UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
CURSO DE CIÊNCIA DA COMPUTAÇÃO
TRABALHO DE GRADUAÇÃO

IVAN ALEXANDRE PAIZ TIERNO

PROTOCOLOS DE ROTEAMENTO PARA RSSFs

ORIENTADORA: PROFª. ROSECLEA DUARTE MEDINA

PROTOCOLOS DE ROTEAMENTO PARA RSSFs

por

Ivan Alexandre Paiz Tierno

Trabalho de Graduação apresentado ao Curso de Ciência da Computação - Bacharelado, da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para obtenção do grau de **Bacharel em Ciência da Computação**

Orientadora: Profª. Roseclea Duarte Medina

Trabalho de graduação n° 264
Santa Maria, RS, Brasil
2008

Universidade Federal de Santa Maria

Centro de Tecnologia
Curso de Ciência da Computação

A Comissão Examinadora, abaixo assinada, aprova o Trabalho de Graduação

PROTOCOLOS DE ROTEAMENTO PARA RSSFS

elaborado por
Ivan Alexandre Paiz Tierno

como requisito parcial para obtenção do grau de
Bacharel em Ciência da Computação.

COMISSÃO EXAMINADORA:

Roseclea Duarte Medina, Dra.
(Presidente/Orientador)

Raul Ceretta Nunes, Dr. (UFSM)

Oni Reasilvia de Almeida Oliveira Sichonany, Msc. (UFSM)

Santa Maria, 24 de Julho de 2008.

LISTA DE TABELAS

LISTA DE ILUSTRAÇÕES

móveis

LISTA DE ABREVIATURAS

CDMA – Code Division Multiple Access

DoS – Denial of Service

FDMA – Frequency Division Multiple Access

JiST – Java in Simulation Time

JVM – Java Virtual Machine

LoS – Line of Sight

MAC – Medium Access Control

MANET - Mobile Ad hoc Network

MEMS – Micro Electro-Mechanical Systems

OFDM – Orthogonal Frequency Division Multiplexing

RSSF – Rede de Sensores Sem Fio

SDMA – Space Division Multiple Access

SWANS – Scalable Wireless Ad hoc Network Simulator

TDMA – Time Division Multiple Access

RESUMO

Trabalho de Graduação
Curso de Ciência da Computação
Universidade Federal de Santa Maria

PROTOCOLOS DE ROTEAMENTO PARA RSSFs
AUTOR: IVAN ALEXANDRE PAIZ TIERNO
ORIENTADORA: ROSECLEA DUARTE MEDINA
Data e Local da Defesa: Santa Maria, 25 de Julho de 2008

As RSSFs são redes que monitoram um ambiente através de dispositivos autônomos (nós-sensores) que coletam dados e cooperam entre si. Estas redes estão sujeitas a condições e restrições especiais que tornam necessário o desenvolvimento de protocolos adaptados a essas diferenças. Devido a importância que os protocolos têm na eficiência da rede este assunto é bastante estudado e muitos protocolos são desenvolvidos. Neste contexto o trabalho tem o intuito de inspecionar e classificar os diferentes protocolos de roteamento disponíveis para posterior análise de um protocolo em um simulador de redes.

Palavras-chave: rede de sensores sem fio; nós sensores; protocolos para RSSFs; simulador de redes;

ABSTRACT

Graduation Work
Computer Science
Federal University of Santa Maria

ROUTING PROTOCOLS FOR WSNs

AUTHOR: IVAN ALEXANDRE PAIZ TIERNO
ADVISER: ROSECLEA DUARTE MEDINA

WSNs are networks that monitor an environment through the use of autonomous devices (sensor nodes) which collect data and cooperate. These networks are exposed to special conditions and constraints that require the development of protocols adapted to these differences. Due to the importance of the protocols in the network's efficiency this subject is widely studied and several protocols are designed. In such context this work has the intent of inspecting and classifying the different routing protocols for subsequent protocol analysis in a network simulator.

Keywords: wireless sensor networks; sensor nodes; WSNs protocols; network simulator;

SUMÁRIO

1 INTRODUÇÃO

Com os avanços tecnológicos das áreas da microeletrônica, comunicação sem fio e micro sistemas eletromecânicos (MEMS – *Micro Electro-Mechanical Systems*) da última década, viabilizou-se de maneira bastante promissora a ascensão das redes de sensores sem fio (RSSFs) (Loureiro et al, 2003).

As RSSFs são redes de dispositivos autônomos (nós-sensores) que coletam dados através de sensores e cooperam entre si. Esses dados podem ser variáveis como temperatura, som, vibração, pressão, etc. Como exemplos de aplicação, dentre inúmeros, pode-se citar :

- na medicina: monitoramento dos sinais vitais de um paciente;
- na agricultura: monitoramento de uma lavoura (temperatura, umidade, concentração de pesticidas..);
- em confrontos militares: detecção e monitoramento de tropas inimigas;
- no meio-ambiente: detecção de incêndios, monitoramento de animais selvagens

As RSSFs diferem das redes comuns, possuindo algumas características peculiares. A comunicação entre os nós é feita através de uma rede *ad-hoc* sem fio. Cada nó-sensor possui capacidade de processamento e armazenamento limitadas e também, espera-se, devem ser baratos para que possam ser adquiridos em grandes quantidades, para que quando combinados possam realizar uma tarefa maior. As condições do ambiente em que venham a atuar podem ser bastante rigorosas(tempestades, nevascas, etc.). Podem ocorrer falhas dos nós e também falhas de comunicação. Apesar desses fatores operacionais, instalar e manter os nós deve permanecer barato. Pelo fato de a configuração manual de uma grande rede de pequenos dispositivos ser impraticável, os nós devem se organizar sozinhos e prover meios de se programar e se gerenciar a rede como um só dispositivo, ao invés de se administrar cada dispositivo individualmente (Culler et al, 2004).

As RSSFs tendem a ser compostas por uma grande quantidade de nós sem posição pré-definida, já que, por exemplo, estes nós-sensores poderiam ser simplesmente soltos por um helicóptero na região do fenômeno a ser monitorado.

Dessa maneira, essas posições devem ser tratadas pelos protocolos de comunicação e gerenciamento da rede, sendo este um vasto campo de pesquisa. Esta área está em franca evolução e as RSSFs são cada vez mais usadas em aplicações comerciais e industriais em que seria difícil ou excessivamente caro fazer uso de sensores com fio comuns.

Este trabalho tem como objetivo o estudo da transmissão de dados em RSSFs. O estudo abrange a revisão bibliográfica sobre a transmissão de dados e posterior execução de testes com um simulador de redes.

2 CARACTERIZAÇÃO DAS RSSFs

Esta seção é um resumo das principais características de uma RSSF. Aqui expõe-se uma série de abordagens que de acordo com Loureiro et al (2003), classificam as redes de sensores segundo sua configuração, sensoriamento, comunicação e processamento. Cada tipo de RSSF se classifica nos itens que veremos a seguir e tais características influenciam diretamente na escolha de um protocolo de roteamento ideal para a aplicação.

2.1 Caracterização segundo a configuração

Composição

Quanto a composição, as RSSFs, podem ser classificadas em homogênea e heterogênea. A rede é homogênea quando for composta por nós com a mesma capacidade física, podendo executar softwares diferentes. A rede é heterogênea quando os nós possuem diferentes capacidades físicas.

Organização

As redes podem ser classificadas em planas, hierárquicas e geográficas. Nas redes hierárquicas os nós estão organizados em *clusters* com um nó líder, nas planas esta distinção não ocorre e na geográfica o roteamento é baseado na localização geográfica dos nós.

Mobilidade

A rede pode ser estacionária, quando os nós permanecem no mesmo local do início ao fim da vida da rede, ou móvel, quando os nós não possuem posição fixa.

Densidade

Pode ser balanceada, quando a distribuição e concentração dos nós em uma região é ideal para o objetivo da rede, densa, quando há alta concentração de nós

por região, ou esparsa, quando a concentração de nós é baixa.

Distribuição

Pode ser irregular, quando a distribuição dos nós não é uniforme em uma área de monitoramento, ou regular, quando a distribuição é uniforme.

2.2 Caracterização segundo o sensoriamento

Coleta

Pode ser periódica quando a coleta de dados ocorre em intervalos regulares, contínua quando a coleta não sofre interrupções, reativa quando a coleta é feita apenas na ocorrência de um evento ou quando solicitado por um observador ou finalmente, pode ser de tempo real, quando são impostas aos nós restrições de tempo para o funcionamento correto da aplicação.

2.3 Caracterização segundo a comunicação

Disseminação

Quanto à disseminação a rede pode ser programada, quando os nós fazem a(s) transmissões em intervalos regulares, contínua quando os dados são transmitidos continuamente ou sob demanda quando os dados são disseminados mediante consulta do observador e à ocorrência de eventos.

Tipo Conexão

Quanto ao tipo de conexão, a rede pode ser simétrica quando todas as conexões entre os nós-sensores, a exceção do nó sorvedouro, têm o mesmo alcance ou assimétrica quando as conexões possuem alcances diferentes.

Transmissão

Quanto à transmissão, a rede pode ser simplex quando os nós possuem

transceptor capaz apenas de fazer a transmissão da informação, half-duplex quando os nós possuem transceptores capazes de transmitir ou receber em um determinado instante ou ainda, full-duplex quando os nós possuem transceptores capazes de transmitir e receber dados ao mesmo tempo.

Alocação de Canal

Quanto à alocação de canal a rede pode ser estática ou dinâmica. A rede é estática quando a largura de banda é dividida igualmente na frequência (FDMA – Frequency Division Multiple Access), no tempo (TDMA – Time Division Multiple Access), no código (CDMA – Code Division Multiple Access), no espaço (SDMA – Space Division Multiple Access) ou ortogonal (OFDM – Orthogonal Frequency Division Multiplexing). A cada nó é atribuída uma parte privada da comunicação, minimizando interferência.

Fluxo de Informação

Quanto ao fluxo de informação as redes podem ser unicast, multicast, gossiping, bargaining ou inundação(*flooding*). Na inundação a comunicação é baseada em *broadcast* gerando alto *overhead* na rede, mas permanecendo imune a mudanças dinâmicas de topologia e alguns ataques de impedimento de serviço (DoS – Denial of Service). Nas redes multicast os nós formam grupos e usam o multicast para comunicação entre os membros do grupo. Nas redes unicast, os nós-sensores podem se comunicar diretamente com o ponto de acesso usando protocolos de roteamento multi-saltos. Nas redes *gossiping*, os nós sensores selecionam os nós para os quais enviamos dados. Finalmente, nas redes *bargaining*, os nós enviam os dados somente se o nó destino manifestar interesse, isto é, existe um processo de negociação.

2.4 Caracterização segundo o processamento

Cooperação

Quanto a cooperação a rede pode ser via a infra-estrutura quando os nós sensores executam procedimentos relacionados à infra-estrutura da rede como, por exemplo, algoritmos de controle de acesso ao meio, roteamento, eleição de líderes, descoberta de localização e criptografia. Localizada, quando os nós sensores executam além dos procedimentos de infra-estrutura, algum tipo de processamento local básico como, por exemplo, a tradução dos dados coletados pelos sensores baseado na calibração. Via correlação, quando os nós estão envolvidos em procedimentos de correlação de dados como fusão, supressão seletiva, contagem, compressão, multi-resolução e agregação.

3 PROTOCOLOS DE ROTEAMENTO

Neste capítulo serão vistos protocolos desenvolvidos para RSSFs. Os protocolos podem ser separados por camadas assim como no modelo OSI. Esta delimitação em camadas serve para que as tarefas de diferentes níveis sejam bem isoladas. Este isolamento permite, por exemplo, definir uma aplicação independentemente do meio de transmissão utilizado, que fica sob encargo da camada física.

Na seção 3.1 é feita uma introdução, com algumas técnicas de roteamento importantes para que, a seguir, nas seções 3.2 e 3.3 sejam descritas brevemente a camada física e de enlace, na seção 3.4 alguns protocolos de roteamento da camada de rede e finalmente na seção 3.5, considerações sobre o desempenho dos protocolos.

3.1 Técnicas de roteamento

Flooding e *gossiping* são duas técnicas clássicas para transmitir informações em RSSFs sem a necessidade de algoritmos de roteamento e manutenção da topologia (Akkaya e Younis, 2005). No *flooding* , cada sensor que recebe os dados faz um *broadcast* para seus nós vizinhos, e assim sucessivamente, até que a informação alcance o seu destino. Já no *gossiping* os dados são transmitidos a apenas um nó vizinho escolhido aleatoriamente, e assim sucessivamente até que alcancem o destino.

Apesar de serem técnicas de fácil implementação elas possuem várias fraquezas. No *flooding* ocorrem problemas como a implosão e a sobreposição. A implosão ocorre quando mensagens duplicadas são enviadas para um mesmo nó e a sobreposição se dá quando um mesmo evento é detectado por mais de um nó gerando mensagens duplicadas. O *gossiping* evita a implosão no entanto causa um atraso na propagação dos dados e é inviável para redes maiores. Uma vantagem do *flooding* é que apesar do alto *overhead* na rede, há imunidade a mudanças

dinâmicas de topologia e alguns ataques de impedimento de serviço (DoS – Denial of Service) (Loureiro et al, 2003).

Em muitas aplicações não é praticável a manutenção de identificadores globais para os nós devido ao grande número de nós e ao *overhead* que isto causaria. Com o surgimento de algoritmos como o SPIN e o *Directed Diffusion* a agregação de dados estabeleceu-se como um paradigma bastante útil no roteamento de RSSFs. A idéia consiste em combinar os dados disseminados por diferentes fontes, diminuindo a redundância, para diminuir as transmissões e o consumo de energia (Krishnamachari et al, 2002). A agregação de dados é muito estudada e existem várias maneiras diferentes de implementá-la, como exemplifica Pinto (2004). A figura 3.1 ilustra dois cenários, sendo um com agregação de dados e o outro sem.

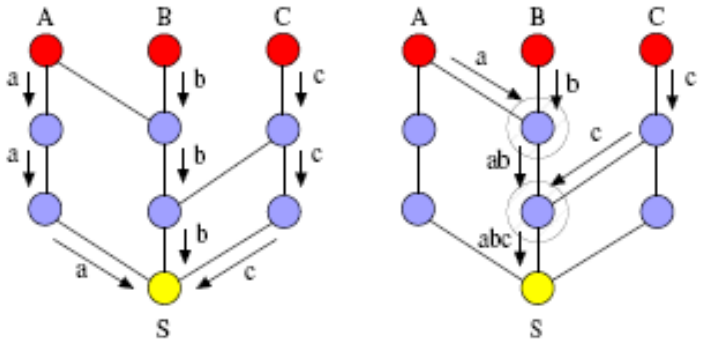


Figura 3.1 – (a) roteamento sem agregação (b) roteamento com agregação (Loureiro et al 2003)

Inicialmente deve ser formada uma árvore ligando os nós fontes ao sorvedouro. Uma referência de desempenho para estes algoritmos é o problema da árvore de Steiner (Hwang et al, 1992). O objetivo é a formação de um grafo cuja soma das arestas seja a menor possível. Portanto, a árvore de Steiner representa a agregação ótima, com menor número de transmissões e consumo de energia. Existem diversos algoritmos para a formação de árvores como CNS, GIT e SPT analisados em Krishnamachari et al (2002). O melhor até hoje conhecido é apresentado em Robins e Zelikovsky (2005), e apresenta uma relação de

aproximação de 1,55 a árvore de Steiner para os casos comuns.

3.2 Camada física

A tarefa da camada física é a transmissão de mensagens entre sensores. Esta camada é responsável por selecionar as freqüências que serão utilizadas, gerar a portadora, detectar, modular e codificar o sinal. Como a minimização do consumo de energia é uma das principais preocupações numa rede de sensores sem fio, a camada física deve tratar de problemas que são comuns em qualquer transmissão sem fio, como a reflexão do sinal. O projeto da camada física também deve levar em conta o meio de transmissão. A comunicação entre os sensores da rede pode utilizar sinais ópticos, infravermelhos ou de rádio-freqüência (RF).

A comunicação óptica tem como principais vantagens o menor consumo de energia que as outras tecnologias, e o fato de não precisar de área física para antenas. Porém, esta comunicação exige uma linha de visada do sinal (LOS - Line of Sight), ou seja, o transmissor e o receptor devem estar alinhados, além de ser sensível às condições atmosféricas (Ruiz et al, 2004).

A comunicação através de sinais infravermelhos também exige uma linha de visada do sinal entre o transmissor e o receptor. A vantagem é, assim como na comunicação óptica, que dispensa área para a antena. Contudo este tipo de comunicação é raramente usado.

As redes Bluetooth e 802.15.4/ZigBee representam as tecnologias atuais mais próximas das RSSFs (Pinto, 2004). A tecnologia Bluetooth foi projetada para atender uma variedade de dispositivos, o que pode provocar desperdício de energia em alguns casos. Por isso um consórcio de empresas, ZigBee Alliance, projetou o protocolo 802.15.4/ZigBee, que permite conectar de forma mais eficiente dispositivos que requerem baixo consumo de energia.

3.3 Camada de enlace

Como as redes de sensores sem fio não exigem a definição prévia de uma

infra-estrutura, os sensores devem possuir algum mecanismo que permita a identificação dos demais sensores na rede. Esta tarefa é realizada pela camada de enlace. Além do controle de acesso ao meio (MAC), esta camada realiza as tarefas de controle de erros, detecção de quadros e multiplexação do fluxo de dados.

Como os sensores têm liberdade para deslocarem-se na rede, o controle de acesso ao meio é responsável pelo estabelecimento da comunicação *multihop*, como forma de organizar a rede e estabelecer rotas. Outra função do MAC é a distribuição dos meios de transmissão entre os sensores que fazem parte da rede. Para isso, existem diversos protocolos, como o CSMA/CA, o EAR, o SMACS e um híbrido TDMA/FDMA. O mais comum é o CSMA, também utilizado no padrão *Ethernet*, com um mecanismo que evita colisões. No caso do *Ethernet*, é comum utilizar-se o padrão CSMA/CD, que evita colisões escutando o meio antes e durante a transmissão. Porém, como isto não é possível nas redes sem fio, utiliza-se o CSMA/CA. A principal diferença é que no CSMA/CD a estação pode detectar se há algum outro nó transmitindo, cancelando imediatamente a sua transmissão. Já no CSMA/CA procura-se apenas evitar as colisões verificando se há algum nó transmitindo antes de começar a transmitir. Em um meio de transmissão com fio cada nó pode receber sinais enquanto faz a transmissão e então detectar essas colisões (com um cabo todas as transmissões tem aproximadamente a mesma força de sinal). Mesmo que um nó-rádio pudesse detectar sinais enquanto fizesse a transmissão, o seu próprio sinal iria mascarar os outros sinais no ar.

O controle de erros também é uma tarefa muito importante da camada de enlace, pois permite a retransmissão de dados que não foram recebidos corretamente, geralmente em função de erros de transmissão, o que é mais comum em redes sem fio. Os principais protocolos de controle de erro utilizados nas redes de sensores sem fio são o ARQ (*Automatic Repeat Request*) e o FEC (*Forward Error Correction*). No ARQ, o receptor envia mensagens de reconhecimento do tipo ACK quando recebe um pacote corretamente e NAK quando detecta a perda de um pacote.

Desta forma, o transmissor é informado dos erros que ocorrem na

transmissão, e pode reenviar os pacotes nos quais houve erros. O protocolo FEC utiliza códigos corretores de erro e transmite dados redundantes. Assim, reduz-se a probabilidade de ocorrerem erros, e, quando estes ocorrerem, é possível corrigí-los. Porém, o tamanho dos pacotes é bastante aumentado, em função dos códigos corretores inseridos no cabeçalho e dos dados redundantes transmitidos.

3.4 Camada de rede

O objetivo dos protocolos de roteamento em RSSFs é a economia de energia, podendo-se sacrificar outras métricas de performance. Este objetivo pode ser almejado através de diferentes idéias. Pode-se procurar as menores rotas, os nós com mais energia disponível e as rotas com o menor número de saltos ou uma abordagem híbrida das anteriores. A seguir expõe-se os protocolos sob suas respectivas topologias.

3.4.1 Roteamento plano

No roteamento plano todos os nós são considerados iguais do ponto de vista funcional, ou seja, a atividade de roteamento é tratada de forma idêntica por todos os nós da rede. Alguns representantes importantes desta classe de algoritmos são apresentados a seguir.

3.4.1.1 SPIN – Um dos projetos pioneiros em roteamento para RSSFs foi a família de protocolos SPIN (*Sensor Protocols for Information via Negotiation)* proposto por Heinzelman et al (1999) .

O SPIN baseia-se em *flooding,* no entanto, evitando a implosão, a sobreposição e com a vantagem de poder se adaptar a disponibilidade de recursos da rede. No *flooding* os dados são enviados em *broadcast* para todos os nós, independentemente da energia disponível em cada um. Estes protocolos podem evitar a transmissão para nós comprometidos, em termos de energia disponível,

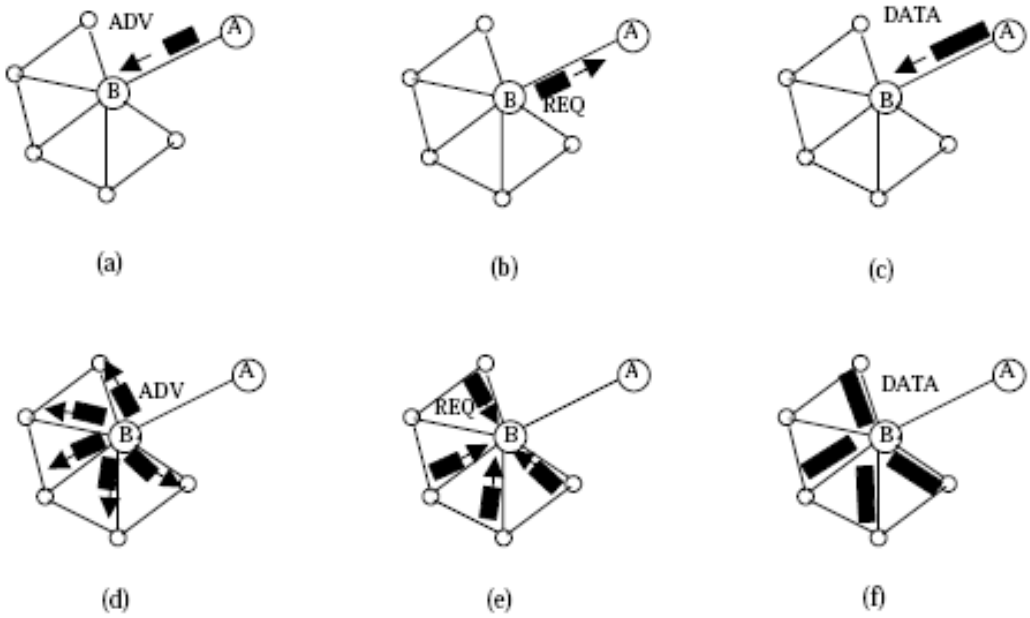
consumindo a energia da rede mais homogeneamente, prolongando assim, a sua longevidade.

O protocolo nomeia seus dados usando descritores de dados de alto nível chamados meta-dados, para fazer a negociação. A negociação ocorre em três fases. Na primeira fase do algoritmo, um nó envia uma mensagem ADV (*advertise*) para os nós vizinhos, procurando interessados. Os nós que recebem a mensagem, respondem a mensagem com um REQ (*request*) caso haja interesse, então finalmente o nó fonte envia os dados. As fases do algoritmo estão representadas na figura 3.2.

O SPIN gera apenas 30%, aproximadamente, da dissipação de energia comparado ao *flooding* e a negociação de meta-dados reduz pela metade a quantidade de dados redundantes (Akkaya e Younis, 2005). No entanto este protocolo não pode assegurar que os dados alcançarão o destino. Por exemplo, se os nós que estiverem interessados nos dados estiverem muito distantes dos nós fontes e não houverem nós intermediários interessados na informação, não haverá como ocorrer o roteamento. Portanto, este protocolo não é uma boa escolha para aplicações como detecção de intrusão, que requerem transmissão confiável dos pacotes em períodos regulares de tempo (Kamal e Al-Karaki, 2004).

Os protocolos SPIN-PP e SPIN-BC foram projetados para redes que funcionam sob condições ideais, onde a energia é abundante e os pacotes nunca são perdidos, sendo que o primeiro atende redes de meios de transmissão ponto-a-ponto (camada física), e o segundo atende redes com meios de transmissão via *broadcasting*.

Os protocolos SPIN-EC e SPIN-RL são modificações dos anteriores, e foram projetados para operar em redes não ideais. O primeiro é um evolução do SPIN-PP que reduz a troca de mensagens quando a energia do sistema é baixa. O segundo é uma versão confiável do SPIN-BC que se recupera de perdas na rede através da retransmissão seletiva de mensagens (Heinzelman et al, 1999).



(a) O nó 'A' envia uma mensagem ADV contendo meta-dados

(b) O nó 'B' então envia uma mensagem REQ manifestando interesse

(c) Os dados são enviados a B

(d) O nó 'B' mostra aos nós vizinhos que possui dados novos

(e-f) Os nós interessados fazem a requisição e recebem os dados

Figura 3.2 – SPIN (Akkaya e Younis, 2005)

3.4.1.2 Difusão Direcionada – Intanagonwiwat et al (2000) propuseram um popular paradigma de agregação de dados, chamado de difusão direcionada. Este algoritmo introduziu o conceito da agregação de dados (Ruiz et al, 2004).

A difusão direcionada propõe o uso de pares atributo-valor para os dados e consulta os sensores, sob demanda, através destes pares. Para fazer uma consulta, é criado um interesse, que é composto por uma lista de pares atributo-valor que podem ser informações como nomes de objetos, intervalos, localização geográfica, etc. A figura 3.3 ilustra um interesse. Este interesse é difundido pela rede na procura de eventos. Os nós podem fazer *caching* dos interesses para uso posterior. A cada transmissão de um interesse é formado um gradiente no nó receptor, que é um link de retorno para o nó remetente do interesse. Assim são formadas as rotas entre o

sorvedouro e o nó fonte e uma delas será escolhida através de repetições, que são feitas pelo sorvedouro, enviando vários interesses, com frequência, através do caminho escolhido, fazendo com que o nó fonte reforce esse caminho.

tipo = veículo com rodas
intervalo = 1 s
area = [100; 200; 200; 400]
timestamp = 01 : 20 : 40 // hh:mm:ss
venceEm = 01 : 30 : 40.

Figura 3.3 – Representação de um interesse

Após a formação das rotas, os *loops* podem ser removidos através da informação armazenada com o cacheamento de dados. O *caching* dos dados proporciona vários usos em potencial, sendo um deles a prevenção de *loops*. Se um nó receber uma mensagem com dados que foram recebidos anteriormente e que estão na cache, a mensagem é silenciosamente excluída, se não os dados são colocados na cache e enviados para os vizinhos (Intanagonwiwat et al, 2000).

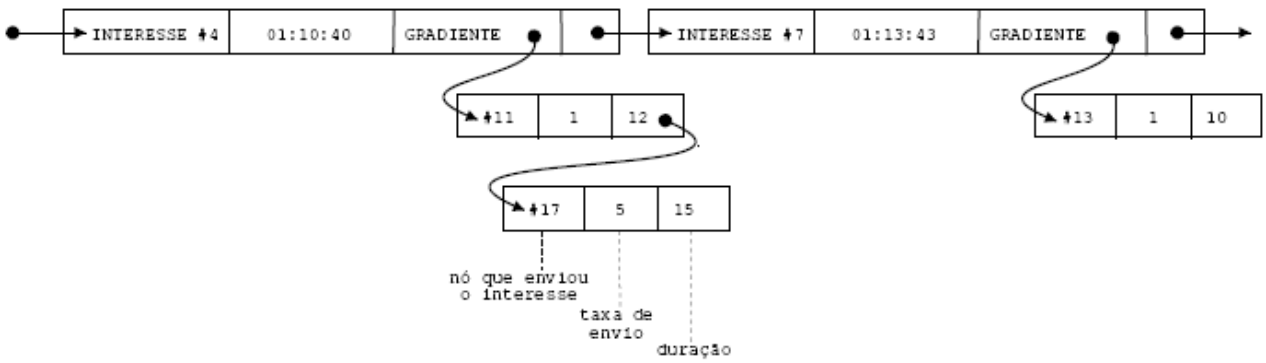
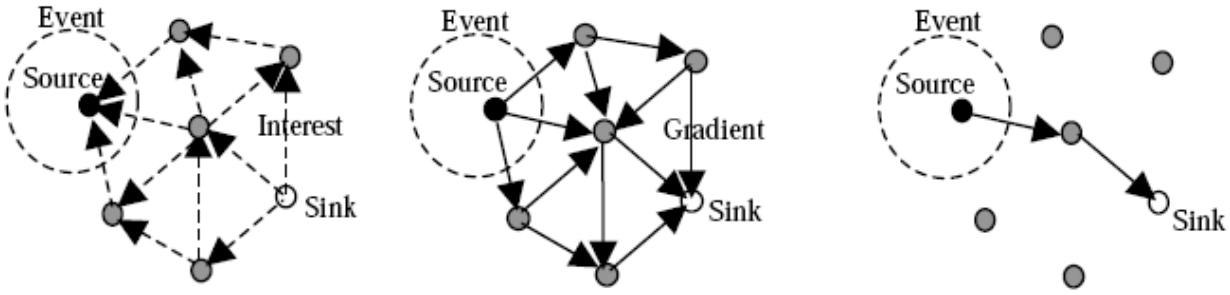


Figura 3.4– Cache de interesses (Pinto, 2004)

Ainda, para reduzir os custos da comunicação Intanagonwiwat et al (2000) propuseram a agregação de dados, que foi explicada no início do capítulo 3.



(a) propagação do interesse (b) retorno dos gradientes (c) reforço da rota escolhida

Figura 3.5 – Difusão Direcionada (Akkaya e Younis, 2005)

A difusão direcionada tem muitas vantagens. Pelo fato de ser centrada em dados toda comunicação é feita entre vizinhos sem haver a necessidade de um mecanismo de endereçamento, logo se adapta bem a redes móveis (Kamal e Al-Karaki, 2004). No entanto, a difusão direcionada não pode ser usada em todas as aplicações de redes de sensores já que é baseada em um modelo de disseminação de dados sob demanda, baseado em eventos. As aplicações que requerem transmissão contínua de dados para o sorvedouro não funcionam eficientemente com um protocolo sob demanda. Então este protocolo não é uma boa escolha para aplicações como monitoramento ambiental. Outro problema é que o processo de combinação entre as consultas e os eventos correspondentes causa uma sobrecarga extra nos nós-sensores (Akkaya e Younis, 2005).

3.4.1.3 Rumour routing – (Braginsky e Estrin, 2002) Rumour routing é uma variação da Difusão Direcionada e foi projetada principalmente para aplicações onde o roteamento geográfico não é praticável. No geral, a difusão direcionada usa o *flooding* para difundir os interesses na rede. No entanto, em alguns casos, existem poucos eventos na rede (para formação dos gradientes) e é desnecessário que haja uma inundação de toda a rede com consultas para receber estes eventos. É mais conveniente que os eventos, ao invés das consultas, sejam espalhados, (Haenselmann, 2006).

Este protocolo usa pacotes longevos chamados agentes. Quando um evento é

detectado por um nó, este é colocado em um tabela local e um agente é gerado. O agente percorre a rede informando sobre o evento. Quando uma consulta é gerada, é também gerado um agente, e este procura na rede por nós que tenham informação sobre o evento desejado. Os nós que sabem a rota para o evento, quando encontrados, respondem a consulta, evitando assim a inundação da rede. Este algoritmo comprovou-se eficiente para redes em que ocorrem poucos eventos (Kamal e Al-Karaki, 2004). Mas a medida que o número de eventos aumenta, a manutenção das tabelas locais e dos agentes se torna insustentável.

Na figura 3.6, os eventos A e B enviam um agente cada. Ambos deixam seu traçado em seu passeio aleatório. O agente B atravessa o traçado do agente A e informa os novos nós sobre ambos os eventos.

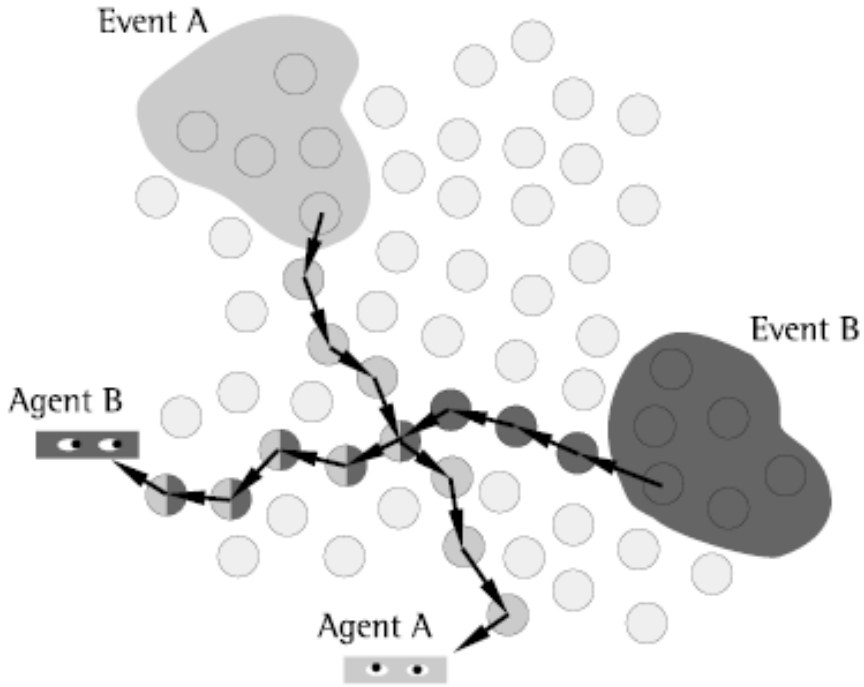


Figura 3.6 – Rumour Routing (Haenselmann, 2006)

3.4.1.4 SAR (Sequential Assignment Routing) - O protocolo SAR (Sequential Assignment Routing), visa facilitar o roteamento multi-saltos. O objetivo é minimizar a

média ponderada de métricas de qualidade de serviço (QoS - Quality of Service) através do tempo de vida da rede. Ele leva em consideração os recursos de energia e as métricas de QoS de cada caminho e a prioridade dos pacotes.

A seleção do caminho é feita pelo nó que gera o pacote a não ser que a topologia mude o caminho fazendo com que o pacote tenha que ser desviado. Tal seleção é baseada em tabelas de roteamento e envia o overhead em caso de falhas.

3.4.1.5 AODV - O AODV (Ad hoc On Demand Vector) é um algoritmo de roteamento genérico para MANETs (Mobile *ad hoc* Networks) frequentemente adaptado e usado em RSSFs. Este é um algoritmo sob demanda capaz de roteamento unicast e multicast. As rotas são mantidas enquanto requerido pelas fontes. Existem algoritmos com mais escalabilidade. Este algoritmo, como será observado nos testes apresentados ao final deste trabalho, causa um considerável aumento de sobrecarga a medida que o tamanho da rede aumenta.

Cada roteador no AODV é essencialmente uma máquina de estados que processa as requisições. Quando for necessário fazer a transmissão de uma mensagem a outro nó o módulo AODV é chamado para determinar o próximo salto (hop). Cada nó possui uma tabela de roteamento que deverá indicar o próximo salto para que a mensagem atinja o destino. Se não houver uma entrada com uma rota o nó armazenará a mensagem em uma fila e iniciará o processo de descoberta. Essa tarefa é realizada através dos pacotes RREQ (*Route request*) para espalhar a requisição, aguardando a resposta que deverá retornar em mensagens RREP (*Route Reply*). As mensagens RREQ são espalhadas através de uma inundação da rede. Cada nó faz um *broadcast* para os seus vizinhos e estes fazem *broadcast* para os próximos e assim sucessivamente.

Para prevenir ciclos cada nó armazena RREQs recentemente encaminhados em um *buffer,* evitando que requisições idênticas sejam espalhadas. Quando a mensagem alcançar o nó destino, ou um nó que saiba a rota para o nó destino, uma mensagem RREP é formada. A mensagem RREP não será transmitida novamente para RREQs iguais, a não ser que a contagem de saltos seja menor. Neste caso a

rota será substituída pela nova rota conhecida.

Evidentemente, esta inundação, representa uma ameaça para a escalabilidade da rede. O autor procurou amenizar o problema implementando um mecanismo que limita o tempo de vida das mensagens (TTL), evitando assim o congestionamento excessivo da rede.

Além destas mensagens existem mais duas, HELLO e RERR que atuam na manutenção e monitoramento da rota, identificando nós que não são mais alcançáveis (dano físico, ou simples esgotamento de energia por exemplo).

3.4.2 Roteamento hierárquico

No roteamento hierárquico são estabelecidas duas classes distintas de nós: nós fontes e líderes de grupo (*cluster heads*). Os nós fontes simplesmente coletam e enviam os dados para o líder de seu grupo que pode executar uma fusão/agregação destes dados antes de enviá-los para o ponto de acesso. Alguns algoritmos desta classe são apresentados abaixo.

3.4.2.1- LEACH (Low Energy Adaptive Clustering Hierarchy)- (Heinzelman et al, 2000) O LEACH foi um dos primeiros protocolos hierárquicos e é um dos mais populares. A idéia consiste em formar *clusters* de nós-sensores baseados na qualidade do sinal recebido e usar apenas os líderes para fazer a comunicação com o sorvedouro.

Para o funcionamento da rede alguns nós são definidos, aleatoriamente, como os líderes, ou *cluster-heads.* Esta eleição é feita periodicamente para distribuir a dissipação de energia homogeneamente na rede. O nó líder então recebe os dados dos nós que compõe o seu cluster e comprime-os através da agregação. A coleta de dados ocorre periodicamente, portanto este protocolo se adapta melhor a redes em que se necessite monitoramento constante, já que se essa condição não for presente ocorrerá desperdício de energia com comunicações desnecessárias. A nova eleição ocorre após um intervalo pré-definido. A figura 3.7 ilustra a formação

dos *clusters*.

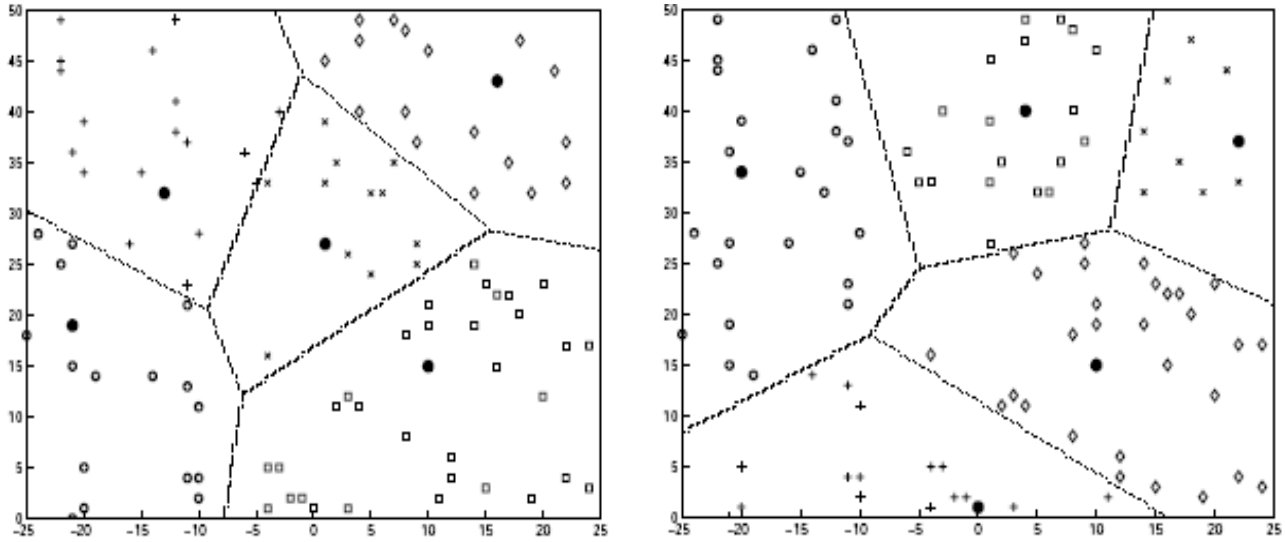


Figura 3.7 – LEACH (Heinzelman, 2000)

Após a eleição, os vencedores fazem *broadcast* de uma mensagem avisando que são os líderes. Ao receber estas mensagens, os nós não-líderes escolhem seus respectivos líderes baseados na qualidade do sinal recebido. Os nós comunicam-se então com seus respectivos líderes. Ao receber estas mensagens cada líder estabelece um intervalo de tempo para que ocorra a comunicação com cada nó do seu grupo, e este agendamento é transmitido aos nós do *cluster*.

Este protocolo no entanto tem suas limitações. Como é necessária a comunicação em um único salto(single hop) entre os líderes dos clusters e o sorvedouro, estes não podem ser espalhados em uma área muito extensa (Akkaya e Younis, 2005).

3.4.2.2- TEEN (Threshold-sensitive Energy Efficient Network)- O TEEN é um algoritmo de roteamento hierárquico similar ao LEACH exceto pelo fato de que os nós sensores podem não possuir dados a serem transmitidos regularmente, a uma taxa constante. Uma das idéias deste protocolo é classificar as redes de sensores em redes pró-ativas e redes reativas. Este protocolo é mais adequado para redes reativas. Uma rede pró-ativa monitora o ambiente continuamente e possui dados a

serem enviados a uma taxa constante. Em uma rede reativa os nós somente enviam dados quando a variável sendo monitorada se incrementa acima de um certo limite. TEEN utiliza a estratégia de formação de cluster do LEACH, mas adota uma estratégia diferente na fase de transmissão de dados.

Ele faz o uso de dois parâmetros, trocados durante a troca da base local:

•Hard Threshold (HT): Limiar no qual o valor continuamente sentido deve ser transmitido.

•Soft Threshold (ST): Variação mínima que justifique o valor a ser transmitido após a primeira vez.

Se o valor exceder HT pela primeira vez, ele é armazenado em uma variável e transmitido durante o intervalo de tempo alocado a transmissão do nó. Em seguida, se o valor monitorado exceder o valor armazenado por uma magnitude de ST o nó transmite o dado imediatamente. O valor enviado é armazenado para comparações futuras. Como a transmissão consome bem mais energia do que o sensoriamento esta característica reduz o consumo de energia. Uma desvantagem que há neste protocolo é que se o HT não é alcançado o nó jamais transmitirá. Para evitar colisões neste protocolo pode-se utilizar escalonamento TDMA ou de CMDA.

3.4.2.3- PEGASIS- Power-Efficient GAthering in Sensor Information Systems (PEGASIS) é um melhoramento do protocolo LEACH. Ao invés de formar vários *clusters,* o PEGASIS forma correntes dos nós sensores de tal maneira que cada nó transmite e recebe de um vizinho e apenas um nó é selecionado dessa corrente para transmitir a estação base (sorvedouro). A informação coletada move-se de nó em nó, sendo eventualmente agregada e finalmente enviada para o sorvedouro.

Portanto, o número de trocas de mensagens será baixo e a comunicação será realizada entre nós próximos uns dos outros. Espera-se com isso que a energia gasta seja menor, se comparada a outros protocolos que requerem muitas trocas de mensagens para eleger líderes e formar grupos, e protocolos em que os nós constantemente trocam mensagens com o nó gateway de forma direta (o gateway geralmente se encontra distante dos nós). Foi mostrado que o PEGASIS pode oferecer um desempenho de 100% a 300% superior em relação ao LEACH,

dependendo do tamanho e da topologia da rede. Tal vantagem de desempenho é alcançada através da eliminação da sobrecarga causada pela formação dinâmica de *clusters* e através da diminuição de transmissões com a implementação da agregação de dados. No entanto o PEGASIS introduz um atraso excessivo para nós distantes na corrente. Ainda, o líder único pode se tornar um gargalo (Akkaya e Younis, 2005).

3.4.3 Roteamento geográfico (baseado em localização)

O roteamento geográfico utiliza informações geográficas para rotear seus dados. Estas informações costumam incluir a localização dos nós vizinhos. Os dados de localização podem ser definidos a partir de um sistema de coordenadas globais (GPS - Global Position System) ou mesmo de um sistema local válido somente para os nós da rede ou válidos somente para subconjuntos de nós vizinhos.

3.4.3.1- GEAR - (Yu et al, 2001) O GEAR (Geographical and Energy Aware Routing) é um protocolo de roteamento geográfico que procura minimizar o consumo de energia da rede. São endereçadas regiões da rede através de retângulos. O repasse de dados utiliza um algoritmo guloso, onde o nó que irá repassar os dados é aquele que possui o menor custo de envio até a região desejada. O custo de envio é calculado em função da distância e energia residual dos nós que compõem a menor rota até a região especificada.

Inicialmente, a função custo é aproximada. A cada pacote enviado para a região, a função custo é recalculada, de forma a otimizar o caminho de repasse dos dados. Ao encontrar a região destinatária dos dados, o protocolo difunde os pacotes através de uma partição recursiva da região em quatro seções. O pacote é enviado para um nó de cada uma das seções, e o algoritmo é aplicado recursivamente, até que as subseções sejam vazias. Em regiões onde a densidade dos nós é pequena, a difusão dos dados é feita via *broadcast*.

O protocolo GEAR se destaca dos demais algoritmos geográficos encontrados na literatura por utilizar informações de toda a rota até o destinatário. O

uso de informações de nós distantes permite uma rota mais eficiente, ao custo de um maior tempo de convergência. Em redes onde há mobilidade de nós, o protocolo irá prover rotas menos eficientes que aquelas encontradas em cenários fixos. Além disto, existem vários casos críticos que necessitam de mecanismos específicos para seu tratamento, o que aumenta a complexidade do protocolo.

3.5 Considerações sobre desempenho dos protocolos

A eficiência dos protocolos de roteamento é largamente dependente da aplicação da RSSF em questão. Não existe uma solução que atenda bem a todas as aplicações, (Tilak et al, 2000; Li et al, 2006). A diversidade de aplicações possíveis proporciona um desafio para os projetistas da rede na escolha do protocolo adequado. Tilak et al (2000) propõe separar a comunicação da aplicação, usada para disseminação dos dados, da comunicação infraestrutural, usada para configurar e otimizar a rede. Esta é uma forma de facilitar a seleção da arquitetura adequada para o tráfego de dados da aplicação. Li et al (2006) analisam os protocolos de roteamento avaliando a que projetos de RSSFs específicos cada um se adapta melhor. A tabela 3.1 ilustra as diferentes opções para as aplicações, e a tabela 3.2 faz uma comparação dos protocolos de roteamento.

Tabela 3.1- Seleção de protocolos de roteamento para aplicações específicas, adaptada de Li et al (2006)

Tipo de aplicação	Projeto	Implanta-ção dos nós	Topologia	Dissemi-nação dos dados	Tamanho	Quantida-de de dados	QoS	Protoco-los
Monitora-mento de habitat	Great Duck	Manual, uma vez	Cluster-heads	Periódica, baseada em localização	10-100	Mínima	Não	SPAN, GAF
Monitora-mento de ambiente	PODS Hawaii	Manual, uma vez	Multi-hop Multi-path	Sob demanda	30-50	Grande	Tolerância a falhas	Difusão Direcionada
	Flood detection	Manual	Multi-hop	Sob demanda	200	Mínima	Tempo-real	COUGAR, ACQUIRE
Saúde	Artificial Retina	Manual, uma vez	Cluster-head	Contínua	100	Máxima	Tempo-real	LEACH
	Vital Sign	Manual	Star	Periódica	10-20	Moderada	Tempo-real	GBR, SAR
Militar	Object Tracking	Aleatória	Multi-hop	Baseada em localização	200	Grande	Colaborativa	GAF
Casa / Escritório	Aware Home	Manual, iterativa	Three-tiered	Híbrida	20-100	Grande	Colaborativa	APTEEN, GEAR
Produção /Comercial	Cold chain	Manual, iterativa	Three-tiered	Contínua	55	Moderada	Confiável	SAR

Ainda, os protocolos da camada física e de enlace podem ter implicações no protocolo da camada de rede, logo estas questões devem ser estudadas pelos projetistas em suas tarefas de conceber a arquitetura.

Tabela 3.2- Comparação de protocolos de roteamento, adaptada de Li et al (2006)

	Classificação	Consumo Energia	Agregação de dados	Escalabilidade	Baseado em consulta	Latência	Overhead	Disseminação dos dados
SPIN	Plana	Ltd.	Sim	Ltd.	Sim	Mod.	Baixa	Eventos
Difusão Direcionada	Plana	Ltd.	Sim	Ltd.	Sim	Mod.	Baixa	Sob demanda
Rumour Routing	Plana	N/D	Sim	Boa	Sim	Mod.	Baixa	Sob demanda
SAR	Plana	N/D	Sim	Ltd.	Sim	Baixa	Alta	Por tabelas
LEACH	Hierárquica	Alto	Sim	Boa	Não	Baixa	Alta	Cluster-head
TEEN	Hierárquica	Alto	Sim	Boa	Não	Mod.	Alta	Threshold
PEGASIS	Hierárquica	Máximo	Não	Boa	Não	Alta	Baixa	Correntes
GEAR	Geográfica	Ltd.	Não	Ltd.	Possível	Mod.	Mod.	Sob demanda

4 SIMULAÇÃO

A escalabilidade deste tipo de rede é ainda um problema aberto a pesquisas. A medida que o tamanho da rede aumenta as dificuldades também crescem. Neste contexto os simuladores desempenham um importante papel na pesquisa e evolução dos protocolos de roteamento. Realizar testes em sensores reais muitas vezes é caro e impraticável (Barr et al, 2005). Adquirir centenas de dispositivos, gerenciar seu software e configuração, encontrar área física para realização dos experimentos e isolá-los de interferência são algumas das dificuldades que tornam esta tarefa custosa. O progresso nesta área depende fundamentalmente da capacidade das ferramentas de simulação e mais especificamente da escalabilidade dos simuladores de RSSFs. Tendo como base estas premissas foi desenvolvido na universidade Cornell o simulador de redes JiST. Este foi o simulador escolhido para a execução dos testes.

Primeiramente, na seção 4.1, expõe-se uma breve dissertação comparando e mostrando algumas vantagens do JiST sobre os populares concorrentes GlomoSim e ns-2. Na seção 4.2 o assunto é o SWANS, uma aplicação projetada para o JiST que simula redes ad hoc sem fio. Finalmente na seção 4.3 são relatados alguns testes feitos neste simulador, mostrando-se a forma de uso da aplicação.

4.1 Comparação entre os simuladores

O JiST apresenta vantagens em relação aos concorrentes mais populares como o ns-2 e o GlomoSim em relação a escabilidade, memória usada e desempenho (Barr et al, 2004; Schoch et al, 2008).

Além destas vantagens, pode-se observar as diferenças de arquitetura. O JiST por ser projetado em java, oferece robustez, muitas biliotecas, facilidade de uso, flexibilidade, portabilidade, coletor de lixo (garbage collector) que evita os vazamentos de memória (memory leaks) que são um problema em simulações maiores, bom suporte a IDE's e não exige o aprendizado de novas linguagens como

no ns-2.

Em Schoch et al (2008) os pesquisadores confrontaram o ns-2 e o JiST, apontando que a comparação dos simuladores usando diferentes implementações de algoritmos/protocolos é delicada, já que a implementação por si só pode fazer diferença. A superioridade de desempenho do JiST foi comprovada, observando-se que o JiST permite mais riqueza de detalhes com um tempo de simulação menor e que tem uma escalabilidade melhor. Com o aumento de número de nós nas simulações, o desempenho do ns-2 cai acentuadamente, enquanto o JiST mostrou-se muito mais escalável. Os pesquisadores também observaram que a versão da JVM (Java Virtual Machine) influencia no desempenho do JiST.

Na tabela 4.1 ilustra-se uma comparação do tempo necessário para simular 5 milhões de eventos e na tabela 4.2 compara-se o uso de memória para uma simulação de 10 mil nós.

Tabela 4.1 - Sobrecarga de memória na simulação de eventos (Barr, 2004)

5×10^6 events	time (sec)	vs. reference	vs. JiST
reference	0.738	1.00x	0.76x
JiST	**0.970**	**1.31x**	**1.00x**
Parsec	1.907	2.59x	1.97x
ns2-C	3.260	4.42x	3.36x
GloMoSim	9.539	12.93x	9.84x
ns2-Tcl	76.558	103.81x	78.97x

Tabela 4.2 – Comparação de uso de memória (Barr, 2004)

memory	entity	event	10K nodes sim.
JiST	36 B	36 B	21 MB
GloMoSim	36 B	64 B	35 MB
ns2	544 B	40 B*	74 MB*
Parsec	28536 B	64 B	2885 MB

4.2 SWANS

O SWANS (Scalable Ad hoc Network Simulator), é uma aplicação desenvolvida para o JIST que simula redes *ad hoc* sem fio, executando aplicações de rede em JAVA sobre redes simuladas. O autor anuncia que este complemento do JiST pode simular redes de até um milhão de nós, uma escalabilidade bastante superior a dos concorrentes. O SWANS executa sobre o JIST como ilustra a figura 4.1. Na camada mais baixa está a JVM (Java Virtual Machine), logo acima o JiST em seguida o SWANS e finalmente a aplicação do programador que pode ser um protocolo de roteamento ou um algoritmo de detecção de falhas por exemplo.

Figura 4.1 - Pilha de Simulação (Barr, 2004)

Com o fim de avaliar a eficiência do SWANS os autores implementaram o protocolo de detecção de vizinhos Heartbeat no SWANS, ns-2 e GlomoSim. O gráfico da figura 4.3 apresenta os resultados da comparação. A aplicação simulada tem duração de 15 minutos e o teste é repetido para diferentes números de nós.

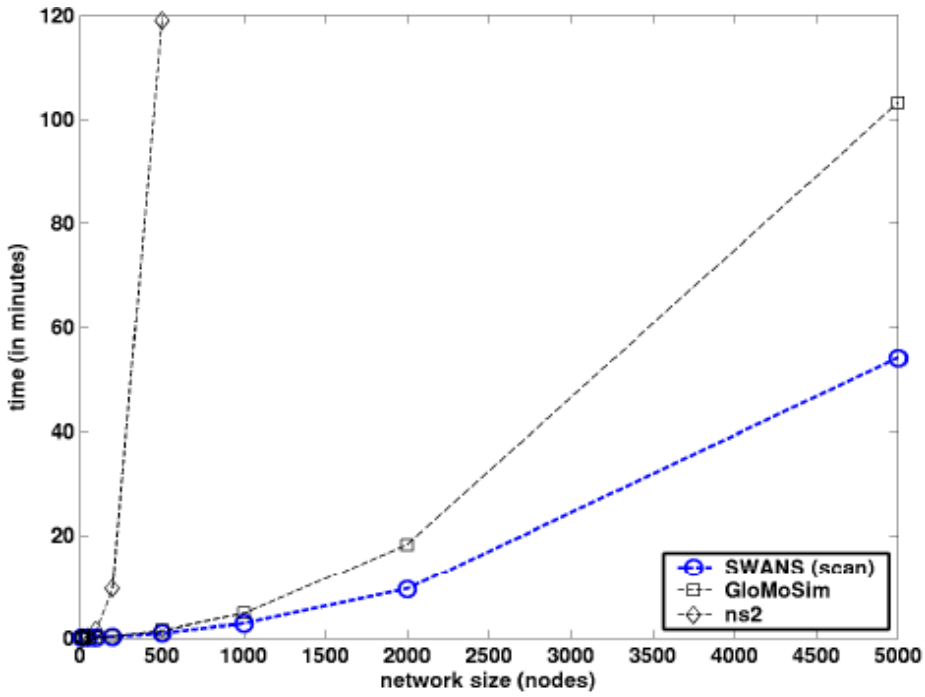


Figura 4.2 - Tempo gasto na simulação da aplicação Heartbeat (Barr , 2004)

O autor em Barr (2004) descreve com detalhes as diferenças entre os simuladores e expõe mais uma série de resultados na forma de gráficos e tabelas ilustrando a vantagem de escalabilidade do JiST/SWANS.

4.3 TESTES

Após o estudo foram feitos alguns testes em um Pentium 3 800mhz com 256MB RAM, sistema operacional Windows XP. A JVM utilizada foi a 1.6.0 no ambiente de desenvolvimento Netbeans. Nesta seção expõe-se a forma de uso da aplicação. Como parâmetros de entrada para a simulação tem-se: o número de nós, a forma em que estes estão arranjados (em grid ou distribuidos aleatoriamente), as

dimensões do campo, a duração do funcionamento da rede, tempo para início da coleta de dados após o início da aplicação, tempo de inatividade após a última transmissão, frequência de transmissão das mensagens por minuto, a mobilidade da rede e se há perdas de pacotes na transmissão.

Na figura 4.3 é ilustrado um exemplo de uso correspondente à linha de comando "**jist.swans.Main driver.aodvsim -n 49 -a grid:7x7 -f 3000x3000 -t 10,600,60 -s 1.0 -m static -l none**" no ambiente de desenvolvimento Netbeans.

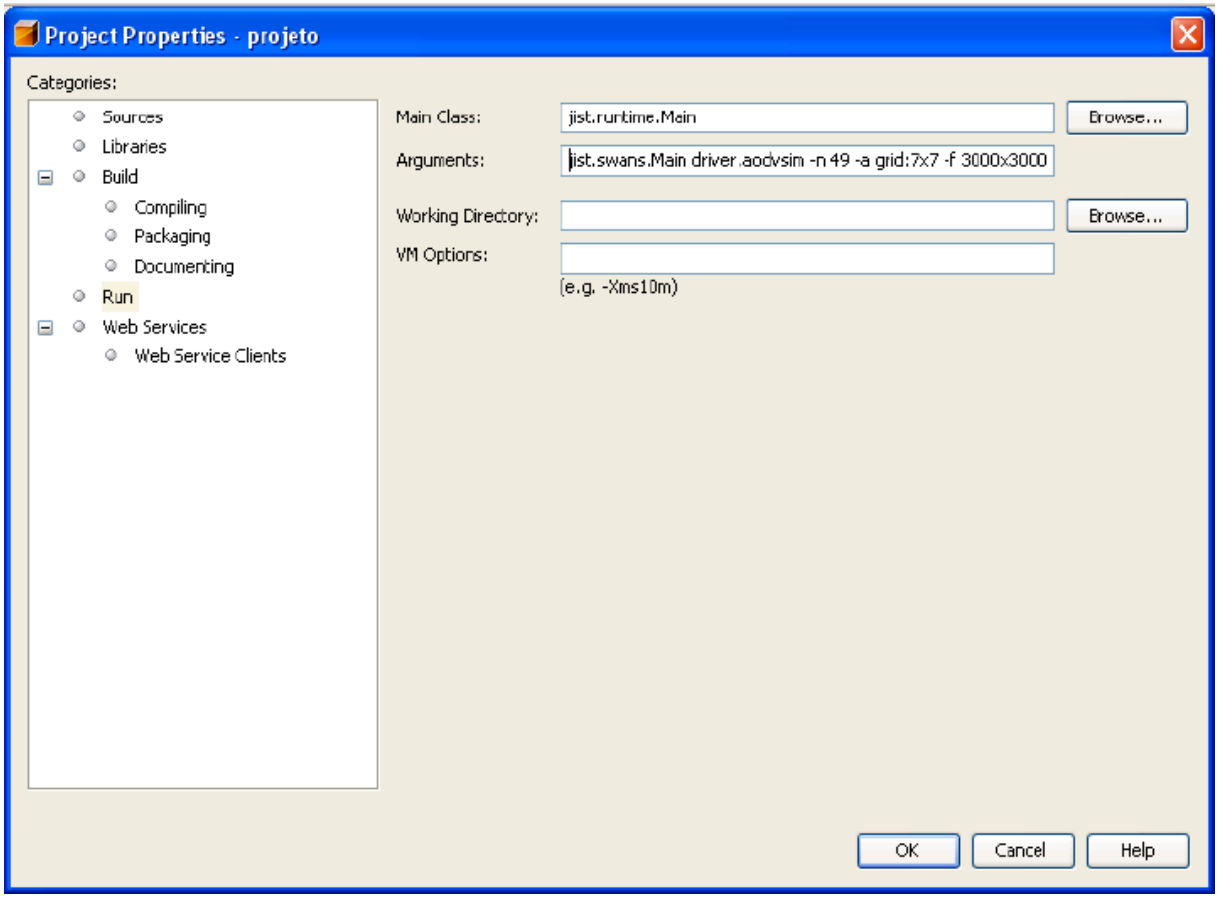


Figura 4.3 – Exemplo de uso no Netbeans

Os pacotes devem estar organizados como explicado no site do JiST para que a aplicação funcione**.** A figura 4.4 ilustra a organização dos pacotes no projeto criado no Netbeans.

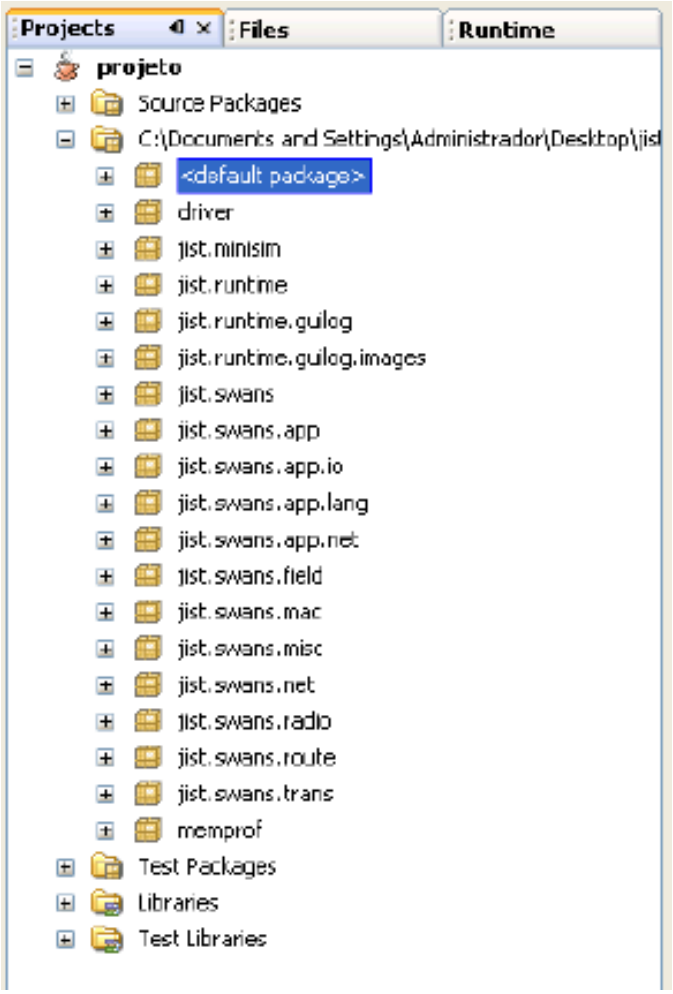
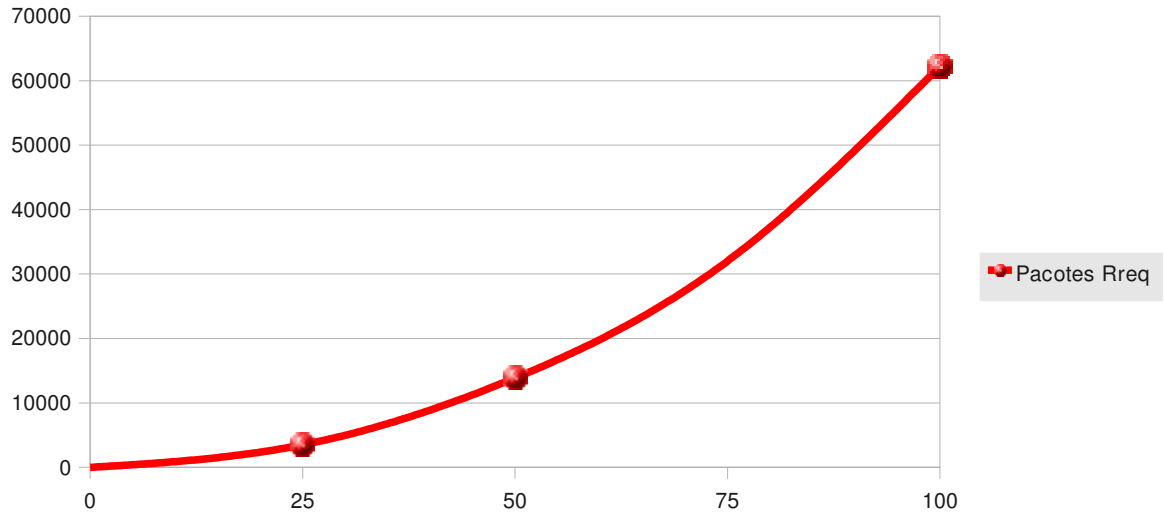


Figura 4.4 – Pacotes do JiST no Netbeans

5 RESULTADOS

Os testes relatados a seguir foram executados em uma área de 3000x3000. A duração da aplicação simulada é de 600 segundos, frequência de mensagens de uma por minuto, em uma rede estática e sem perda de pacotes, variando apenas o número de nós e consequentemente a densidade. A rede está arranjada em um esquema de grid (distribuição homogênea pelo campo) para evitar a aleatoriedade. Os resultados retornados pelo simulador são mostrados no apêndice, ao final do trabalho.

Os gráficos a seguir são baseados nos resultados expostos no apêndice. O gráfico da figura 5.1 ilustra o volume de pacotes Rreq enviados em cada simulação. Estes pacotes são usados na construção das rotas deste algoritmo, como explicado na seção 3.4.1.5. Observa-se que o número de pacotes evolui em progressão geométrica conforme o aumento do número de nós mesmo com o mecanismo que o autor implementou para amenizar o problema.



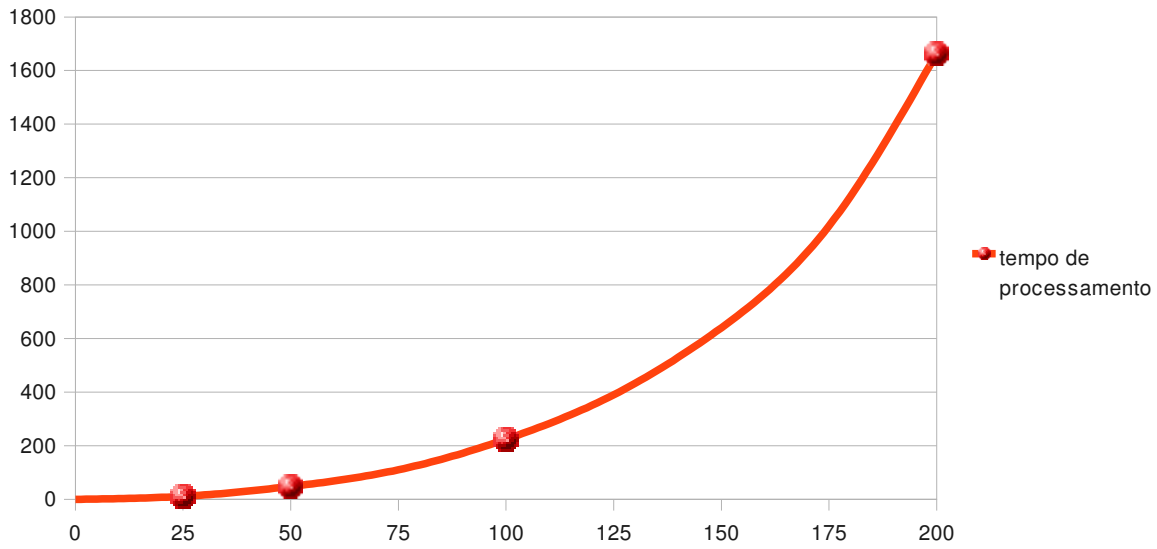
eixo y: Pacotes Rreq eixo x: número de nós

Figura 5.1 – Pacotes Rreq nas redes estáticas

O gráfico da figura 5.2 ilustra o tempo de processamento em cada simulação.

Observa-se novamente que a curva é aproximadamente quadrática. O tempo necessário para a simulação de 196 nós foi de quase 28 minutos.

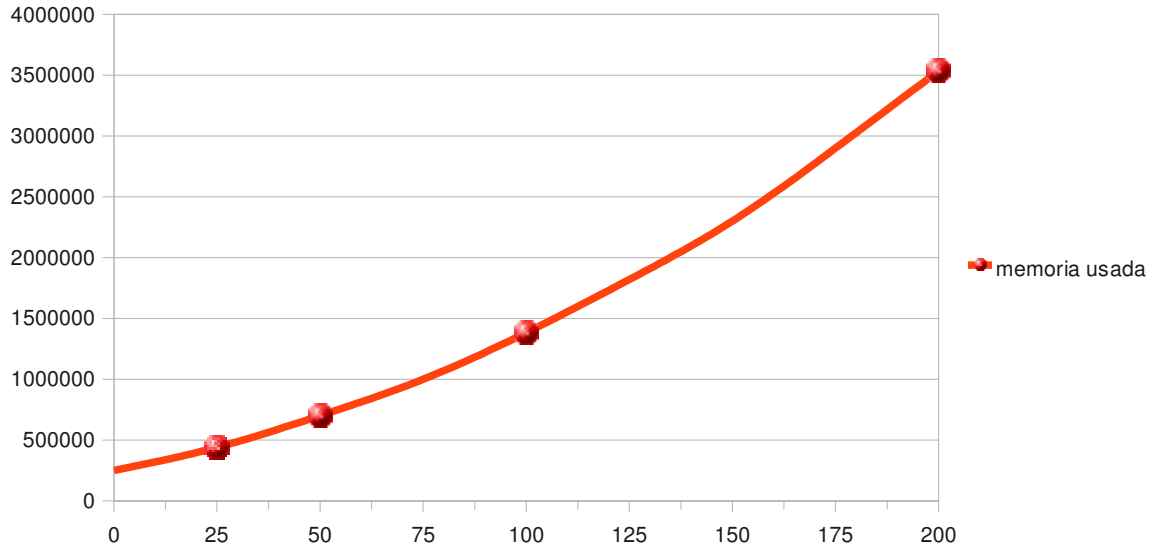
Finalmente, no gráfico da figura 5.3, observa-se o uso de memória de acordo com o número de nós. Na simulação de 196 nós foi necessário mais de 3MB, havendo também uma progressão geométrica, embora neste gráfico, devido a evolução mais lenta da quantidade de memória necessária, a curva não seja tão acentuada quanto nos gráficos anteriores.



eixo y: tempo em segundos eixo x: numero de nós

Figura 5.2 - Tempo de processamento da simulação nas redes estáticas

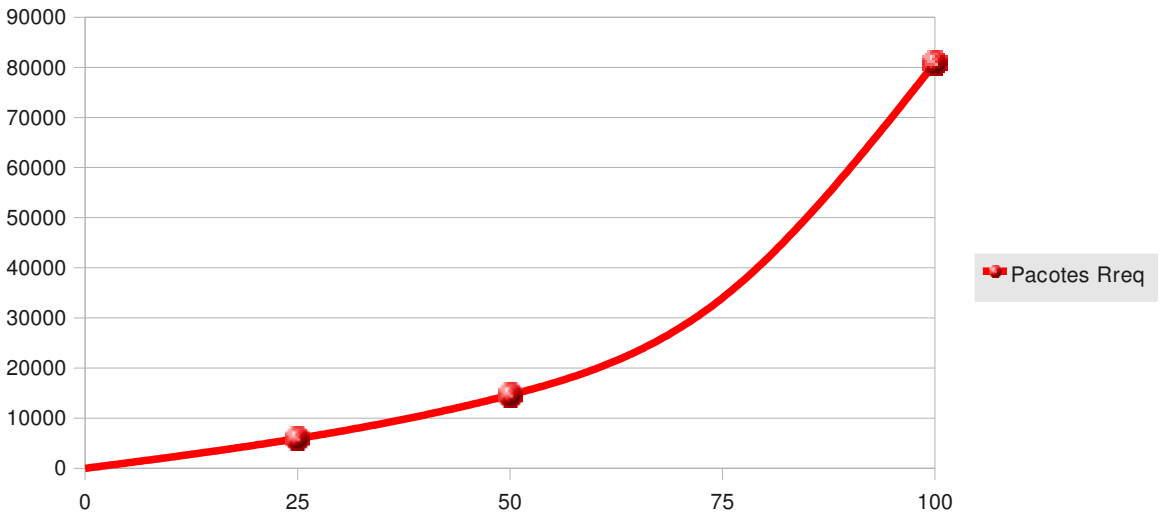
Além destes, foi testado também o desempenho quando a rede for móvel e foi possível constatar que, como esperado, em uma rede com mobilidade, o número de mensagens e a sobrecarga de memória aumentam em relação às redes estáticas. Isso se deve as reconfigurações de rotas que se fazem necessárias devido a movimentação dos nós.



eixo y: memoria usada, em bytes *eixo x: numero de nós*

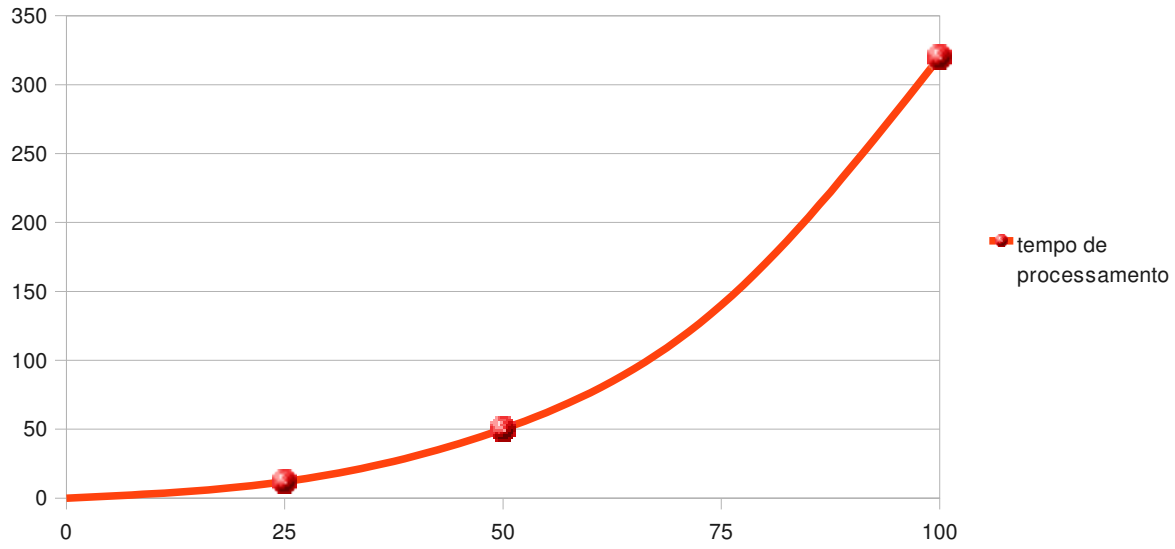
Figura 5.3 – Uso da memória nas redes estáticas

Os resultados retornados pelo simulador para a simulação de redes móveis são apresentados no apêndice ao final do trabalho. Os gráficos das figuras 5.4 e 5.5, assim como os anteriores para redes estáticas, ilustram os resultados.



eixo y: Pacotes Rreq *eixo x: número de nós*

Figura 5.4 – Pacotes Rreq nas redes móveis



eixo y: tempo em segundos eixo x: numero de nós

Figura 5.5 - Tempo de processamento da simulação para as redes móveis

O AODV tem como vantagem pelo fato de ser um protocolo sob demanda, a diminuição do desperdício de energia quando a rede não precisa estar funcionando. Outra vantagem é que este protocolo não tem sobrecarga de roteamento para rotas já estabelecidas, diferentemente de outros protocolos classificados como pró-ativos. No entanto, há uma latência maior no estabelecimento das conexões.

6 CONCLUSÃO

Neste trabalho foi estudada a transmissão de dados em RSSFs. Este é um assunto bastante investigado já que os protocolos podem causar grande impacto na eficiência da rede. Tanto o desenvolvimento e uso de simuladores quanto o estudo de protocolos são temas importantes no projeto de redes e motivam muitas pesquisas.

Este estudo abrangeu a caracterização e classificação das RSSFs e uma pesquisa sobre os protocolos de roteamento. Ao final, para sedimentar esses conhecimentos o simulador JiST/SWANS foi estudado e usado nos testes para subsequente análise.

Ao longo da realização deste trabalho pôde-se verificar a importância de se investigar as peculiaridades e requisitos de uma aplicação antes de projetar-se a arquitetura da rede. O desempenho de cada protocolo é largamente dependente da aplicação, logo o projetista deve procurar os protocolos que melhor se adaptem ao contexto.

Pode-se concluir que esta pesquisa alcançou as metas projetadas e poderá servir em trabalhos futuros na área, sejam estes trabalhos com o simulador JiST ou trabalhos com sensores reais. Inclusive poderá auxiliar no projeto “Uso de telemetria para transmissão de dados de desempenho de máquinas agrícolas visando o gerenciamento”.

REFERÊNCIAS BIBLIOGRÁFICAS

Akkaya, K.; Younis, M. **A Survey on Routing Protocols for Wireless Sensor Networks** Elsevier Ad Hoc Network Journal, vol. 3, no. 3, pag. 325--349, 2005**.** Disponível em: **http://www.cs.umbc.edu/~kemal1/mypapers/Akkaya_Younis_JoAdHocRevised.pdf**

Al-Karaki, J.; Kamal, A. - **Routing techniques in wireless sensor networks: a survey** – Wireless Comunications, IEEE, vol. 11 pag. 6-28, 2004. Disponível em: **http://www.ece.iastate.edu/~kamal/Docs/kk04.pdf**

Barr, R. - **An efficient, unifying approach to simulation using virtual machines -** PhD dissertation, Cornell University, Maio de 2004. Disponível em: **http://jist.ece.cornell.edu/docs/040517-thesis.pdf**

Barr, R.; Haas, Z.; Renesse, R.; - **Handbook on Theoretical and Algorithmic Aspects of Sensor, Ad hoc Wireless, and Peer-to-Peer Networks -** Cap. 19, pag. 297-311, CRC Press, 2005. Disponível em: **http://jist.ece.cornell.edu/docs/050416-bookch-camera.pdf**

Culler, D.; Estrin, D.; Srivastava, M. **Overview of Sensor Networks** publicação em Computer, volume 37, págs. 41-49, Agosto de 2004. Disponível em: **http://csdl2.computer.org/comp/mags/co/2004/08/r8041.pdf**

Estrin, D. **Rumour Routing Algorithm for Sensor Networks –** em Proceedings of the First Workshop on Sensor Networks and Applications (WSNA), Atlanta, GA, Outubro de 2002.Disponível em: **http://research.cens.ucla.edu/people/estrin/resources/conferences/2002sept-Braginsky-Estrin-Rumor.pdf**

GLOMOSIM - **Global Mobile Information Systems Simulation Library**. Disponível em: **http://pcl.cs.ucla.edu/projects/glomosim/**

Haenselmann, T. **Sensornetworks** Livro publicado pelo autor na internet, 2006. Disponível em:

http://www.informatik.uni-mannheim.de/~haensel/sn_book/

Heinzelman, W. **Energy-Efficient Communication Protocol for Wireless Microsensor Networks** em Proceeding of the Hawaii International Conference System Sciences, Hawaii, 2000. Disponível em: **http://pdos.csail.mit.edu/decouto/papers/heinzelman00.pdf**

Hwang, F. K.; Richards, D. S.; Winter, P. **The Steiner Tree Problem** Elsevier, North-Holland, ISBN 0-444-89098-X (Annals of Discrete Mathematics, vol. 53), 1992.

Intanagonwiwat, C.; Govindan, R.; Estrin, R.; Heidemann, J.; Silva, F. **Directed Diffusion for Wireless Sensor Networking** em Proceedings of the 6th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'00), Boston, MA, Agosto de 2000.
Disponível em:
http://www.cs.huji.ac.il/labs/danss/sensor/sensors/intanagonwiwat_03directed diffusionforwireless.pdf

Krishnamachari, B.; Estrin, D.; Wicker, S. **Modelling Data Centric Routing In Wireless Sensor Networks** in the Proceedings of IEEE INFOCOM, New York, NY, Junho de 2002. Disponível em:
http://www.parc.com/zhao/stanford-cs428/readings/Networking/Krishnamachari_infocom02.pdf

Kulik, J.; Heinzelman, W.; Balakrishnan, H. **Negotiation-based protocols for disseminating information in wireless sensor networks** (1999, revisado em 2002) Disponível em: **http://lecs.cs.ucla.edu/lecs-reading/spring2001/spin-monet.ps**

Lin, C. **AODV Routing Implementation for Scalable Wireless Ad-Hoc Network Simulation (SWANS)** relatório de projeto, Abril 2004 – Disponível em: **http://jist.ece.cornell.edu/docs.html**

Loureiro, A. A. F.; Nogueira, J. M. S.; Ruiz, L. B.; Mini, R. A. de F.; Nakamura, E. F.; Figueiredo, C. M. S. **Redes de sensores sem fio** SBRC 2003. Disponível em: **http://www.dcc.ufmg.br/~loureiro/cm/docs/sbrc03.pdf**

Narasimha, N.; Gopinath, K. **A survey of routing algorithms for wireless sensor**

networks , 2005. Disponível em:
http://journal.library.iisc.ernet.in/vol200606/paper2/Datta_NN.PDF

ns-2 - **Network Simulator 2**. Disponível em: **http://www.isi.edu/nsnam/ns/**

Pinto, A. J. G. **MECANISMO DE AGREGAÇÃO DE DADOS EMPREGANDO TÉCNICAS. PARAMÉTRICAS EM REDES DE SENSORES** , 2004. Disponível em:
www.gta.ufrj.br/ftp/gta/TechReports/Antonio04/Antonio04.pdf

Robins, G.; Zelikovsky, A. **TIGHTER BOUNDS FOR GRAPH STEINER TREE APPROXIMATION** SIAM Journal on Discrete Mathematics Philadelphia, USA Volume 19 , Issue 1 Pages: 122 – 134, 2005.
Disponível em:
http://www.cs.virginia.edu/~robins/papers/Robins_Graph_Steiner_Approximation.pdf

Ruiz, L. B., Correia, L. H., Vieira, L. F., Macedo, D. F., Nakamura, E. F., Figueiredo, C. M., Vieira, M. A., Bechelane, E. H., Câmara, D., Loureiro, A. F., Nogueira, J. M., da Silva, D. C. **Arquiteturas para Redes de Sensores Sem Fio** (SBRC 2004) , 2004. Disponível em:
http://homepages.dcc.ufmg.br/~linnyer/linnyerSBRC04.zip

Schoch, E., Feiri, M., Kargl, F., Weber, M. - ***Simulation of Ad Hoc Networks: ns-2 compared to JiST/SWANS* -** First International Conference on Simulation Tools and Techniques for Communications, Networks and Systems (Simutools 2008), Março 2008 – Disponível em:
http://medien.informatik.uni-ulm.de/~frank/research/simutools2008.pdf

APÊNDICE - RESULTADOS DO SIMULADOR

Redes estáticas:

para n=25 em grid 5x5 :

run:

Packet stats:

Rreq packets sent = 3498
Rreq packets recv = 7107
Rrep packets sent = 536
Rrep packets recv = 391
Rerr packets sent = 0
Rerr packets recv = 0
Hello packets sent = 92
Hello packets recv = 20
Total aodv packets sent = 4126
Total aodv packets recv = 7518
Non-hello packets sent = 4034
Non-hello packets recv = 7498

Overall stats:

Messages to deliver = 2499
Route requests = 103
Route replies = 249
Routes added = 101

freemem: 1513184
maxmem: 66650112
totalmem: 2031616
used: 519176

elapsed time: 10946

para n=49 em grid 7x7 :

run:

Packet stats:

Rreq packets sent = 13867
Rreq packets recv = 37739
Rrep packets sent = 2321
Rrep packets recv = 1435
Rerr packets sent = 46
Rerr packets recv = 6
Hello packets sent = 457
Hello packets recv = 63
Total aodv packets sent = 16691
Total aodv packets recv = 39243

Non-hello packets sent = 16234
Non-hello packets recv = 39180

Overall stats:

Messages to deliver = 4900
Route requests = 324
Route replies = 1242
Routes added = 323

freemem: 1921080
maxmem: 66650112
totalmem: 2621440
used: 701104

elapsed time: 48029

para n=100 em grid 10x10 :

run:

Packet stats:

Rreq packets sent = 62220
Rreq packets recv = 182062
Rrep packets sent = 6887
Rrep packets recv = 3554
Rerr packets sent = 186
Rerr packets recv = 13
Hello packets sent = 1045
Hello packets recv = 46
Total aodv packets sent = 70338
Total aodv packets recv = 185675
Non-hello packets sent = 69293
Non-hello packets recv = 185629

Overall stats:

Messages to deliver = 9999
Route requests = 645
Route replies = 4099
Routes added = 640

freemem: 3274912
maxmem: 66650112
totalmem: 4657152
used: 1382984

elapsed time: 224243

para n=196 em grid 14x14 :

run:

```
-------------
Packet stats:
-------------
Rreq packets sent = 330469
Rreq packets recv = 1089976
Rrep packets sent = 26909
Rrep packets recv = 9299
Rerr packets sent = 1338
Rerr packets recv = 204
Hello packets sent = 3646
Hello packets recv = 37
Total aodv packets sent = 362362
Total aodv packets recv = 1099516
Non-hello packets sent = 358716
Non-hello packets recv = 1099479
--------------
Overall stats:
--------------
Messages to deliver = 19600
Route requests      = 1487
Route replies       = 19421
Routes added        = 1458

freemem:  6689576
maxmem:   66650112
totalmem: 10223616
used:     3534784

elapsed time: 1664393
```

Redes móveis:

para n=25 em grid 5x5 :

```
run:
-------------
Packet stats:
-------------
Rreq packets sent = 5950
Rreq packets recv = 10368
Rrep packets sent = 740
Rrep packets recv = 581
Rerr packets sent = 40
Rerr packets recv = 24
Hello packets sent = 77
Hello packets recv = 11
Total aodv packets sent = 6807
Total aodv packets recv = 10984
Non-hello packets sent = 6730
Non-hello packets recv = 10973
--------------
Overall stats:
```

```
--------------
Messages to deliver = 2499
Route requests      = 245
Route replies       = 302
Routes added        = 160

freemem:  1511864
maxmem:   66650112
totalmem: 2031616
used:     520496

elapsed time: 12267
```

para n=49 em grid 7x7:

```
run:
-------------
Packet stats:
-------------
Rreq packets sent = 14638
Rreq packets recv = 39701
Rrep packets sent = 3050
Rrep packets recv = 1984
Rerr packets sent = 274
Rerr packets recv = 77
Hello packets sent = 279
Hello packets recv = 26
Total aodv packets sent = 18241
Total aodv packets recv = 41788
Non-hello packets sent = 17962
Non-hello packets recv = 41762

--------------
Overall stats:
--------------

Messages to deliver = 4900
Route requests      = 425
Route replies       = 1548
Routes added        = 425

freemem:  1959184
maxmem:   66650112
totalmem: 2629632
used:     671192

elapsed time: 50542
```

para n=100 em grid 10x10:

```
run:
-------------
Packet stats:
```

```
-------------
Rreq packets sent = 80822
Rreq packets recv = 237540
Rrep packets sent = 9720
Rrep packets recv = 4897
Rerr packets sent = 956
Rerr packets recv = 184
Hello packets sent = 11
Hello packets recv = 10
Total aodv packets sent = 91509
Total aodv packets recv = 242631
Non-hello packets sent = 91498
Non-hello packets recv = 242621

--------------
Overall stats:
--------------
Messages to deliver = 9999
Route requests      = 846
Route replies       = 5841
Routes added        = 834

freemem:  3324176
maxmem:   66650112
totalmem: 4677632
used:     1354200

elapsed time: 320211
```