasionando redundância nos dados coletados além de acarretar um tráfego desnecessário à rede conseqüentemente aumentando o consumo de energia (Tilak et al., 2002). O mecanismo de controle de densidade deve utilizar o menor número de sensores possível na tarefa de cobertura da área sensoriada.

- **Mobilidade do sorvedouro:** O controle da mobilidade do sorvedouro na coleta de dado dos sensores, para as duas formas, é considerado neste trabalho. As duas formas de coleta de dados podem gerar redes desconexas, onde alguns nós sensores ficam sem comunicação com o resto da rede, obrigando o sorvedouro a movimentar-se sobre a área sensoriada para a coleta de dados de todos os nós sensores. Dado que a velocidade de movimentação do sorvedouro é ordens de magnitude menor que a velocidade de comunicação sem fio dos sensores, a otimização da rota do sorvedouro é fator importante para a minimização do atraso na entrega das mensagens. Além disso, nesse trabalho consideramos o caso de RSSFs com um ou mais sorvedouro móvel.

Apesar desses problemas serem independentes, neste trabalho os trataremos em um arcabouço integrado, visando principalmente o aumento da vida útil das RSSFs e a redução do atraso na entrega das mensagens, principal problema relacionado a utilização de sorvedouros móveis (Gandham et al., 2003).

Cabe ressaltar a complexidade dos problemas e algoritmos tratados. O objetivo deste trabalho é uma avaliação dos problemas em RSSF não se preocupando com a eficiência dos algoritmos utilizados, focando nas melhorias que tais algoritmos podem trazer para as RSSF.

Neste trabalho consideramos os problemas citados, de forma integrada. Os resultados

apresentados tratam um cenário onde os sensores possuem uma taxa de geração de mensagens constante e igual em cada nó sensor. Além disso, os nós sensores são distribuidos de forma aleatória, com função densidade de probabilidade uniforme, por toda a área de sensoriamento.

1.3 Contribuições

São propostos dois métodos integrados para a solução dos problemas tratados: o SHS - **S**ingle-**H**op **S**trategy e o MHS - **M**ulti-**H**op **S**trategy. No primeiro é considerado uma RSSF com estratégia de coleta de dados onde as mensagens são entregues ao sorvedouro com um único salto. O segundo considera uma RSSF com estratégia de coleta de dados onde as mensagens são entregues ao sorvedouro com multiplos saltos, porém o número de saltos é limitado.

O SHS otimiza ao máximo o consumo de energia da RSSF reduzindo o consumo de dados com roteamento de mensagens. Esse método estende as soluções existentes na literatura para organização de RSSF com sorvedouros móveis com o objetivo de reduzir o atraso esperado na entrega de mensagens apresentado por esses métodos, mantendo o baixo consumo de energia.

O MHS apresenta uma solução intermediária ao SHS e métodos onde as mensagens são roteadas através dos sensores, de forma convencional, dos nós sensores até o sorvedouro (sem limite). O método MHS estende o SHS, relaxando a restrição de um único salto possibilitando o roteamento de mensagens pelos nós sensores, porém o número de saltos que uma mensagem pode atravessar até chegar ao sorvedouro é limitado. O MHS considera os objetivos conflitantes entre a otimização do consumo de energia e o atraso na entrega das mensagens.

O cenário tratado é mostrado na figura 1.1.

Este trabalho apresenta as seguintes contribuições:

- a integração do controle de densidade com a mobilidade do sorvedouro, onde este é o responsável pela implantação da solução do controle de densidade na RSSF.

- roteamento do sorvedouro visando a redução do tempo de ciclo de coleta do mesmo, sendo o problema modelado como o Problema do Caixeiro Viajante;
- um controle eficiente do estado dos três dispositivos básicos do nó sensor (rádio, bateria e função de sensoriamento) aumentando a eficiência energética da RSSF;
- além de extensiva avaliação experimental dos resultados.

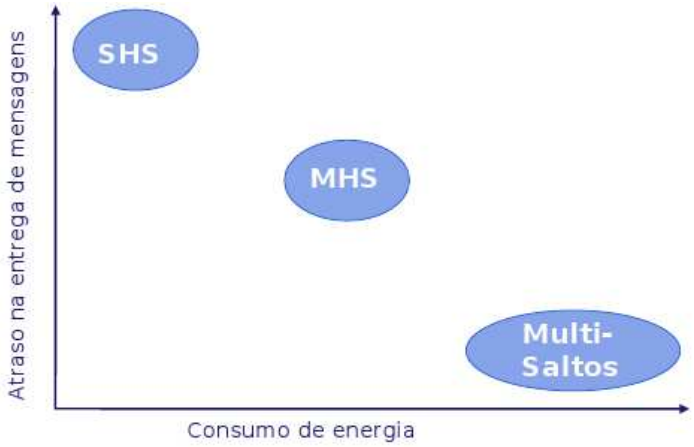

Figura 1.1: Cenário de pesquisa

1.4 Organização da dissertação

Esse documento está organizado como segue. O Capítulo 2 apresenta os fundamentos das RSSF, seus principais problemas, as simplificações e suposições feitas nesse trabalho. O Capítulo 3 apresenta uma revisão bibliográfica detalhada de cada um dos problemas tratados nesse trabalho, tanto da área de RSSF como a revisão de todos os modelos e algoritmos de otimização combinatória utilizados como base para a solução de tais problemas. No Capítulo 4 são apresentadas as diversas abordagens utilizadas para a solução integrada dos problemas apresentados, tanto para um ou mais *sorvedouros*, com redes um-salto e redes multi-saltos. No Capítulo 5 é apresentada a metodologia de avaliação experimental utilizada para validar e avaliar os métodos propostos. No Capítulo 6 os principais resultados computacionais das nossas avaliações são apresentados. Finalmente, no Capítulo 7 é apresentado as conclusões e considerações finais além de possíveis extensões à este trabalho.

Capítulo 2

Rede de sensores sem fio

As RSSFs são compostas por um grande número de elementos computacionais, os **nós sensores**, dotados de capacidade de monitorar algum tipo de evento físico e reportar através de comunicação sem fio dados sobre esses eventos a um observador.

Além disso, nessas redes existe um ou mais elementos computacionais responsáveis por coletar os dados enviados pelos nós sensores e disponibilizá-los para um observador externo a RSSF, o **nó sorvedouro**.

Entre as possíveis aplicações das RSSFs podemos citar:

- **Aplicações militares:** As RSSF podem ser utilizadas em operações militares para diversos fins, como: rastreamento e monitoração de tropas inimigas; monitoramento de áreas para detecção de radioatividade e gases tóxicos; controle de fronteiras, captação de comunicação sem fio de tropas inimigas, dentre outras.
- **Monitoração de ambientes:** Devido às suas características, como pequenas dimensões, capacidade de leitura de dados físicos e comunicação sem fio, os nós sensores podem ser utilizados na monitoração de ambientes de difícil penetração humana ou de grandes dispositivos de monitoramento. As RSSF podem ser utilizadas por exemplo, para a monitoração de temperatura, luminosidade, pressão, níveis de gases no ar entre outros

eventos físicos de uma floresta tropical. Esses dados podem ajudar na preservação de tais florestas, ajudando no controle de focos de incêndio entre outros.

- **Aplicações médicas:** À medida que os nós sensores tem seu tamanho reduzido, novas aplicações se tornarão possíveis, como por exemplo aplicações médicas. No futuro, será possível colocar nós sensores dentro do corpo de um animal ou até mesmo de um ser humano, permitindo que eles formem uma RSSF que poderá monitorar vários aspectos como níveis de oxigênio, insulina ou colesterol, além do volume e tamanho dos órgãos.

Para todas as aplicações já conhecidas de RSSF, uma característica se destaca: a cooperação entre os nós sensores. As tarefas de uma RSSF são basicamente executadas de forma cooperativa entre todos os nós sensores. Cada sensor é responsável por uma parte da tarefa principal da RSSF, e eventualmente responsável por algumas tarefas administrativas, como por exemplo rotear dados de outros nós sensores ou ser o líder de um agrupamento de nós sensores (*cluster-head*), responsável por organizar os nós sensores do seu agrupamento e reportar ao sorvedouro os dados recolhidos pelos nós sensores do agrupamento.

Os nós sensores, principais componentes da RSSF, são elementos computacionais de pequenas dimensões, com baixo poder de processamento, memória limitada e comunicação sem fio. Devido às suas dimensões reduzidas, a quantidade de energia que um nó sensor pode carregar é limitada. A pouca quantidade de energia é o principal fator limitante do uso de um nó sensor, e a economia de energia é o principal foco de pesquisa na área de RSSF.

As características das RSSF somadas às restrições inerentes ao nó sensor abrem inúmeras possibilidades de pesquisa, visando aprimorar técnicas de redes *ad hoc* a realidade das RSSF.

A estrutura e a modelagem dos elementos que compõem as RSSFs, o nó sensor e o sorvedouro, são de fundamental relevância para esse trabalho. Nas próximas seções descreveremos os principais componentes desses elementos, bem como alguns modelos utilizados para simular seus comportamentos em nosso trabalho.

2.1 Sensor

Nesta seção descreveremos o modelo de sensor utilizado nesse trabalho e os modelos de consumo de energia e propagação de sinal adotados.

Nosso modelo é baseado na linha de nós sensores comercialmente disponíveis, a linha Motes. A linha Motes foi desenvolvida por pesquisadores da Universidade de Berkeley (cotsdust, 2003). Atualmente a linha é comercializada pelo fabricante Xbow (XBOW, 2006b). Dentre os modelos disponíveis comercialmente destacam-se os nós sensores Mica2 (Figura 2.2), Mica2DOT e a mais nova geração o MicaZ (Figura 2.3). Na Figura 2.1 é mostrado o RF mote, o primeiro sensor da linha.

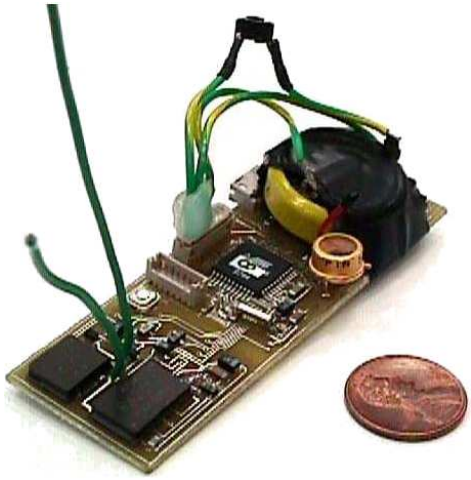

Figura 2.1: Primeira geração da linha Mote (RF Mote).

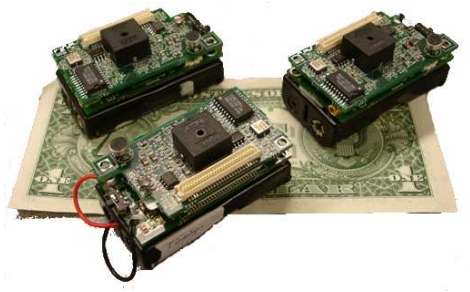

Figura 2.2: Mica mote 2.

O nó sensor é o principal componente computacional de uma RSSF. Cada um desses elementos é composto por 5 unidades básicas: processador, memória, placa de sensoriamento, rádio e bateria como mostra a Figura 2.4.

Figura 2.3: MicaZ.

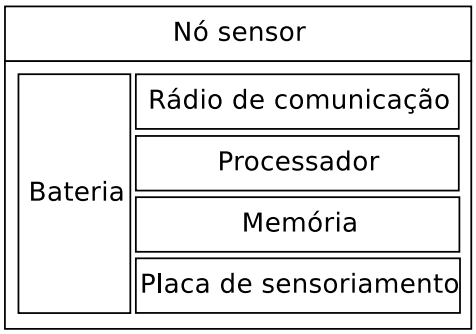

Figura 2.4: Elementos que compõem um nó sensor.

2.1.1 Componentes

2.1.1.1 Processador

O processador de um nó sensor apresenta inúmeras restrições quanto a instruções e principalmente quanto a capacidade computacional. A linha Motes é equipada com processadores Atmel ATMega 128L cujas características estão listadas na tabela 2.1. Esse processador possui uma arquitetura RISC com um conjunto de 133 instruções, a maioria sendo executada em um único ciclo de *clock*.

Freqüência	4 ou 8 MHz
Memória de programa	128 Kbytes
Memória de dados	512 Kbytes

Tabela 2.1: Características do processador Atmel ATMega 128L

2.1.1.2 Placa de sensoriamento

A placa de sensoriamento é responsável pela função de monitoramento do nó sensor. Ela é a responsável por extrair alguma informação do ambiente sobre eventos de interesse. Existem hoje uma grande variedade de tipos de placas de sensoriamento que colotam diferentes dados como: temperatura, luminosidade, pressão, níveis de gases no ar, aceleração, magnetismo, GPS, acústico entre outros.

Um sensor geralmente possui um tipo específico de monitoramento, porém algumas pesquisas recentes sugerem a utilização de mais de uma dessas placas por sensor dando a cada sensor a possibilidade de monitorar mais de um fenômeno (Dasgupta et al., 2003). Existem no mercado algumas alternativas para multi-sensoriamento por um único sensor, por exemplo a placa de aquisição de dados *MICA2DOT Multi-Sensor Module* da XBOW (2006b).

2.1.1.3 Rádio

O rádio dos nós sensores da linha Motes operam em múltiplos canais, divididos em faixas de freqüência distintas. O rádio possui capacidade de envio de dados com diferentes potências de transmissão, aumentando ou diminuindo o raio de alcance do sinal. Segundo o fabricante, o rádio da linha Motes pode alcançar de 75 a 100 metros em um ambiente aberto.

2.1.1.4 Bateria

Todos os elementos descritos anteriormente são consumidores de energia do nó sensor. A energia para esses dispositivos é fornecida pela bateria. A linha Motes utiliza como fonte de energia duas pilhas do tipo AA. Na próxima seção é descrito o modelo de consumo de energia adotado para cada nó sensor.

2.1.2 Modelo de consumo de energia

O nó sensor é um sistema complexo de hardware. Para esse trabalho, o consumo de energia desse sistema foi modelado de forma simplificada para a avaliação experimental, via simulação, de nossas propostas.

Nesse trabalho consideramos cinco fontes de consumo de energia entre as tarefas do nó sensor: transmissão de mensagem, recepção de mensagem, escuta do canal de comunicação, processamento, e sensoriamento.

Park et al. (2001) propõem três modelos de bateria baseados em seu comportamento de descarga da energia. Nesse trabalho utilizamos o modelo de bateria com descarregamento linear, onde a diferença de potencial (medida em Volts) permanece constante ao longo do tempo de vida útil da bateria. Esse modelo é uma boa aproximação para o comportamento de descarregamento de baterias, sua limitação fica na consideração da diferença de potencial constante, o que não acontece na prática em geral.

A capacidade de uma bateria é medida em Ah (Ampere * Hora). Dado um bateria com uma capacidade C, e um sistema com corrente de descarga I, temos o tempo de vida útil da bateria T dado por:

$$T = \frac{C}{I} \tag{2.1}$$

Como utilizamos um modelo de descarga linear, depois de um período de operação t_d, a carga restante R na bateria é dado por:

$$R = C_{t_0} - \int_{t_0}^{t_0+t_d} I(t)dt \tag{2.2}$$

Onde $I(t)$ é a corrente consumida no instante t e C_{t_0} é a carga da bateria no período t_0. Considerando-se uma corrente constante, temos a equação 2.2 reduzida a:

$$R = C_{t_0} - It_d \tag{2.3}$$

Nas próximas sub-seções descreveremos as equações utilizadas para o cálculo do consumo de energia do nó sensor para cada uma das fontes de consumo tratadas nesse trabalho.

2.1.2.1 Transmissão

O processo de transmissão, em termos de energia, pode ser simplificado em uma tarefa do sensor de transmitir dados por um tempo t_t (h) a uma potência especificada p (dBm).

A carga consumida para transmissão (C_t) pode ser calculada como:

$$C_t(t_t) = f_t(p) * t_t \tag{2.4}$$

Onde a função $f_t(p)$ relaciona a potência de transmissão com a corrente necessária, dado uma diferença de potencial fixa. Para o Mica2, essa relação pode ser encontrada na especificação do produto em XBOW (2006a).

2.1.2.2 Recepção

O processo de recepção, em termos de energia, é muito parecido com o processo de transmissão.

Porém, vale lembrar que para uma única transmissão podemos ter várias recepções. Uma mensagem enviada por um nó sensor é ouvida por todos os nós sensores que estiverem dentro do raio de comunicação do nó sensor transmissor.

Considerando a recepção de uma mensagem durante o tempo t_r (h) em um nó sensor, a carga consumida é dada por:

$$C_r(t_r) = I_r * t_r \tag{2.5}$$

Onde I_r é a corrente requerida para a recepção de dados em um nó sensor.

2.1.2.3 Escuta do canal de comunicação

Além dos gastos por transmissão e recepção de dados, o rádio apresenta outra fonte de consumo de energia, a escuta do canal de comunicação.

O rádio necessita verificar continuamente o canal de comunicação para detectar a chegada de novas mensagens. Para isso ele deve manter ligado o seu sistema de rádio, a um consumo de corrente I_e.

Podemos determinar a carga consumida por um nó sensor na escuta do canal pelo tempo t_e (h) por:

$$C_e(t_e) = I_e * t_e \tag{2.6}$$

Consideramos um consumo de energia nulo quando o rádio do nó sensor está desativado.

2.1.2.4 Processamento

O processador possui um consumo de corrente dado por I_p durante o tempo de seu funcionamento.

A carga necessária para manter o processador do nó sensor ligado durante um tempo t_p é dado por:

$$C_p(t_p) = I_p * t_p \tag{2.7}$$

Consideramos um consumo de energia nulo quando o processador do nó sensor está desativado. Nesse trabalho o processador estará ativo durante todo o tempo em que o rádio ou a placa de sensoriamento estiverem ativos.

2.1.2.5 Sensoriamento

Por fim, a função de sensoriamento também consome energia na sua tarefa de monitorar o ambiente para a captura de valores referentes a possíveis eventos físicos observados.

Dado uma corrente I_s necessária para realizar o sensoriamento durante um tempo t_s, a carga consumida é dada por:

$$C_s(t_s) = I_s * t_s \tag{2.8}$$

Nesse trabalho, consideramos que um sensor ativo realiza sensoriamentos a uma taxa constante de 1/20 Hz, e cada sensoriamento dura 3 s, ou seja, $t_s = 3$.

2.1.2.6 Cálculo do consumo

Dadas as equações 2.4, 2.5, 2.6, 2.7 e 2.8 temos o consumo de energia de um nó sensor durante um tempo de funcionamento t dado por:

$$C = C_t(t_t) + C_r(t_r) + C_e(t_e) + C_p(t_p) + C_s(t_s) \tag{2.9}$$

onde t_t, t_r, t_e, t_p e t_s são frações do tempo de funcionamento t.

2.1.3 Modelo de propagação de sinal

O modelo adotado para a propagação do sinal de comunicação sem fio foi o *Free-space* (Balanis, 2005; Lin, 1998).

O *free-space* é um modelo de atenuação da potência do sinal onde os efeitos de absorção, difração, obstrução, refração e reflexão são suficientemente removidos e não têm efeito sobre a propagação do sinal. Como o nome indica, o *free-space* assume que tanto o transmissor quanto o receptor do sinal está em um ambiente livre. Sendo assim, a perda de sinal nesse modelo é proporcional ao quadrado da distância entre o transmissor e o receptor e também proporcional ao quadrado da freqüência do sinal de rádio transmitido.

A perda no *free-space* é causada pela dissipação da energia sobre áreas grandes, proporcional à distância a fonte emissora do sinal e inversamente proporcional à lei dos quadrados da radiação eletromagnética.

A equação da perda no *free-space*, para um tamanho de onda λ, uma freqüência de propagação de sinal f e uma distância do emissor R, e sendo c a velocidade de propagação da luz no ambiente considerado, é dada por (Lin, 1998):

$$FS = \left(\frac{4\pi R}{\lambda}\right)^2 = \left(\frac{4\pi Rf}{c}\right)^2 \tag{2.10}$$

Uma forma conveniente de expressar em termos de dB, a perda de potência do sinal de propagação em função da distância d entre o transmissor e receptor, é dada pela seguinte equação:

$$FS(d) = 20\log_{10} d + 20\log_{10} f + K \tag{2.11}$$

Onde K é uma constante que depende dos rádios usados.

2.2 Sorvedouro

O nó sorvedouro é um tipo especial de nó sensor, responsável por coletar os dados dos outros nós sensores da rede. O sorvedouro apresenta praticamente as mesmas características do nó sensor apresentadas na seção 2.1. Nessa seção apresentaremos as diferenças entre o nó sensor e o sorvedouro.

Em modelos de RSSFs atuais, o nó sorvedouro não possui as restrições de energia do nó sensor. Ele é considerado um dispositivo com capacidade infinita de energia, e logo, esse fator não é considerado nesse trabalho. Isso se deve ao fato de em redes com sorvedouros fixos estarem posicionado na borda do ambiente de sensoriamento eles podem ser inspecionados, reparados ou substituídos a qualquer momento. Em redes que consideram os sorvedouros móveis eles geralmente estão associado a um dispositivo com capacidades sofisticadas, como robos terrestres ou aéreos, e como acontece no primeiro caso, os sorvedouros podem ser inspecionados, reparados ou substituídos.

Além disso, consideramos o sorvedouro como um dispositivo móvel. Ele possui a capacidade de se movimentar em todas as direções. Independente da forma de movimentação, consideramos aqui que o sorvedouro é capaz de se movimentar em linha reta de um ponto a outro de um espaço plano a uma velocidade constante.

Porém, neste trabalho não tratamos questões inerentes a essa movimentação como por exemplo instruções para desvio de obstáculos. Consideramos, sempre, que o caminho entre dois pontos no espaço é uma linha reta e nenhum obstáculo existe nesse caminho.

Alguns trabalhos de pesquisa, na área de robótica, oferecem suporte ao requisito de movimentação do sorvedouro. O projeto *CotsBots* (CotsBots, 2006) é um exemplo. Nele, um nó sensor é acoplado a um veículo terrestre de pequenas dimensões como mostra a figura 2.5.

Uma outra possibilidade é a utilização de veículos aéreos, como por exemplo um dirigível. Em uma estrutura inflável, de pequenas dimensões com navegação controlável um sorvedouro pode ser acoplado, como mostrado pela Figura 2.6.

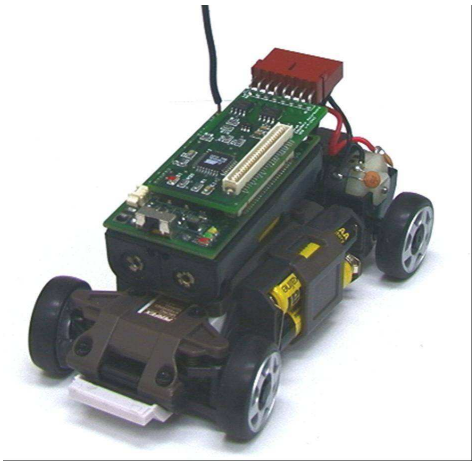

Figura 2.5: Sorvedouro móvel terrestre - Projeto CotsBots.

Figura 2.6: Sorvedouro móvel aéreo - Dirigível

Capítulo 3

Trabalhos relacionados

Nesse capítulo faremos uma revisão bibliográfica dos diversos problemas abordados nesse trabalho, abordando os artigos mais relevantes e referências a trabalhos recentes sobre cada um deles. Aqui tratamos algumas publicações sobre mobilidade, controle de densidade e protocolos de comunicação em RSSF, além dos problemas de otimização combinatória que dão base para nossas soluções.

3.1 Mobilidade em RSSF

Diversos trabalhos vem utilizando dispositivos móveis, principalmente sorvedouro móvel, para melhorar o desempenho das RSSF. Geralmente o objetivo desses estudos é o aumento do tempo de vida dessas redes (Wang et al., 2005a; Jea et al., 2005; Gandham et al., 2003; Wang et al., 2005b; Chakrabarti et al., 2003; Jain et al., 2006).

Esses trabalhos podem ser classificados de acordo com o tipo de mobilidade do dispositivo móvel: controlável e não controlável; previsível e não previsível.

Um dispositivo móvel pode ter sua mobilidade controlável ou não, podendo por exemplo ser instalado um sorvedouro móvel sobre um animal cuja mobilidade não é controlável.

Para os dispositivos com mobilidade não controláveis, podemos dividi-los em com mobilidade previsível e não previsível. No exemplo anterior é muito complicado prever a mobilidade de um animal em um ambiente livre, mas é fácil prever a mobilidade de um ônibus, por exemplo, apesar de um ônibus em uma linha publica de transporte não poder ser controlado pela aplicação da RSSF para melhor coletar seus dados.

Em Shah et al. (2003); Jain et al. (2006); Small e Haas (2003); Juang et al. (2002) elementos móveis presentes no ambiente de sensoriamento são utilizados para movimentação do sorvedouro, como animais, carros, entre outros. Esses trabalhos utilizam movimentação não controlável e não previsível.

Utilizando um esquema de movimentação não previsível temos uma impossibilidade de determinar um limite máximo para a latência das mensagens. Isso porque não podemos determinar o caminho do dispositivo móvel assim não podemos determinar quando ele entregará as mensagens para o destino final.

Em Chakrabarti et al. (2003) dispositivos móveis não controláveis são utilizados, porém a movimentação desses dispositivos pode ser prevista. Uma rede de sorvedouros montados sobre uma rede de transportes públicos, com ônibus com padrão de movimentação bem definidos, é utilizada. Os nós sensores dessa rede determinam os períodos em que têm conectividade com um desses dispositivos móveis e se programam para ativar seus rádios nesses períodos para reportar seus dados.

Dispositivos móveis controláveis são utilizados em alguns trabalhos como Kansal et al. (2004); Jea et al. (2005) onde robôs são utilizados para movimentação do sorvedouro. A velocidade é controlada para aumentar a eficiência da RSSF.

Alguns trabalhos utilizam elementos móveis somente para transportar dados de um nó para outro, em redes onde cada nó tem um pequeno raio de comunicação. Em Zhao et al. (2004) todos os elementos são móveis, tanto o sorvedouro quanto os nós sensores. Eles se organizam para tornar a rede mais eficiente na coleta dos dados.

Já em Wang et al. (2005a) somente alguns dispositivos são móveis e trabalham como um

relay. Ele se posiciona ao lado de um nó sensor para substituir esse nó nas suas tarefas de roteamento de dados durante um certo período. Assim durante esse período o nó sensor fica desobrigado de rotear mensagens podendo economizar energia. Esse trabalho apresenta um limite teórico para a melhoria do tempo de vida da rede em um fator de 4, comparado a uma RSSF totalmente estática. Esse aumento é encontrado com o relay móvel a no máximo dois hops de distância do sorvedouro fixo.

Muitos trabalhos tratam a mobilidade do sorvedouro, propondo protocolos de comunicação que dêem suporte para a mobilidade, mas pouca atenção é dada ao problema de planejamento de trajetória do sorvedouro. O planejamento de trajetória do sorvedouro móvel é um problema pouco explorado na literatura. Em Jea et al. (2005) é assumido uma trajetória linear para o sorvedouro móvel, reduzindo o problema do planejamento de trajetória ao controle de velocidade com a qual o sorvedouro se move. Também em Kansal et al. (2004) é apresentado um novo esquema de controle de velocidade, porém somente isso é considerado no controle de trajetória.

Nesse trabalho, abordaremos o planejamento de trajetória do sorvedouro utilizando problemas clássicos de otimização combinatória.

3.2 Controle de densidade

Em uma RSSF densa, podemos ter vários nós sensores trabalhando sobre uma mesma região em um mesmo intervalo de tempo. Isso gera dados redundante e, principalmente, mais tráfego na rede (Cardei et al., 2002).

Mecanismos de controle de densidade gerenciam o estado dos diversos nós da rede deixando somente um pequeno conjunto deles ativos, reduzindo essa redundância. Diversas propostas de controle de densidade existem na literatura, tanto centralizadas (Cardei et al., 2002; Slijepcevic e Potkonjak, 2001; Nakamura et al., 2005b) quanto distribídas (Ye et al., 2002; Zhang e Hou, 2005; Siqueira et al., 2006; Cerpa e Estrin, 2004; Xing et al., 2005).

Em Slijepcevic e Potkonjak (2001) uma heurística centralizada é proposta para dividir o conjunto de nós sensores em sub-conjuntos mutuamente exclusivos, onde cada sub-conjunto de nós sensores cobre totalmente a área de sensoriamento. O objetivo é maximizar o número de sub-conjuntos. Essa estratégia é capaz de aumentar o tempo de vida da RSSF em até n vezes, onde n é o número de sub-conjuntos em que a rede foi dividida.

Zhang e Hou (2005) propuseram o algoritmo distribuído OGDC. A idéia básica do algoritmo é manter nós temporariamente inativos, enquanto eles não são necessários para garantir cobertura e conectividade da RSSF. Cada nó sensor sabe sua posição geográfica, mas utiliza somente informações locais para decidir se estará ativo ou não no próximo período de funcionamento da rede. O OGDC requer um relógio global sincronizado para iniciar cada período simultaneamente em todos os nós sensores.

Em Siqueira et al. (2006) uma proposta de integração *cross-layer* entre o algoritmo de controle de densidade OGDC e o algoritmo de roteamento EFTREE (Figueiredo et al., 2004) é apresentado. Como resultado dessa integração há um ganho considerável na taxa de entrega de mensagens, já que sensores desativados pelo controle de densidade não são considerados para o roteamento, sendo a decisão do controle de densidade e roteamento integrada.

Em Megerian e Potkonjak (2003); Nakamura et al. (2005b) são propostos alguns modelos de Programação Linear Inteira (PLI) para cobertura e agendamento de tarefas para resolver o problema de cobertura em RSSFs. Li et al. (2002) propõem um algoritmo distribuído utilizando um grafo da vizinhança do nó para resolver o problema de cobertura como definido em Meguerdichian et al. (2001).

Nesse trabalho, o problema de controle de densidade será modelado como o problema de otimização clássico de cobertura de conjuntos. Através dessa técnica a cobertura ótima será encontrada.

3.3 Protocolos de comunicação

Nesse trabalho, abordaremos muito pouco os protocolos de comunicação. Apoiaremos nossas abordagens em protocolos já existentes, como protocolos de roteamento de mensagens em árvores, como o algoritmo apresentado em Figueiredo et al. (2004), e protocolos de camada de enlace como o 802.11 e esquemas de transmissão de dados como o TDM (*Time Division Multiplexing*).

A maioria dos trabalhos em RSSF com sorvedouro móvel utiliza comunicação um-salto, onde os nós sensores comunicam-se somente com o sorvedouro de forma direta. Porém, essa restrição impõe a rede um alto atraso na entrega das mensagens.

Algumas propostas recentes utilizam comunicação multi-saltos com sorvedouro móvel, como por exemplo o trabalho Gandham et al. (2003). A comunicação multi-saltos é a mais utilizada em RSSF sem mobilidade do sorvedouro. Nessas redes, os nós sensores devem rotear mensagens de outros nós para que essas cheguem a seu destino final. Alguns algoritmos de roteamento destacam-se nesse cenário.

A área de algoritmos de roteamento provavelmente é a mais estudada em RSSF. Nesse campo temos diversas propostas de roteamento para diferentes fins. Revisão de publicações sobre algoritmos de roteamento podem ser encontradas em Rajaraman (2002)

3.4 Agrupamento

Em muitas pesquisas, o agrupamento (*cluster*) de nós sensores é posposto para organizar hierarquicamente a topologia das RSSFs. Essa organização tem levado a uma variedade de melhorias nas RSSFs (Heinzelman et al., 2002).

Nossas abordagens possuem duas estratégias de agrupamento de nós sensores. Na primeira o agrupamento é feito em agrupamentos com raio máximo R, onde esse raio é definido pelo raio de comunicação dos nós sensores. Na segunda o agrupamento é feito em árvores com no

máximo um número λ de saltos, com raiz em p sensores.

A primeira abordagem é modelada como o problema *min-size k-clustering problem* (Bilò et al., 2005), onde é dado um conjunto S de sensores e $dist(s_i, s_j)$ é uma função que determina a distância entre os sensores s_i e s_j, e o objetivo é construir o menor número de agrupamentos possível com raio de cada agrupamento limitado por R. Essa estratégia é bem similar a adotada em Heinzelman et al. (2002), porém naquele trabalho a construção dos agrupamentos é distribuído sem garantia de ótimalidade.

A segunda abordagem é modelada como o problema *inverse p-Center* (Mirchandani e Francis, 1990), onde é dado um conjunto S de sensores, e o objetivo é identificar um número mínimo p de raízes de árvores tal que todos os nós sensores podem se comunicar, com um número limitado λ de saltos.

A formação de p árvores é similar a formação da árvore de coleta de dados dos algoritmos de roteamento em árvore como em Figueiredo et al. (2004). Nossa abordagem explora uma ou mais árvores de coleta, sendo esse cenário o preferencial. A existência de mais de uma árvore de coleta está ligada ao limite máximo de saltos.

3.5 Problemas de Otimização combinatória

Para a solução dos problemas de RSSF abordados nesse trabalho, basicamente utilizaremos a solução de alguns problemas de otimização combinatória. Problemas, como controle de densidade, planejamento de rota e agrupamento de nós sensores serão modelados como problemas clássicos e resolvidos através algoritmos apresentados na literatura.

3.5.1 Caixeiro viajante

Nesse trabalho, tratamos o problema de planejamento de rotas para o sorvedouro móvel. A estratégia é modelá-lo como o problema do Caixeiro Viajante.

O Problema do Caixeiro Viajante (PCV) (*Traveling Salesman Problem*) é um dos mais tradicionais e conhecidos problemas de otimização combinatória. Dado um conjunto de cidades e suas conexões com seus respectivos pesos, o problema consiste em encontrar o ciclo hamiltoniano (ciclo que passe uma única vez por todas as cidades) de menor custo. Mais formalmente, dado um grafo $G = (V, E)$, onde $V = \{1, ..., n\}$ é um conjunto de nós e E é o conjunto de arestas que ligam esses nós com seus respectivos pesos d_{ij}, o problema consiste em achar o menor ciclo hamiltoniano possível em G (Wagner, 1969; Christofides, 1979; Nemhauser e Wolsey, 1988).

Nas décadas de 1930 a 1950 diversos artigos trataram o problema. O mais significativo deles (Dantzig et al., 1954) apresenta a solução para instâncias do problema com até 49 cidades utilizando uma formulação de Programação Linear Inteira Mista (PLIM). Porém a formulação apresentada por eles possui um número exponencial de restrições em relação à quantidade de vértices.

De 1950 até os dias de hoje o PCV continua sendo extensivamente estudado. Atualmente a maior instância cuja solução ótima é conhecida possui 24.978 cidades, representando um ciclo sobre pontos da Suécia (Traveling Salesman Problem Home-Page - Princeton University, 2003). Nessa página podem ser encontradas informações recentes sobre o TSP, como instâncias de teste, instâncias reais com cidades de determinadas nações do mundo como: Alemanha, Suíça, Estados Unidos e Argentina. A solução dessas instâncias, informações sobre algoritmos e soluções para o problema também podem ser encontradas no mesmo endereço eletrônico.

Em nosso trabalho, devido ao agrupamento dos nós sensores, o problema de planejamento de rotas é limitado a poucas cidades, sendo possível tratar a solução dessas instâncias com algoritmos exatos, sem portanto acarretar um tempo de computação elevado.

3.5.2 Cobertura de conjuntos

Em nossas abordagens, o problema de controle de densidade será modelado como um problema de cobertura mínima de conjuntos e toda informação necessária para realizar um controle de

cobertura centralizado é disponibilizada.

O problema de cobertura de conjuntos (*Set cover problem*) consiste em encontrar um sub-conjunto de pontos que cubram todo o espaço de observação. Mais formalmente, o problema é defindo como segue: Seja um conjunto U de pontos e um conjuto S de sub-conjuntos de U, uma cobertura mínima é dada por um sub-conjunto $C \subseteq S$ tal que $\{\bigcup C_i, \forall_{C_i \in C}\} = U$ e $|C| = k$.

A solução do problema pode ser dada por meio de programação matemática, resolvendo-se o seguinte modelo:

$$minimizar \sum_{s \in S} y_s$$

Sujeito a:

$$\sum_{s \in S \text{ e } d \in D_s} x_s^d \geq 1, \qquad \forall d \in D$$

$$x_s^d \leq y_s, \quad \forall s \in S \; \forall d \in D_s$$

$$x \in \{0, 1\}$$

$$y \in \{0, 1\}$$

onde S é o conjunto de sensores e c é o raio de sensoriamento de cada sensor. Cada sensor cobre um sub-conjunto de pontos de demanda $D_s = \{d \in D | dist(d, s) \leq c\}$, onde $dist(d, s)$ é a distância euclidiana entre o ponto de demanda $d \in D$ e o sensor $s \in S$.

Uma variante do problema de cobertura de conjuntos é o problema de cobertura de conjuntos ponderado, onde cada elemento possui um peso (w_s) para sua utilização na cobertura. Com isso, o objetivo é minimizar a soma dos pesos ao invés de minimizar o número de elementos como no problema original. O modelo matemático é apresentado no Capítulo 4.

3.5.3 p-Centros

O problema de p-Centros consiste em localizar p centros em um grafo de modo a minimizar a distância máxima entre um nó qualquer desse grafo a algum dos p centros.

Um aplicação comum do problema de p-Centros é a localização de unidades de serviços de emergência em redes de transporte, por exemplo, localizar p batalhões do corpo de bombeiros em uma comunidade rural de maneira a minimizar o tempo máximo de resposta de um desses p batalhões para todas as fazendas da comunidade.

Um problema inverso ao p-Centros, o *inverse p-Center* é utilizado em nossas abordagens para a localização de p pontos de parada para o sorvedouro móvel. Diferentemente do problema de p-Centros, onde o objetivo é localizar p nós, no p-Centros inverso o objetivo é determinar e localizar o menor número p de centros que atendam todos os nós de demanda do grafo com uma distância máxima λ.

O problema de p-Centros invertido é comumente resolvido por uma redução ao problema de cobertura mínima de conjuntos, como descrito em Mirchandani e Francis (1990). Com isso podemos resolver esse problema com um dos dois modelos apresentados na secção 3.5.2.

Capítulo 4

Abordagens

Neste capítulo descreveremos os problemas tratados e a forma de integração entre eles que dá origem a dois métodos para a organização de RSSFs com controle de densidade e sorvedouro móvel, o SHS e o MHS.

O SHS utiliza uma rede um-salto explorando ao máximo a energia disponível nos nós sensores para a coleta de dados. Já o MHS utiliza um esquema de comunicação multi-saltos com número de saltos limitados, explorando a energia disponível nos nós sensores para coleta de dados e tarefas de roteamento de mensagens.

Nos métodos propostos aqui, é assumido que cada nó sensor sabe sua posição geográfica e que o sorvedouro conhece a posição de todos os nós sensores. Apesar de não tratarmos o problema de localização dos nós sensores, existem diversas propostas na literatura para tal, como Oliveira et al. (2005); Ye (2006). Além disso, o raio de comunicação do sorvedouro e dos nós sensores são fixos e não variam com a carga da bateria.

Na seção 4.1 definiremos os sub-problemas tratados nesse trabalho. Na seção 4.2 descrevemos as duas abordagens propostas, uma para redes um-salto e outra para redes multi-saltos.

4.1 Definição dos problemas

Nesse trabalho, consideramos os problemas de controle de densidade e mobilidade do sorvedouro em RSSFs. A solução individual de cada um desses problemas traz diferentes benefícios para às RSSFs, como aumento no tempo de vida, menor atraso na entrega de mensagens, capacidade de operação em redes desconexas, entre outros. Nas próximas sub-seções detalharemos como cada um desses sub-problemas foi abordado e as soluções adotadas para cada um.

4.1.1 Controle de densidade

O mecanismo de controle de densidade gerencia a redundância da rede, deixando um conjunto mínimo de nós sensores em atividade em determinado período de tempo (Ye et al., 2002; Quintão et al., 2005). Como os sensores não ficam ativos o tempo todo, o tempo de vida dos nós sensores é estendido e conseqüentemente o tempo de vida da rede também. Os dois métodos utilizam o mesmo mecanismo de controle de densidade.

Em nossa abordagem, o controle de densidade é feito pelo sorvedouro, de forma centralizada. Assumimos que o sorvedouro sabe a posição de todos os nós sensores, e é capaz de coletar informações sobre o estado de energia de todos os nós. Com isso o sorvedouro pode realizar um controle de densidade centralizado de forma ótima, considerando o estado de energia de cada nó sensor.

O problema do controle de densidade é modelado como o Problema de Cobertura de Conjuntos (Garey e Jonhson, 1979). Primeiramente a área de monitoramento é discretizada, de forma que cada ponto representa uma pequena porção dessa área, formando um conjunto de pontos D que devem ser monitorados.

Seja S o conjunto de sensores e C o raio de sensoriamento de cada sensor. Cada sensor cobre um sub-conjunto de pontos de demanda $D_s = \{d \in D | dist(d, s) \leq C\}$, onde $dist(d, s)$ é a distância euclidiana entre o ponto de demanda $d \in D$ e o sensor $s \in S$.

Associado a cada sensor temos um custo de ativação w_s, que é dado por uma função inversamente proporcional à energia de cada sensor $s \in S$, priorizando assim sensores com mais energia para ativação. O objetivo é minimizar o custo de ativação dos nós sensores, sujeito à restrição de cobertura de todos os pontos de demanda.

A saída do problema é um conjunto Y, onde y_s indica se o sensor $s \in S$ estará ativo no próximo período de operação da RSSF. Com a definição acima podemos representar o problema através do modelo de Programação Matemática para o problema de cobertura de conjuntos:

$$minimize\ z\ = \sum_{s \in S} w_s y_s \tag{4.1}$$

Sujeito a:

$$\sum_{s} x_s^d \geq 1, \forall d \in D \tag{4.2}$$

$$x_s^d \leq y_s, \forall s \in S\ \forall d \in D_s \tag{4.3}$$

$$x \in \{0, 1\} \tag{4.4}$$

$$y \in \{0, 1\} \tag{4.5}$$

No modelo as restrições 4.2 garantem a cobertura de todos os pontos de demanda ($d \in D$) que podem ser atendidos por pelo menos um nó sensor. As restrições 4.3 garantem que se um nó sensor ($s \in S$) for utilizado para cobrir um ponto de demanda ($d \in D$) então $y_s = 1$.

As decisões de controle de densidade são implantadas na rede pelo sorvedouro móvel. O sorvedouro, em nossa abordagem, é o único elemento que tem conectividade com todos os nós sensores da RSSF. Porém essa conectividade é ativada ao longo do tempo. No momento em que o sorvedouro está localizado em um agrupamento de nós sensores, todos os nós sensores desse agrupamento tem conectividade com o sorvedouro, e recebem as decisões de controle de densidade.

No início de cada ciclo (para uma definição formal de ciclo, consulte a seção 4.1.3), o

sorvedouro toma as decisões de controle de densidade e as implementa em cada sensor à medida que percorre todos os agrupamentos. Quando o sorvedouro requisita os dados do nó sensor, ele o avisa se ficará ativo após reportar seus dados.

Essa implantação das decisões de controle de densidade podem levar a falhas momentâneas na cobertura da área monitorada. Por exemplo, considere que o sensor s_1 é desativado no tempo t_1 e o sensor s_2 é ativado no tempo $t_2 > t_1$ e $D_{s_1} = D_{s_2}$. Assim, o sub-conjunto de pontos de demanda $D^1 \subseteq D_{s_1}$ que não é sensoriado por outro sensor ativo ficará descoberto durante o intervalo de tempo $[t_1, t_2]$.

Quando um sensor é desativado pelo controle de densidade, a sua função de sensoriamento é desativada. O processador estará ativo sempre que a função de sensoriamento ou o rádio estiverem ativos. O rádio dos nós sensores é ativado somente quando o sorvedouro está no agrupamento que esse sensor se encontra. Isso pode ser feito através de métodos que deixam o rádio em baixa energia até receber um estímulo externo para ativar o seu rádio (Polastre et al., 2004; Correia et al., 2005).

4.1.2 Agrupamento

Os dois métodos propostos possuem requisitos para agrupamento dos nós sensores para facilitar a coleta de dados pelo sorvedouro móvel. Tanto em um quanto em outro método é desejável que a área de monitoramento seja coberta por um conjunto mínimo de agrupamentos, o que reduziria a quantidade de pontos de coleta para o sorvedouro possivelmente reduzindo o tamanho de suas rotas.

4.1.2.1 SHS - Redes single-hop

No SHS, os nós sensores devem ser agrupados em agrupamentos com raio máximo R, ficando os nós sensores geograficamente próximos no mesmo agrupamento. Nesse método o problema de agrupamento foi modelado como o problema *min-size k-clustering problem* (Bilò et al., 2005), onde é dado um conjunto S de nós sensores e $dist(s_i, s_j)$ uma função que determina a

distância entre os sensores s_i e s_j. O objetivo é construir o menor número de agrupamentos possível, onde o raio de cada agrupamento é limitado por R.

Para construir tais agrupamentos utilizamos um algoritmo baseado no método da árvore geradora mínima (*Minimum spanning tree method*), que é uma técnica de agrupamento aglomerativa hierárquica (Jain, 1991). Dado um conjunto S de nós sensores, o algoritmo guloso começa construindo $|S|$ agrupamentos centrados em cada um dos nós sensores em S.

Iterativamente, os agrupamentos mais próximos c_i e c_j são unidos em um único agrupamento c_r se e somente se $raio(c_r) \leq R$, onde $raio(.)$ indica o raio do agrupamento indicado. O processo iterativo é interrompido quando nenhum agrupamento c_i e c_j pode ser unido. O procedimento é mostrado no Algoritmo 4.1.

Algoritmo 4.1: Clusterização SHS(**Nós sensores** S)

```
1  CLUTERS = S;
2  repeat
3  |  Calcula o centro do i-ésimo cluster, ∀i ∈ CLUSTERS;
4  |  Calcula a matrix de distância entre os clusters M;
5  |  Encontra o menor elemento não nulo (c_i, c_j) da matriz M;
6  |  R_r = raio(c_r), onde c_r é o cluster formado pela junção dos clusters c_i e c_j;
7  |  if (R_r ≤ R) then
8  |  |  CLUSTERS = CLUSTERS − c_i − c_j ∪ c_r;
9  |  end
10 until R_r > R ;
11 return CLUSTERS
```

A figura 4.1 mostra um cenário com aglomeração para o método SHS.

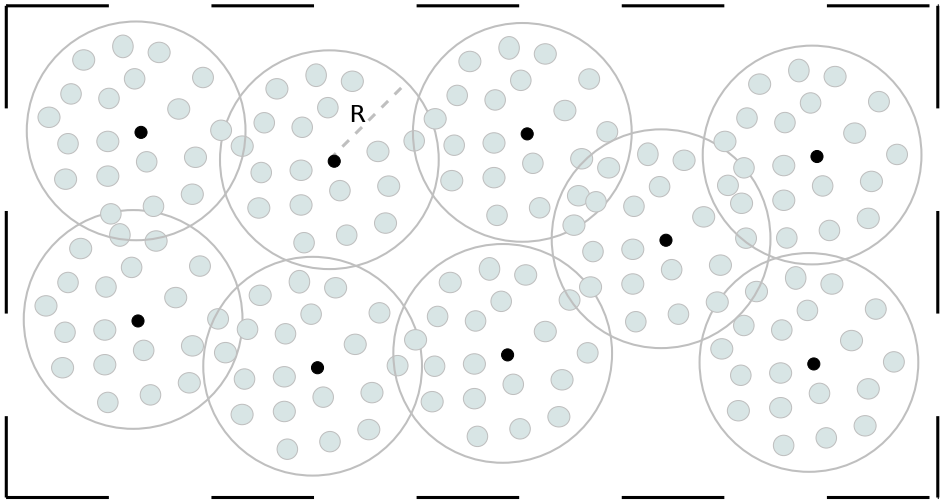

Figura 4.1: Cenário com agrupamento para o SHS.

4.1.2.2 MHS - Redes multi-salto

No MHS, os nós sensores são agrupados formando árvores de coleta de dados com profundidade limitada. Nesse método o problema de agrupamento foi modelado como o problema *inverse p-Center* (Mirchandani e Francis, 1990), onde é dado um conjunto S de sensores, e o objetivo é identificar um número mínimo p de centros tal que os elementos de cada centro podem se comunicar, com um número limitado λ de saltos.

Para construir os agrupamentos em forma de árvore exigidos pelo MHS, utilizamos o algoritmo descrito em (Mirchandani e Francis, 1990), onde o problema *inverse p-Center* é reduzido ao problema de cobertura de conjunto. Dado um grafo $G = (S, B)$, onde S é o conjunto de nós sensores e B as arestas que representam os enlaces de comunicação existentes entre os nós sensores, construímos a matriz A onde a_{ij} é igual a 1 se o sensor $i \in S$ tem um caminho em G até o sensor $j \in S$ com no máximo λ saltos e 0 caso contrário. Seja X_p o sub-conjunto de p sensores escolhidos como centros. O sub-conjunto X_p é obtido resolvendo o problema de otimização (cobertura mínima de conjuntos) $X_P = min\{k \subseteq S \mid \sum a_{ik}x_k \geq 1 \; \forall i, \; x_k \in \{0,1\}\}$.

Um cenário com aglomeração para o método MHS é mostrado na figura 4.2.

O sorvedouro utiliza esse procedimento para construir todas as árvores de coleta de dados.

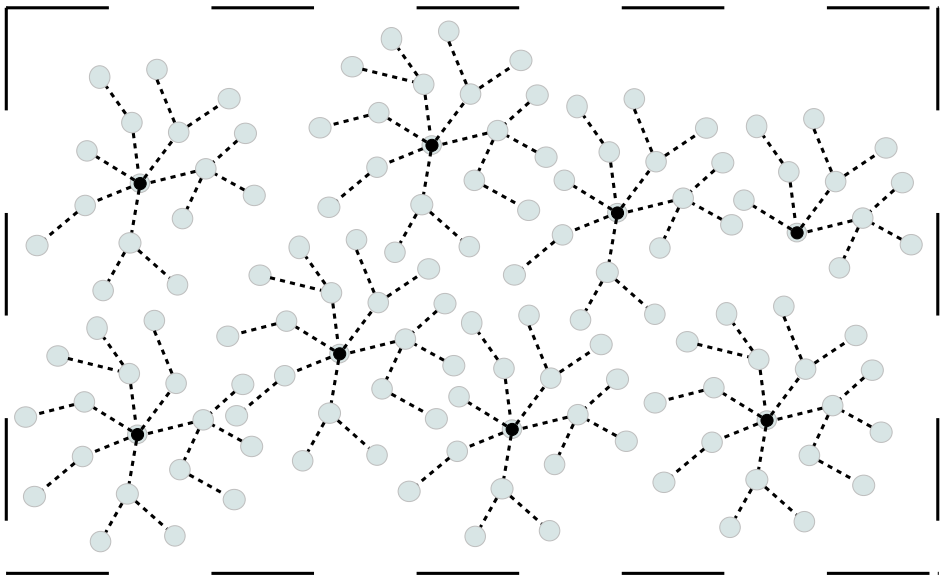

Figura 4.2: Cenário com agrupamento para o MHS.

Essa configuração topológica da rede é disseminada ao agrupamento quando o sorvedouro está posicionado sobre a raiz de uma árvore específica e então envia a configuração para os nós sensores da árvore. O sorvedouro envia uma mensagem com a configuração da árvore para os nós sensores, os nós sensores que recebem a mensagem e fazem parte da árvore, guardam essa mensagem e a propaga para seus vizinhos, os nós sensores que não fazem parte somente ignoram essa mensagem. Esse procedimento recursivo constroi a ávore de coleta de dados de forma parecida como ocorre em métodos de roteamento em árvore como o EFTREE (Figueiredo et al., 2004).

4.1.3 Controle de mobilidade do sorvedouro

O planejamento eficiente de rotas para mover o sorvedouro sobre os agrupamentos de nós sensores é um problema central em nossos modelos. Esse planejamento tem um impacto significativo em algumas das mais importantes métricas em RSSFs, como o atraso na entrega das mensagens e a taxa de sucesso na entrega de mensagens. O sorvedouro móvel precisa visitar cada agrupamento de modo a minimizar a perda de mensagens por falta de espaço em seus *buffers* e minimizar o atraso na entrega das mensagens.

O maior empecilho ao uso de sorvedouro móvel e comunicação um-salto é o considerável

atraso na entrega dos dados coletados pelos nós sensores, pois os nós sensores precisam esperar pelo sorvedouro para reportá-los.

Se o tempo que um sensor tem de esperar pelo sorvedouro para reportar seus dados for muito longo, alguns dados podem ter de ser retirados do *buffer* para acomodar outros novos. Por isso, o planejamento de rota para movimentação do sorvedouro é uma importante parte deste trabalho.

Os dois métodos propostos dividem o conjunto de nós sensores S em sub-conjuntos (agrupamentos) disjuntos, e o sorvedouro precisa visitar cada um desses agrupamentos para coletar dados. No método SHS, o sorvedouro precisa visitar o centro de cada agrupamento, e no método MHS o sorvedouro precisa visitar a raiz de cada árvore.

Nós modelamos o problema de planejamento das rotas como o Problema do Caixeiro Viajante (Dantzig et al., 1954) (PCV) para os dois métodos. Cada agrupamento é modelado como uma cidade do PCV e as distâncias euclidianas entre os agrupamentos são usadas como medidas de distância. A solução do PCV fornece o planejamento da rota para o sorvedouro móvel.

Existem diversos modelos matemáticos e algoritmos para a solução do PCV de forma ótima. Utilizamos o modelo abaixo para sua solução, apesar de suas limitações, ele é capaz de resolver instâncias do tamanho das encontradas nos nossos modelos computacionais. Seja o grafo $G = (V, E)$ então o PCV pode ser formulado como:

$$minimizar \sum_{(i,j)\in E} d_{ij}x_{ij}$$

Sujeito a:

$$\sum_{i\in V} x_{ij} = 1, \qquad \forall j \in V$$
$$\sum_{j\in V} x_{ij} = 1, \qquad \forall i \in V$$

$$\sum_{ij \in E} x_{ij} < |N|, \quad \forall N \subset V$$
$$x_{ij} \in \{0,1\}, \quad \forall (i,j) \in E$$

O modelo acima tem como saída o planejamento da rota do sorvedouro sobre os agrupamentos de nós sensores para os dois métodos propostos.

4.2 Arquiteturas de rede

Os sub-problemas apresentados são tratados em dois métodos para otimização do controle de densidade e controle de mobilidade do sorvedouro. Os dois métodos, o SHS e o MHS, fazem uso da solução dos sub-problemas apresentados de forma integrada. Nas duas próximas subseções apresentamos o método SHS e o MHS detalhadamente.

4.2.1 SHS - Redes um-salto

Alguns trabalhos da literatura, como (Kim et al., 2003), mostram que em RSSFs com comunicação multi-saltos o roteamento de mensagens é um grande responsável pelo consumo de energia. O custo energético de manter o rádio ligado, o custo para a transmissão e para a recepção de dados para o roteamento de dados de outros sensores contribuem para o consumo de energia de nós sensores.

Para eliminar o consumo de energia provocado pelo roteamento de dados, diversos autores (Shah et al., 2003; Gandham et al., 2003; Kansal et al., 2004; Jea et al., 2005; Kim et al., 2003; Wang et al., 2005a) utilizam sorvedouro móvel para buscar os dados dos nós sensores com somente um-salto, evitando o roteamento de dados pelos nós sensores.

Neste trabalho propomos o método SHS (*single-hop strategy*), que utiliza uma estratégia de comunicação um-salto, agrupando os nós sensores para minimizar o caminho de coleta de dados pelo sorvedouro. Os nós sensores são organizados em agrupamentos de diâmetro $2R$,

onde R é o raio de comunicação dos nós sensores e do sorvedouro.

O preço pago pela redução do consumo de energia é o maior atraso na entrega das mensagens ao sorvedouro móvel. A velocidade de um dispositivo móvel geralmente é muito menor que a velocidade que um dado pode trafegar via comunicação sem fio. Por isso, a utilização de comunicação um-salto implica em uma maior atraso na entrega da mensagens.

Dado a configuração em agrupamentos de sensores, onde o sorvedouro é capaz de comunicar com os sensores com um-salto, podemos implementar um sistema de comunicação onde as colisões de pacotes são evitadas, reduzindo ainda mais a quantidade de energia consumida na entrega de mensagens. No método SHS, os nós sensores comunicam-se com o sorvedouro através de um protocolo baseado no TDM (*Time Division Multiplexing*). Esse mecanismo elimina as colisões, aumentando a eficiência da comunicação e sua implementação necessita trocar poucas mensagens para gerenciamento.

A comunicação entre o sorvedouro e um nó sensor começa quando o sorvedouro envia um sinal avisando para um sensor específico que ele pode transmitir seus dados. Quando o sensor terminar, o sorvedouro é avisado e pode enviar o sinal autorizando o início de transmissão de dados para outro nó sensor. O processo é repetido até que todos os sensores do agrupamento tenham seus dados solicitados.

Os estados da RSSF com o método SHS estão descritos na figura 4.3, com suas respectivas transições.

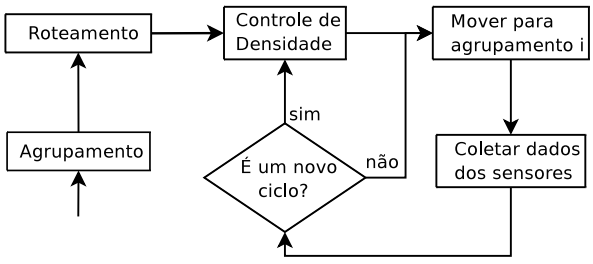

Figura 4.3: Método SHS.

No SHS, no início da operação da RSSF, o sorvedouro define os agrupamentos e planeja a rota sobre ele. Depois desse procedimento inicial, o sorvedouro inicia seu ciclo para coleta de dados.

A primeira tarefa ao iniciar um novo ciclo de coleta de dados é a solução do controle de densidade. O sorvedouro executa o mecanismo de controle de densidade para implantá-lo na RSSF à medida que vai coletando os dados dos sensores.

Após esse procedimento inicial, o sorvedouro começa a se mover para o primeiro agrupamento do ciclo. Ao receber o estímulo do sorvedouro, todos os nós sensores do agrupamento ativam seus rádios. Respeitando o modelo de comunicação TDM, o sorvedouro começa o processo de requisição de dados de cada um dos nós sensores. Cada vez que o sorvedouro inicia a comunicação com um nó sensor específico, o sorvedouro envia a decisão de controle de densidade para aquele nó específico, informando-o se sua função de sensoriamento ficará ativa depois que reportar seus dados. Nesse método o tamanho de cada ciclo de atividade é definido por duas requisições consecutivas de dados pelo sorvedouro. Com isso, os ciclos são assíncronos, definidos individualmente para cada nó sensor e de duração não conhecida *a priori*

Após terminar o envio dos seus dados ao sorvedouro, cada nó sensor desativa seu rádio, e também desativa suas funções de sensoriamento se ele não for utilizado para cobertura no ciclo seguinte.

Quando finaliza a coleta de dados de todos os nós sensores do agrupamento, o sorvedouro move-se para o próximo agrupamento no ciclo e assim por diante. Ao retornar ao primeiro agrupamento do ciclo, é iniciado um novo ciclo e uma nova solução do controle de densidade é gerado para o próximo ciclo.

A figura 4.4 mostra um cenário onde o método SHS é utilizado para a organização da RSSF, o ciclo do sorvedouro é definido entre o centro de cada um dos agrupamentos existentes.

4.2.2 MHS - Redes multi-salto

A restrição de comunicação um-salto com sorvedouro móvel em RSSFs acarreta um atraso na entrega dos dados ao sorvedouro (Wang et al., 2005a) maior que o apresentado por métodos de roteamento multi-saltos.. Entretanto, o uso de comunicação multi-saltos implica um maior

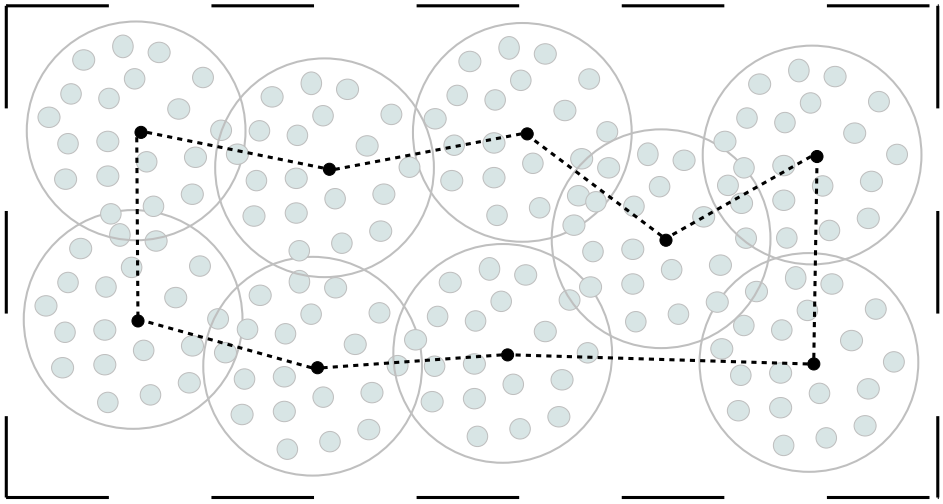

Figura 4.4: Cenário SHS.

consumo de energia pelos nós sensores no roteamento de mensagens (Kim et al., 2003). Para obter uma solução intermediária a esses dois extremos, propomos o método MHS (*multi-hop strategy*).

O MHS utiliza comunicação multi-saltos, porém com um número limitado de saltos que uma mensagem pode percorrer. Ou seja, limitamos o caminho de uma mensagem ao sorvedouro em λ saltos, todas as mensagens que atravessam λ saltos e não chegam ao sorvedouro são descartadas.

Com a utilização de comunicação multi-saltos, podemos construir agrupamentos maiores que no método SHS, reduzindo assim o número de agrupamentos necessário para cobrir uma área de sensoriamento. Com a redução no número de agrupamentos, temos uma redução no número de pontos de parada do sorvedouro, reduzindo assim o tamanho da rota e conseqüentemente o atraso na entrega das mensagens.

No método MHS, a RSSF é dividida em algumas árvores (uma floresta) de coleta de dados com a propriedade de que cada nó sensor está a no máximo λ saltos da raiz de sua árvore. Nosso objetivo na construção dessas árvores é, respeitando a restrição de número máximo de saltos, produzir uma floresta com o menor número de árvores possível.

Mais formalmente, dado um conjunto de nós sensores S, seja X_p o sub-conjunto de p nós

sensores e $D(y, X_p) = min_{x \in X_p}\ hops(y, x),\ \forall y \in S$, onde $hops(y, x)$ é a distância mínima em saltos entre os nós sensores y e x. Finalmente, seja $H(X_p) = max_{y \in S}\ D(y, X_p)$ a maior distância entre um nó sensor e um dos pontos escolhidos como raiz. As árvores são construídas resolvendo o problema de otimização $p_\lambda = min\{p : X_p \subseteq S, H(X_p) \leq \lambda, p \geq 0\}$, conhecido como o problema p-Centro invertido (*inverse p-Center*) (Mirchandani e Francis, 1990).

O problema do p-Centro é resolvido por uma redução ao problema de cobertura de conjuntos como descrito na seção 4.1.2.2.

O sorvedouro utiliza esse procedimento para construir todas as árvores de coleta de dados. Essa configuração topológica da rede é disseminada ao agrupamento quando o sorvedouro está posicionado sobre a raiz de uma árvore específica e então envia a configuração para os nós sensores da árvore. Os estados da RSSF com o método MHS estão descritos na figura 4.5, com suas respectivas transições.

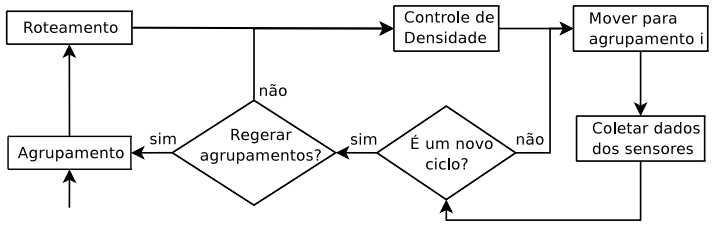

Figura 4.5: Método MHS.

No MHS, no início da operação das RSSFs, o sorvedouro executa o procedimento de construção das árvores e planeja a rota sobre essas árvores. Depois desse procedimento, o sorvedouro inicia seu ciclo para coleta de dados. A primeira tarefa ao iniciar um novo ciclo de coleta de dados, como no SHS, é a solução do controle de densidade. O sorvedouro executa o mecanismo de controle de densidade para implantá-lo na RSSF à medida que vai coletando os dados dos nós sensores.

Após esse procedimento inicial, o sorvedouro começa a se mover para a primeira árvore do ciclo. Ao receber o estímulo do sorvedouro, todos os nós sensores que pertencem a árvore ativam seus rádios. Primeiramente o sorvedouro envia as decisões de controle de densidade para os sensores daquele agrupamento, e dissemina a decisão de topologia daquela árvore específica. A partir dai os nós sensores podem enviar seus dados. No MHS, como utiliza

comunicação com mais de um-salto, não é utilizado um esquema de comunicação como o TDM. No MHS, os sensores comunicam-se com os outros nós sensores e com o sorvedouro utilizando um protocolo que permite colisão de pacotes, mas existe um mecanismo para reenviar pacotes perdidos. Nas simulações utilizamos o protocolo 802.11 para a coleta de dados dos nós sensores para o sorvedouro nesse método.

Sem a organização existente no SHS, o sorvedouro mantém registro de todos os nós sensores da árvore, e mantém para cada um deles um contador regressivo que limita o tempo de espera por uma mensagem de cada nó sensor. Esse contador é reinicializado sempre que uma nova mensagem chega ao sorvedouro e essa mensagem não indica o fim da transmissão dos dados do nó sensor. Quando todos os contadores terminarem, o sorvedouro termina sua espera por dados e parte para a próxima árvore da rota.

Como no SHS, após terminar o envio dos seus dados ao sorvedouro, os nós sensores desligam seus rádios, também desativam suas funções de sensoriamento se eles não forem utilizados para cobertura no período seguinte (decisão do controle de densidade).

Ao retornar à primeira árvore do ciclo é iniciado um novo ciclo, e como no SHS, uma nova solução do controle de densidade é gerada para o próximo ciclo. Além disso, o procedimento para a construção das árvores de coleta de dados pode ser realizado novamente caso algum sensor tenha sido declarado morto pelo sorvedouro. Caso um novo conjunto de árvores tenha sido gerado, uma nova rota de coleta é construída para as novas árvores. As árvores são reconstruídas para garantir que nós sensores sem energia não sejam utilizados para fazer roteamento de mensagens, o que implicaria em uma rota inválida ocasionando perda de dados.

A figura 4.6 mostra um cenário onde o método MHS é utilizado para a organização da RSSF, o ciclo do sorvedouro é definido entre as raízes de cada um das árvores de coleta de dados.

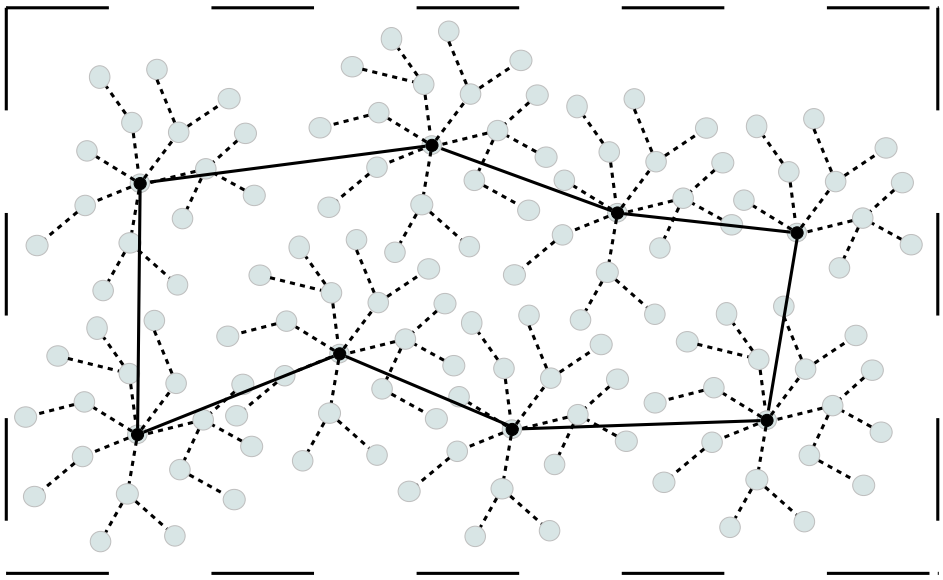

Figura 4.6: Cenário MHS.

4.3 Complexidade dos algoritmos

Os problemas tratados neste trabalho são em sua maioria problemas da classe NP. Porém, o problema de cobertura de conjuntos e o p-Centros invertido possuem algoritmos polinomiais que apresentam soluções satisfatórias para os mesmos. Também o problema do caixeiro viajante, um conhecido problema NP possui hoje métodos eficiêntes para a solução de instâncias com mais de 32.000 cidades. Neste trabalho não é tratada detalhadamente a complexidade dos algoritmos de solução para estes problemas. Informações sobre esses problemas, suas variantes e complexidades podem ser encontradas nas referências deste trabalho.

4.4 Mais de um sorvedouro móvel

Todos os dois métodos apresentados, o SHS e o MHS, podem utilizar mais de um sorvedouro para a coleta de dados. O sorvedouro possui uma seqüência de agrupamentos/árvores a serem visitados. Essas visitas podem ser divididas entre diversos *sorvedouros*, onde cada um deles realiza a coleta de dados de vários agrupamentos/árvores simultaneamente. Essa divisão de tarefas traz alguns benefícios para a RSSF principalmente decorrentes da redução do tempo entre duas coletas sucessivas do sorvedouro a um agrupamento. Essa redução leva

a uma coleta mais rápida dos dados dos nós sensores, aumentando a eficiência da rede com relação a taxa de dados efetivamente entregues ao sorvedouro, já que com um menor tempo de coleta, a possibilidade do *bufffer* de armazenamento de dados dos nós sensores ser totalmente ocupado é menor. Além disso, essa redução no tempo de coleta reduz o atraso na entrega de mensagens ao sorvedouro, principal deficiência das RSSFs com sorvedouro móvel e restrição de comunicação um-salto.

Tanto no SHS como no MHS, utilizamos mais de um sorvedouro móvel dividindo-os sobre a rota de forma defasada de modo a ficarem o mais afastados possível. Nos dois métodos existe um sorvedouro principal que é responsável pelas tarefas de controle de densidade, agrupamento de nós sensores e roteamento sobre os agrupamentos. Os outros *sorvedouros* existentes são utilizados somente para coleta de dados, seguindo a rota e os agrupamentos definidos pelo sorvedouro principal.

Nesse trabalhos tratamos somente o fluxo de dados dos nós sensores até os sorvedouros. O problema de como levar esses dados até um observador fora da rede não é tratado, sendo assim a comunicação entre os sorvedouros móveis não existe nos modelos considerados.

Capítulo 5

Avaliação experimental

Nesse capítulo descreveremos todos os passos do processo de avaliação experimental. Serão definidas as métricas avaliadas, parâmetros considerados, a técnica de avaliação e a carga de trabalho utilizada.

5.1 Técnica de avaliação

Para avaliação experimental das abordagens de solução propostas utilizamos simulação. Essa escolha é a mais usual em avaliação experimental de redes de computadores. Devido à sua complexidade, as redes de computadores de um modo geral são difíceis de modelar analiticamente, e por se tratar de uma nova solução, a monitoração de uma RSSF rodando nossas abordagens também é inviável.

Para a simulação, utilizamos um arcabouço construído para esse propósito sobre o simulador de redes *ad-hoc* (Java in Simulation Time - Scalable Wireless Ad hoc Network Simulator, 2006). O arcabouço é descrito em detalhes no Apêndice A.

5.2 Métricas

Algumas métricas foram utilizadas com o propósito de medir e comparar o desempenho das abordagens propostas. Dentre as mais importantes métricas para RSSFs, as seguintes foram selecionadas para análise:

- **Atraso na entrega de mensagens:** Tempo médio gasto por todas as mensagens entre o tempo em que o dado da mensagem é criado pelo sensor e o tempo de sua recepção por um dos *sorvedouros*. Se o atraso na entrega de mensagens é alto, a informação contida nas mensagens entregues possui um atraso alto com relação ao evento monitorado e pode não ser mais útil.
- **Cobertura:** Porcentagem da área sobre monitoramento que é realmente sensoriada pela RSSF. Uma porção da área é dita está coberta se essa porção está na área de sensoriamento de ao menos um nó sensor ativo. Para verificação dessa métrica, a área de sensoriamento é dividida em pequenas áreas quadradas. Cada uma dessas pequenas áreas é dita está coberta se o centro dela está no raio de sensoriamento de um nó sensor ativo. Essa métrica varia com o tempo, devido à falha em nós sensores, e dependendo do estado de ativação desses nós. Essa métrica é coletada em períodos de 10 minutos, o valor da cobertura da área em um período é a média de medidas feitas dentro do período a cada 20 segundos.
- **Taxa de mensagens entregues:** Mede a relação entre o número de mensagens recebidas pelo sorvedouro e o número de mensagem enviadas pelos nós sensores ao sorvedouro. Essa métrica indica a quantidade de informação do ambiente que é efetivamente entregue ao sorvedouro.
- **Tempo de vida da rede:** Mede o tempo de vida da rede. O tempo de vida é determinado pelo intervalo de tempo tempo entre o início de operação da RSSF até a morte do primeiro nó sensor.

Essas métricas foram escolhidas por serem as mais importantes métricas das RSSFs que são afetadas pelos métodos propostos. O consumo de energia é verificado implicitamente em

todas essas métricas.

5.3 Parâmetros

Entre todos os parâmetros de uma RSSF, avaliaremos o impacto dos mais relevantes aos nossos métodos: densidade da rede, velocidade do sorvedouro, tamanho da área monitorada e número de sorvedouros móveis. Esses parâmetros são descritos a seguir.

- **Densidade da rede:** Esse parâmetro é calculado como o número de sensores monitorando uma área divido pelo tamanho da área. Esse parâmetro é avaliado através da variação do número de sensores de uma área mantendo a área fixa. Esse parâmetro possui impacto em diversas métricas da RSSF, principalmente atraso na entrega de mensagens, tempo de vida, taxa de entrega e cobertura.

- **Número de sorvedouros móveis:** O número de sorvedouros móveis indica o número de sorvedouros que são responsáveis por coletar os dados da RSSF. O seu principal impacto, novamente, é sobre a métrica do atraso na entrega das mensagens. A utilização de mais de um sorvedouro móvel reduz o tempo entre duas visitas de um sorvedouro a um agrupamento de nós sensores, com isso reduzindo o atraso na entrega das mensagens.

- **Número de saltos λ:** O número de saltos que o uma mensagem pode atravessar até chegar ao sorvedouro. O principal impacto desse parâmetro é o número de árvores de coleta de dados gerada, influênciando, assim, diretamente o atraso e taxa de entrega das mensagens.

- **Velocidade do sorvedouro:** Velocidade com a qual o sorvedouro se move entre os agrupamentos de nós sensores. A velocidade do sorvedouro tem impacto principalmente no atraso na entrega das mensagens. Quanto mais rápido o sorvedouro se move, menor é o tempo de ciclo e conseqüentemente menor é o atraso na entrega de mensagens.

- **Tamanho da área monitorada:** Nas simulações utilizamos áreas sob monitoramento quadradas. Esse parâmetro mede o tamanho dos lados dessa área quadrada. Como a

métrica anterior, o tamanho da área monitorada tem impacto principalmente no atraso na entrega das mensagens.

5.4 Carga de trabalho e definição de cenário

Como um tópico muito recente de pesquisa em ciência da computação, as RSSFs até o presente momento não apresentam cargas de trabalhos bem definidas e difundidas no meio acadêmico. Todos o experimentos assumem condições e cenários baseados basicamente no sentimento dos pesquisadores quanto a topologia, distribuição geográfica e tráfego nas RSSFs. Serão utilizados cenários onde os sensores são dispostos na região de forma completamente aleatória, obedecendo uma distribuição uniforme sobre uma área quadrada de tamanho fixo.

O trafego de dados na rede é basicamente determinado pela taxa na qual os nós sensores coletam dados do ambiente. Os nós sensores coletam dados a uma taxa constante, em intervalos de 20 segundos. Os dados são enviados aos *sorvedouros* de forma assíncrona, sendo o processo de transmissão sempre disparado pelo sorvedouro. Quando um nó sensor estiver em uma organização topológica que o permita enviar dados ao sorvedouro, ele será informado e a transmissão se iniciará.

Além disso, como utilizamos na camada de enlace o protocolo IEEE 802.11, algumas mensagens de controle são trocadas entre os elementos de rede, porém essas mensagens não são computadas nas métricas, mas são consideras para cálculo de consumo de energia do nó sensor.

5.5 Problemas combinatórios

As abordagens propostas utilizam intensivamente problemas de otimização combinatória para sua solução. As soluções para esses problemas foram vastamente estudados na literatura (vide Capítulo 3). Nesse trabalho utilizamos soluções conhecidas para cada um desses problemas, principalmente utilizando modelos de programação matemática. Esses modelos foram resolvi-

dos via técnicas de otimização combinatória implementadas pelo ILOG Cplex Solver (2006), principal pacote comercial de otimização.

A solução desses problemas até a otimalidade em larga escala é custosa e inviável em termos computacionais. Porém, os tamanhos de tais problemas nos ambientes simulados nos permite utilizar tais técnicas. Com exceção do problema do Caixeiro Viajante, todos os problemas são resolvidos até a otimalidade em todos os experimentos executados.

O problema do Caixeiro Viajante é resolvido até a otimalidade para um limite máximo de 11 cidades, a partir desse limite a solução do problema é dada pela heurística do vizinho mais próximo, apresentada em Garfinkel (1985).

Vale lembrar aqui que esses problemas são todos resolvidos pelo sorvedouro. Esse elemento geralmente possui mais recursos computacionais que os nós sensores. Além disso, esse elemento tem capacidade de comunicação com a rede externa a RSSF e pode utilizar-se dela para a solução desses problemas.

O tempo de solução desses problemas está na casa de segundos, o que pode ser desprezado tratando de simulações de mais de 10 horas (em tempo de simulação). Além disso, algumas simples modificações nas abordagens propostas podem ser realizadas, alocando, em intervalos que o sorvedouro viaja de um ponto para outro, tarefas de solução desses problemas de otimização, eliminando-se assim o atraso no início de algumas tarefas devido ao tempo de solução de tais problemas.

Capítulo 6

Resultados computacionais

Neste capítulo analisaremos os resultados computacionais dos métodos propostos na seção 4. Entre todos os parâmetros de uma RSSF, avaliaremos os mais relevantes aos nossos métodos: densidade da rede, velocidade do sorvedouro,tamanho da área monitorada e número de sorvedouros móveis. Para cada um desses parâmetros verificamos a influência em cada métrica.

Como descrito no capítulo 5, avaliamos nossos métodos via simulação. Utilizamos um simulador de RSSFs construído sobre o simulador SWANS[1] detalhado no Apêndice A. Os parâmetros de simulação foram escolhidos com base no hardware do nó sensor Mica2[2].

Todos os problemas de otimização combinatória são resolvidos usando pacote de otimização CPLEX[3]. Para o tamanho dos ambientes simulados, com até 600 nós sensores e área quadrada de até 200 metros de lado, a solução desses problemas é computada em alguns segundos, o que acreditamos ser aceitável para a simulação. Porém, todos os problemas aqui apresentados possuem algoritmos ótimos para a solução de instâncias de larga escala.

Cada um dos métodos apresentados possui configurações específicas. Os valores dos parâmetros foram escolhidos baseados em trabalhos anteriores de simulação de RSSFs, como o

[1]Scalable Wireless Ad hoc Network Simulation. http://jist.ece.cornell.edu/
[2]XBOW MICA2 - Wireless Measurement System. http://www.xbow.com/
[3]CPLEX Solver - www.ilog.com

modelo de consumo de energia dos nós sensores. Outros valores foram retirados de especificações técnicas do nó sensor Mica Motes.

Para cada método escolhemos alguns valores desses parâmetros formando algumas instâncias dessas configurações, que chamaremos aqui de versões. Além disso, utilizaremos uma implementação de um algoritmo de roteamento em árvore. Também apresentamos um cenário onde o sorvedouro não é móvel, para ele escolhemos o algoritmo EFTREE (Figueiredo et al., 2004), um algoritmo que monta uma árvore com raiz no sorvedouro para a coleta de dados dos nós sensores. A árvore é atualizada periodicamente a cada 100 segundos. As versões dos métodos avaliados nesse trabalho estão descritos abaixo:

- **R**T - Implementa o algoritmo de roteamento de dados (Figueiredo et al., 2004) com atualização da árvore a cada 100s;
- **S**HS - Implementa o método SHS;
- **M**HS-2 - Implementa o método MHS com $\lambda = 2$;
- **M**HS-3 - Implementa o método MHS com $\lambda = 3$; e
- **M**HS-4 - Implementa o método MHS com $\lambda = 4$.

A menos que sejam re-definidos em algum experimento específico, os valores dos parâmetros da RSSF utilizados na simulação estão listados na Tabela 6.1.

As RSSFs simuladas executam uma aplicação de monitoramento de temperatura. Nessa aplicação, os nós sensores monitoram a temperatura periodicamente, a cada 20 segundos. Cada leitura de temperatura tem 32 bits de informação, gerando a cada 1 hora 720 Bytes de informação.

Os nós sensores são espalhados sobre a área de sensoriamento de forma aleatória, com função densidade de probabilidade uniforme, sendo assim cada ponto da área de sensoriamento possui probabilidade igual de ter um nó sensor sobre ele.

Parâmetro	Valor
Área monitorada	40000 m^2
Energia inicial do sensor	50 mAh
Raio de sensoriamento	15 m
Raio de comunicação	30 m
Energia gasta na transmissão	8.9 mA
Energia gasta na recepção	7 mA
Energia gasta com rádio ativo	7 mA
Energia gasta no processamento	8 mA
Energia gasta na função de sensoriamento	5 mA
Velocidade do sorvedouro	1 m/s
Largura de banda	250 kbps

Tabela 6.1: Parâmetros de simulação

Esse cenário é um dos possíveis cenários onde as RSSFs podem ser aplicadas. Outros cenários possíveis incluem, diferentes formas de geração de dados, por exemplo uma aplicação onde uma mensagem é enviada somente se algum evento é detectado na área de sensoriamento, ou até mesmo, aplicações onde o fluxo de informação é contínuo e as mensagens tem uma taxa de geração maior que a utilizada aqui. Além disso, outras formas de disposição dos nós sensores podem ser utilizadas, por exemplo depositando sensores sobre áreas de maior interesse e deixando menos sensores sobre áreas de pouco interesse.

A memória do nó sensor pode armazenar 4 Kbytes, o que é suficiente para suportar 3,4 horas de coleta de dados sem reportar ao sorvedouro. Portanto, se o tempo do ciclo do sorvedouro for maior que esse tempo, o nó sensor passará a perder dados por falta de memória para armazenamento.

Na simulação, a camada MAC utilizada é a IEEE 802.11, já que o Mica2 utiliza um protocolo de camada MAC CSMA/CA. Essa escolha se deve ao fato de nossos métodos trabalharem em camadas superiores, tendo a escolha da camada de rede pouco impacto nos resultados. Porém o IEEE 802.11 é muito semelhante ao protocolo de camada MAC utilizado no nó sensor Mica2.

Os experimentos executados simulam o funcionamento de uma RSSF por 10 horas. Cada simulação é repetida 33 vezes, o que nos permite calcular o tamanho do erro atribuído a aleatoriedade, assim permitindo analisarmos nossos resultados com 95% de confiança tanto

na diferença entre médias como na confiança das próprias médias.

6.1 Impacto da densidade da rede

Esse parâmetro é calculado como o número de sensores monitorando uma área divido pelo tamanho da área. Esse parâmetro possui impacto em diversas métricas da RSSF, principalmente atraso na entrega de mensagens, tempo de vida, taxa de entrega e cobertura.

Nesse experimento mantemos a área fixa em 40000 m^2 e variamos o número de sensores depositados na área de monitoramento de 50 a 600 nós sensores, ou seja, variamos a densidade de 0,00125 sensores/m^2 a 0,015 sensores/m^2, ou de 1 sensor a cada 800 m^2 a 1 sensor a cada 66 m^2.

6.1.1 Atraso na entrega de mensagens

A principal desvantagem de usar sorvedouro móvel em RSSF é o alto atraso na entrega das mensagens (Wang et al., 2005a). Analisamos as diferentes versões em função da densidade da rede.

A figura 6.1 mostra os resultados. No gráfico, não são mostrados os resultados do RT, pois eles são 4 ordens de magnitude menor que as outras versões. Entretanto, o RT apresenta um alto crescimento, de 0,1s a 0,8s no atraso das mensagens, ou seja, o atraso cresce 729% quando a rede cresce de 50 para 600 sensores, um crescimento de 1100%.

O SHS apresenta também uma grande sensibilidade a densidade da rede, nele o atraso cresce 93% com o aumento no número de nós sensores, porém o crescimento é muito inferior ao RT, mostrando uma maior escalabilidade do SHS. Esse crescimento é esperado, já que com o maior número de nós sensores numa área mais dados o sorvedouro móvel tem para coletar, aumentando o tempo do ciclo e conseqüentemente aumentando o atraso na entrega.

Apesar da diferença de desempenho entre o RT e o SHS, essa diferença se reduz de forma

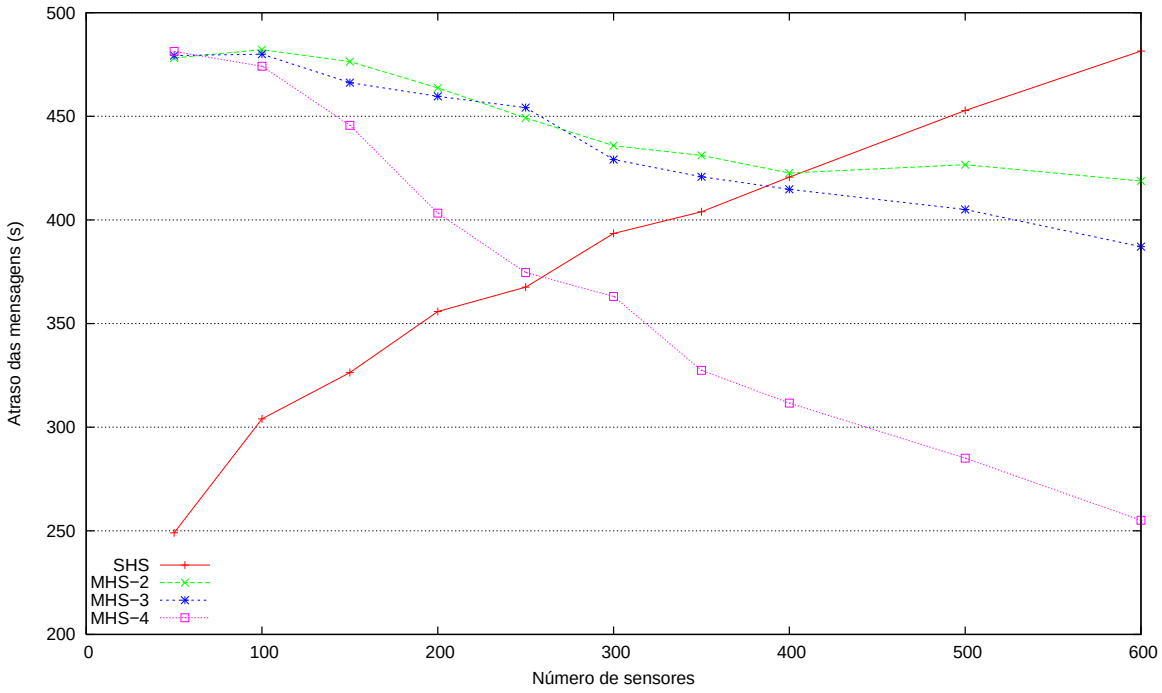

Figura 6.1: Atraso x Número de sensores.

acentuada com o crescimento da rede, no intervalo de densidade analisado, a diferença de desempenho entre o RT e o SHS se reduz em 329% com o aumento da densidade.

Diferentemente do SHS, as versões que utilizam o MHS tendem a ter o atraso na entrega das mensagens reduzido na faixa de valores de densidade de rede avaliados. Isso se deve principalmente a maior conectividade entre os nós sensores quando a rede é mais densa.

Além disso, esse efeito é potencializado pelo número de saltos, como mostra o resultado do MHS-4, que tem uma redução de 47% com o aumento da densidade da rede, em comparação com o MHS-2 que tem uma redução de 12%.

Apesar de não termos analisado para densidades maiores, acreditamos que a partir de uma certa densidade o comportamento do MHS passe a ser semelhante ao SHS, apresentando um crescimento do atraso na entrega de mensagens com o aumento da densidade da rede.

6.1.2 Tempo de vida da rede

Agora avaliamos as versões em termos de tempo de vida. O tempo de vida é comumente tratado como o tempo de morte do primeiro sensor da rede, e essa é a definição utilizada nesse trabalho. Na seção 6.1.4 avaliaremos a cobertura da rede, uma outra métrica de tempo de vida da rede.

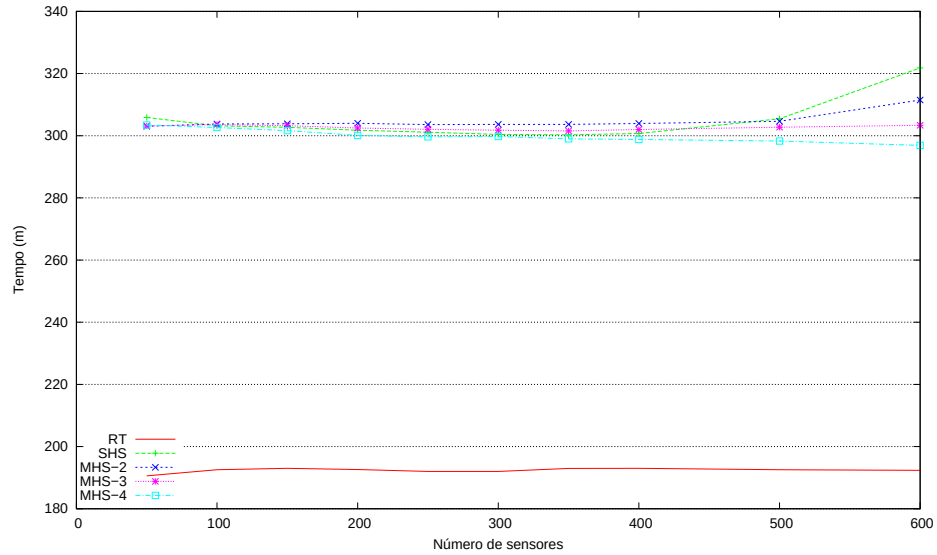

Figura 6.2: Tempo de vida da rede x Número de sensores.

A figura 6.2 mostra o tempo de vida da rede em função da densidade da rede. Como podemos ver, as diferentes versões não são muito sensíveis a esse parâmetro na faixa de valores testados. De forma geral, o RT não apresenta nenhuma variação considerável, tendo tempo de vida médio de 192 minutos. Já as versões com sorvedouro móvel apresentam um tempo de vida significativamente maior. O SHS é 58% em média melhor que o RT, chegando a 67% para rede com 600 sensores.

Vale notar que a partir de 400 sensores, o tempo de vida das redes com sorvedouro móvel apresenta um crescimento como mostrado no gráfico. Para o SHS, o tempo de vida cresce 21% quando a rede aumenta de 400 para 600 nós sensores. Já o MHS apresenta o mesmo comportamento, mas para $\lambda = \{3, 4\}$, como podemos notar no gráfico, esse crescimento é menor. Essa melhora pode ser atribuída a melhor divisão de tarefas entre os diversos sensores que cobrem simultaneamente a mesma área. O que acarreta um maior tempo de desativação

para todos os nós sensores, devido ao controle de densidade, o que aumenta o tempo de vida da rede. Porém, não simulamos redes maiores para verificar essa tendência de crescimento além dos limites testados.

6.1.3 Taxa de mensagens entregues

Nessa seção avaliamos as versões em termos de confiabilidade da RSSFs, através da métrica de taxa de mensagens entregues.

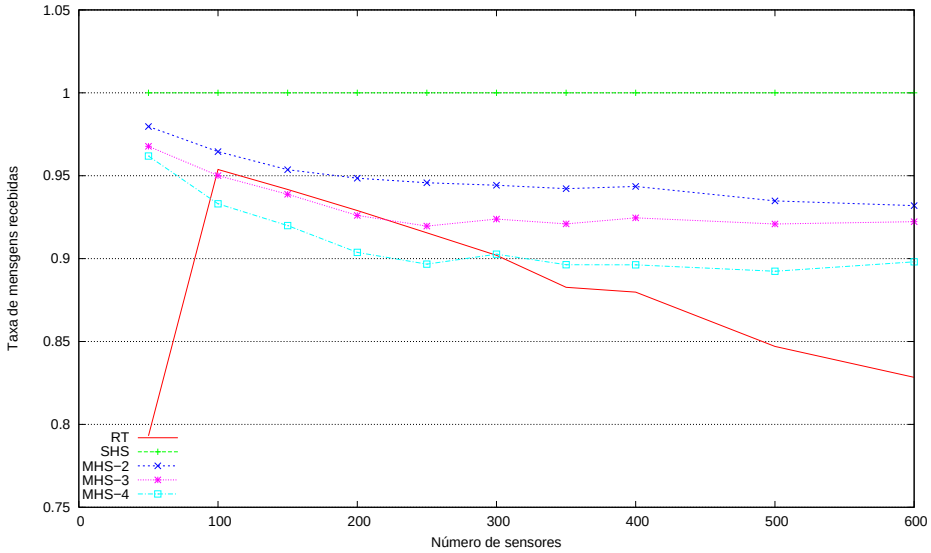

Figura 6.3: Taxa de entrega x Número de sensores.

Outra métrica importante em redes *ad-hoc* em geral é a taxa de mensagens entregues ao destino, no caso de RSSFs taxa de mensagens entregues ao sorvedouro, medindo a qualidade do mecanismo de coleta de dados.

Como mostra o gráfico da figura 6.3 nas versões baseadas no MHS, não temos uma variação muito significativa na taxa de entrega de mensagens com o aumento na densidade da rede, sendo a redução na taxa de entrega de 5%, 5% e 7% para o MHS-2, MHS-3 e MHS-4 respectivamente. Esse resultado se deve principalmente ao fato de no MHS as árvores de coleta de dados terem limite de profundidade, limitando o caminho de uma mensagem até o sorvedouro, limitando por sua vez a probabilidade de uma mensagem ser perdida durante sua transmissão.

Já no SHS, a taxa de entrega é muito próxima de 100%. Devido ao processo de transmissão de dados desse método ser organizado de forma a não permitir colisões de mensagens. Com isso as mensagens somente são perdidas, nesse método, devido à morte do nó sensor. Isso ocorre quando o sensor morre com mensagens em seu *buffer*. Porém, devido ao alto número de mensagens entregue, essas mensagens perdidas no *buffer* são praticamente despresíveis, o que torna a eficiência do SHS próxima de 100%nessa métrica.

No RT, temos uma diferença muito acentuada entre a taxa de entrega para 50 e 100 sensores. Essa diferença se deve ao fato de com 50 sensores, a RSSF formada não é densa o suficiente para conectar todos os nós sensores, com isso as mensagens enviadas por nós sensores desconexos são sempre perdidas.

Considerando-se que um nó sensor possui um raio de sensoriamento de 15 m, cada um cobre uma área circular de aproximadamente 706 $m2$. Sendo a área de sensoriamento de 40.000 $m2$, o limite inferior de nós sensores para cobrir toda a área é dado por:

$$LI = \frac{40000}{706} \approx 56 \tag{6.1}$$

sendo esse limite inferior, o número mínimo de nós sensores necessários, se não existir sobreposição das áreas de cobertura por cada nó sensor (o que é impossível com área de cobertura circular).

Isso evidencia um dos principais benefícios de se utilizar sorvedouro móvel para coleta de dados. O fato da rede não ser suficientemente densa, não afeta a conectividade do sorvedouro aos nós sensores, já que o sorvedouro possui mobilidade para alcançar sensores sem conexão com outros nós sensores.

Por outro lado, o fato da rede ser pouco densa ajuda na confiabilidade de entrega de mensagens nas versões com sorvedouro móvel já que evita um número grande de colisões de pacotes no método MHS, e evita gasto de energia na recepção de mensagens por nós sensores nos métodos SHS e MHS. Vale lembrar que somente o SHS utiliza TDM para transmissão de

dados e nesse método não existe colisão de dados.

Outro ponto importante no comportamento da versão RT é a queda acentuada da confiabilidade de entrega quando a rede cresce. Com o crescimento da rede de 100 para 600 sensores, a taxa de entrega de mensagens cai 13%, enquanto cai muito pouco nas outras versões. Isso ocorre pois com a densidade de sensores alta, muitos sensores concorrem pelo meio de transmissão, ocasionando muitas colisões e conseqüentemente perda de mensagens.

O gráfico ainda mostra uma tendência de queda do RT para redes maiores que 600 nós sensores, enquanto que para o MHS mostra uma tendência de queda mas menos acentuada que o RT, mostrando mais confiabilidade desse método.

6.1.4 Cobertura

Nesta seção avaliaremos a cobertura da área monitorada em função da densidade da rede. Para avaliarmos essa métrica verificamos a cobertura da rede em 4 diferentes tempos de simulação: com 3, 5, 7 e 9 horas de simulação. Os resultados são mostrados nas Figuras 6.4, 6.5, 6.6, 6.7.

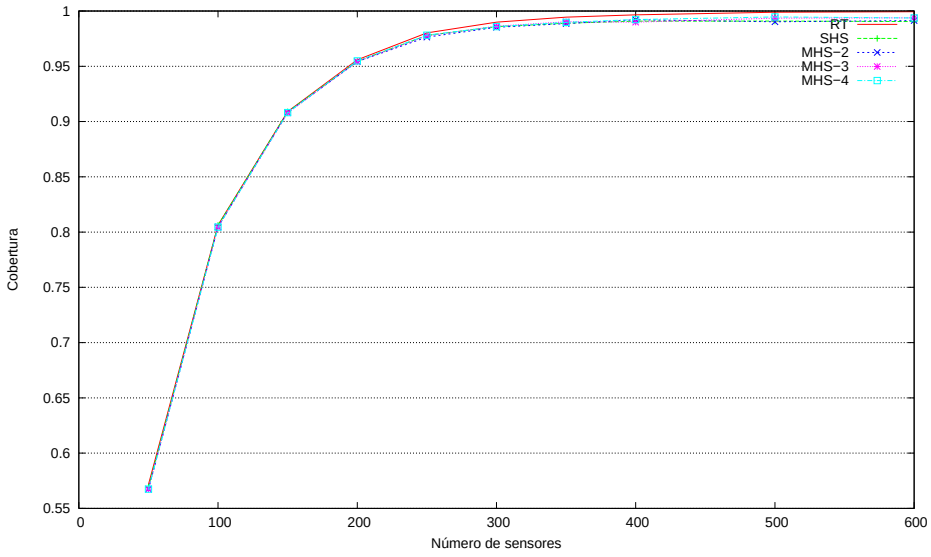

Figura 6.4: Cobertura x Número de sensores: 3 horas de simulação.

O resultado mais visível nos gráficos é a perda de cobertura da rede após 3 horas de simulação do RT. Como mostrado na figura 6.2, o primeiro sensor morre pouco depois de 3

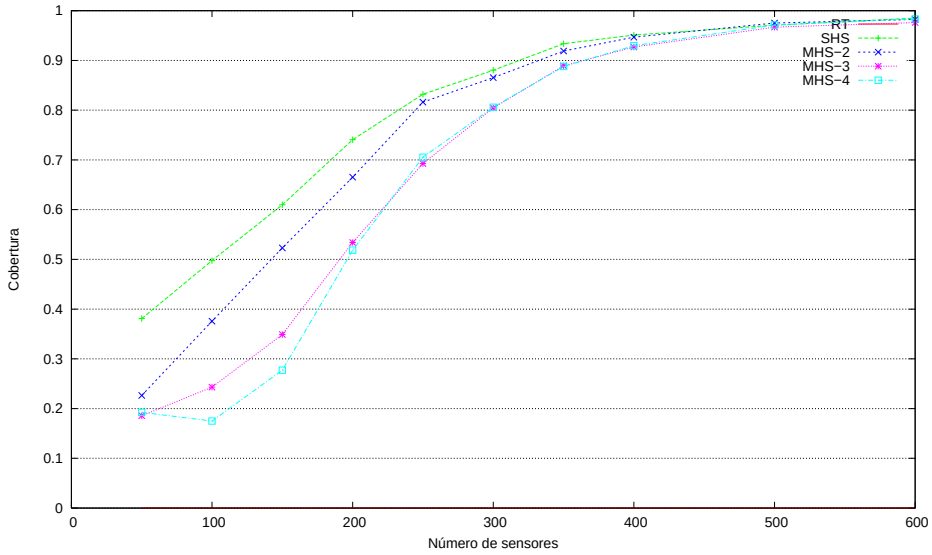

Figura 6.5: Cobertura x Número de sensores: 5 horas de simulação.

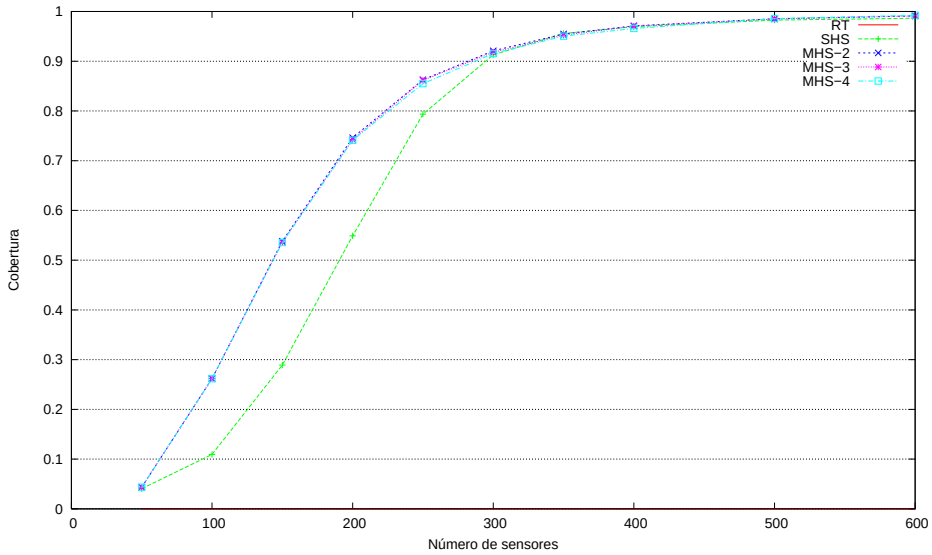

Figura 6.6: Cobertura x Número de sensores: 7 horas de simulação.

horas de simulação no RT, após 3 horas todos os nós sensores estão mortos e a cobertura é toda perdida. Nessa versão todos os nós sensores morrem praticamente ao mesmo tempo, perdendo a cobertura total da rede ao mesmo tempo.

Já as versões dos métodos SHS e MHS com controle de densidade mantém uma cobertura alta da área, acima de 3 horas de simulação. Com redes mais densas, como por exemplo, com 600 nós sensores, a área de observação possui cobertura de mais 98% da área monitorada. Além disso nas versões dos métodos SHS e MHS, os primeiros nós sensores morrem por volta

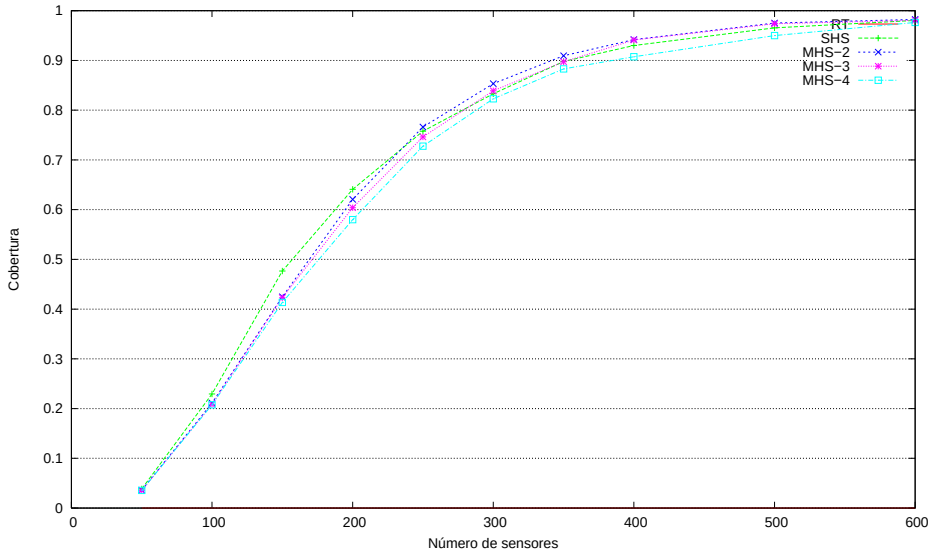

Figura 6.7: Cobertura x Número de sensores: 9 horas de simulação.

de 5 horas de simulação, como mostra a figura 6.2.

Como mostra o gráfico da figura 6.5 temos uma certa diferenciação entre as versões. O resultado mostra o SHS mais estável nesse horário, onde os primeiros nós sensores morrem, mantendo uma maior cobertura da área para redes menos densas (número de sensores < 400). Isso se deve principalmente a utilização mais inteligente de energia pelo método SHS, o que leva os primeiros sensores a morrer mais tarde do que nos outros métodos.

6.2 Impacto da velocidade do sorvedouro

A velocidade com a qual o sorvedouro se move entre os agrupamentos de nós sensores tem impacto principalmente no atraso na entrega das mensagens. Quanto mais rápido o sorvedouro se move menos os nós sensores esperarão para reportar seus dados, resultando assim em um menor atraso na entrega de mensagens.

Nessa seção variamos a velocidade do sorvedouro de 0,5 m/s a 5 m/s e avaliamos o impacto dessa variação no atraso na entrega de mensagens.

6.2.1 Atraso na entrega de mensagens

A figura 6.8 mostra os resultados em função da velocidade do sorvedouro para uma rede de 400 sensores.

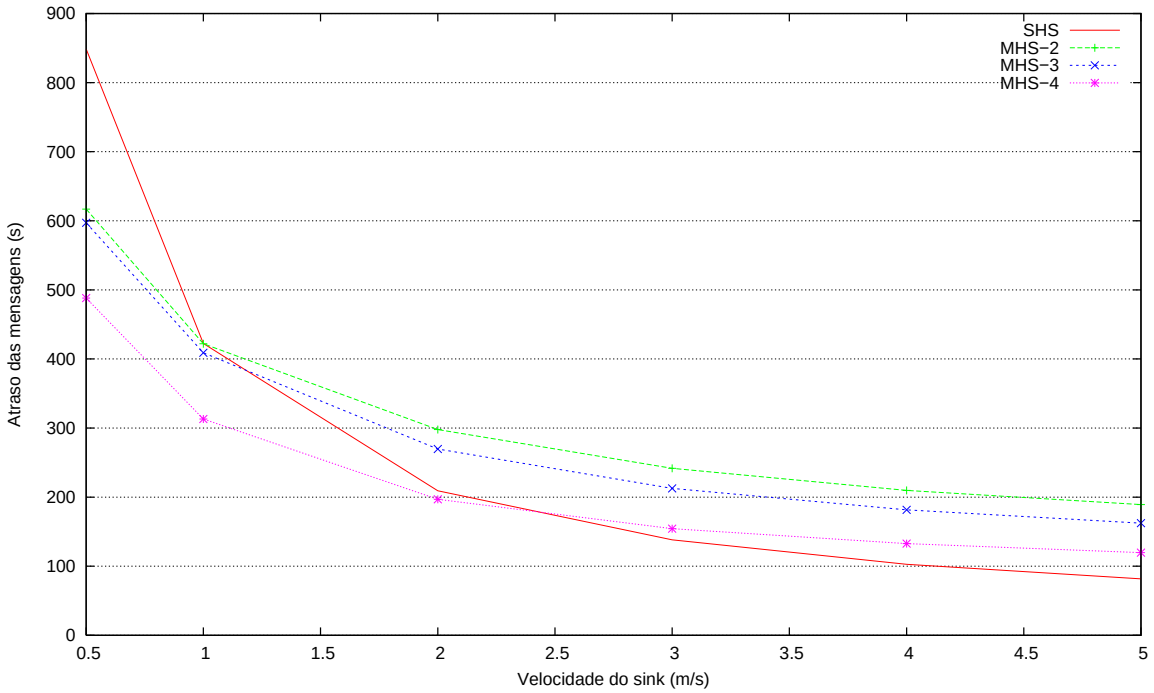

Figura 6.8: Atraso x Velocidade do sorvedouro.

Todas as versões com sorvedouro móvel apresentaram uma enorme sensibilidade à velocidade do sorvedouro, tendo uma redução no tempo de atraso de entrega de mensagens. Principalmente a versão SHS, que com um aumento na velocidade do sorvedouro de 0,5 m/s para 5 m/s tem o atraso médio das mensagens reduzido em 90%.

As versões que utilizam o MHS têm o mesmo efeito, mas menos potencializado, 75% foi o melhor ganho obtido entre todas as versões do MHS, obtido pelo MHS-4.

O maior ganho do SHS é explicado pela forma de comunicação desse método. No SHS o sorvedouro sabe exatamente quando pode sair do agrupamento de sensores para o próximo agrupamento. Já no MHS o sorvedouro só pode deixar o cluster atual quando todos os sensores responderem ou quando todos os tempos de espera por mensagens acabarem. Com um maior tempo de espera no MHS, a maior velocidade do sorvedouro tem efeito reduzido já que a

parte do atraso na entrega da mensagens devido ao tempo de movimentação do sorvedouro é menor que no SHS.

6.3 Impacto do tamanho da área de monitoramento

Nas simulações utilizamos áreas quadradas sobre monitoramento. O tamanho da área monitorada aqui se refere ao tamanho do lado dessa área quadrada. O tamanho da área monitorada tem impacto principalmente no atraso na entrega das mensagens. Nessas simulações, o número de sensores é fixado em 400 nós sensores.

6.3.1 Atraso na entrega de mensagens

O atraso na entrega de mensagens está estritamente ligado ao tempo gasto pelo sorvedouro para coletar os dados. Esse tempo é determinado pela velocidade do sorvedouro, analisado na seção 6.2.1, e também pelo tamanho da rota do sorvedouro.

Nessa seção avaliaremos o impacto do tamanho da rota do sorvedouro, influenciado pelo tamanho da área monitorada.

A figura 6.9 mostra o comportamento do atraso na entrega de mensagens em função do tamanho da área. Como podemos notar, o atraso na entrega de mensagens aumenta com o crescimento do tamanho da área, o que ocorre devido ao aumento do tamanho ciclo do sorvedour necessário para visitar todos os agrupamentos. Nas configurações com sorvedouro móvel, o fator de crescimento é considerável, para o SHS, MHS-2, MHS-3 e MHS-4 temos um crescimento de 658%, 546%, 4320% e 3499% respectivamente. Já no RT o crescimento no atraso é de apenas 8%, já que a influência do tamanho da área é minimizada devido à alta velocidade de transmissão.

Podemos notar que do MHS-2 para o MHS-3 e MHS-4, existe uma enorme diferença de sensibilidade do atraso ao tamanho da área. Isso leva a conclusão que o parâmetro λ

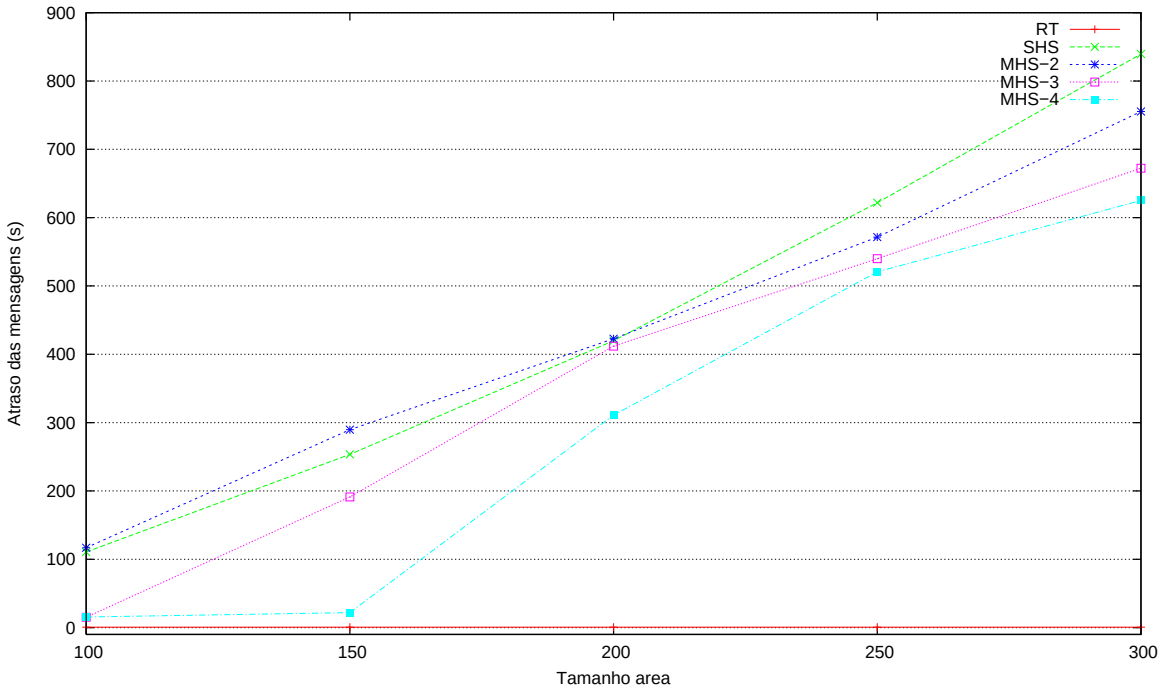

Figura 6.9: Atraso x Tamanho área.

deve ser escolhido em função do tamanho da área, provendo um menor atraso na entrega das mensagens preferencialmente com o gasto de energia aceitável. Porém, esse grande crescimento das versões MHS-3 e MHS-4 está ligado a grande conectividade da rede para o tamanho da área de 50 metros. Nessa configuração, com $\lambda = \{3, 4\}$ a área é toda coberta por poucos agrupamentos de nós sensores, o que reduz a distância percorrida pelo sorvedouro, que por sua vez reduz o atraso na entrega das mensagens.

6.3.2 Cobertura

Agora avaliamos a cobertura em função do tamanho da área. A cobertura é detalhada ao longo do tempo de simulação para três tamanhos de área diferentes: 100, 200 e 300 metros de lado, como mostra as Figuras 6.10, 6.11 e 6.12.

Para uma área de 100 metros de lado o método MHS com $\lambda = \{3, 4\}$ se comporta muito próximo ao RT. Devido à conectividade dos nós sensores, em uma área de pequenas dimensões, essas configurações com um número de saltos maiores pode cobrir a área monitorada com poucos agrupamentos de nós sensores. Esse mesmo comportamento é observado na métrica

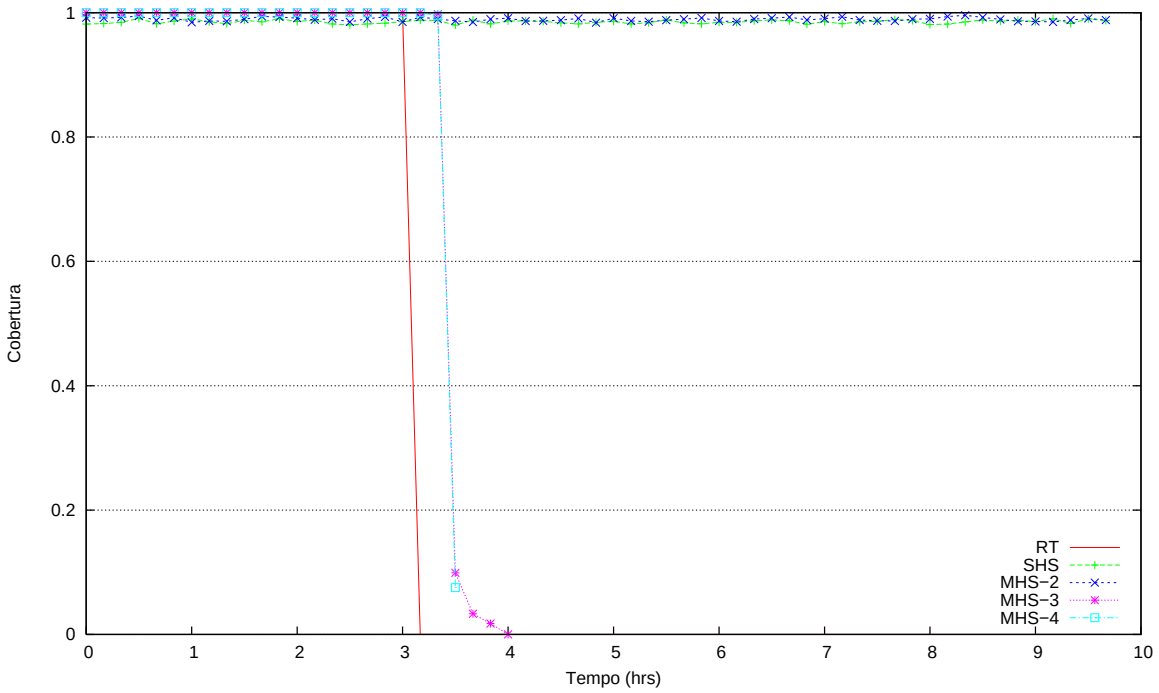

Figura 6.10: Cobertura x Tempo: 100 metros de lado

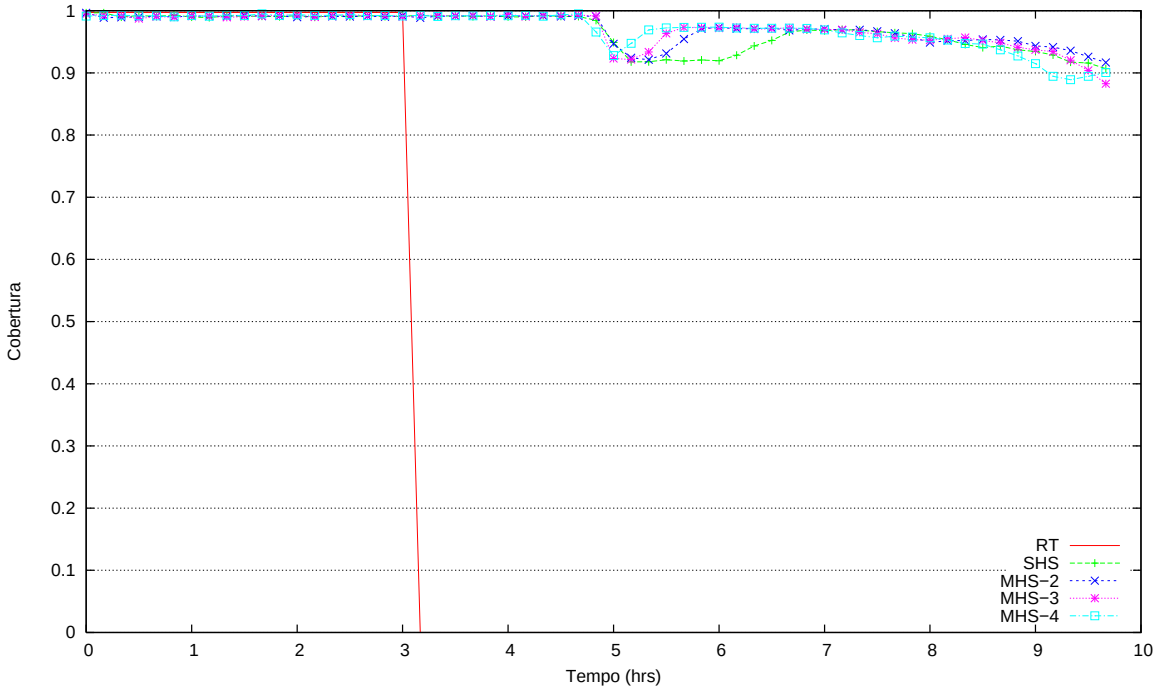

Figura 6.11: Cobertura x Tempo: 200 metros de lado

atraso na entrega de mensagens.

Nessa configuração, o MHS com poucos agrupamentos, gasta-se mais tempo requisitando

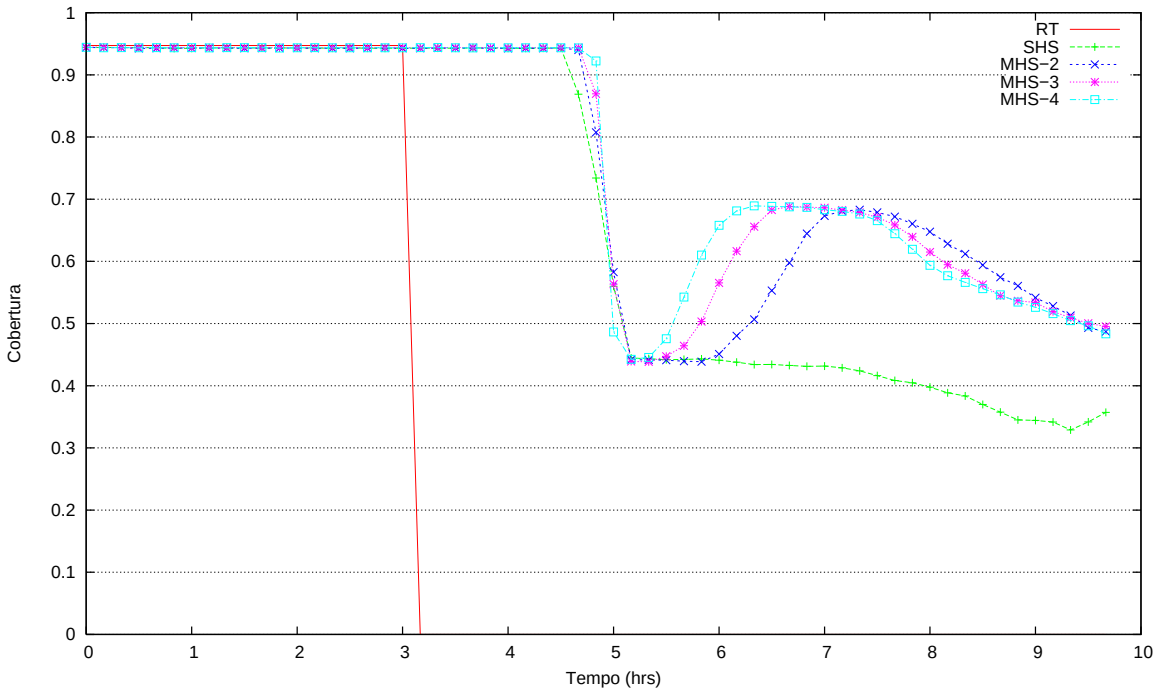

Figura 6.12: Cobertura x Tempo: 300 metros de lado

dados dos nós sensores que viajando entre os agrupamentos. Durante o processo de leitura do agrupamento, o nó sensor gasta energia com seu rádio e seu processador ligados, com isso o efeito do controle de densidade é minimizado e a rede acaba tendo um comportamento em termos de consumo de energia parecidos com o RT, resultando em um comportamento em relação a cobertura também parecido com o RT.

Já para uma rede com 200 metros de lado, o MHS com $\lambda = \{3, 4\}$ se comporta como o MHS-2 e o SHS, mantendo uma alta cobertura da rede durante mais de 9 horas de funcionamento da rede. Esse tamanho de área, se mostrou mais eficiente, dentre todos testados para a configuração de 400 nós sensores.

Com 300 metros de lado, as configurações com sorvedouro móvel apresentam uma grande baixa na cobertura por volta de 5 horas de funcionamento. Com esse tamanho de área, o tempo de ciclo do sorvedouro móvel é maior e a RSSF demora a recuperar a cobertura, já isso é feito à medida que o sorvedouro vai passando sobre os agrupamentos de nós sensores.

Com o aumento no tamanho da área, temos uma redução na densidade da rede. Com isso, reduzimos a redundância de cobertura e mais sensores são utilizados simultaneamente

em cada ciclo. Assim, a economia de energia proveniente do controle de densidade é reduzido levando as versões SHS e MHS terem seu consumo de energia muito parecidos com o RT, e um comportamento em relação a cobertura também parecido com o RT.

6.4 Múltiplos sorvedouros

Dos resultados apresentados até aqui, o que apresenta a principal discrepância entre os valores do RT e das outras versões é o atraso na entrega das mensagens, 4 ordens de grandeza menor. Isso se deve, em grande parte, a velocidade de movimentação do sorvedouro, que é muito menor que a velocidade de propagação das mensagens via comunicação sem fio.

Para reduzir essa diferença, que mesmo utilizando comunicação multi-saltos nas versões MHS, é muito significativa, tentamos a utilização de mais de um sorvedouro. Para isso, somente acrescentamos aos nossos métodos *sorvedouros* auxiliares. Eles só terão a função de coletar dados, utilizando a mesma rota definida para o sorvedouro principal. A distância entre os *sorvedouros* não será controlada, entretanto, para tentar mantê-los o mais afastado possível, iniciaremos o primeiro ciclo com os *sorvedouros* posicionados com o maior espaçamento possível entre eles (considerando a distância sobre a rota).

Os resultados são apresentados separadamente para cada método.

6.4.1 SHS com mais de um sorvedouro

Na figura 6.13 os resultados para o SHS é mostrado. Nele temos o SHS com 1 sorvedouro, com 2 sorvedouros (SHS/2) e com 3 sorvedouros (SHS/3).

Como era de ser esperar, o atraso é reduzido significativamente. Em média o SHS/2 é 51,8% melhor que o SHS e o SHS/3 é 67,5% que o SHS, mostrando um fator de redução do atraso na entrega de mensagens proporcional ao número de *sorvedouros* utilizados.

Além disso, para os tamanhos de rede simulados, a utilização de mais de um sorvedouro

torna a rede mais escalável. A utilização de mais *sorvedouros* reduz o impacto da densidade da rede no atraso na entrega das mensagens. Com 50 sensores, o SHS/2 é aproximadamente 100% melhor que o SHS, já para a rede com 600 sensores essa diferença aumenta para 125%, o que mostra essa maior escalabilidade.

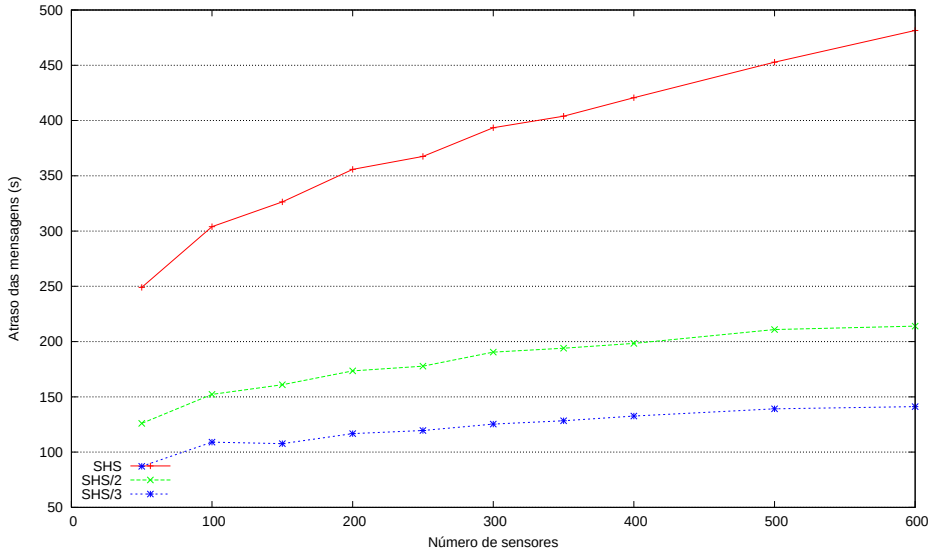

Figura 6.13: Atraso x Número de sensores (SHS)

6.4.2 MHS com mais de um sorvedouro

Na figura 6.14 os resultados para o MHS é mostrado. Nela temos as versões MHS-2 como nos resultados anteriores, MHS-2/2 com o $\lambda = 2$ e 2 *sorvedouros* móveis e MHS-2/3 com o $\lambda = 2$ e 3 *sorvedouros* móveis.

Como no resultado para o SHS, existe uma redução significativa no atraso das mensagens. Para todos os tamanhos de rede, a relação entre as 3 versões permaneceu praticamente constante. Na média, os MHS-2/2 foi 38% melhor que o MHS-2, enquanto que o MHS-2/3 foi 56% melhor que o MHS-2.

Mas diferentemente do SHS, nesse método não há, na faixa de tamanhos de rede simulados, uma tendência de maior escalabilidade das versões com mais sorvedouro móvel.

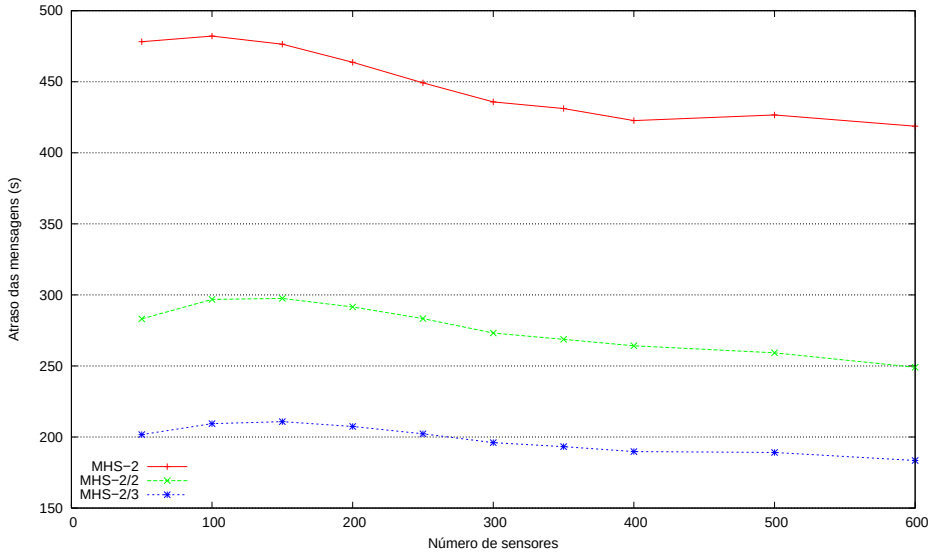

Figura 6.14: Atraso x Número de sensores (MHS).

6.5 Outros resultados

Nessa seção apresentaremos outros resultados, não necessariamente ligados à avaliação de um parâmetro específico. Aqui avaliaremos a cobertura da rede, em um funcionamento mais longo que o apresentado até aqui.

6.5.1 Cobertura

Nessa seção avaliaremos a cobertura da rede em função do tempo para uma rede com 400 nós sensores e, diferentemente dos outros experimentos, aumentamos o tempo de simulação para 25 horas, vendo o comportamento da rede até praticamente todos os nós sensores estarem mortos.

A figura 6.15 mostra como a cobertura da área monitorada varia ao longo do tempo. Para a carga de dados imposta, e a quantidade de energia definida, temos um incremento significativo na cobertura da área com os métodos propostos.

No RT todos os nós sensores morrem em pouco mais de 3 horas de simulação. Vale salientar que praticamente todos os sensores dessa versão morrem ao mesmo tempo, já que

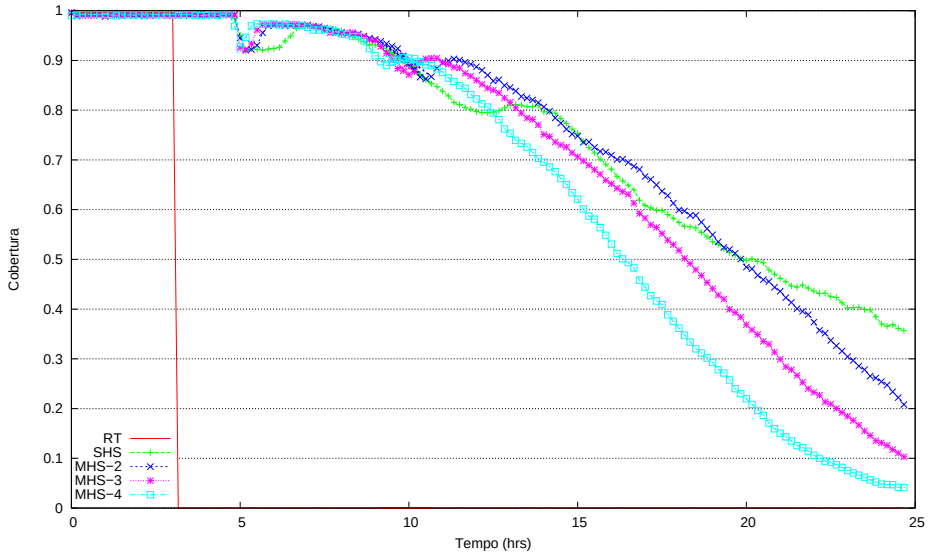

Figura 6.15: Cobertura da área x Tempo

essa versão não possui controle de densidade. Comparado com o SHS, que é a versão que mantêm maior cobertura por um maior período de tempo, com 15 horas de simulação 75% da área ainda está coberta por ao menos um sensor, e com 25 horas de simulação essa mesma versão ainda mantém 36% da área coberta.

As versões que utilizam o método MHS têm um consumo maior de energia se comparados ao SHS, devido ao consumo com roteamento de mensagens, tendem a morrer antes. Porém, vale lembrar que as versões baseados no MHS têm um menor atraso na entrega das mensagens.

Além disso, quanto maior λ mais próximo será o comportamento do MHS e do RT. Para λ menores, o MHS terá comportamento mais próximo ao SHS, o que pode ser visto no gráfico. Quanto maior o λ mais rapidamente a área perde cobertura.

Devido à forma de implementação das decisões do controle de densidade na rede, podemos ter falhas de cobertura momentâneas, como descrito na seção 4.1.1. No gráfico da figura 6.15, podemos ver logo após as 5 horas de simulação uma queda brusca de cobertura que é recuperada logo em seguida.

Capítulo 7

Conclusões

Nesse trabalho apresentamos dois novos métodos para organização de RSSF. Esses métodos integram o controle de densidade e mobilidade do sorvedouro para otimizar algumas métricas importantes dessas redes.

Em nossos métodos, os nós sensores são organizados em agrupamentos para reduzir o caminho de coleta de dados do sorvedouro móvel, reduzindo o tempo necessário para coletar os dados de todos os nós sensores.

No método SHS, os nós sensores são agrupados de maneira que cada agrupamento tenha no máximo um raio R, onde R é o alcance de comunicação dos nós sensores e do sorvedouro. Assim, quando o sorvedouro estiver posicionado sobre o centro desse agrupamento, ele é capaz de se comunicar com todos os nós sensores do agrupamento usando apenas um-salto. Esse agrupamento é feito de maneira a minimizar o número de agrupamentos. Além disso, devido à comunicação um-salto, esse método também utiliza um protocolo de comunicação baseado em TDM (*Time Division Multiplexing*).

A combinação de comunicação um-salto com TDM leva esse método a uma enorme eficiência energética nas tarefas de coleta de dados, mas o torna ineficiente no tempo de entrega de mensagens ao sorvedouro.

Para tratar essa duas métricas concorrentes propomos o método MHS, que é uma solução intermediária entre a comunicação um-salto, implementada pelo SHS e comunicação multi-saltos, implementadas em métodos que não utilizam sorvedouro móvel e onde as mensagens, para atingir o sorvedouro devem ser roteadas pela RSSF até ele.

O MHS utiliza um esquema de comunicação multi-saltos mas com o número de saltos limitado. Com isso, podemos construir agrupamentos de raio maior que no SHS o que reduz o tamanho da rota do sorvedouro móvel, reduzindo o tempo para a coleta dos dados dos nós sensores. Para construir os agrupamentos em forma de árvore, exigidos pelo MHS, utilizamos o algoritmo descrito em Mirchandani e Francis (1990) para o problema p-Centros invertido.

Além do agrupamento de nós sensores, os dois métodos apresentam um mecanismo de controle de densidade comum. Esse mecanismo através de um modelo centralizado, determina quais nós sensores devem ficar ativos para que todos os pontos da área sobre monitoramento fossem cobertos com o menor número de nós sensores possível.

Esse mecanismo centralizado leva a um controle ótimo da densidade da rede, reduzindo ao máximo a redundância de cobertura. Como os sensores não ficam ativos o tempo todo, o tempo de vida da rede é estendido. Em nossa abordagem, o controle de densidade é feito pelo sorvedouro. Para resolver o problema de controle de densidade, esse foi modelado como o Problema de Cobertura de Conjuntos (Garey e Jonhson, 1979).

Ao dividir os nós sensores em agrupamentos, criamos um problema logístico para o sorvedouro móvel. Precisamos determinar sua rota de movimentação para que ele possa coletar os dados desses diversos agrupamentos. O planejamento eficiente de rotas para mover o sorvedouro sobre os agrupamentos de nós sensores é um problema central em nossos modelos. Esse planejamento tem um impacto significativo em algumas das mais importantes métricas em RSSFs, como o atraso na entrega das mensagens e a taxa de sucesso na entrega de mensagens.

Modelamos o problema de planejamento das rotas como o Problema do Caixeiro Viajante (Dantzig et al., 1954) (PCV) para os dois métodos. Cada agrupamento é modelado como uma cidade do PCV e as distâncias euclidianas entre os agrupamentos são usadas como

medidas de distância. A solução do PCV fornece o planejamento da rota para o sorvedouro móvel.

Para avaliação experimental das abordagens de solução propostas utilizamos simulação. Essa escolha é a mais usual em avaliação experimental de redes de computadores. Devido à sua complexidade, as redes de computadores de um modo geral são difíceis de modelar analiticamente, e por se tratar de uma nova solução, a monitoração de uma RSSF rodando nossas abordagens também é inviável.

As abordagens propostas utilizam intensivamente problemas de otimização combinatória para sua solução. A solução desses problemas foram vastamente estudadas na literatura. Nesse trabalhos utilizamos soluções conhecidas para cada um desses problemas, principalmente utilizando modelos de programação matemática. Esses modelos foram resolvidos via técnicas de otimização combinatória implementadas pelo ILOG Cplex Solver (2006), principal pacote comercial de otimização.

Os resultados computacionais mostram uma significativa melhora em métricas como tempo de vida da RSSFs, cobertura e conectividade. Entretanto na métrica de atraso na entrega de mensagens os métodos com sorvedouro móvel é significativamente menos eficiente que estratégias como o roteamento de dados RT (Figueiredo et al., 2004). Entretanto o ajuste de alguns parâmetros como velocidade do sorvedouro, densidade da rede e número de sorvedouro móvel podem melhorar significativamente essa métrica.

Nas próximas seções apresentamos uma compilação dos resultados computacionais aprentados no Capítulo 6, separados por métrica avaliada.

7.1 Atraso na entrega de mensagens

O atraso na entrega de mensagens em um RSSF é sensível a muitos parâmetros da rede como: densidade da rede, tamanho da área monitorada, velocidade do sorvedouro e número de *sorvedouros* móveis. Apesar do atraso do RT, versão que implementa o algoritmo (Figueiredo et al.,

2004), ser ordens de magnitude menor que as outras versões, ele apresenta um grande crescimento no atraso quando a densidade da rede cresce. Enquanto o SHS tem seu atraso incrementado em 93%, o RT cresce 729% quando a rede passa de 50 para 600 sensores, i.e. crescimento de 1100%.

Com o aumento na velocidade do sorvedouro, as versões com sorvedouro móvel apresentam uma enorme variação no atraso. Com um aumento na velocidade de 0,5 m/s para 5 m/s, o SHS tem o atraso na entrega de mensagens reduzido em 90%, apresentando um atraso médio de aproximadamente 81 milisegundos, enquanto o MHS reduzi o atraso em 75%, apresentando um atraso médio de aproximadamente 119 milisegundos. O SHS se mostrou melhor que o MHS com sorvedouro com velocidade maior que 2,5 m/s (para as condições simuladas).

Com mais de um sorvedouro móvel, a melhoria no atraso na entrega de mensagens foi expressiva. O SHS com a inclusão de mais um sorvedouro móvel (rede com 2 *sorvedouros*) tem o seu atraso médio reduzido em 51,8% e com a inclusão de mais 2 *sorvedouros* móveis (rede com 3 *sorvedouros*) a redução é de 67,5%, para diferentes tamanhos de rede.

O MHS possui basicamente o mesmo comportamento, sendo sua melhora no atraso de 38% com 2 *sorvedouros* móveis, e de 56% com 3 *sorvedouros* móveis com relação a versão com 1 sorvedouro móvel e profundidade da árvore limitada a 2 saltos.

7.2 Tempo de vida da rede

Com relação ao tempo de vida, os métodos propostos aumentam significantemente essa métrica. De forma geral, o RT não apresenta nenhuma variação considerável tendo tempo de vida médio de 192 minutos com o crescimento da rede, assim como as outras versões.

Já o SHS é 58% em média melhor que o RT, chegando a 67% para rede com 600 sensores. A partir de 400 sensores, o tempo de vida das redes com sorvedouro móvel apresentam um crescimento razoável, o que mostra o ganho obtido pela utilização do controle de densidade com redes mais densas.

O MHS apresenta comportamento similar ao SHS, porém o crescimento da eficiência a partir de 400 nós sensores é menor quanto maior for a profundidade máxima das árvores de coleta de dados.

7.3 Taxa de entrega de mensagens

A taxa de entrega de mensagens mede a eficiência na transmissão de dados dos nós sensores para o sorvedouro. Nesse aspecto, todas as versões possuem processos de transmissão eficientes, com destaque para o método SHS.

A versão RT possui o pior desempenho nessa métrica enquanto que o SHS possui o melhor, ficando o MHS com um desempenho intermediário. Isso é esperado devido à forma de funcionamento dos métodos, sendo o MHS uma versão intermediária ao SHS e o RT.

O SHS apresenta uma taxa de entrega de quase 100%, devido ao seu protocolo de comunicação um-salto e ao TDM que evita perda de dados por colisões de pacotes. Enquanto que o MHS apresenta um desempenho quase sempre acima de 90% de mensagens entregues com diferentes densidades de rede.

7.4 Cobertura

A cobertura mede quanto da área monitorada está sobre monitoramento de pelo menos um nó sensor. E nesse quesito, os métodos propostos são extremamente mais eficientes que o RT.

No RT todos os nós sensores morrem em pouco mais de 3 horas de simulação. Vale salientar que praticamente todos os sensores dessa versão morrem ao mesmo tempo, já que essa versão não possui controle de densidade.

Comparado com o SHS, que é a versão que mantêm maior cobertura por um maior período de tempo, com 15 horas de simulação até 75% da área ainda está coberta, e com 25 horas de

simulação essa mesma versão ainda mantêm 36% da área coberta.

Além disso, quanto maior o parâmetro λ mais próximo será o comportamento do MHS e do RT e para λ menores o MHS terá comportamento mais próximo ao SHS. Isso implica que quanto maior o λ mais cedo a área perderá cobertura.

7.5 Considerações para outros cenários

Nos resultados computacionais apresentados no Capítulo 6, é considerado um cenário onde os nós sensores geram mensagens a uma taxa constante e igual em cada nó sensor e nós sensores distribuídos de forma uniforme, como descrito na seção 5.4.

Porém, outros cenários podem ser encontrados em aplicações de RSSF. Como por exemplo, a taxa de geração de mensagens podem não ser constante ou os nós sensores podem ser distribuídos sobre a área de monitoramento de forma uniforme. Os métodos propostos podem ser aplicados sobre outros cenários, além disso algumas alterações podem ser feitas para tornar os métodos mais aderentes a cada cenário.

Um outro cenário possível para as RSSF pode considerar que as mensagens não são geradas a uma taxa constante. Um exemplo são as aplicações orientadas a evento onde mensagens são geradas somente por nós sensores que observam algum evento de interesse em sua área de sensoriamento. Nesse tipo de redes existe a tendência das mensagens serem geradas em diferentes intencidades de acordo com a área geográfica.

Nesse cenário, a utilização da informação do número de mensagens geradas em cada agrupamento de nós sensores poderia guiar a construção da roda do sorvedouro. Uma maneira de fazer isso é a utilização do Problema de Mínima Latência (PML) (Blum et al., 1994). O PML é uma extenção do Problema do Caixeiro Viajante, onde cada cidade tem um peso, e o objetivo é minimizar uma ponderação entre os pesos com o tamanho do caminho do caixeiro da cidade inicial até a cidade visitada. Ou seja, a ordem de visitação das cidades no PML é considerada. Com isso, poderia priorizar os agrupamentos com maior geração de mensagens.

Da mesma forma, a distribuição geográfica dos nós sensores pode ser considerada. Por exemplo, se em RSSF os agrupamentos, formados tanto pelo SHS quanto pelo MHS, tem um número de nós sensores muito variável, uma solução com o PML (como o exemplo anterior) poderia ser utilizada para melhorar a solução dos métodos.

Enfim, os modelos apresentados foram testados para um cenário específico, mas podem ser aplicados com o sem modificações ou adaptações em outros cenários.

7.6 Trabalhos futuros

Alguns trabalhos futuros são vislumbrados para a melhoria principalmente do atraso na entrega de mensagens.

Melhorar a divisão de tarefas entre os *sorvedouros* móveis buscando a redução no atraso das mensagens. Uma possível solução seria utilizar o problema de roteamento de veículos, uma generalização do TSP para mais de um elemento móvel. Essa solução pode levar a construção de rotas menores o que reduziria o atraso. Porém, a grande diferença entre a velocidade de movimentação do sorvedouro e a velocidade de propagação de uma mensagem via rádio torna muito difícil construir métodos que tenha o mesmo desempenho que o RT para a métrica de atraso.

Outro caminho para a redução no atraso seria modificar a forma de construção dos agrupamentos. Uma possível alteração, seria resolver o problema de agrupamento com objetivo de minimizar a distância entre tais agrupamentos. Se uma redução na rota for encontrada com outros métodos de agrupamento, a métrica de atraso na entrega de mensagens será afetada positivamente.

O controle de densidade proposto é estático dentro de cada ciclo do sorvedouro móvel. Porém, isso leva a algumas perdas de cobertura momentâneas. A utilização de métodos de otimização *on-line* poderia ajudar a reduzir essas perdas. A otimização *on-line* lida com problemas onde a solução é dada ao longo do tempo de forma dinâmica. Essas técnicas

poderiam ser utilizadas para solução do problema de controle de densidade de forma dinâmica.

Ainda nos problemas de otimização, a avaliação do impacto da qualidade de solução desses problemas no desempenho dos modelos propostos pode ser interessante. Tratando de redes maiores que as avaliadas nesse trabalho, possíveis perdas de qualidade na solução dos problemas de otimização combinatória podem ser necessárias para tornar a sua resolução viável. Uma maneira de torná-la viável é a redução da qualidade de suas soluções aceitando um certo *gap* na otimalidade da mesma. O impacto de um *gap* de otimalidade deve ser medido nas diferentes métricas de maneira a verificar o seu impacto.

Outra linha para trabalhos futuros é a investigação dos métodos propostos em outros cenários e cargas de trabalho. Além disso, analisar alterações nos modelos para torná-los mais aderentes a cada cenário.

Um outro ponto não explorado nesse trabalho é a agregação de dados, Nakamura et al. (2005a). Os dados em uma RSSF podem ser agregados de forma a minimizar o tráfego na rede. Principalmente no MHS, onde os dados são coletados em uma árvore, a agregação de dados pode ser explorada para melhorar o desempenho da rede.

Apêndice A

Simulador para RSSF

Para a realização do projeto dessa dissertação foi necessária a utilização de um simulador para RSSF. Existem algumas implementações que não atendiam completamente aos requisitos, ou quando atendiam, tinham históricos de difícil utilização por parte de pesquisadores do meio acadêmico. Para evitar tais problemas, uma ferramenta de simulação foi implementada para dar suporte ao trabalho. Essa ferramenta foi desenvolvida como parte desse trabalho de mestrado com um projeto orientado e está descrita em detalhes nesse apêndice.

A.1 Introdução

Nesse projeto orientado foi proposto a especificação e implementação de um arcabouço para simulação de redes sem fio, tanto *ad-hoc* quanto estruturadas.

Esse projeto foi proposto para facilitar os projetos de pesquisa em redes sem fio que tem grande demanda por simulação, para a verificação experimental de seus resultados.

O LaPO, Laboratório de Pesquisa Operacional, possui um grande número de pesquisadores trabalhando na área de rede de computadores, tendo necessidades constantes de realizar simulações. Essas simulações esbarram em problemas recorrentes como definição de ambiente

de simulação, definição de carga de trabalho e utilização de ferramentas.

Esse projeto tentou minimizar esses problemas, tornando o processo de simulação mais eficiente, reduzindo o custo dessas simulações e tornando o processo de simulação mais confiável.

Como resultado, foram levantados os requisitos de simulação para as redes sem fio, um arcabouço de simulação foi implementado sobre o simulador de eventos discreto JiST(Barr et al., 2005; Java in Simulation Time - Scalable Wireless Ad hoc Network Simulator, 2006).

Na próxima seção discutiremos os problemas mais comuns em simulação, esses problemas foram abordados nesse projeto.

A.2 Problemas em simulação

Simulação é o método de avaliação experimental mais comum em projetos em ciência da computação. Em especial, o LaPO possui uma demanda considerável por simulação de propostas em redes de computadores, principalmente em rede celulares e redes sem fio *ad-hoc*.

Essas simulações em rede de computadores esbarram em dois problemas recorrentes a todas: definição de cenário de simulação e ferramentas utilizadas. Nas próximas duas seções trataremos de cada um desses problemas de forma detalhada.

A.2.1 Definição de cenário de simulação

A definição de cenário a ser simulado inclui todos os parâmetros que definem o ambiente que se deseja simular como: distribuição dos usuários móveis, modelos de mobilidade para os usuários, modelos de propagação de sinal de rádio, modelos de consumo de energia e modelos de utilização da rede pelos usuários entre outros.

Esses modelos e parâmetros, não possuem valores reais facilmente encontrados, sendo

definidos estatisticamente através de modelos difundidos na literatura. Diversos modelos para esses parâmetros podem ser encontrados em projetos como o *Momentum*(Models et al., 2006). A implementação dos modelos mais utilizados em uma ferramenta de simulação comum poderia trazer enormes benefícios aos pesquisadores do laboratório e a outros pesquisadores que tenham as mesmas necessidades.

Embora todo trabalho de pesquisa no laboratório tenha um foco diferente, esses parâmetros são quase todos compartilhados. Por exemplo, em trabalhos de Rede de Sensor Sem Fio (RSSF), o modelo de propagação de sinal geralmente é o mesmo para muitos trabalhos, onde esse modelo não é o foco principal. Modelos de mobilidade em simulações de redes celulares são quase sempre necessários e geralmente complexos de serem implementados.

Entretanto, esses modelos podem ser compartilhados por todas as simulações, dependendo unicamente de uma padronização de ferramenta de simulação e implementações reutilizáveis desses modelos.

A.2.2 Ferramentas de simulação

O outro problema é a ferramenta utilizada na simulação. Atualmente existem inúmeras dessas disponíveis para uso, entre elas o ns2(NS2 - Network Simulator 2, 2006) e GloMoSim(Bajaj et al., 1999). O primeiro é o simulador de rede mais utilizado em trabalhos acadêmicos e o segundo é um popular simulador de rede sem fio comercial.

A falta de uma padronização na ferramenta de simulação praticamente impossibilita a implementação dos modelos citados anteriormente. Além disso, os simuladores citados não possuem todos os requisitos para uma utilização direta dos mesmos. Características específicas de simulação de RSSF, simulação de redes celulares, por exemplo, ainda não são bem definidas no ns2, apesar de existirem algumas propostas.

Alguns grupos propuseram extensões para o ns2, incorporando a ele características de simulação que originalmente não possuía, por exemplo, no projeto SensorNet do DCC foi construída uma extensão do ns2 para simulação de redes sensores sem fio.

Ainda referente a ferramenta de simulação, existe uma demanda crescente de escalabilidade dessas ferramentas. Processo de simulação de rede de computadores são extremamente intensivos em processamento e memória. Um novo simulador, desenvolvido pela universidade de Cornell chamado JiST(Java in Simulation Time - Scalable Wireless Ad hoc Network Simulator, 2006) tem como objetivo principal ser um simulador eficiente na utilização desses recursos.

Além disso, o JiST é implementado em ambiente Java, o que o torna portável e principalmente torna sua utilização menos árdua, devido às diversas ferramentas de auxilio disponíveis para esse ambiente.

Em um ambiente Java, o simulador roda em um ambiente controlado tornando sua depuração muito mais simples, diferentemente do ns2, por exemplo, onde um erro pode ser indicado simplesmente por uma mensagem de "Segmentation fault"sem nenhuma informação adicional.

Testes publicados em (Barr et al., 2005) mostram que esse simulador é mais escalável que o ns2 e o GloMoSim. Entretanto, o JiST ainda não possui a maturidade e a mesma quantidade de recursos disponíveis nos outros simuladores.

Atualmente o JiST possui alguns recursos para simulação de redes *ad-hoc* implementados, como modelos de propagação de sinal e o protocolo de camada física 802.11.

O JiST foi a ferramenta de simulação escolhida por prover o ambiente mais fácil de implementação. Essa escolha foi baseada na expectativa de reduzir o tempo de desenvolvimento de cada simulação.

Não escolhendo o ns2, perdemos uma gama de protocolos e algoritmos que já estão disponíveis para esse ambiente, mas ganhamos na facilidade de uso. O ns2 é hoje um simulador muito grande, do mesmo tamanho das queixas de seus usuários. Ele possui muitos componentes implementados, porém essa implementação é geralmente muito difícil de ser utilizada.

Com a escolha do JiST, desenvolvido em Java com a filosofia de ser o mais escalável possível, temos a expectativa de fornecer um arcabouço robusto, com as funcionalidade básicas

e principalmente extensível.

A.3 JiST/SWANS

O JiST é um simulador de eventos discretos desenvolvido na Universidade de Cornell. Ele utiliza reescrita de código executável Java para transformar a linguagem orientada a objeto Java em uma linguagem de simulação.

Através de definições de entidades de simulação, é possível definir métodos nas entidades que representam eventos. Como eventos, tais métodos são executados em tempo de simulação.

O tempo de simulação é controlado pelo núcleo do JiST. Cada método de evento é executado no tempo de simulação agendado pelo usuário. Ao mandar executar um evento, o programador da simulação pode avançar o tempo para que o evento seja executado no tempo de simulação especificado.

O JiST é um simulador de eventos discretos para simulações gerais. Sobre ele foi implementado o SWANS, um simulador de redes *ad-hoc*.

O SWANS possui uma arquitetura inspirada na divisão de camadas comum em elementos de rede, o modelo de camada de referência OSI mostrado na Figura A.1.

A arquitetura do SWANS é um espelho dessa divisão. A Figura A.2 mostra alguns elemento de rede, composto pela junção de diversos componentes. Esses componentes são módulos que representam alguma implementação específica para a camada correspondente.

A interação entre os componentes do elemento de rede segue a mesma idéia do modelo de camadas. Os módulos só têm acesso aos módulos das camadas imediatamente superior e inferior a ele. A divisão de responsabilidades na transmissão é a mesma que no modelo de camadas OSI.

Além dos elementos de rede, o SWANS modela o ambiente da rede na classe *Field*. Essa

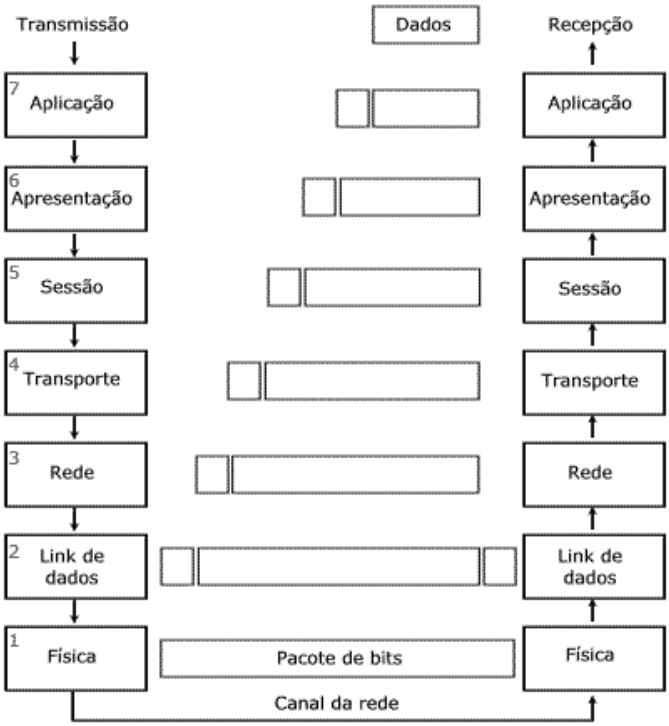

Figura A.1: Modelo de camadas OSI

classe contem todos os elementos de rede da simulação e é a classe responsável pela transmissão de mensagens entre esses elementos.

Na próxima seção será detalhada a extensão feita do SWANS para a geração de um arcabouço para simulação de redes sem fio, especialmente redes RSSF.

A.4 Arcabouço

Nessa seção será mostrado o resultado do projeto. O produto final do projeto foi a implementação de um arcabouço de simulação para rede sem fio. Implícito no desenvolvimento desse arcabouço foram levantados os requisitos básicos que tal arcabouço deveria atender.

Será descrita a arquitetura do arcabouço, mostrando a implementação dos módulos que formam o arcabouço, como esses módulos interagem para realizar a simulação. Também será detalhado os principais componentes do arcabouço, seus objetivos e suas funcionalidades.

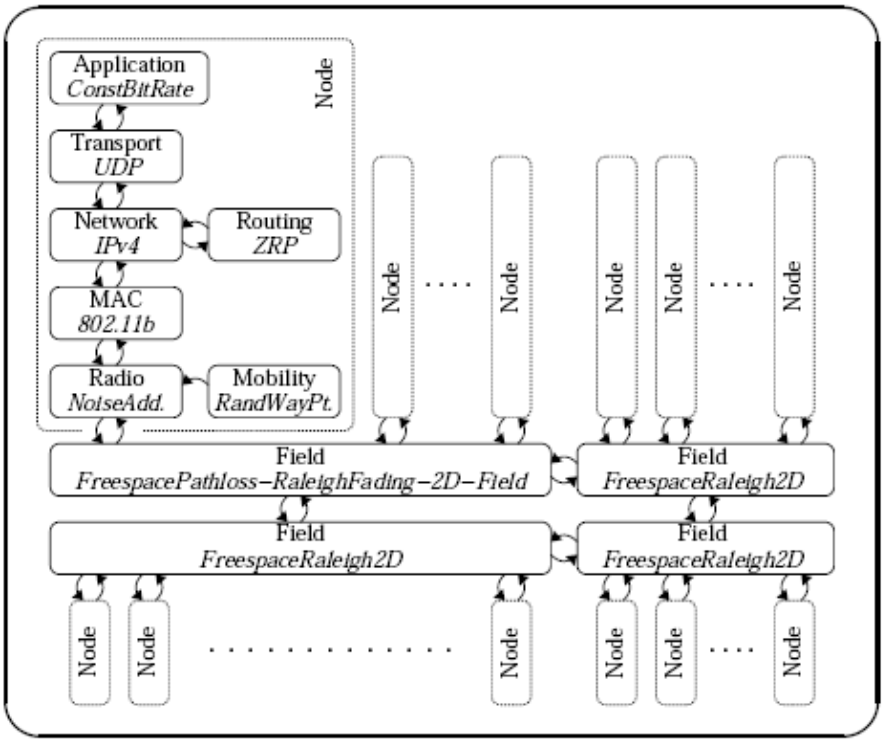

Figura A.2: Arquitetura do JiST/SWANS

A.4.1 Decisões de implementação

A escolha do JIST/SWANS implica na utilização da linguagem Java, linguagem na qual esse simulador foi construído.

A linguagem Java executa sobre uma maquina virtual, o que para muitos a torna lenta. Porém, com melhorias recentes na implementação dessas maquinas o desempenho na execução de uma aplicação Java está muito próximo do desempenho de aplicações executada em código de máquina.

Alguns estudos realizados e publicados pelos autores do simulador(Barr et al., 2005), reportam um desempenho superior ao seus concorrentes, tanto em termos de tempo de processamento quanto em quantidade de memória utilizada. Isso mostra que ao menos a implementação do simulador em Java não diminui sua competitividade em relação a outras ferramentas.

O código do arcabouço foi implementado seguindo as melhores práticas de orientação por objeto, e todo o seu código possui comentários relevantes ao entendimento dos diversos componentes e principalmente da interação entre esses componentes.

O inglês foi a linguagem utilizada para nomenclatura e documentação de tal código Essa escolha foi feita visando uma melhor visibilidade do projeto. Tal visibilidade pode atrair ao projeto outros pesquisadores dispostos a melhorar e acrescentar novas funcionalidades, tornando-o uma ferramenta de simulação mais completa.

A.4.2 Arquitetura

A arquitetura do arcabouço segue a divisão de camadas padrão de um elemento de rede. Além dos módulos das camadas de rede, alguns módulos representam elementos de simulação ou elementos de um ambiente de rede sem fio, como seus usuários, sua forma de movimentação, forma de propagação de sinal entre outros elementos.

Na FiguraA.3 é mostrado a organização em alto nível da arquitetura do arcabouço. Essa figura é um diagrama de classes UML representando a relação entre as classes do arcabouço, suas especialização e relacionamentos.

Nas subseções seguintes será discutido cada um dos componentes dessa arquitetura em mais detalhes.

A.4.3 Elemento de rede

Na arquitetura foram definidos três tipos de elementos de rede: *Sensor*, *FixedSinkNode* e *MobileSinkNode*. Devido a componentes comuns a esses três elementos, foi proposta uma hieraquia de classes de modo a tornar a implementação desses elementos mais simples. Praticamente toda a implementação desses elementos está na classe abstrata *AbstractSensor*.

Essa classe representa o elemento de rede e agrega todos os seus componentes: *Monito-*

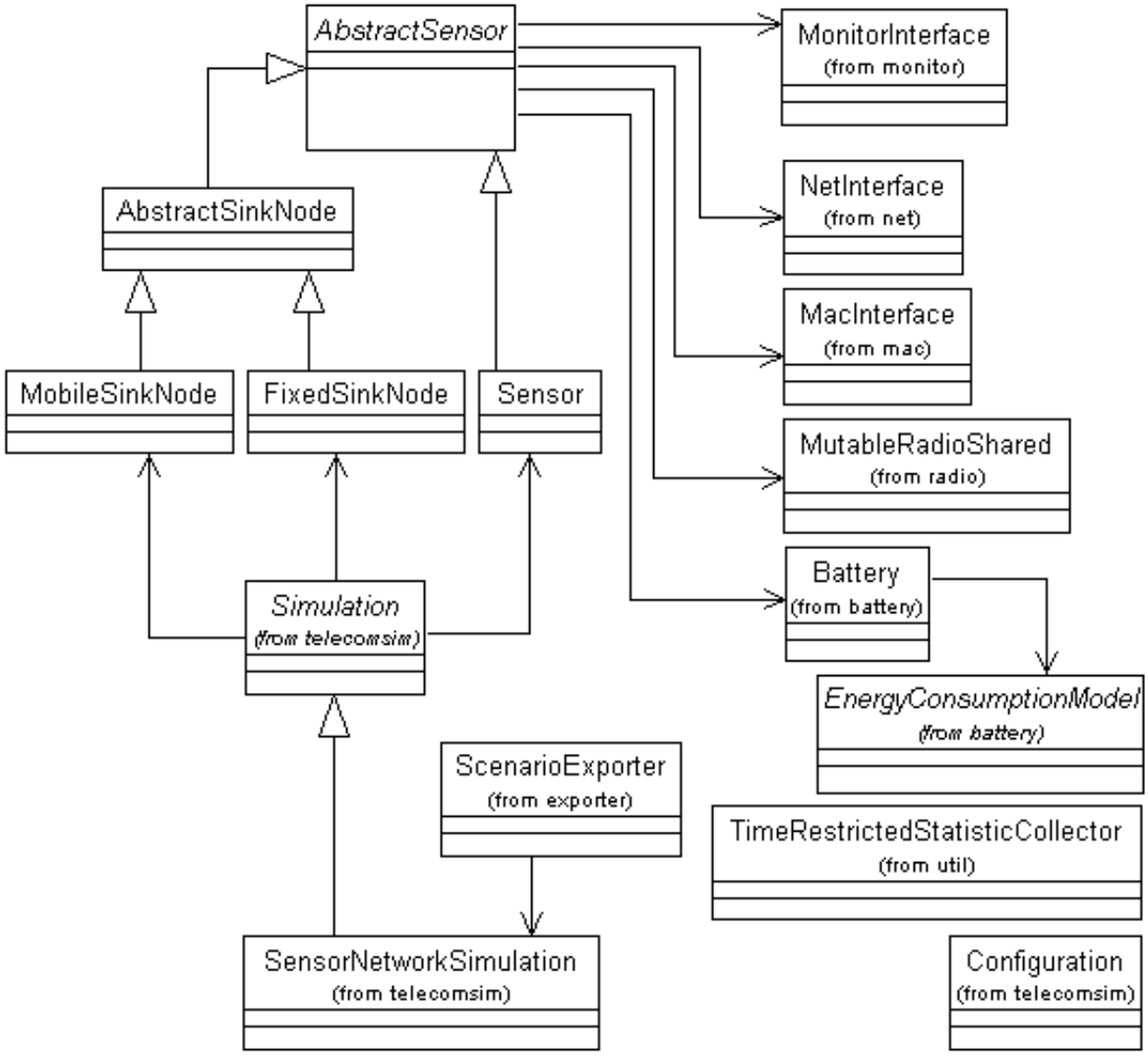

Figura A.3: Arquitetura do arcabouço

rInterface, *Battery*, *MutableRadioShared*, *MacInterface* e *NetInterface*. Estes elementos são montados e integrados pelo elemento para realizar suas funções.

Além das classes que compõem os elementos de rede, as classes que representam os próprios elementos de rede, existem outras classes utilitárias que cumprem objetivos de simulação específicos.

A classe abstrata *AbstractSensor* possui a implementação quase que total do elementos de rede até a camada de enlace. Nessa camada é utilizado o protocolo IEEE 802.11, que já está disponível no simulador JiST/SWANS. Porém, deixa para as classes que a implementam

a definição das camadas superiores, especialmente a camada de rede. Tal camada é o alvo principal dos projetos de pesquisa desenvolvidos no LaPO. Sua implementação geralmente é atrelada a aspectos específicos dos projetos.

As classes *Sensor*, *FixedSinkNode* e *MobileSinkNode* são classes concretas que possuem uma implementação completa para uma RSSF. Essas implementações utilizam como protocolo de comunicação na camada de rede o *DynamicSpanningTree*, uma implementação do algoritmo EF-Tree(Figueiredo et al., 2004). Nesse protocolo, existe sempre um sorvedouro fixo na rede. Esse sorvedouro emite periodicamente mensagens de *broadcast* para todos os elementos da rede para a formação de uma árvore de disseminação de informação dos sensores para esse sorvedouro.

Em implementações de rede de sensores que utilizem características diferentes da especificada nas classes *Sensor*, *FixedSinkNode* e *MobileSinkNode*, as mesmas devem ser estendidas. Porém, essas classes foram criadas para facilitar a troca de pequenos módulos. A implementação delas, torna possível a troca do protocolo de rede com a sobrescrita de um único método, responsável pela criação da camada de rede.

Além disso, algumas características mais variáveis entre simulações diferentes foram parametrizadas, evitando a especialização desnecessária dessas classes. Parâmetros como quantidade de energia, características dos rádios desses sensores são configurados na sua criação através de parâmetros passados pelo construtor dessas classes.

A classe abstrata *AbstractSensor* também controla os demais dispositivos, podendo desligar partes do sensor. O sensor foi quebrado em três módulos que podem ser desligados independentemente: processador, rádio e monitor. É possível especificar o modo de funcionamento do sensor de forma a indicar quais desses três módulos estará ligado em determinado tempo.

A.4.4 Bateria

A bateria foi implementada na classe *Battery*. Essa contém informação de quantidade de energia do sensor a que está associada. Além disso, a bateria encapsula os métodos que realizam o consumo de energia, conforme operação realizada pelo sensor.

A bateria possui métodos específicos para cada operação do sensor, e realiza o decréscimo de energia de acordo com essa operação e informações descritas no modelo de consumo de energia que é implementado na classe abstrata *EnergyConsumptionModel*.

Essa classe descreve o modelo de consumo de energia do sensor para a bateria. O arcabouço possui uma implementação do modelo de consumo de energia do nó sensor Mica Mote 2 *Mica2EnergyConsumptionModel*. Esse modelo de consumo foi derivado da especificação técnica desse nó sensor.

Alguns elementos de rede, como sorvedouro, geralmente não tem associados a ele uma bateria limitada. Em geral assume-se que esses elementos possuem uma quantidade de energia suficientemente grande para realizar todas as suas tarefas sem preocupações. Para esses elementos foi criado um tipo especial de bateria que possui sempre uma quantidade de energia infinita. Essa bateria é implementada na classe *UnlimitedBattery*

A.4.5 Rádio

O JiST/SWANS em sua implementação padrão já possuí implementações de interfaces de comunicação sem fio via radio. Porém, essa interface possui algumas restrições que limitam seu uso. O arcabouço resolve esses problemas especializando a implementação padrão e incluindo as funcionalidades necessárias, dando origem a classe *MutableRadioShared*.

A implementação padrão do JiST/SWANS tem dois problemas principais em relação a seu rádio: controle de potência e integração com a bateria do sensor.

Alguns projetos de pesquisa tem de controlar a potência que cada mensagem é enviada, e

isso não é fornecido pelo JiST/SWANS. A potência de transmissão é definida uma vez para cada sensor e não pode ser alterada na implementação padrão. Na nossa implementação existe um método auxiliar que envia a mensagem em uma potência especificada como parâmetro. Assim cada mensagem pode ter sua potência de envio especificada no momento da transmissão.

Outra limitação da implementação padrão é a integração com a bateria do elemento de rede. A bateria foi criada no arcabouço e sua integração com o rádio é necessária para computar e debitar a energia gasta nas transmissões e recepções de dados do rádio. Essa integração também implementada na classe *MutableRadioShared* permite ao rádio debitar a energia gasta em suas operações.

A.4.5.1 Propagação de sinal

O JiST/SWANS possui alguns modelos de propagação de sinal. Esses modelos são especificados na construção do cenário de simulação, mais especificamente na classe *Field*. Essa classe representa o ambiente onde a rede funcionada. Ela contem os métodos para inserir elementos de rede nesse ambiente e métodos para a transmissão de uma mensagem de um elemento para outro.

Esses modelos são suficientes para um grande número de projetos de pesquisa diferentes, o arcabouço não implementa nenhuma outro modelo de propagação de sinal, somente utiliza os já implementado na distribuição padrão do JiST/SWANS.

A.4.6 Camada de enlace

A camada de enlace é alvo de muitas pesquisas em redes sem fio, sua implementação geralmente é custosa. O JIST/SWANS já fornece a implementação do protocolo IEEE 802.11 para a camada de enlace.

O IEE 802.11 foi desenvolvido para a utilização em rede *ad-hoc* gerais, não é específico para RSSF. Apesar disso, ele é muito utilizado na camada de enlace nas simulações de projetos

de pesquisa nessa área.

O arcabouço não apresenta nenhum novo protocolo para a camada de enlace. Ele somente integra os outros componentes do elemento de rede no protocolo disponível no JiST/SWANS.

A.4.7 Camada de rede

A camada de rede é a mais rica no JiST/SWANS em número de protocolos disponíveis. Nele temos a implementação do TCP/IP, ZRP, DSR e o AODV

Com isso, nessa camada definimos somente métodos de integração com o arcabouço. Novos protocolos de rede criados devem seguir padrões para a integração.

Além de definir a integração, o arcabouço possui a implementação do protocolo EF-Tree(Figueiredo et al., 2004). Essa implementação é bem comum em redes sem fio *ad-hoc* e pode servir de base para comparações com novas propostas.

O JiST/SWANS também possuí a implementação do pacote TCP/IP, disponibilizando para a camada de rede o protocolo de transporte IP. Os protocolos implementados pelo JiST/SWANS também podem ser utilizados livremente no arcabouço.

A.4.8 Monitor

Os sensores são elementos que além de um módulo de comunicação a rádio, possui também um modulo sensor. Esse módulo é responsável por monitorar o ambiente, recolhendo dele alguma medida de interesse que pode ser enviada ao sorvedouro.

Existem hoje diferentes módulos sensores para nós sensores. Esses módulos podem monitorar temperatura, pressão, umidade, luminosidade, vibração entre outros fenômenos físicos.

O arcabouço não modela cada um desses dispositivos sensores, ao invés disso ele modela o funcionamento de todos esses sensores. Apesar das aparentes diferenças de funcionalidades,

esses dispositivos são essencialmente leitores de valores, ou seja, eles são responsáveis por verificar o ambiente e retornar um valor para uma característica desse ambiente.

A definição do monitor é abstrata no modelo, definida pela interface *MonitorInterface*. Essa interface define os métodos necessários para a implementação de um monitor real. O arcabouço possui uma implementação de um monitor que verifica o ambiente em uma taxa constante. A classe *ConstantRateMonitor* implementa esse monitor, sua implementação não efetua leitura de dado, ela somente gera um dado e envia ao sensor a uma taxa constante.

Esse monitor simula um sensor qualquer, e pode ser utilizado em pesquisas onde o valor lido não é uma informação importante, somente é relevante a carga de dados que esse dado impõem na rede.

A.4.9 Módulos utilitários

Além dos módulos relacionados diretamente com a simulação de componentes de rede, o arcabouço oferece alguns componentes que facilitam o processo de simulação. Esses componentes realizam tarefas desvinculadas do processo de comunicação de dados em uma rede, mas estão fortemente ligados a tarefas rotineiras no processo de simulação.

A.4.9.1 Simulation

Esse componente é o concentrador de toda simulação. Implementado pela classe abstrata *Simulation*, possui métodos auxiliares ao processo de simulação. Funciona como ponto de partida para a simulação. Todo processo de simulação deve conter uma classe principal que estende essa classe ou alguma de suas filhas.

Uma especialização dessa classe foi criada para a simulação de RSSF. A classe *SensorNetworkSimulation* implementa alguns métodos auxiliares específicos para RSSF. Essa classe possui métodos que auxiliam a criação do ambiente de simulação, definido pelo tamanho da área onde os sensores serão espalhados e o número de sensores.

Esse é um importante componente do arcabouço. Nele está concentrado todos os elementos da simulação, a partir dele é possível obter informações de todos esses elementos, independente de comunicação entre eles. Isso torna esse componente ideal para a coleta de informações sobre a simulação

Porém, sua utilização requer cuidado. As informações que podem ser obtidas a partir desse objeto podem representar um problema quando sua utilização não for cuidadosa. Por exemplo, um desenvolvedor de uma simulação pode utilizar dados de um elemento de rede A em um elemento de rede B mesmo sem nunca ter requisitado esse dado ao elemento B. Com isso, uma informação de um elemento B está presente em um elemento A, sem que haja comunicação para isso, o que tornaria o processo de decisão com essa informação impossível de se implementar em um ambiente não simulado.

Via de regra, as informações obtidas por esse módulo devem ser utilizada somente com fins de avaliação da simulação nunca em algum processo de decisão da simulação.

A.4.9.2 RandomNumberGenerator

Esse é o módulo responsável pela geração de números aleatório para o resto do arcabouço. A necessidade de números aleatórios é rotineira em simulação. Praticamente todo processo de simulação envolve variáveis aleatórias que seguem as mais variadas distribuições de probabilidade.

A centralização da geração de números aleatórios torna mais simples o processo de simulação já que o processo geralmente é fortemente dependente de números aleatórios. Em sistemas computadorizados a aleatoriedade é dependente da semente passada ao gerador de números aleatórios. Com essa centralização a definição da semente aleatória é feita em um único lugar.

O gerador possui métodos para a geração de números aleatórios baseados na distribuição uniforme e normal. Outras distribuições de probabilidade são necessárias em vários tipos de simulação, porém por limitação de prazo não foi possível implementá-las na primeira versão

do arcabouço.

Distribuições como, a distribuição Exponencial, Binomial, Bernoulli e outras serão implementadas em versões futuras do arcabouço. Essas implementações são fáceis e geralmente baseadas em algoritmos de inversão da função densidade das distribuições. Mais informações sobre esses algoritmos podem ser encontradas em Jain (1991)

A.4.9.3 TimeRestrictedStatisticCollector

A coleta de informações é outra tarefa constante em simulação. Devido ao isolamento das entidades de simulação imposto pelo JiST/SWANS, esse processo não é tão simples. A classe *Simulation* apesar de poder acessar diversas informações de todos elementos de rede, pode não ser capaz de ler algumas informações sobre esses elementos devido ao controle de visibilidade dessas informações.

Para facilitar esse processo, foi desenvolvido um módulo com objetivo único de coletar informações. A classe *TimeRestrictedStatisticCollector* é responsável por coletar todas as informações relevantes ao processo de simulação.

Esse módulo trabalha como uma tabela, onde é relacionado para cada chave um valor. Essa chave representa uma informação que se tem interesse em contabilizar. O valor associado a essa chave representa essa informação no sistema. Esse valor pode ser manipulado e armazenado nessa tabela. Com essa formação podemos concentrar todas informações sobre a simulação em um único lugar e manipular essas informações de forma centralizada.

É muito comum em simulação ignorar dados do início e do final da simulação. Isso é feito, pois esses períodos contem dados instáveis, geralmente associados ao processo de iniciação e finalização da simulação. A eliminação desses dados é necessária para a qualidade do resultado.

Esse módulo oferece a capacidade de ignorar informações que ocorrem fora de um intervalo de tempo. Com isso, pode ser configurado o período de tempo que esse módulo aceitará alterações nas suas informações, tornando possível ao arcabouço ignorar informações geradas

no início e ao final da simulação.

A.4.9.4 Configuration

Uma simulação típica possui alguns parâmetros que devem ser configurados. Esses parâmetros geralmente são variados durante o projeto experimental para verificar o comportamento do sistema com diferentes valores.

Para facilitar a parametrização das simulações o arcabouço possui um módulo, implementado pela classe *Configuration*, que centraliza esses parâmetros. Esse módulo funciona de forma semelhante ao módulo de coleta de informações. Ao invés de coletar informações esse módulo funciona como uma fonte de informações ao sistema.

Como no módulo de coleta de informações, o módulo de configuração é basicamente uma tabela que relaciona uma chave a um valor. A chave identifica a configuração e o valor armazena o valor corrente da configuração.

A.4.9.5 ScenarioExporter

O módulo implementado pela classe *ScenarioExporter* tem por objetivo exteriorizar de forma gráfica as informações da simulação ao longo do tempo. Esse módulo possui todos os recursos necessários para gerar periodicamente imagens com informações da simulação.

Esse módulo é acoplado ao módulo *Simulation* para a extração de informações sobre a rede em simulação. Essas informações são formatadas de forma gráfica e gravadas em arquivos indexados pelo tempo de simulação.

A implementação padrão gera uma imagem mostrando o posicionamento de todos os sensores da rede no espaço. Além disso, é informado o estado do sensor, indicando se ele está ligado ou desligado e também a quantidade de energia de cada sensor.

Os sensores são representados por pequenos quadrados. No canto superior esquerdo desse

quadrado existe um pequeno ponto indicando o estado do sensor, verde para ligado e vermelho para desligado. A cor do sensor indica a quantidade de energia em sua bateria. Com a bateria totalmente carregada sua cor é preto, com a bateria descarregada sua cor é branca. Uma quantidade de energia intermediária é mostrada com um degrade entre preto e branco.

Esse é a imagem padrão formada por esse módulo. Seu comportamento pode ser estendido de duas formas: pela especialização da classe *ScenarioExporter* ou por dois métodos da classe simulação.

Quando a simulação necessitar de uma re-estruturação completa da imagem gerada a especialização da classe *ScenarioExporter* deverá ser a opção do desenvolvedor. Com a especialização, a geração da imagem pode ser totalmente redefinida para as necessidades da simulação. Uma nova implementação deve sobrescrever o método de desenho da imagem e pode definir exatamente o que será desenhado nela.

Porém, para inclusão de mais informações sobre a imagem, existem dois métodos na classe *Simulation* para adição de informações na imagem: um executada antes da classe *ScenarioExporter* começar a desenhar a imagem e outro executado depois da classe *ScenarioExporter* terminar o seu trabalho.

Com isso, existem duas soluções possíveis para estender a geração de imagens do arcabouço, uma para cada tipo de situação necessária a escolha do desenvolvedor da simulação.

Além disso, a geração da imagem é opcional cabendo ao desenvolvedor indicar a sua utilização. No seu construtor ainda é possível indicar a periodicidade da geração da imagem, indicando de quanto em quanto tempo (tempo de simulação) a imagem é gerada.

A.4.9.6 InterfaceMobileUnit

O arcabouço não contém implementação de modelos de mobilidade que são muito úteis em alguns tipos de rede *ad-hoc*, especialmente rede celulares onde o comportamento do usuários dessas redes é de extrema importância para as simulações.

No entanto, o arcabouço define uma forma de solução para a implementação desses modelos. A interface *InterfaceMobileUnit* define os métodos necessários para transformar um elemento de rede em uma unidade móvel.

A implementação dessa interface por um elemento de rede o torna móvel. Por exemplo, a implementação de uma nova classe que estende a classe *Sensor* do arcabouço e que implemente a interface *InterfaceMobileUnit* define um novo elemento de rede móvel.

Essa solução apesar de não implementar um modelo concreto deixa definida sua forma de implementação utilizando o arcabouço.

A.4.10 Celular

A principal diferença de uma RSSF e uma rede celular é a estruturação da comunicação. A RSSF é uma rede sem fio *ad-hoc*, ou seja, é uma rede onde os elementos se comunicam de uma forma não estruturada. A comunicação entre eles ocorre sem a suposição da existência de uma infra-estrutura preparada para isso.

A rede celular se caracteriza pela estruturação da comunicação. Cada elemento de rede só precisa comunicar-se com uma Estação Rádio Base ERB, quando um elemento de rede se comunicar com outro elemento de rede, ao menos uma ERB funciona como intermediária na comunicação. A comunicação entre dois elementos de rede celular nunca ocorre diretamente, sempre ocorre com intermédio de uma ou mais ERBs.

O arcabouço foi desenvolvido tomando por base as necessidades de simulação de RSSF. Fugindo um pouco do objetivo inicial. Porém, sua extensão para a simulação de redes celulares é simples.

Considerando a arquitetura em camadas toda a camada física, que envolve o ambiente de simulação, o rádio e os módulos utilitários podem ser utilizados sem alteração alguma. Novos elementos de rede celulares podem ser construídos utilizando esses componentes.

A principal mudança começa na camada de enlace. A camada de enlace em uma rede celular geralmente feita através de algoritmos do tipo CA, "collision avoidance"ao contrário dos protocolos de enlace disponíveis no arcabouço, como o IEEE 802.11.

A extensão do arcabouço para a simulação de rede celulares passa basicamente pela implementação de camadas de enlace comuns a essas redes, como o CDMA, WCDMA, TDMA entre outras. Com esses novos módulos é possível construir novos elementos de rede que possibilitem a simulação de redes celulares.

A.4.11 Integração bibliotecas externas

Esse projeto possui alguns problemas que podem ser modelados como problemas clássicos de computação, como o Caixeiro Viajante (Dantzig et al., 1954), Cobertura de conjuntos(Cormen et al., 2001; Garey e Jonhson, 1979) e Clusterização (Garey e Jonhson, 1979).

A solução desses problemas é recorrente em projetos do LaPO, por isso a implementação de métodos exatos e aproximados de solução para esses problemas foram desmembrados em um biblioteca genérica independente do projeto do arcabouço de simulação.

Essa biblioteca já conta com soluções exatas para o problema do caixeiro viajante utilizando o CPLEX (ILOG Cplex Solver, 2006), solução exata para o problema de cobertura de conjuntos também utilizando o CPLEX e uma solução heurística para o problema de clusterização utilizando o algoritmo de árvore geradora mínima (Hartigan, 1975).

O processo de simulação utiliza essa bibliotecas externas sem alterações, conforme a arquitetura Java. Essas bibliotecas são importadas para o ambiente de execução da simulação e podem ser utilizadas como classes do próprio projeto.

A.5 Conclusão e trabalhos futuros

O arcabouço possui em sua primeira versão, módulos suficientes para a sua imediata utilização. Para RSSF, a utilização é bem simples, como demonstra o projeto de mestrado que originou esse projeto. Nesse projeto de mestrado, novos elementos de rede foram criados, com uma implementação de camada de rede diferente, incluindo a utilização de elementos móveis.

A comparação de novas propostas com propostas antigas da literatura é uma prática comum em projetos de pesquisa. A utilização do arcabouço pode dificultar essa comparação visto que sua implementação ainda não contém muitas opções de protocolos a disposição. A extensão do arcabouço com a implementação de novos protocolos, novas arquiteturas de rede é um trabalho futuro que deve ser realizado aos poucos, a medida que novas pessoas aderirem ao projeto e novos projetos de pesquisa forem desenvolvidos utilizando-o.

Como citado, a rede celular não foi muito bem referenciada no arcabouço. Nas versões futuras do arcabouço, novos elementos de rede e protocolos devem ser implementados para uma melhor cobertura dos projetos de simulação desse tipo de rede.

O tratamento estatístico de dados também é um processo comum em simulação. Nosso arcabouço tratou, superficialmente, o problema de geração de números aleatórios. Além da implementação de novas distribuições de probabilidade, a implementação de alguns métodos estatísticos podem ser úteis para o arcabouço. Algumas bibliotecas open-source podem ser consideradas como a biblioteca *Apache commons math* (Apache commons math, 2006). Essa biblioteca possui entre outras coisas, métodos para análise numérica, álgebra linear, métodos estatísticos e implementação de algumas distribuições como: Binomial, Chi-squared, Exponencial, F, Gamma, Poisson, t e Weibull.

Outro ponto levantado nesse projeto é a necessidade de padronização de uma nova biblioteca para armazenamento de soluções a alguns problemas clássicos de computação. Propostas complexas, em parte são resolvidas por pequenos problemas clássicos, como caminho mínimo, caixeiro viajante, cobertura de conjunto entre outros.

A solução para esses problemas já foi implementada, tanto soluções exatas como soluções aproximadas, inúmeras vezes no LaPO. Essas soluções, pela falta de padronização e principalmente por falta de uma centralização não são reutilizadas. Como em qualquer projeto de software, e geralmente uma pesquisa em ciência da computação implica no projeto de um ou alguns, o reuso de soluções já implementadas trás muitos benefícios, como a redução no tempo de desenvolvimento desse software e melhor qualidade entre eles. A criação dessa biblioteca poderá trazer benefícios futuros para os projetos de pesquisa.

Essa biblioteca já conta com soluções exatas para o problema do caixeiro viajante utilizando o CPLEX (ILOG Cplex Solver, 2006), solução exata para o problema de cobertura de conjuntos também utilizando o CPLEX e uma solução heurística para o problema de clusterização utilizando o algoritmo de árvore geradora mínima (Hartigan, 1975).

Essa biblioteca também conta com métodos de consolidação de resultados de experimentos. Esses métodos possuem funcionalidades para, a partir de um log de um resultado, gerar uma representação tabular que permite a análise ponto a ponto desses dados, gerando relatórios com algumas análises recorrentes como: diferenças entre diferentes soluções para níveis de parâmetros diferentes, entre outros.

Referências Bibliográficas

Akyildiz, I.; Su, W.; Sankarasubramaniam, Y. e Cayirci, E. (2002). A survey on sensor networks. In A, editor, *IEEE Comm. Magazine*, volume 40, pp. 102–114.

Apache commons math (2006). http://jakarta.apache.org/commons/math/.

Bajaj, L.; Takai, M.; Ahuja, R.; Bagrodia, R. e Gerla, M. (1999). Glomosim: A scalable network simulation environment. Technical Report 990027.

Balanis, C. A. (2005). *Antenna Theory: Analysis and Design, 3rd Edition.* John Wiley and Sons.

Barr, R.; Haas, Z. e Renesse, R. (2005). Scalable wireless ad hoc network simulation. *Handbook on Theoretical and Algorithmic Aspects of Sensor, Ad hoc Wireless, and Peer-to-Peer Networks. Ch. 19*, pp. 297–311.

Bilò, V.; Caragiannis, I.; Kaklamanis, C. e Kanellopoulos, P. (2005). Geometric clustering to minimize the sum of cluster sizes. In *13th Eur. Symp. Algorithms*, volume 3669, pp. 460–471. LNCS.

Blum, A.; Chalasani, P.; Coppersmith, D.; Pulleyblank, B.; Raghavan, P. e Sudan, M. (1994). The minimum latency problem. pp. 163–171.

Cardei, M.; Maccallum, D.; Cheng, M.; Min, M.; Jia, X.; Li, D. e Du, D. (2002). Wireless sensor networks with energy efficient organization. *Journal of Interconnection Networks*, 3(3-4):213–229.

Cerpa, A. e Estrin, D. (2004). Ascent: Adaptive self-configuring sensor networks topologies. *IEEE Trans. on Mobile Computing*, 3(3):272–285.

Chakrabarti, A.; Sabharwal, A. e Aazhang, B. (2003). Using predictable observer mobility for power efficient design of sensor networks. In *The 2nd Intl. Workshop on Information Processing in Sensor Networks (IPSN)*.

Christofides, N. (1979). *The Traveling Salesman Problem*. Wiley Chichester.

Cormen, T.; Leiserson, E.; Rivest, R. e Stein, C. (2001). *Introduction to Algorithms*. MIT Press and McGraw-Hill.

Correia, L. H. A.; Macedo, D. F.; Silva, D. A. C.; dos Santos, A. L.; Loureiro, A. A. F. e Nogueira, J. M. S. (2005). Transmission power control in mac protocols for wireless sensor networks. In *MSWiM '05: Proceedings of the 8th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*, pp. 282–289, New York, NY, USA. ACM Press.

CotsBots (2006). http://www-bsac.eecs.berkeley.edu/projects/cotsbots/ (data último acesso: março de 2007).

cotsdust (2003). Cots dust, large scale models for smart dust. fonte: http://www-bsac.eecs.berkeley.edu/archive/users/hollar-seth/macro_motes/macromotes.html (data último acesso: março de 2007).

Dantzig, R.; Fulkerson, R. e Johnson, S. (1954). Solution of a large-scale traveling-salesman problem. *Operations Research*, 2:393–410.

Dasgupta, K.; Kukreja, M. e Kalpakis, K. (2003). Topology-aware placement and role assignment for energy-efficient information gathering in sensor networks. In *ISCC '03: Proceedings of the Eighth IEEE International Symposium on Computers and Communications*, p. 341, Washington, DC, USA. IEEE Computer Society.

Figueiredo, C.; Nakamura, E. e Loureiro, A. (2004). Protocolo adaptativo híbrido para a disseminação de dados em redes de sensores sem fio auto-organizáveis. *22 Simpósio Brasileiro de Redes de Computadores*, 1:43–56.

Gandham, S.; Dawande, M.; Prakash, R. e Venkatesan, S. (2003). Energy efficient schemes for wireless sensor networks with multiple mobile base stations. In *IEEE GLOBECOM*, San Fransisco, USA.

Garey, M. e Jonhson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W H Freeman.

Garfinkel, R. (1985). *The Traveling Salesman Problem*, chapter 2. Wiley-Interscience Series in Discrete Mathematics and Optimization.

Hartigan, J. (1975). *Clustering Algorithms.* John Wiley & Sons.

Heinzelman, W.; Chandrakasan, A. e Balakrishnan, H. (2002). An application-specific protocol architecture for. *IEEE Trans. on Wireless Communications*, 1(4):660–670.

ILOG Cplex Solver (2006).

Jain, R. (1991). *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling.* John Wiley and Sons.

Jain, S.; Shah, R.; Brunette, W.; Borriello, G. e Roy, S. (2006). Exploiting mobility for energy efficient data collection in wireless sensor networks. *Mobile Networks and Applications*, 11(3):327–339.

Java in Simulation Time - Scalable Wireless Ad hoc Network Simulator (2006).

Jea, D.; Somasundara, A. e Srivastava, M. (2005). Multiple controlled mobile elements (data mules) for data collection in sensor networks. In *IEEE/ACM Intl. Conf. on Distributed Computing in Sensor Systems.*

Juang, P.; Oki, H.; Wang, Y.; Martonosi, M.; Peh, L. S. e Rubenstein, D. (2002). Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with zebranet. *SIGOPS Oper. Syst. Rev.*, 36(5):96–107.

Kansal, A.; Somasundara, A.; Jea, D.; Srivastava, M. e Estrin, D. (2004). Intelligent fluid infrastructure for embedded networks. In *MobiSys '04*, pp. 111–124, New York, NY, USA. ACM Press.

Kim, H. S.; Abdelzaher, T. F. e Kwon, W. H. (2003). Minimum-energy asynchronous dissemination to mobile sinks in wireless sensor networks. In *SenSys '03: Procs. of the 1st international conference on Embedded networked sensor systems*, pp. 193–204, New York, NY, USA. ACM Press.

Li, X.; Wan, P. e Frieder, O. (2002). Coverage in wireless ad hoc sensor networks. In *IEE International Conference on Comunications (ICC)*.

Lin, Y.-B. (1998). Wireless communication: The interactive multimedia cd-rom. *IEEE Personal Communications*, 5:8–9.

Megerian, S. e Potkonjak, M. (2003). Low power 0/1 Coverage and Scheduling Techniques in Sensor Networks. Technical Report 030001, University of California, Los Angeles.

Meguerdichian, S.; Koushanfar, F.; Potkonjak, M. e Srivastava, M. B. (2001). Coverage Problems in Wireless ad hoc Sensor Networks. In *Proceedings of the IEE Conference on Computer Communications (INFOCOM01)*, pp. 1380–1387.

Mirchandani, P. e Francis, R. (1990). *Discrete Location Theory*. John Wiley and Sons.

Models, M.; simulation for network planning e control of UMTS (2006).

Nakamura, E. F.; Figueiredo, C. M. e Loureiro, A. A. (2005a). Information fusion for data dissemination in self-organizing wireless sensor networks. In Lorenz, P. e Dini, P., editores, *Proceedings of the 4th International Conference on Networking (ICN 2005)*, volume 3420 of *Lecture Notes in Computer Science*, pp. 585–593, Reunion Island, France. Springer-Verlag.

Nakamura, F.; Quintão, F.; Menezes, G. e Mateus, G. (2005b). An Optimal Node Scheduling for flat Wireless Sensor Networks. In *IEEE ICN05*, volume 3420, pp. 475–483.

Nemhauser, G. e Wolsey, L. (1988). *Integer and Combinatorial Optimization*. Wiley Interscience Series in Discrete Mathematics and Optimization.

NS2 - Network Simulator 2 (2006).

Oliveira, H.; Nakamura, E.; Loureiro, A. e Boukerche, A. (2005). Directed position estimation: a recursive localization approach for wireless sensor networks. In *IEEE IC3N'05*, pp. 557–562.

Park, S.; Savvides, A. e Srivastava, M. B. (2001). Simulating networks of wireless sensors. In *Proceedings of the 33nd Conference on Winter Simulation*, pp. 1330–1338. IEEE Computer Society.

Polastre, J.; Hill, J. e Culler, D. (2004). Versatile low power media access for wireless sensor networks. In *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pp. 95–107, New York, NY, USA. ACM Press.

Quintão, F. P.; NAKAMURA, F. G. e Mateus, G. R. (2005). Evolutionary algorithm for the dynamic coverage problem applied to wireless sensor networks design. In *IEEE Congress on Evolutionary Computation*, Edinburg, UK.

Rajaraman, R. (2002). Topology control and routing in ad hoc networks: a survey. *SIGACT News*, 33(2):60–73.

Shah, R.; Roy, S.; Jain, S. e Brunette, W. (2003). Data mules: Modeling a three-tier architecture for sparse sensor networks. In *First IEEE SNPA Workshop*, Anchorage, Alaska.

Siqueira, I.; Fiqueiredo, M.; Loureiro, A.; Nogueira, J. e Ruiz, L. (2006). An integrated approach for density control and routing in wireless sensor networks. In *Parallel and Distributed Processing Symposium*, pp. 10–19, Greece.

Slijepcevic, S. e Potkonjak, M. (2001). Power efficient organization of wireless sensor networks. In *IEEE Intl. Conference on Communications*, volume 2, pp. 472–476, Helsinki, Finland.

Small, T. e Haas, Z. J. (2003). The shared wireless infostation model: a new ad hoc networking paradigm (or where there is a whale, there is a way). In *MobiHoc '03: Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, pp. 233–244, New York, NY, USA. ACM Press.

Tilak, S.; Abu-Ghazaleh, N. e Heinzelman, W. (2002). Infrastructure Tradeoffs for Sensor Networks. In *ACM WSNA'02*, pp. 49–58.

Traveling Salesman Problem Home-Page - Princeton University (2003). http://www.math.princeton.edu/tsp/ (data último acesso: março de 2007).

Wagner, H. M. (1969). *Principles of Operations Research - With Applications to Managerial Decisions*. Prentice-Hall.

Wang, W.; Srinivasan, V. e Chua, K. (2005a). Using mobile relays to prolong the lifetime of wireless sensor networks. In *MobiCom '05*, pp. 270–283, New York, NY, USA. ACM Press.

Wang, Z.; Basagni, S.; Melachrinoudis, E. e Petrioli, C. (2005b). Exploiting sink mobility for maximizing sensor networks lifetime. In *HICSS '05*, p. 287.1, Washington, DC, USA. IEEE Computer Society.

XBOW (2006a). http://www.xbow.com/support/support_pdf_files/mpr-mib_series_users_manual.pdf (data último acesso: março de 2007).

XBOW (2006b). Mica2 - wireless measurement system. fonte: http:www.xbow.com (data último acesso: março de 2007).

Xing, G.; Wang, X.; Zhang, Y.; Lu, C.; Pless, R. e Gill, C. (2005). Integrated coverage and connectivity configuration for energy conservation in sensor networks. *ACM Trans. Sen. Netw.*, 1(1):36–72.

Ye, F.; Zhong, G.; Lu, S. e Zhang, L. (2002). Peas: A robust energy conserving protocol for long-lived sensor networks. In *ICNP '02: Proc. of the 10th IEEE Intl. Conf. on Network Protocols*, pp. 200–201, Washington, DC, USA. IEEE Computer Society.

Ye, J. (2006). A Scalable Sensor Localization Algorithm based on SDP subproblems. In *International Symposium on Mathematical Programming*, p. 125.

Zhang, H. e Hou, J. C. (2005). Maintaining sensing coverage and connectivity in large sensor networks. *Intl. Journal of Wireless Ad Hoc and Sensor Networks*, 1(1-2):89–124.

Zhao, W.; Ammar, M. e Zegura, E. (2004). A message ferrying approach for data delivery in sparse mobile ad hoc networks. In *MobiHoc '04*, pp. 187–198, New York, NY, USA. ACM Press.

Zhou, C. e Krishnamachari, B. (2003). Localized topology generation mechanisms for wireless sensor networks. In *IEEE GLOBECOM*, San Fransisco, USA.