Universidade Federal do Ceará

Centro de Ciências

Mestrado e Doutorado em Ciência da Computação (MDCC)

BiO4SeL: Uma Abordagem Baseada em Colônia de Formigas para a Otimização do Tempo de Vida de Redes de Sensores Sem Fio

Dissertação de Mestrado

Levi Bayde Ribeiro

Orientador: Prof. Dr. Miguel Franklin de Castro

Fortaleza

Agosto - 2009

Levi Bayde Ribeiro

BiO4SeL: Uma Abordagem Baseada em Colônia de Formigas para a Otimização do Tempo de Vida de Redes de Sensores Sem Fio

Dissertação de Mestrado submetida à Coordenação do Mestrado e Doutorado em Ciência da Computação (MDCC) da Universidade Federal do Ceará como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Miguel Franklin de Castro

Fortaleza
Agosto - 2009

BiO4SeL: Uma Abordagem Baseada em Colônia de Formigas para a Otimização do Tempo de Vida de Redes de Sensores Sem Fio

Dissertação de Mestrado submetida à Coordenação do Mestrado e Doutorado em Ciência da Computação (MDCC) da Universidade Federal do Ceará como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Aprovado em 19 de agosto de 2009

Banca Examinadora

Prof. Dr. Miguel Franklin de Castro (Orientador)
Universidade Federal do Ceará – UFC

Prof. Antônio Alfredo Ferreira Loureiro, PhD.
Universidade Federal de Minas Gerais – UFMG

Prof. José Neuman de Souza, PhD.
Universidade Federal do Ceará – UFC

Prof. Danielo Gonçalves Gomes, PhD.
Universidade Federal do Ceará – UFC

Prof. Fernando Antônio de Carvalho Gomes, PhD.
Universidade Federal do Ceará – UFC

Ao desenvolvimento científico.

Agradecimentos

Por receio de esquecer alguém, agradeço a todos os integrantes desta minha caminhada mundana. Afinal, como diria um grande ser, “dai a César o que é de César”.

Tudo havia mudado, menos o espírito humano.
Albert Einstein, em resposta à bomba atômica em
Hiroshima e Nagasaki.

Resumo

A miniaturização e imersão dos aparatos computacionais no ambiente é uma tendência dos tempos atuais. Um exemplo de tais aparelhos são os sensores, destinados a coletar dados de um ambiente e enviá-los a um centro de processamento. As chamadas Redes de Sensores Sem Fio têm sido utilizadas em diversos âmbitos, de monitoração de tráfego a prevenção de incêndios. Estes aparelhos, entretanto, têm algumas limitações inerentes, como baixo poder de processamento, pouca mobilidade e limitada bateria. A baixa capacidade energética torna-se mais crítica em algumas aplicações, como em ambientes hostis à sobrevivência humana. Nestas situações, os sensores devem otimizar seu gasto ao máximo, a fim de necessitarem o mínimo possível de trocas ou novos sensores. Além disso, devem atuar por si mesmos, sem a necessidade de intervenção humana direta. Tendo-se estas restrições em vista, este trabalho propõe o BiO4SeL (*Biologically-inspired Optimization for Sensor network Lifetime*, ou otimização de tempo de vida de Redes de Sensores biologicamente inspirada), um protocolo de roteamento autônomo utilizando Colônias de Formigas cujo objetivo é maximizar o tempo de vida de Redes de Sensores. As Colônias de Formigas são utilizadas como uma forma de alcançar esta autonomia. Para demonstrar sua eficiência, o protocolo é comparado com outros da área.

Palavras-chave: Redes de Sensores Sem Fio; Protocolo de Roteamento; Otimização de tempo de vida; Sistemas Biologicamente Inspirados; Côlonias de Formigas; Autonomia.

Abstract

The miniaturization and imersion of computacional devices in the environment is one trend of the current times. An example of such devices is the sensors, destined to collect data from an environment and send it to a processing center. The so called Sensor Networks have been used for a set of different tasks, from traffic monitoring to fire prevent. These devices, however, have some inerent limitations, as low power of processing, little mobility and limited battery. The low energy capacity becomes more critical in some applications, as in hostile environments to the survival human being. In these situations, the sensors must optimize it's expense to the maximum, in order to need the possible minimum of exchanges or new sensors. Moreover, they must operate by themselves, without direct humam intervention. Having these restrictions in sight, this work proposes the BiO4SeL (*Biologically-inspired Optimization for Sensor network Lifetime*), an autonomous Ant Colony routing protocol for optimization of Sensor Network lifetime. The Ant Colony are used as a method to reach this autonomy. To demonstrate it's efficiency, the protocol is compared with others of the area.

Palavras-chave: Wireless Sensor Networks; Routing Protocol; Lifetime Optimization ; Biologically Inspired Systems ; Ant Colony; Autonomy

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Algoritmos

Capítulo 1

Introdução

Este trabalho apresenta o BiO4SeL, *Biologically-inspired Optimization for Sensor network Lifetime*, ou Otimização de Tempo de Vida em Redes de Sensores Biologicamente Inspirada. A Seção 1.1 apresenta uma introdução a redes de sensores e a contextualização do problema a ser abordado. A Seção 1.2 traz a motivação para o desenvolvimento do trabalho. Na Seção 1.3, serão apresentados os motivos para o desenvolvimento desta solução. A Seção 1.4 traz a estrutura do restante da dissertação.

1.1 Contextualização

Com o avanço da tecnologia, as pessoas vivem, cada vez mais, imersas em computação. De relógios digitais a computadores de mesa, passando por geladeiras e até mesmo nas paredes, tudo tende a funcionar por meio de chips.

Assim, à medida que o aprimoramento tecnológico permite, os aparatos computacionais começam a tomar espaço no nosso dia-a-dia. Ninguém mais estranha hoje caminhar tranqüilamente escutando seu MP3 *Player*, ter um relógio digital ou escrever em seu PDA. Alguns destes avanços tenológicas estão tão imersos em nosso cotidiano a ponto de quase esquecermo-nos deles.

A esta imersão computacional deu-se o nome de computação pervasiva. Nela, tenta-se alcançar o sonho de conectar aparelhos computacionais diversos, entre si, na ordem de milhões de aparatos. Devem, obviamente, conectar-se através de uma rede, como a Internet.

A maioria destes dispositivos, se devem estar tão imersos em nosso cotidiano, devem passar a funcionar sem intervenção humana. Um relógio, por exemplo, não deve necessitar de constantes ajustes. Também não deve ser necessário preocupar-se com desligar um ar-condicionado caso a temperatura do ambiente já tenha atingido o desejado, pois ele tem

um termostato para isso. Indo mais além, pode-se imaginar uma rede de computadores de uma grande empresa, com 10.000 máquinas. Se, a cada pequeno problema de rede, o gerente fosse chamado, ele não daria conta do trabalho. Inclusive, um ambiente inteiro, como todas as suas particularidades, pode ser gerido por aparatos interligados [Qiao et al. 2006].

Mais especificamente, há um problema relacionado à integração de softwares [Kephart and Chess 2003]. Em outubro de 2001, a IBM lançou um manifesto alertando para a complexidade em configurar softwares para integrar tais aparelhos. Os softwares, em geral, não são planejados para integrar-se entre si. Assim, com o aumento de sua quantidade precisando trabalhar em conjunto, torna-se um problema para profissionais especializados em integração transformar um conjunto softwares dispersos em um sistema empresarial. No futuro, com o aumento irrefreável da complexidade de tal tarefa, nem mesmo o melhor e mais experiente profissional conseguirá integrar tão complexo sistema.

Assim, é possível notar a tendência inerente, assim como em todo o nosso cotidiano, da computação, a automação. Máquinas substituem pessoas desde a Revolução Industrial. Tal processo não se refreou nem tende a fazê-lo em um futuro próximo, já passado quase um século do início desta tendência. Com as redes, em especial dentro da computação, deve ocorrer o mesmo.

Tal processo, para a computação pervasiva, deve compor-se de unir os aparelhos computacionais sob uma única, grande e robusta rede, a Internet. O gerenciamento, claramente, não poderá ser realizado por pessoas, pois a complexidade é demasiadamente grande. Cada camada de aparelhos deverá gerenciar-se, facilitando o gerenciamento das camadas superiores sobre as inferiores. A este gerenciamento automatizado dá-se o nome de computação autonômica.

A computação autonômica, embora surgida da problemática da automatização da configuração de softwares e de sua integração, não se restringe somente a aplicações empresariais. Como supra-citado, ela deve abranger qualquer tipo de aparelho, e em uma escala mundial. Para interconectar todos eles, uma rede é necessária, uma mundial com suporte para os trilhões de aparelhos por vir. Assim, a Internet parece uma opção mais intuitiva.

Todos estes aparatos, ligados entre si, trabalhando entre si, devem poder comunicar-se. Afinal, a alma da computação autonômica é o gerenciamento dos aparelhos. Para alcançar tal gerenciamento de forma autonômica, ele foi divido em quatro partes. Tais partes formam a comunicação autonômica da rede [Dobson et al. 2006]. Elas são concisamente resumidas a auto-configuração, auto-otimização, auto-reparação e auto-proteção. Elas são chamadas, coletivamente, em inglês, de *self-* properties*.

Como frisado anteriormente, com a imersão computacional em meio à vida das pessoas, pode-se esperar pela comunicação entre os dispositivos. Se elas têm a possibilidade, por exemplo, de trocar músicas entre seus celulares, o farão. Precisarão, no entanto, de uma rede para suportar tais trocas. Esta rede, é claro, não pode ser cabeada, nem pode ser fixa, pois as pessoas estarão sempre em movimento. Tal rede deve ser flexível o suficiente para adaptar-se ao meio, e às disponibilidades de seus usuários. Tais redes foram chamadas de redes ad-hoc.

A expressão latina *ad hoc* significa, literalmente, ”para isto”. Assim, uma situação *ad hoc* é aquela sem nenhuma preparação prévia, e somente ocorre quando a situação é propícia. Assim, uma rede ad-hoc é formada, geralmente, por um conjunto de aparatos computacionais móveis, conectados sem utilizar meios cabeados. Tal rede somente ocorre se eles necessitarem comunicar-se, tendo início e fim dependentes desta troca de informações. Tais dispositivos podem ser computadores de mesa, *notebooks*, PDAs, celulares, ou qualquer outro capaz de processamento computacional e comunicação com outros dispositivos.

Um tipo especial de rede ad-hoc é a rede de sensores, utilizada para medir algum parâmetro (temperatura, movimento, tráfego) em um ambiente. Redes de sensores são compostas por muitos nós, geralmente interconectados sem utilizar fios. Sua principal função é captar informação do ambiente, repassando-a a uma estação base, ou *sink*, que fará algum processamento com estes dados.

Os sensores foram desenvolvidos com propósitos militares, mas hoje sua utilização dá-se em uma grande variedade de meios. Há muitos exemplos de aplicações comuns de rede de sensores [Akyildiz et al. 2002a]. Medição de temperatura, tráfego, umidade, pressão, níveis de ruído e sinais vitais de pacientes, além de regulação hidrográfica e sensoriamento de presença ou ausência de objetos. Estas são algumas aplicações mais comuns, revelando o propósito maior de utilização das redes de sensores. Como o próptio nome sugere, elas devem sensoriar ambientes, objetos ou indivíduos em busca de alguma informação.

1.2 Motivação

Um dos principais problemas em redes de sensores é mantê-los funcionando. Por sua baixa capacidade energética, em meio ao fluxo informacional, podem ficar sem energia. Assim, a área coberta por aquele sensor não terá mais cobertura, ou seja, não mais gerará dados, criando uma falha na rede. Manter uma rede de sensores com a maior área de cobertura durante um tempo de vida útil máximo é o principal objetivo dessas redes.

Diversas soluções para este problema, ou para parte dele, podem ser encontrados na literatura. Mais especificamente, a maior parte das soluções envolve somente aumentar o tempo de vida da rede. Este chamado tempo de vida é comumente tratado, na literatura da área, como o tempo até o primeiro nó morrer, ou seja, ter sua energia reduzida a zero.

Estas soluções são diversas. Há três grupos principais de estudos para aumentar este tempo de vida. O primeiro trata de melhoria da bateria [Royer and Toh 1999, Zhang and Huang 2006a, Ma and Yang 2006]. Assim, o foco é desenvolver e utilizar novas tecnologias para desenvolver baterias cada vez mais duráveis e com menor preço para construção. O segundo grupo trata de otimizar as funções de sensoriamento dos nós [Slama et al. 2006, Shen et al. 2005a, Haapola et al. 2005]. Eles objetivam gastar menos energia durante a busca e captação de informações do ambiente, além de regular o gasto energético nas camadas mais baixas da rede, enlace e redes. Assim, trata de diminuição de ruído, otimização de espaçamento dos nós, dentre outras técnicas. O terceiro grupo trata a otimização dentro do roteamento dos dados até a estação base. Como enviar os dados até a Estação Base é a principal atividade de um sensor, o roteamento é alvo certo da otimização do tempo de vida, pois acaba gastando muita energia [Islam and Hussain 2006].

Os nós precisam transmitir seus dados até a EB. Enviá-los diretamente a ela implicaria em um gasto muito elevado para os nós, além de diminuir drasticamente a escalabilidade da rede, pois todos os nós teriam de ter a EB em sua área de cobertura [Dobson et al. 2006]. Assim, tem-se como solução inicial protocolos de roteamento com multi-salto, ou seja, os nós vão repassando a informação recebida de seus vizinhos até atingir seu destino, a EB.

Dentre estes protocolos, há muitas soluções diferentes. A agregação [Akyildiz et al. 2002b], ou *clustering*, consiste em agrupar diversos nós em torno de um nó cabeça, para este receber os dados de todos, retirar as redundâncias e repassar à EB. Na *Spanning tree* [Zhang and Huang 2006a], forma-se uma árvore para enviar as informações, de filhos a pais, até a raiz, a EB. Algumas soluções utilizam estratégias de grafos, como conjuntos dominantes [Misra et al. 2007] ou Programação Linear [Ergen and Varaiya 2007].

Outros protocolos, além de tratarem o aumento do tempo de vida, melhoram o tempo de conectividade da rede. Isto significa extender ao máximo o tempo de cobertura do ambiente pela rede, sem pontos fora de cobertura pela morte de algum sensor. Alguns baseiam-se no mapa energético da rede [Al-Karaki and Al-Mashaqbeh 2007, Huang and Jan 2004] para rotear dados por onde há mais energia, balanceando [Yang and Ho 2006, Qi and Zhao 2007, Liu and Hong 2006] a carga da rede. Pode-se utilizar técnicas de multi-agentes [Biswas et al. 2006, Biswas 2005], trechos de código realizando operações nos nós ou sendo trocados entre eles como mensagens. Algumas técnicas de Inteligência

Artificial, como Algoritmo Genético [Islam and Hussain 2006] ou Inteligência de Enxame [Selvakennedy et al. 2007] podem ser utilizados. As Colônias de Formigas [Hussein and Saadawi 2003, Hussein et al. 2005, Shuang et al. 2007] são uma especialização da Inteligência de Enxame, muito utilizadas no contexto de redes ad-hoc e de sensores.

As técnicas de Inteligência Artificial, aplicadas à descoberta de rotas, geram resultados melhores se comparados aos obtidos por outras técnicas. Há várias razões para isto, como otimização do caminho mínimo e da distribuição de rotas, menor sobrecarga, ausência de infra-estrutura, dentre outras. As técnicas de agentes têm algumas destas vantagens em comum, mas normalmente é necessário um *framework* para seu funcionamento. Assim, é necessário nós com agentes embarcados, códigos para fazê-los funcionar, tipos de dados específicos a serem trocados, implicando em uma maior sobrecarga na rede. Assim, estas abordagens são um dos expoentes do desenvolvimento do roteamento para redes de sensores.

As técnicas de Inteligência de Enxames tendem a agregar as melhores características dessas duas abordagens. Por ser uma estratégia derivada da IA, agrega as características de otimização e pouca sobrecarga da rede. Em [Iyengar et al. 2007], por exemplo, o autor compara otimização com formigas a com algoritmo genético, reforçando a superioridade daquele sobre este. Além disso, por ser um tipo de agente, privilegia o processamento à troca de informações entre os sensores, gerando uma menor sobrecarga. Além disso, dispensa a necessidade de um *framework*. Os sensores, por terem baixa capacidade de processamento e memória, não poderiam suportar um *framework* complexo.

As técnicas de Inteligência de Enxames tendem a agregar as melhores características dessas duas abordagens. Tais técnicas fazem parte de um grupo maior, denominado Sistemas Biologicamente Inpirados (SBI) [Dobson et al. 2006]. Eles trazem soluções para os mais diversos campos, da robótica à algoritmos de roteamento em redes, baseadas na Natureza. Assim, a Inteligência de Enxames baseia-se no conceito de todos os grupos de pequenos insetos da Natureza, como abelhas e formigas. Tais criaturas têm, individualmente, uma inteligência rudimentar. Apesar disso, em conjunto, apresentam uma inteligência bem superior à individual. Baseado nesta idéia, diversos algoritmos para resolução dos mais variados problemas de IA foram desenvolvidos.

Os Sistemas Biologicamente Inspirados, neles incluía da Inteligência de Enxames, estão intimamente relacionados à Computação Autonôma [Balasubramaniam et al. 2006]. Tais sistemas devem, além de auto-gerenciamento, efetuar aprendizado e raciocínio sobre as informações adquiridas. Assim, eles devem evoluir, a fim de tornarem-se mais robustos. Claramente, qualquer sistema biológico natural, como nosso sistema imunológico ou uma colônia de formigas, por exemplo, utiliza esta estratégia.

O principal representante do grupo de Inteligência de Enxame para o roteamento em redes de sensores é a Colônia de Formigas [Dobson et al. 2006].

Uma colônia de formigas, na Natureza, funciona como descrito para a Inteligência de Enxame. Cada formiga tem uma inteligência bem rudimentar, voltada à captação de comida, defesa do formigueiro ou reprodução. Apesar disto, uma formiga só não conseguiria captar comida, por exemplo, eficientemente. Assim, elas deixam fermônios pelos caminhos percorridos, sinalizando para as outras formigas uma rota para encontrar alimento para toda a colônia. Assim, não há comunicação direta entre cada ser, mas elas conseguem inter-comunicar-se pelos feromônios. Com eles, elas conseguem escolher e percorrer os melhores caminhos para sua tarefa.

Transportando o modelo para a computação, pode-se aplicá-lo ao roteamento em redes de sensores. Utiliza-se não somente uma heurística, como o caminho mais curto, mas também um fator (feromônico) de aprimoramento para o cálculo das rotas. Além disto, por suas próprias características, o algoritmo cria diversas rotas, com quantidades feromônicas diferenciadas, aumentando a confiabilidade e distribuição de carga pela rede. Estas características destacam os algoritmos deste grupo além dos de IA e de agentes.

Nesta área, entretanto, há muitos trabalhos desenvolvidos. Assim, para conseguir melhorar o cenário atual, deve-se conhecer o estado da arte. Pode-se utilizar as formigas para calcular os caminhos ótimos da rede [Liu et al. 2007], colocar sensores para dormir [Rui et al. 2006] ou calcular caminhos ótimos, estaticamente [Huang et al. 2006] ou dinamicamente [GhasemAghaei et al. 2007, Hussein and Saadawi 2003, Hussein et al. 2005, Shuang et al. 2007], dentre outros.

Para facilitar o entendimento, suponha-se o seguinte cenário. Uma região é atingida por algum desastre, natural ou não. Tal região torna-se inóspita para qualquer grupo humano, seja de busca por sobreviventes ou para estudo das causas e conseqüências do ocorrido. Assim, algum método independente de controle direto deve ser utilizado na área a fim de conseguir tais informações.

Uma idéia facilmente proponível seria enviar um robô. Tal estratégia poderia solucionar o problema, mas o ambiente poderia ter-se tornado intrasponível ou inospito até mesmo para ele. Assim, alguma outra solução poderia ser levantada.

Uma solução possível é, com a utilização de um avião, “aspergir” sensores pela área afetada. Boa parte deles pode ser destruída, tanto pelo ambiente quanto por dano na queda. Isto não é um problema, pois eles são de baixo custo, e a quantidade não é fator de preocupação. Depois de estabelecidos no solo, os sensores deveriam estabelecer, por si próprios, a rede e os métodos de comunicação, sem interferência alguma dos seres humanos. A rede deve, por si própria, lidar com sensores defeituosos ou destruídos por

condições inóspitas, assim como otimizar o seu consumo de energia, pois tal região não permite constantes adições de sensores à rede ou recarga das baterias.

Em uma outra situação, temos um ambiente de trabalho. Sua ocupação [Qiao et al. 2006] e condições climáticas [Ota et al. 2006, Walla et al. 2007, Yong et al. 2007], além de outros fatores, devem ser monitorados. Em um método clássico, isto seria feito manualmente, por pessoas. Uma idéia para solucionar o problema seria com redes de sensores para fazerem um monitoramento autonômico. Tais redes decidiriam, autonomicamente, sobre economia energética do ambiente, controle das condições e soluções para possíveis problemas.

Se a rede não for capaz de trabalhar sem interferência humana, tais situações, assim como outras, teriam resoluções limitadas ou de alto custo. Assim, é notável ser a autonomicidade um claro passo a frente para as redes de sensores.

1.3 Objetivos

Face a esse cenário, este documento propõe uma pesquisa acerca da utilização de SBI, mais especificamente Colônia de Formigas, para a otimização do tempo de vida em redes de sensores. Esta proposta é o algoritmo de roteamento para redes de sensores BiO4SeL, *Bioligically-inspired Optimization for Sensor network Lifetime*, Otimização de Tempo de Vida em Redes de Sensores utilizando Colônias de Formigas. Como dito, há muitos trabalhos na área. Apesar disto, eles apresentam alguns pontos a serem melhorados, objetivos deste trabalho.

Primeiramente, há a visão autônoma do roteamento. O exemplo citado traz a necessidade da auto-organização, uma das características da autonomia. Os nós devem ser capazes de criar sua própria rede, conectar-se e criar as rotas para enviar seus dados até a(s) estação(ões) base. Enquanto estiver funcionando, a rede deve sempre otimizar suas rotas (auto-otimização). Assim, evita-se gasto energético desnecessário, e uma possível e difícil reposição ou recarga do sensor. Além disso, eles devem poder decidir, para qual estação base enviar seus dados, utilizando-se de alguma métrica. Como estarão longe de qualquer interação humana direta, eles devem ser capazes de contornar problemas, como nós destruídos ou sem energia, redirecionando suas rotas por outros nós (auto-restauração). Todas estas características serão abordadas neste trabalho. Elas não foram abordadas completamente em nenhum dos trabalhos anteriores

O algoritmo pretende otimizar o tempo de vida da rede. Para alcançar tal objetivo, ele irá utilizar-se das características inerentes ao paradigma utilizado, o de Colônia de Formigas. Utilizando-se a deposição feromonial, cria-se diversos caminhos “ótimos” na rede.

Ao enviar os dados, será possível fazê-lo por estes caminhos, alternadamente. Assim, espera-se gastar a energia dos nós do modo mais igualitário possível, evitando a morte prematura de nós na rede.

As rotas deverão, então, ser dinamicamente modificadas, para atender aos objetivos esperados. Esta modificação levará a um balanceamento da carga pela rede, fazendo os dados serem trafegados de modo mais distribuído. Assim, espera-se extender o tempo de vida da rede, além de conseguir uma maior robustez para a rede como um todo.

1.4 Estrutura da dissertação

O restante deste documento está dividido como segue. No Capítulo 2, são apresentadas, em maior detalhe, as redes de sensores, assim como suas utilizações e problemas. No Capítulo 3, a comunicação autonômica é descrita, com seu histórico, funcionalidades e utilizações. O Capítulo 4 contém explicações sobre sistemas biologicamente inspirados. Mais especificamente das formigas, suas características e utilizações. No Capítulo 5 está a apresentação da proposta deste trabalho, uma solução para a economia energética em redes de sensores. O Capítulo 6 traz os resultados obtidos com o protocolo, em formas de gráficos e explicações textuais. O Capítulo 7 traz a conclusão do trabalho e os trabalhos futuros. O Apêndice A traz alguns dos resultados omitidos no Capítulo 6, por serem muito numerosos. Por último, o Apêndice B traz uma complementação das comparações efetuadas no Capítulo 5, por serem demasiado numerosas.

Capítulo 2

Redes de Sensores

Uma rede de sensores é um tipo especial de rede *ad hoc*. Elas, no entanto, não são iguais. As Redes de Sensores têm diversas particularidades. Elas têm uma menor capacidade de armazenagem de dados e energética, além de limitado poder computacional, se comparadas às *ad hoc*. Além disto, são utilizadas para sensoriar parâmetros do ambiente e enviar a uma Estação Base para processamento. As redes *ad hoc* têm outras utilizações, como troca de dados, por exemplo. A infra estrutura das Redes de Sensores costuma ser menos mutável, se comparada a das redes *ad hoc*.

Como citado em [Zhang and Huang 2006a], redes de sensores em larga escala detêm as seguintes propriedades:

- Cada nó é, além de um sensor, também um roteador.
- A rede é muito dinâmica, podendo sofrer modificações durante sua operação.
- A abrangência do sinal de cada sensor é muito pequena, resultando em um ambiente densamente povoado.
- Os enlaces, normalmente, são de sentido único.
- Cada sensor tem limitado poder computacional e pouca memória, impossibilitando operações muito complexas.
- Cada nó tem quantidade energética bem limitada.

O funcionamento básico de uma rede de sensores é o seguinte. Cada sensor capta, constantemente ou em espaços regulares, informações do ambiente. Tais informações devem ser repassadas a uma Estação Base, um nó com maior poder computacional, com o intuito de receberem processamento. Até chegarem à Estação Base, as informações

devem ser repassadas pela rede de sensores, de sensor a sensor, pois o alcance de rede de cada nó é baixo, nem todos alcançando a Estação Base. Ou seja, cada sensor capta informação e a envia para a Estação Base, além de reencaminhar informação de outros sensores, durante toda a execução da rede. Este é o roteamento de informação de uma rede de sensores.

As principais diferenças entre as redes de sensores e as redes *ad hoc* comuns encontram-se nas capacidades dos nós da rede. Na rede de sensores, eles têm menor poder de processamento e capacidade energética, com maior povoamento, além de uma menor necessidade de locomoção. Pode-se classificar, claramente, as duas primeiras particularidades como sendo os principais gargalos destes sistemas.

Há diversos tipos de sensores [Akyildiz et al. 2002a]. Cada sensor tem um método de captar o ambiente, seja por radar, visual, infravermelho, acústico, térmico, sísmico, dentre outros. Cada tipo de sensor é apropriado para uma tarefa específica. O térmico, por exemplo, é mais apropriado para medir temperatura de uma caldeira. Já o infravermelho é mais adequado ao uso em campo de batalha, no qual sua leitura do calor corporal pode ser útil para localização de inimigos.

O restante do capítulo será dividido da seguinte forma. A Seção 2.1 traz uma explanação sobre as utilizações de Redes de Sensores. A Seção 2.2 mostra como o roteamento dos dados é feito nestas redes. Na Seção 2.3, há a discussão principal do capítulo, sobre as formas de economizar energia no funcionamento destas redes. A Seção 2.4 finaliza o capítulo mostrando o possível futuro da área.

2.1 Utilizações de Redes de Sensores

Com uma baixa capacidade computacional, uma rede de sensores não pode exercer funções muito complexas. Como exemplos de algumas aplicações comuns [Akyildiz et al. 2002a], temos, dentre outras, medições de temperatura, tráfego, umidade, pressão e níveis de ruído, além de regulação hidrográfica e da presença ou ausência de objetos. Tais aplicações podem ser completas por si mesmas, como a medição da temperatura de um reator nuclear, ou a da umidade de uma cidade, ou podem desempenhar funções para outras aplicações, como a medição do tráfego de uma rua gerando dados para um programa de medição e remanejamento, ou a presença de inimigos em um campo de batalha, para acionar o sistema de defesa automático.

Podemos classificar alguns grandes ramos de utilização de redes de sensores [Akyildiz et al. 2002a, Estrin et al. 1999, Dobson et al. 2006]. Militar, saúde, ambiental, caseiro, exploração espacial, recuperação de desastres, processamento químico, dentre outros.

Quanto ao uso militar [Sun et al. 2006], as redes de sensores encaixam-se bem por sua confiabilidade e capacidade de recuperação. Em um campo de batalha, se algum dos nós da rede for destruído, ela, como um todo, continuará funcionando. Várias aplicações podem ser utilizadas em campos de batalhas. Os sensores podem ser utilizados para contabilização de munição e recursos aliados, para avisar sobre a aproximação de inimigos, para a descoberta e avaliação de armas biológicas, substituindo uma equipe humana, dentre outras aplicações. Infelizmente para a raça humana, esta parece ser uma proveitosa e crescente aplicação para as redes de sensores.

Em um contraste extremo, outra área de aplicação é a saúde. Dentro de um hospital, cada paciente pode carregar, em sua roupa ou corpo, sensores de peso irrisório a fim de manter um histórico de informações importantes, como batimento cardíaco, funcionamento renal, respiração, dentre outros. Além disso, poder-se-ia minimizar o erro na aplicação de medicamentos, pois estes sensores poderiam guardar as informações alérgicas do paciente. O enfermeiro, ao aproximar-se do paciente trazendo o remédio, também com um sensor, eles se comunicariam e veriam uma possível incompatibilidade. Além disso, como um intento da computação pervasiva, teríamos a casa médica inteligente. Por meio de sensores para captar certas métricas, como um sensor de exame de urina no sanitário, ou medição de batimentos cardíacos na cama ou roupa de dormir, o médico poderia monitorar a saúde do paciente à distância, assim como o sistema poderia prever e apontar certas doenças por si mesmo.

As redes de sensores também são utilizadas para medir certos parâmetros ambientais. Podem ser usadas em florestas, a fim de medir as condições climáticas e de biodiversidade do ambiente com o objetivo de evitar incêndios, desmatamentos, erosão e morte dos animais, além de auxiliar no estudo das áreas e das espécies viventes. Outra aplicação é evitar desastres ambientais como inundações, terremotos, maremotos e outros. Além disso, os sensores podem ser usados na agricultura, a fim de controlar o nível de agrotóxicos, as pragas ou o crescimento da plantação.

O aspecto caseiro das redes de sensores é bem próximo da computação pervasiva. Objetiva-se interligar todos os aparelhos domésticos a fim de eles trabalharem em conjunto, trocando informações. Também é possível colocar sensores em cada compartimento da casa, repassando informações dos acontecimentos dos compartimentos para um processador central. Se colocarmos tais informações em conjunto com informações dos aparelhos, e todas estas comandadas por algum tipo de inteligência, central ou distribuída, teremos o conceito de casa inteligente da computação ubíqua em ação.

Em recuperação de desastres, o exemplo mais claro e recente seria o da queda do *World Trade Center*. Isto também se estende a qualquer ataque terrorista, assim como

a desastres naturais, como terremotos. Este tipo de recuperação consiste em conseguir localizar o sinal emitido por sensores carregados pelas pessoas presas ou perdidas durante o desastre. Tais sinais podem ser de uma *smart badge* [Zussman 2003], um sensor constantemente emitindo informações sobre a localização e o estado de saúde da pessoa. É importante que estes sensores trabalhem como uma rede, pois eles têm pouco alcance, e os equipamentos de recebimento dos sinais, normalmente, ficam na borda de onde o acidente ocorreu. Assim, se os sensores não formarem uma rede, provavelmente alguns não terão seus sinais recebidos. Na falta de um sensor especializado, algum aparelho capaz de emitir sinais, como um *pager*, *bip* ou celular pode ser utilizado, ajudando na localização da pessoa. Uma outra utilização poderia ser a seguinte. Tendo disposto sensores na área na qual um acidente venha a ocorrer, eles podem avisar ao socorro no mesmo instante do ocorrido, como por sensores de fumaça avisando aos bombeiros. Pode, ainda, ajudar a encontrar uma saída livre do prédio, sentindo quais caminhos ainda não foram afetados pelo incidente, ou uma entrada para o grupo de socorro.

Redes de sensores também podem ser usadas no meio comercial. Elas podem ser usadas para rastreamento de carros e objetos, assim como contra roubos. Caso o carro saia do alcance da rede na qual o dono o deixou, este será avisado pela Internet, ou então poderia haver uma trava de segurança que desligasse o carro caso deixasse a área. Sensores podem ser utilizados, também, para gerenciar itens em um depósito, muito útil em grandes áreas, tornando bem simples a localização dos itens. Da mesma forma, pode ser usado em museus, para dar explicações às crianças sobre um objeto ao chegarem próximas a ele. Em empresas, uma utilização possvel seria a redução do gasto com ar-condicionado. A maioria das empresas utiliza uma central, e o ar é distribuído pelas salas. Cada sala, contendo somente um termostato, pode ficar com uma parte quente e outra fria, acarretando maior consumo de ar para igualar a temperatura da sala. Se houvessem vários sensores em cada sala, a economia, segundo, seria, nos EUA, da ordem de 55 bilhões por ano, em 2000.

Há algumas outras aplicações menos diretas e estudadas. Um exemplo é o controle de distância entre satélites, combinando redes de sensores e multi-agentes. Elas também podem ser utilizadas em alguns ambientes mais incomuns, como em sítios arqueológicos, ou em alguns mais hostis, como o glacial.

Todas estas utilizações têm algo em comum. Elas necessitam, para sua utilização, captar dados do ambiente e repassá-los até a Estação Base. Esse encaminhamento é chamado de roteamento.

2.2 Roteamento

No que concerne roteamento, a gama de soluções é bastante vasta. Elas podem ir desde soluções para as tabelas de roteamento [Royer and Toh 1999] até métodos e algoritmos bastante intrincados, por exemplo, utilizando aprendizado [Zhang and Huang 2006a]. Alguma dessas técnicas serão descritas a seguir. Tais técnicas são descritas em diversos artigos. Maior parte delas pode ser encontrada em [Zhang and Huang 2006a], com suas respectivas citações.

Há três elementos básicos em todo protocolo de roteamento [Zhang and Huang 2006a] : especificação de destino, objetivos de roteamento e estratégia de roteamento. A especificação do destino trata de como os nós da rede são especificados um para o outro, seja com uma tabela de vizinhança, uma árvore, etc. Objetivos de roteamento são os parâmetros de escolha da rota. A estratégia de roteamento é como a informação chegará de seu remetente até seu destino.

Uma abordagem para a classificação dos protocolos pode ser dividi-los em dois grandes grupos, os estruturados e os desestruturados. Os estruturados usam uma estrutura em sua base para decidir qual o próximo passo. Os desestruturados fazem essa decisão a cada nó. A seguir, cada divisão será tratada separadamente, começando pelos protocolos estruturados.

A despeito da especificação do destino, há uma ampla gama de soluções e suas utilizações. A mais básica e mais utilizada é a tabela de vizinhanças, como o DSDV (*Destination-Sequenced Distance-Vector Routing*) e o WRP (*Wireless Routing Protocol*) [Royer and Toh 1999]. Cada nó guarda, em sua tabela, seus vizinhos mais próximos, ou seja, aqueles nós alcançados por sua área de cobertura. Cada nó tem um identificador, guardado nas tabelas de seus vizinhos. Outra estratégia são as *spanning trees*. Transformada a rede em um grafo, mais especificamente uma árvore, cujos nós são os sensores e as arestas ligam os vizinhos, a árvore formada designa os caminhos pelos quais as informações devem trafegar de um nó a outro. Algumas estratégias endereçam não somente um, mas vários caminhos possíveis [Ganesan et al. 2001, Tsirigos and Haas 2001, De et al. 2003, Nasipuri and Das 1999], a fim de aumentar a robustez do roteamento. Alguns destes até implementam redundância [Hong et al. 2002, Dulman et al. 2003], ou seja, eles enviam as informações por vários caminhos ao mesmo tempo. Uma solução utilizando inteligência de enxame é o JARA (*Jumping Ant Routing Algorithm*) [Chen et al. 2007b]. Ela permite um modo híbrido entre o reativo e o proativo de manutenção da estrutura de especificação de destino. O método reativo somente modifica a estrutura ao ocorrer alguma modificação externa notificada, enquanto o proativo atualiza, periodicamente, a estrutura.

Objetivos de roteamento geralmente vêm agregados à estratégia de roteamento. Entre os diversos métodos figuram caminho mínimo [Perkins and Royer 1999, Park and Corson 1997], grau de estabilidade de associação [Toh 1996], ou seja, o quão estáveis são as ligações entre o nó e seus vizinhos, dentre outros. Algumas outras estratégias também são utilizadas, como roteamento por meio de um *grid*, ou seja, uma grade dos nós, algo semelhante ao agrupamento, além de por meio de conectividade ou de um *backbone*. Também podem haver dois ou mais objetivos em conjunto, como estabilidade e força do sinal emitido pelo sensor, além de caminho mínimo.

Estratégias desestruturadas, em redes em constante mudança, precisam de constantes reparos por saída ou entrada de novos nós, modificações em suas características, dentre outros. Este sobrepeso pode ser grande demais em uma rede, tornando a utilização de estratégias desestruturadas mais adaptada a tais ambientes.

A respeito das estratégias desestruturadas, tratemos, primeiramente, da especificação do destino. Figura, nesta classe, localização baseada em coordenadas geográficas, ou seja, em posição geográfica real dos sensores.

Quanto aos objetivos de roteamento, temos muitas soluções. Algumas são baseadas em modelos gulosos, sempre escolhendo o melhor em cada passo. Estratégias baseadas em localização geográfica [Zhang and Huang 2006a] ou em informação adquirida nos nós são um bom exemplo de modelos gulosos. Outras exemplos de estratégias são as baseadas em busca, nível energético do sensor, roteamento Q, formigas [Subramanian et al. 1997a], NADV [Royer and Toh 1999], dentre outras. Algumas, ainda, são baseadas em inundação, ou seja, enviar as informações a todos os vizinhos, não só a um caminho.

Como pode-se ver, há muitas formas de rotear os dados pela rede. Toda a utilização da rede utiliza energia, e este é um fator importante. Por sua troca ou recarga ser inviável, a bateria dos nós deve durar o máximo de tempo possível. ASsim, começaram a desenvolver-se diversas soluções, inclusive no roteamento, para economizar a energia dos nós.

2.3 Soluções para capacidade energética

Vários assuntos sobre redes de sensores são estudados e pesquisados. A segurança é sempre um assunto importante em qualquer rede sem fio, dada sua vulnerabilidade a ataques inerentes. Um assunto, no entanto, domina a quantitade de soluções para rede de sensores. Tal problema é a capacidade energética. Por serem sensores simples, suas baterias não têm muita carga, e a troca destas baterias é, na maioria das vezes, inviável. Alguns ambientes são muito hostis para a troca. Em outros os sensores localizam-se fora do alcance, como dentro de construções. Assim, a bateria deve ser economizada ao

máximo a fim de permitir a menor taxa de mudança na rede, além de minimizar o tempo de ausência de cobertura em qualquer parte da rede.

A capacidade energética da rede de sensores é um assunto abordado pelos pesquisadores. A energia de um sensor é gasta em diversos momentos. O fato de ele estar ligado gasta energia, assim como captar, receber, processar e enviar informação. Cada um destes pontos já foram estudados separadamente, ou em conjunto [Slama et al. 2006], por diversas soluções existentes na literatura. O mais abordado, certamente, é o do envio de informações.

Na tentativa de solucionar o problema de energia da rede de sensores, os pesquisadores começaram a desenvolver estratégias para aumentar o tempo de vida de um sensor, objetivando manter a rede no ar por mais tempo. Assim, diversas soluções começaram a ser desenvolvidas, podendo ser divididas em três grandes áreas. A primeira, e mais basal, é aumentar a economia da bateria. Desenvolver novas e melhores baterias, assim como aprimorar seu modelo computacional deve ser uma área em constante desenvolvimento a fim de proporcionar maior capacidade energética aos sensores. Em seguida, temos as funções de sensoriamento, ou seja, captura, recebimento e processamento das informações, assim como a parte física do sensor relacionada a estas funções. A terceira área, sempre mais abordada [Ergen and Varaiya 2007], é constituída por técnicas de roteamento das informações, como e para quem são enviadas dos sensores até a Estação Base ou vice-versa.

2.3.1 Economia da bateria

Por volta do ano de 2000, alguns trabalhos, como [Royer and Toh 1999], mostraram a defasagem do crescimento tecnológico das baterias dos sensores em relação à continuamente crescente demanda das aplicações [Zhang and Huang 2006a]. Tais trabalhos melhoraram o entendimento sobre as baterias, revelando defasados os modelos até a época. As baterias, foi mostrado, não gastam toda a energia provida ao sensor. Por suas próprias características, até 30% da carga da bateria seria restituída pela própria bateria, se comparado aos modelos anteriores [Ma and Yang 2006]. Não só este, mas vários modelos foram desenvolvidos levando-se em consideração estas descobertas.

É notável a necessidade deste tipo de avanço. Tal aprimoramento, no entanto, somente aumenta o tempo de vida dos sensores individualmente. Se a energia continuar sendo subutilizada, a rede continuará com seu tempo de vida, ou seja, o tempo até o primeiro nó ficar sem bateria, muito baixo. Tal constatação será melhor compreendida posteriormente, quando vier a seção de roteamento.

Depois de muita pesquisa na área energética da rede de sensores, os cientistas começaram a notar a falha de suas soluções. Aumentar o tempo de vida de cada sensor individual da rede aumentava o tempo de vida da rede, ou seja, o tempo até o primeiro nó acabar sua bateria, mas não o maximizava [Slama et al. 2006]. Fazia-se necessário soluções a fim de otimizar o tempo de vida de todos os nós da rede, de forma a aumentar o tempo de vida da rede como um todo.

As soluções para este problema podem residir em vários aspectos da rede. Em geral, acumulam-se no roteamento dos pacotes pela rede, mas elas podem estar nas camadas mais baixas da rede, seja transmissão, recepção ou processamento dos pacotes.

Assim, a partir deste ponto, as soluções descritas poderão ser classificadas por dois métodos, otimização de rotas ou maximização do tempo de vida da rede. A otimização de rotas não leva em consideração o tempo de vida da rede. No máximo, ela tenta gastar o mínimo de energia possível, dentro de sua estratégia de roteamento. Quando o termo ”tempo de vida da rede”for citado, estaremos tratando de sua otimização. Este termo significa o tempo até o primeiro nó da rede ficar sem energia, ou seja, o tempo até a rede tornar-se incompleta.

O modelo de bateria clássico utilizado nos algoritmos é calculado antes do seu início, pois é complexo demais para ser calculado em tempo de execução. Como uma recente alternativa a este modelo, foi proposto um utilizando um modelo de restituição de bateria [Ma and Yang 2006]. Tal modelo utiliza a restituição energética, como explicado acima, e, segundo o autor, é um modelo simples e realístico. Simples, pois pode ser calculado em tempo de execução, melhorando sua utilização em cenários mutáveis, pois, nos clássicos, os parâmetros da bateria devem ser pré-calculados e não mais poderiam ser modificados. Realista, pois os anteriores utilizavam tempo contínuo, quando a rede pode utilizar um tráfego em rajadas. Assim, segundo os autores, tal modelo torna-se mais adequado a redes de sensores, aumentando seu tempo de vida.

Uma outra técnica de economia de bateria é a hibernação de sensores. Ela consiste em, sob certas condições determinadas pelo algoritmo, desligar ou ligar sensores, sem atrapalhar a conectividade ou o roteamento da rede. Tal técnica é utilizada para economizar a bateria dos sensores desligando-os. Em [Tynan et al. 2007], acrescenta-se a autonomicidade à solução. Em escalas elevadas, a autonomia de roteamento da rede é indispensável. Por complexidade, tempo ou ambiente perigoso, é aconselhável a rede rotear-se, excluindo o trabalho e preocupação humanos. Nesta solução, os nós recalculam constantemente suas interpolações. Os nós dormentes, após passado seu tempo de hibernação, acordam por si mesmos e recalculam a interpolação, voltando a dormir ou permanecendo acordados conforme a resposta.

2.3.2 Funções dos sensores

As principais funções de um sensor são a captação, processamento, recebimento e envio de informações. Elas gastam a maior parte de sua bateria. Podem ser divididas em dois grupos. As funções de sensoriamento são relacionadas à captação de algum parâmetro do meio e seu processamento em dados. As de roteamento, à recepção, envio e repasse de dados provenientes do mesmo sensor ou de seus vizinhos, no caminho até a Estação Base.

Para capturar as informações do meio, os sensores devem estar funcionando e captando informações. Esta função é a principal e mais recorrente de uma rede de sensores. Cada sensor tem uma pequena área de abrangência, por suas próprias limitações, forçando a rede a ser bem povoada. Estas áreas de abrangência, assim como as de telecomunicações, tendem a ter uma representação próxima à circular, frequentemente cruzando-se. Assim, podemos intuir a necessidade de otimizar a localização dos sensores, a fim de gastar o mínimo possível com a manutenção da área de cobertura, além de deixá-los o mais próximo possível com o intuito de gastar menos nas transmissões de informação [Shen et al. 2005a]. Devemos ter o cuidado, no entanto, para, com esta disposição mínima, não inviabilizar um melhor algoritmo de roteamento, ou isolar um nó da rede se seu único vizinho acabar sua bateria. Assim, deve ser possível modificar a abrangência dos sensores em tempo real, tornando a rede mais dinâmica.

Outras soluções objetivam modificar o modelo de consumo energético nas camadas mais baixas da rede, física, enlace e de redes. Em [Haapola et al. 2005] está um destes métodos. Neste artigo, é criado um modelo energético diferenciado, objetivando a conservação energética da rede. Outro é [Madan et al. 2006]. Ele objetiva otimizar a rede em todas as suas camadas. No sensoriamento, utiliza uma estratégia de regulação dos ruídos de transmissão nos nós, minimizando o seu efeito. Assim, se for possível, dois nós podem transmitir ao mesmo tempo sem interferir um com o outro. Quanto ao roteamento, utiliza estratégia de multi-saltos.

Algumas soluções, entretanto, tratam de inundação. Em [Flammini et al. 2006], tenta-se minimizar a quantitade de energia gasta com o rádio. Para isso, ele almeja minimizar a área de propagação da informação em cada sensor durante uma inundação. Fazendo-se isto, consegue-se uma economia energética na rede como um todo.

2.3.3 Roteamento

As mais abundantes soluções para a escassez energética em uma rede de sensores localizam-se no roteamento. Depois da coleta de informações do ambiente, elas devem

ser enviadas à Estação Base. Assim, o roteamento é uma função de suma importância e extensamente utilizada, sendo um alvo óbvio para a economia energética [Islam and Hussain 2006].

Apesar de alguns autores assim pensarem, não somente o envio de informações causa gasto de bateria [Slama et al. 2006]. O recebimento da informação também deve ser levado em consideração no gasto energético, assim como a captação de informação do ambiente e seu subsequente processamento. O processamento também ocorre em vários métodos de roteamento, no quais as informações enviadas contém mensagens para o nó.

As premissas básicas do roteamento já foram explicitadas em uma sub seção anterior. De início, os primeiros algoritmos de roteamento somente preocupavam-se em rotear os pacotes. Ao emergir a preocupação com a capacidade energética dos sensores, eles passaram a tentar otimizar esta capacidade. A partir deste ponto do texto, somente serão expostas técnicas com esta otimização como objetivo principal.

Algumas estratégias, no entanto, utilizam parte estruturada e parte desestruturada. Na Difusão Direcionada [Intanagonwiwat et al. 2003], cada nó tem certos atributos, ou seja, as informações por ele produzidas. Baseada nesses atributos, a Estação Base envia por *broadcast* o interesse aos nós, ou seja, qual informação ela deseja. Eles o salvam e o utilizam os gradientes do interesse para decidir o caminho a ser tomado. Já na família SPIN (*Sensor Protocols for Information via Negotiation*) [Akyildiz et al. 2002b], quando um sensor tem dados para enviar, ele envia antes uma mensagem aos seus vizinhos avisando qual tipo de dados ele possui. Então, ele envia os dados somente aos interessados.

Agregação e Derivados

As estratégias supra-descritas, no entanto, utilizam muitas mensagens, aumentando, desnecessariamente, a quantidade energética utilizada no roteamento. Com isto em mente, o agrupamento [Akyildiz et al. 2002b], ou *clusterização*, foi desenvolvido. Nesta estratégia, a rede é dividida em grupos, cada um com um cabeça de grupo, ou *cluster head*. Ao transmitir alguma informação, os nós a enviam ao cabeça. Este, por meio de agregação de dados, reduz a quantidade de dados a ser enviada, economizando energia e largura de banda.

Esta estratégia, no entanto, provou ter respondido apenas parcialmente ao problema energético. Nota-se, claramente, a diminuição acentuada da capacidade energética do nó cabeça do grupo, pois todas as mensagens passarão por ele. Assim, mesmo com uma diminuição do gasto energético por diminuição da troca de mensagens, o cabeça da rede rapidamente fica sem energia, inviabilizando um grupo inteiro.

Assim, como já explicitado antes, tais soluções não maximizam o tempo de vida da rede, somente o dos nós. A fim de aumentar o tempo de vida da rede utilizando esta estratégia, vários algoritmos foram propostos. O Leach (*Low-Energy Adaptive Clustering Hierarchy*) [Akyildiz et al. 2002b] tornou os cabeças de grupo, antes estáticos, rotacionais. Por meio de um método local ao nó, embora utilizando informação global, os grupos e seus cabeças são rotacionados pelos nós da rede, objetivando consumir a energia dos nós de forma mais igualitária. Assim, este é um método de meio termo entre a centralização e a descentralização. O Leach-C [Akyildiz et al. 2002b] também foi proposto, de estratégia semelhante ao seu predecessor, embora centralizado na Estação Base. Com informação dos nós, ela distribui os grupos pela rede. O HEED (*Hybrid, Energy-Efficient, Distributed Clustering Approach*) [Misra et al. 2007] utiliza um híbrido de energia residual e grau do nó para escolher seus cabeças. Outra estratégia, Pegasis (*Power Efficient Gathering in Sensor Information Systems*) [Akyildiz et al. 2002b] cria correntes de nós. Os nós somente trocam informação com seus vizinhos de corrente, e um nó escolhido aleatoriamente envia a informação agregada, nó este diferente a cada envio. Em [Jayashree et al. 2007], a estratégia otimiza a distribuição dos cabeças, de forma a minimizar o gasto energético e a possibilidade de retransmissões por colisão.

As quatro estratégias acima têm um problema em comum. Assume-se o envio da informção como dos cabeças dos nós diretamente à Estação Base. Isto limita a rede, pois os nós cabeça devem estar próximos à Estação Base ou ter suas áreas de cobertura grandes demais, gastando energia desnecessária. Pensando neste problema, o DEHCP (*Distributed Energy-efficient Clustering Hierarchy Protocol*) [Akyildiz et al. 2002b], além da rotação dos cabeças de grupo, forma um grupo também entre os cabeças de grupo, permitindo a troca de informações entre eles até um cabeça de grupo deste grupo. Este cabeça de grupo de "segundo nível" fará a agregação dos dados recebidos dos cabeças de grupo de "primeiro nível" e os enviará à Estação Base.

Outra solução para a comunicação entre os nós cabeça é o EEDP (*Even Energy Dissipation Protocol*) [Mandala et al. 2006]. Suponha a rede orientada horizontalmente, da Estação Base para os nós, ou seja, ela se localiza "à esquerda" ou "à direita" dos sensores. O algoritmo cria faixas verticais na rede, colunas de cabeças com distâncias à Estação Base semelhantes. As correntes pelas quais a informação passará serão formadas por um nó em cada uma destas faixas. Assim, se determinado nó da faixa está com baixa energia, outro nó de aproximada distância à Estação Base será escolhido, aumentando o tempo de vida dos cabeças. A figura abaixo, retirada do artigo, melhora a visualização do método.

A estratégia original de agregação de dados leva em consideração a igualdade de todos os sensores. Assim, é de crucial importância rotacionar os cabeças, como já visto. Uma

estratégia alternativa é a adotada em [Mann et al. 2005]. Os nós cabeça da rede seriam nós de tipo diferenciado, com maior capacidade de processamento e de bateria. Tal rede é chamada de rede heterogênea. Assim, não seria necessária nem desejável a rotação, pois estes cabeças, além disto, ainda poderiam concentrar certas operações, como o método de hibernação dos sensores. Isto significa colocar os sensores em modo de espera, sem funções de sensoriamento ou rede, para economizar sua bateria.

Também utilizando uma rede heterogênea, o EDTC (*Energy-Aware Distributed Topology Control*) [Kim et al. 2006] utiliza um método de agrupamento com o fim de aumentar o tempo de vida da rede. Tal método utiliza diagramas de Voronoi e diversos cálculos matemáticos a fim de dividir a rede em grupos. Outra utilização de Voronoi está em [Bash and Desnoyers 2007], que a utiliza, exclusivamente, para a divisão da rede. O autor afirma ser útil em vários casos, pois, sendo um método matemático, a localização dos nós está implícita no método. Se utilizada para medição de temperatura, por exemplo, a localização do sensor cuja área excedeu o máximo de temperatura já seria, automaticamente, de conhecimento da Estação Base.

Os estudos na área de agregação de dados são bem extensos. Tais estudos produziram muitas boas soluções, mas a tecnologia deve sempre seguir em frente. O DSC (*Distributed Source Coding*) [Wang et al. 2006] é uma solução para a redundância dos dados. Nele, cada sensor, utilizando as informações recebidas de outros sensores, faz sua própria eliminação de redundância. Nenhuma informação é trocada mutuamente, diminuindo a passagem de dados pela rede, como acontece na agregação de dados. Assim, tal estratégia consegue diminuir a quantidade de dados trafegando pela rede, pois não há redundância nos dados saindo de nenhum nó na rede.

Várias estratégias utilizam o DSC. Entre elas, temos o ROR (*Rate-Oriented Routing*) [Wang et al. 2006]. Ela parte do pressuposto de os sensores da rede serem capazes de modular seu máximo de fluxo. Esta estratégia, além do DSC, utiliza um controle de múltiplas taxas de tráfego, objetivando dividir, igual e estaticamente, o fluxo pela rede. Assim, segundo o autor, melhora-se, em relação a um algoritmo de taxas fixas, o tempo de vida da rede.

Outra estratégia semelhante à agregação é a dos conjuntos d-dominantes. Tais conjuntos detêm a característica de ter todos os seus nós a até d saltos de distância de qualquer outro nó. Alguns métodos o utilizam, como o d-CDS (*d-hop Connected D-dominating Sets*) [Ko and Huang 2005] e o Dd-CDS (*Densitybased d-hop Connected D-dominating Sets* [Ko and Huang 2005]. Aquele baseia sua estratégia em criar os conjuntos dominantes de modo ótimo, ou seja, com a razão da distância pela quantidade de conjuntos ótima. Em cada um destes conjuntos, usa-se a hibernação dos nós, e, ao acabar a bateria de um nó,

um dos dormentes o substitui. No Dd-CDS, a energia potencial entra como um parâmetro de escolha deste nó acordado no conjunto.

Spanning Tree

A *spanning tree* com aprendizado [Zhang and Huang 2006a] utiliza uma estratégia de especificação de destino estruturada, uma *spanning tree*, mas o pai de cada nó pode mudar a cada informação enviada, pois o cáculo é refeito a cada envio. Assim, informação de controle torna-se desnecessária, aumentando a autonomia do método.

Soluções Baseadas em Grafos

Algumas soluções vêm por meio da teoria dos conjuntos. O problema dos conjuntos dominantes, transposto para redes, consiste particionar um grafo em vários conjuntos cujos nós devem estar ativos e o resto desativado, ou eles devem estar usando energia total e, o resto, em modo econômico. Em [Misra et al. 2007], utiliza-se agrupamento e o *domatic partitioning problem*, problema de encontrar o número máximo de conjuntos dominantes disjuntos em uma rede. Cada conjunto dominante será composto pelos cabeças dos grupos. Assim, ao determinar um conjunto dominante, determinamos os cabeças dos grupos, e basta construir os grupos ao seu redor. A fim de aumentar o tempo de vida da rede como um todo, deve-se rotacionar os cabeças. Assim, maximizar o tempo de vida da rede significa encontrar a maior quantitade de conjuntos dominantes disjuntos possível. Com este método, o autor almeja minimizar o custo de rotação dos cabeças.

Outras são efetuadas por meio de problemas de Programação Linear, ou seja, uma solução de otimização de algum parâmetro, nesse caso, o gasto energético. Em [Ergen and Varaiya 2007], o aumento do tempo de vida é formulado como um problema de Programação Linear, transformado em uma série de árvores a fim de rotear os pacotes. Esta solução objetiva otimizar o tempo de vida e, ao mesmo tempo, garantir um atraso máximo de espera. O atraso é contabilizado em quantidade de saltos nas árvores. Neste mesmo artigo, vários outros são apontados, anteriores a ele, e com soluções mais simples e básicas, baseadas em grafos, além de várias funções de custo diferenciadas. O autor afirma, no entanto, aprimorá-las.

Teoria dos Jogos

Algumas soluções utilizam a teoria dos jogos como carro principal. Em [Nurmi 2006], o objetivo é desenvolver um roteamento a fim de aumentar o tempo de vida da rede. Seu método baseia-se na teoria dos jogos e em um método de aprendizado, pelo qual os nós

aprendem as melhores rotas com o tempo. Não utiliza ifraestrutura. Parte do princípio egoísta de roteamento, no qual nem todos os nós podem querer rotear um pacote, visando aumentar seu próprio tempo de vida. A teoria de jogos usada é inovadora, segundo o autor, pois é utilizada em um ambiente de rede dinâmica, sendo as tradicionais somente usadas em redes estáticas. Por fim, o autor utiliza os resultados obtidos a fim de provar o equilíbrio da distribuição energética na rede, provando um aumento do tempo de vida.

Mapa energético

Tendo-se em vista a importância da energia em uma rede de sensores, alguns métodos de medição energética [Al-Karaki and Al-Mashaqbeh 2007] foram criados. A medição energética básica consiste em conseguir, constantemente, informações dos sensores a fim de montar um mapa da rede. Alguns métodos baseiam-se nestas medições para fazer engendrar seu roteamento [Al-Karaki and Al-Mashaqbeh 2007]. Esta técnica utiliza agrupamento a fim de recuperar as informações dos nós. Concentradas na Estação Base, esta faz um mapa de áreas da rede e da situação energética destas áreas. Então, as informações são roteadas, preferencialmente, por áreas com alta capacidade energética.

O nivelamento consiste em minimizar a distância de todos os nós à Estação Base. O MCP-PS (*Maximum Capacity Path, Path Switching*) [Huang and Jan 2004], utiliza esta estratégia, unida a uma tática de mudança de caminhos baseada no mapa energético da rede, para melhorar o tempo de vida da rede. Sendo uma estratégia multi-caminhos, pode escolher e mudar entre eles a fim de economizar a energia de nós com baixa energia, gastando a dos outros nós.

Balanceamento de Carga

Balanceamento de carga é uma solução de redes de comunicações. Balancear a carga significa dividir as informações passadas pela rede de modo igualitário, nela inteira. Assim, nenhum caminho na rede fica sobrecarregado enquanto outros ficam sub-utilizados.

Um exemplo de utilização de balanceamento de carga na rede de sensores. O ELFR (*Energy-aware Load-balanced Fault-tolerant Routing*) [Yang and Ho 2006] utiliza a estrutura da rede como uma importante métrica nas decisões. A rede é hierarquizada por quantidade de saltos a partir da Estação Base, divida em caminhos e em nós de junta, os nós pelos quais passam mais de um caminho. Estes nós guardam informação de congestionamento e energética dos caminhos, fazendo as escolhas de roteamento dependendo de suas informações.

Em [Qi and Zhao 2007], a inteligência de enxame é utilizada para alcançar o balance-

amento de carga. São utilizadas várias Colônias de Formigas. Quando a formiga de uma colônia é enviada, ela fortalece a trilha feromônica da sua colônia, ao mesmo tempo enfraquecendo a das outras colônias. Assim, as colônias tendem a dividir os caminhos da rede, pois tendem a ter uma força feromônica equiparada em cada caminho. Assim, ao enviar informações, um nó tende a enviá-las por caminhos diferentes. Segundo o autor, melhora a utilização da banda, mas não prova um aumento do tempo de vida. Já em [Guoying et al. 2000], apesar de ser um estratégia com informação global, leva-se em conta não somente o balanceamento de carga, mas também a capacidade energética.

No TAEE (*Trafic-Aware Energy Efficient*) [Liu and Hong 2006], é utilizada uma métrica mista. Ela baseia-se em três fatores, a capacidade da rede, ou seja, o seu consumo de poder total, a energia remanescente nos nós, a fim de aumentar o tempo de vida da rede, e a reserva de tráfego, baseada no padrão de carga da rede. Assim, o balanceamento de carga tem um papel importante neste método, cujo roteamento é feito baseado em reseva de tráfego, ou seja, reservando uma largura de banda para passar as informações. A reserva é feita por meio de uma *spanning tree*, formada utilizando os três fatores supra-citados.

Em [Xu et al. 2005], uma solução baseada em vários caminhos é usada. Segundo o autor, os nós mais próximos da Estação Base são de importância crucial para o tempo de vida da rede, pois as informações dos caminhos passarão por eles para chegar à Estação Base. Assim, seu consumo tende a ser maior, se comparado ao dos outros nós. Este método, então, utiliza o DER (*Decisive Energy Ratio*), uma proporção da energia sobrando nos nós da rede com relação ao total possível, ou seja, uma média geral da energia sobrando nos nós. Se a energia em um nó for inferior ao DER, a informação não será passada por este nó. Assim, tenta-se igualar o consumo energético na rede como um todo. Um método baseado no supra-citado, baseado no DEROA (*DER with Overload Avoidance*) [Zongkai et al. 2005] melhora, segundo o autor, o método no qual se baseia. Ele tenta evitar a sobrecarga de um caminho sendo escolhido por vários nós.

Em [Akkaya and Younis 2005], uma solução utilizando QoS (*Quality of Service*) é proposta. O roteamento baseado em QoS parte do pressuposto de atribuir, a certos tipos de tráfego, uma precedência especial. Tais tráfegos seriam os de tempo real, enquanto os outros estariam em segunda precedência. Um exemplo de utilização seria no campo de batalha. Ao avistar algum veículo, a rede deveria proceder com seu encaminhamento de dados em tempo real, alertando sobre o inimigo e possibilitando o monitoramento. Assim, segundo os testes realizados pelo autor, o método desenvolvido seria mais rápido se comparado ao mesmo método sem utilizar QoS.

O LBR (*Load Balanced Routing*) [Iqbal et al. 2006] utiliza não somente o estado da

rede em um dado momento, mas sim um histórico dos estados da carga na rede. Assim, ele pode fazer uma estimativa mais acurada do consumo energético na rede. Depois de agrupar a rede, faz a rotação dos cabeças de acordo com o histórico de carga, podendo atribuir uma melhor, mais duradoura e mais econômica distribuição de grupos pela rede. Em [Liu 2006b], também é utilizado o agrupamento. Os nós utilizados, entretanto, são bem simples, impossibilitando a compactação no grupo. Para o roteamento, utiliza distribuição de carga, baseada na densidade dos grupos.

Inteligência Artificial

Pode-se utilizar um Algoritmo Genético para calcular a programação de roteamento [Islam and Hussain 2006]. Esta programação diz, a cada turno do algoritmo, os caminhos a serem seguidos pelas informações. Tais caminhos usam *spanning trees* nomeadas por árvoes de agregação, *aggregation trees* [Islam and Hussain 2006], pois utilizam agregação de dados intra-rede.

Esta, entretanto, não é a única solução para o problema. O T-ANT [Selvakennedy et al. 2007] utiliza agrupamento, mas de um modo diferenciado. Ele utiliza inteligência de enxame, formigas para a fomação dos grupos, e *Voronoi tesselation* para descobrir o número ótimo de grupos. Consegue aumentar o tempo de vida criando grupos ótimos.

Em [Iyengar et al. 2007], o atuor faz uma comparação entre algoritmos de roteamento baseados em inteligência de enxame. Ele compara os baseados em ACO, *Ant Colony Optimization*, com aqueles baseados em Algoritmo Genético. Aqueles, segundo o autor, por não serem centralizados, diminuem o atraso ponto-a-ponto.

Sistemas Multi-Agentes

Sistemas Multi-Agente são derivados da Inteligência Artificial e da computação distribuída orientada a objetos [Biswas et al. 2006, Biswas 2005]. Um agente é um software inteligente com capacidades autônomas. Um sistema baseado em agentes têm vários deles dentro do sistema, comunicando-se e atingindo objetivos, conjugadamente, cuja capacidade computacional de um só agente nunca permitiria. A inteligência de enxame é um tipo de sistema multi-agente, sendo cada formiga um agente.

O padrão organizacional de um sistema multi-agentes é sua parte mais importante. Ele define os tipos de agentes existentes, suas funções e como eles interagem entre si. Assim, como o autor afirma em [Horling et al. 2004], o padrão organizacional tem influência na eficiência da rede de sensores, sendo necessário escolhê-lo com cuidado. Além disso, ele afirma serem os agentes uma boa solução para aumentar a eficiência de uma rede de

sensores, motivo pelo qual há muitos artigos no tema.

Diversas soluções constituem uma arquitetura para utilizar sistemas multi-agentes em uma rede de sensores, como em [Biswas 2005, Biswas et al. 2006, Hussain and Matin 2006]. Nestes artigos descreve-se as categorias de agentes utilizados, sendo alguns nas EBs, para comunicação com o mundo externo, e alguns nos próprios sensores. Eles designam a infra-estrutura como sendo o agrupamento, e designam os pacotes trocados entre agentes. Estes pacotes são demasiado grandes se comparados a outros, como formigas. Além disso, elas não descrevem o roteamento ou o fazem de forma rápida e pouco profunda. Outra arquitetura é a MULE (*Mobile Ubiquitous LAN Extensions*) [Jain et al. 2006]. Ela utiliza sensores esparsos com abrangência de rádio baixa. Para a sua comunicação com a Estação Base, utiliza-se elementos móveis no ambiente. Estes elementos, sejam carros, pessoas, animais, dentre outros, ao passarem próximo aos sensores, recebem informações deles. Para tal, precisam estar equipados com aparatos sem fio. Ao passarem perto da Estação Base, entregam as informações. Assim, cada sensor gasta bem menos energia, pois não tem que reencaminhar mensagens nem ter muito alcance de rádio.

Uma divisão possível é quanto à localização dos agentes. Em [Shakshuki et al. 2006], os agentes são softwares localizados dentro da Estação Base. Este artigo trata de problemas nos quais usuários externos à rede sejam capazes de requisitar algum tipo de informação, ou mesmo a própira Estação Base o faça. Ao ser requisitada, a informação passa por um processamento interno aos agentes. Eles estão dispostos em uma hierarquia, e dispõe de informações sobre a rede a fim de realizar um roteamento economizando energia. Apesar da economia, esta é uma solução centralizada, com um claro problema de escalabilidade.

Outra opção pode ser o sensor conter os agentes. Em [Cicirelli et al. 2007], utiliza-se dois protocolos. O primeiro utiliza um método de conseguir informações da vizinhança do sensor para saber se ele deve ou não entrar em modo de espera. O segundo utiliza um esquema baseado em *Minority Game* para conservação energética. Assim também é em [Obayashi et al. 2005]. Ele utiliza uma linguagem de comunicação entre agentes baseada na UML e, embora não aumente o tempo de vida da rede, implementa um *framework* para segurança na rede de sensores com consumo energético baixo.

Eles podem, ainda, locomover-se de um nó para outro. Como já citado, as formigas podem ser tratadas como agentes. Assim, em alguns trabalhos, as formigas são mostradas como uma construção diferente, enquanto em outros, elas são agentes com feromônios. Em [Tamaki et al. 2007], o objetivo é conseguir a topologia de uma rede de sensores utilizando formigas. Utiliza três fases. A primeira é a de geração feromônica, na qual também

ocorre o cálculo da distância. A segunda é a de movimentação das formigas e descarga dos feromônios nos caminhos. A terceira é a de evaporação feromônica. Segundo o autor, o método feromônico alcança maior similaridade topológica se comparado a um sem utilização de feromônios. Pode-se, também, adicionar memória e mudança aos agentes. Em [Lerman and Galstyan 2003], o objetivo é alcançar adptação para redes multi-agentes. Impulsionados por mudanças ambientais ou por mensagens de outros agentes, os agentes utilizam sua memória para mudarem seu próprio comportamento. Assim, pode tornar-se um método semelhante ou complementar à "memória implícita" dos feromônios das formigas.

Em alguns casos, estas duas estratégias mesclam-se, formando um sistema multi-agentes com agentes nos sensores e trafegando pela rede. Uma situação possível é transmitir imagens em tempo real dos sensores à Estação Base. Em [Sun et al. 2006], propõe-se uma solução para o problema da fusão de dados. Ela consiste em utilizar as "fotos" de estado dos sensores da rede a fim de conseguir uma "foto" da rede como um todo. A solução consiste em utiliza mais de um tipo de agente. Os agentes estão tanto nos sensores como transitando pela rede. Ao receber um agente andarilho, o agente do sensor pode captar, dele, e liberar, para ele, informações sobre o estado dos sensores. Assim, de modo cooperativo, alcança-se uma estimativa do estado global da rede. O autor, entretanto, não utiliza esta alternativa a fim de melhorar o tempo de vida da rede.

Assim como no anterior, em [Saoseng and Tham 2006] temos uma estratégia híbrida de agentes dentro e fora dos sensores. Nele, a estratégia é otimizar a taxa de envio de mensagens pelos sensores. Ele utiliza uma arquitetura com agentes e aprendizagem por reforço, *reinforcement learning*. A comunicação entre sensores será utilizada para otimizar a taxa de envio, fazendo com que a banda da rede seja melhor utilizada. Com a aprendizagem, os agentes aprendem sobre seus vizinhos, prevendo seu comportamento. Otimizando a taxa de envio, minimiza-se o gasto energético extra por reenvio. Além disso, otimiza-se o uso da largura de banda, recurso escasso em redes de sensores. Apesar disso, o tempo de vida da rede não é levado em consideração. Em [Hussain and Matin 2006], estratégia ainda aborda, além dos agentes móveis e dentro dos sensores, agentes fora da rede, de interface para consultas na rede. O objetivo, aqui, é detectar os tipos de agentes importantes para o desenvolvimento de redes de sensores, segundo o autor. Agentes de consulta, de interface, regionais e de grupos são utilizados. Além disto, utilizando a comunicação entre eles e as capacidades derivadas das informações trocadas, consegue-se diminuir a quantidade de mensagens trocadas, aumentando o tempo de vida dos sensores. Em contrapartida, a solução não aumenta o tempo de vida da rede, um aspecto desejável, além de utilizar grupos, ultrapassado.

No aspecto anterior, ou seja, agentes móveis, há várias soluções para aumento do tempo de vida de uma rede de sensores baseadas em Sistemas Multi-Agentes [Malik et al. 2007, Taylor et al. 2004, Jamont and Occello 2003]. Nas soluções, os agentes podem desempenhar papéis diferentes na rede. Os algoritmos apresentados a seguir surgem como soluções para o roteamento em redes de sensores utilizando um paradigma diferenciado do cliente-servidor. Por sua abordagem na utilização dos agentes para a coleta de informação, eles desenvolvem uma comunicação a partir de agentes agregando informações de um certo interesse. Assim, eles utilizam uma abordagem orientada a interesses, agregamento e agentes para o roteamento dos dados.

As soluções para roteamento em redes de sensores são diversas, algumas bem simples, como no MADSN (*Mobile-Agent-based Distributed Sensor Networks*) [Qi et al. 2001]. Ela parte de uma rede sem roteamento, ou seja, a um único salto para fazê-lo a partir de grupos e agentes dentro deles. Esta, entretanto, é muito simples. Em [Yongtao et al. 2006], a proposta é melhorar o MADSN [Qi et al. 2001]. Neste, a rede é agrupada de forma a reduzir o tráfego de dados pela rede. Ele, entretanto, considera a troca de mensagens, dentro de cada grupo e entre eles, como somente um salto. [Yongtao et al. 2006] utiliza roteamento dentro dos grupos para diminuir a energia gasta nas trocas de informações entre os nós e seu cabeça. Em [Shakshuki and Malik 2007], a estratégia é, depois de agrupada a rede segundo um algoritmo chamado S-MAC, localizar os nós de borda, ou seja, aqueles pertencentes a dois grupos contíguos. Utilizando agentes, deve-se decidir quanto a deixá-los em um grupo ou em outro, dependendo do nível energético dos grupos.

Outras utilizam a disseminação de interesses como base para a solução [Intanagonwiwat et al. 2003]. Assim, conseguem agregar de dados sem a necessidade de agrupamento dos nós. Esta estratégia é utilizada em [Gan et al. 2004], cujo roteamento é herdado do EAR (*Energy Aware Routing*) [Shah and Rabaey 2002]. Neste, leva-se em conta, para a escolha da rota, o custo, em cada caminho, de alcançar a EB, além da energia restante no nó. Ele utiliza uma tabela de roteamento probabilística na qual não figuram todos os vizinhos, somente aqueles com baixo custo. Assim, ele consegue economizar parte da energia da rede criando diversos bons caminhos, ao invés de um só melhor caminho, no qual as baterias dos nós acabariam rapidamente. Em [Gan et al. 2004], o autor utiliza o EAR como base para o desenvolvimento de seu próprio protocolo. Ele somente leva em consideração, para seus testes, a fase de criação das rotas. Assim como no EAR, o autor conclui, depois de testes, ser, a melhor heurística para a criação das rotas, o custo do caminho e a bateria restante no nó. Comparando-se a somente a cada um destes parâmetros separados, esta estratégia mostra-se quase tão boa quanto somente a energia em tempo de vida da rede,

mas melhor no quesito de bateria média dos sensores. Se utiliza-se somente a energia, os caminhos criados tornam-se demasiado grandes, gastando energia demais. Além disto, este artigo utiliza uma ”agregação implícita”. Cada nó, ao receber uma mensagem, testa se ela veio de algum nó próximo ao nó de uma mensagem previamente recebida. Caso positivo, essa mensagem não é repassada, pois essa informação já fora enviada. O problema com esta estratégia é a utilização de um tempo global, inexistente em uma rede real.

Também é possível utilizar a disseminação de interesses para a agregação, mas os agentes agem de uma maneira distinta. Assim como em [Shakshuki et al. 2007], após a descoberta dos nós fonte, ou seja, os captadores de informação, os agentes são utilizados para percorrer estes nós, agregarem a informação e retornarem. Neste, entretanto, a desvantagem deste algoritmo é o conhecimento geral da rede necessário à EB a fim de rotear o agente pelos nós fonte. O MADD (*Mobile Agent Directed Difusion*) [Chen et al. 2007a] é semelhante ao anterior. A diferença, aqui, é o fato de existir uma sequência de visita dos nós fonte. Assim, o código do agente é salvo no primeiro fonte a ser visitado, e ele mantém um temporizador. Uma vez acionado, ele envia um agente, sem o código, pois os fontes já o têm da primeira passagem, passando por eles e levando informação à EB. O AbDD (*Agent based Directed Difusion*) [Malik et al. 2007], além de ter as vantagens do MADD, acrescenta algo a mais. O caminho do agente entre a EB e o primeiro fonte, e entre o último fonte e a EB, é escolhido por meio de uma tabela de roteamento probabilística baseada no custo do caminho e no nível energético dos vizinhos, assim como em [Gan et al. 2004]. Já no EMA (*Energy-efficient Mobile Agent*) [Pan et al. 2007], tenta-se resolver um problema intrínseco a todos os agentes. Por eles carregarem trechos de código a serem executados, os agentes acabam tornando-se muito grandes, gastando muita energia da rede. Neste algoritmo, o autor enumera cada tipo de código. Ao passar um agente com determinado tipo de de código, ele fica salvo no sensor, e aquele tipo de código não será mais carregado, por agentes, para aquele sensor. Assim, carrega-se menos carga nos pacotes. Em compensação, o diferencial dos agentes, a inocupação da memória do sensor com código, desaparece. O problema destas estratégias é o conhecimento geral da rede necessário para dizer qual a rota a se tomar para que o agente percorra os nós fonte para a agregação dos dados.

No IAR (*Intelligent Agent Routing*) [Liu 2006a] cada agente é um pacote trafegando pela rede. A Estação Base envia o agente por inundação. Quando o agente passa por um nó, ele desencadeia um temporizador. O agente continua sendo inundado pela rede, angariando informação de caminho, até o primeiro temporizador expirar. Nesta hora, a inundação pára e os agentes retornam, salvando nos nós o menor caminho, em saltos, até a Estação Base. Caso hajam empates, há um algoritmo de coloração para criar

mais de um caminho. Este algoritmo, no entanto, não almeja melhorar o tempo de vida da rede, pois não se preocupa com energia, engendrando somente um simples caminho mínimo. Em [Biswas et al. 2006], os agentes também são os pacotes. Neste artigo, o autor preocupa-se com o consumo energético. Ele cria um método de fusão informacional intra-nó, similar ao DSC já citado. O autor afirma ser a abordagem de agentes colaborativos melhor se comparada à clássica, cliente servidor. Tal abordagem já fora utilizada anteriormente. Em [Qi et al. 2001], ela fora proposta, embora de modo mais simples.

Quando um estudo começa tomar maiores proporções, a tendência é criar-se uma solução genérica, em algum aspecto. Em [Anthony and Jannett 2005], desenvolve-se um *framework* modular para testes de agentes em redes de sensores. Tal estratégia é desenvolvida em módulos, possibilitando a troca simples do sistema agentes ou da rede de sensores. Assim, possibilita-se uma fácil ferramenta de testes para agentes em redes de sensores.

A maior parte das estratégias com formigas, no entanto, concentra-se no primeiro tipo de sensores, os trafegantes pela rede. Neste intuito, muitas soluções diferenciadas foram construídas utilizando-se a inteligência de enxame.

Uma delas é a auto-organização utilizando formigas [Rui et al. 2006]. Nesta solução, o autor utiliza os feromônios deixados pelas formigas como uma métrica para colocar os sensores para acordar ou dormir. Com isto, a organização dos sensores acordados seria feita automaticamente, sem a necessidade de troca de mensagens diferentes das próprias formigas.

À moda dos algoritmos de IA, há algumas soluções com formigas com o intuito de calcular os caminhos ótimos em uma rede de sensores. Em [Liu et al. 2007], uma abordagem deste metodo é desenvolvida. Por meio da deposição de feromônios, as formigas traçam os melhores caminhos das classes de eventos até a Estação Base. Uma classe de evento é um conjunto de nós captando informações semelhantes. Depois de estabelecidas as rotas, elas tornam-se estáticas durante toda a utilização da rede. Em [Huang et al. 2006], uma estratégia bem semelhante é utilizada. A diferença são os fatores de avaliação do caminho. Nesta, utiliza-se a largura de banda, a energia do nó e a dos nós circundantes, enquanto naquela, uma estratégia de ângulo de deflexão. Tal estratégia objetiva minimizar o distanciamento do caminho de uma linha reta entre o nó e a Estação Base.

Estes algoritmos, no entanto, são estáticos. A estratégia calculada inicialmente é seguida até o fim, sem modificações. Esta estaticidade é indesejável ao algoritmo. Se alguma seção da rede começar a ficar sem energia, seria útil poder transferir as rotas pertinentes àqueles nós para outros, a fim de não extinguir sua energia. Também há, entretanto, algumas soluções dinâmicas, como a proposta no IAR (*Improved Adaptive*

Routing) [GhasemAghaei et al. 2007]. Nesta solução, constrói-se uma tabela de roteamento cujas linhas são os vizinhos do nó, e as tabelas, as Estações Base. Preenche-se a tabela com a probabilidade de certo nó, no próximo passo, enviar o pacote a certo vizinho, a fim de alcançar certo destino. A distância entre o nó atual e o destino é levada em consideração, assim como a soma da distância de cada um de seus vizinhos até o destino com a do nó até cada um de seus vizinhos. Também utiliza formigas voltando, cuja função é atualizar as tabelas de roteamento, somando probabilidade no caminho utilizado pelas formigas indo, e diminuindo nos adjacentes. Assim, prioriza o melhor caminho encontrado na ida. Solução semelhante é encontrada no *AntHocNet* (*Ant inspired algorithm for routing in MANETS*) [Caro et al. 2005].

Outras soluções para redes *ad hoc* [Hussein and Saadawi 2003, Hussein et al. 2005, Shuang et al. 2007] são ligeiramente diferentes da anterior. Apesar de utilizarem as tabelas de roteamento, suas probabilidades são compostas por mais de um fator, como a distância até a Estação Base e a energia nos nós do caminho. Este aprimoramento leva em conta a capacidade dos nós, e não envia informações por nós cujas baterias estejam em baixa, arriscando acabá-las.

Em [Jaikaeo et al. 2005, Yuan et al. 2004], as formigas são utilizadas para soluções de roteamento *multicast* na rede de sensores. Estas soluções são utilizadas no modo inverso das apresentadas até o momento. Enquanto estas focam-se em disseminar a informação dos sensores para a Estação Base, aquelas o fazem no caminho inverso, da Estação Base para os sensores. Geralmente, as *multicast* são utilizadas para disseminação de interesse, já apresentada.

Broadcast

O roteamento de informação em uma rede de sensores segue dois padrões básicos. Ela pode ser enviada da Estação Base para os nós ou destes para aquela. A primeira forma consiste, comumente, em enviar alguma informação de controle, algo necessário aos nós. Geralmente utiliza *broadcast*, pois este tipo de informação, normalmente, é destinada a todos os sensores da rede. A segunda, em enviar as informações coletadas do meio para processamento na Estação Base. Esta forma é utilizada pela maioria das soluções, por ser a mais utilizada em aplicações reais.

Os métodos apresentados até o exato momento, a exceção de [Mann et al. 2005], foram para redes *unicast*. Nesta, a comunicação é feita de uma fonte para um destino. Em *broadcast*, de uma fonte para a a rede inteira. Para fins de economia de energia, há uam diferença crucial entre os métodos de economia para as duas estratégias. Para *unicast*,

a energia deve ser economizada fazendo os caminhos se revezarem, gastando a energia da rede toda por igual. Em uma rede *broacast*, a mensagem já é enviada a toda a rede, inviabilizando este tipo de solução.

2.4 Redes de Sensores e Desafios

Supõe-se o seguinte cenário. Uma região é atingida por algum desastre, natural ou não. Tal região torna-se inospita para qualquer grupo de busca humano por sobreviventes, ou para estudo das causas e consequências do ocorrido. Assim, algum método longe do controle do ser humano deve ser utilizado na área a fim de conseguir tais informações.

Uma ideia facilmente proponível seria enviar um robô. Tal estratégia poderia solucionar o problema, mas o ambiente poderia ter-se tornado intrasponível ou inospito até mesmo para circuitos e metal. Assim, alguma outra solução poderia ser proposta.

Uma solução possível é, com a utilização de um avião, dispersar sensores pela área inteira. Boa parte deles pode ser destruída, tanto pelo ambiente quanto por dano na queda. Depois de estabelecidos no solo, os sensores deveriam estabelecer, por si próprios, a rede e os métodos de comunicação, sem interferência alguma dos seres humanos. A rede deve, por si própria, lidar com sensores defeituosos ou destruídos por condições inóspitas, assim como otimizar o seu consumo de energia, pois tal região não permite constantes adições de sensores à rede.

Em uma outra situação, temos um ambiente de trabalho. Sua ocupação [Qiao et al. 2006] e condições climáticas [Ota et al. 2006, Walla et al. 2007, Yong et al. 2007], além de outros fatores, devem ser monitorados. Em um método clássico, isto seria feito manualmente, por pessoas. Uma ideia para solucionar o problema seria com redes de sensores para fazerem um monitoramento autonômico. Tais redes decidiriam, autonomicamente, sobre economia energética do ambiente, controle das condições e soluções para possíveis problemas.

Às situações citadas, foram dadas soluções para a rede trabalhar sem interferência humana. Se ela não for capaz de trabalhar por si mesma, tais situações, assim como outras, teriam resoluções limitadas ou de alto custo. Estas soluções, por suas características, são chamadas de autonômicas. Assim, é notável ser a autonomicidade um claro passo a frente para as redes de sensores.

Capítulo 3

Comunicação autonômica

Em meados de outubro de 2001, por meio de um manifesto, a IBM externou sua preocupação para com o aumento da complexidade dos sistemas. Com tal aumento, tornavam-se necessários métodos a fim de automatizar o seu gerenciamento e integrar diversos sistemas. Este foi o início da computação autonômica.

Como citado anteriormente, a essência da computação autonômica é o auto-gerenciamento. De acordo com [Kephart and Chess 2003], esta, assim como outras características, devem ser levadas em consideração para a construção de um sistema computacional autonômico. Assim, no contexto deste trabalho, não é interessante entrar em detalhes sobre estas outras caraterísticas. O auto-gerenciamento, entretanto, deve ser melhor descrito. Ele baseia-se em quatro diferentes aspectos.

3.1 Características do Auto Gerenciamento

- **Auto configuração**

  O sistema deve conseguir, por si mesmo, realizar qualquer tipo de configuração necessária ao seu funcionamento. Nisto enquadra-se qualquer estado do sistema antes do início de seu funcionamento, assim como qualquer mudança imprevista. A adição e subsequente instalação, configuração e teste de uma nova componente do sistema é um bom exemplo de como a configuração autonômica deve funcionar. De acordo com políticas de alto nível, o sistema auto-configurar-se-á a fim de funcionar da forma requerida. Estas políticas descrevem o que sistema deve fazer, mas ele decide como implementará tais decisões.

- **Auto otimização**

Os sistemas autonômicos devem, também, otimizar seu próprio funcionamento. Assim, eles estão continuamente buscando novas alternativas de melhorar e aprimorar seu desempenho. Isto pode variar desde modificar informações de configuração das ferramentas sendo utilizadas até instalar novas versões dos programas já existentes ou novas componentes. O sistema deve, continuamente, buscar seu aprimoramento.

- **Auto reparação**

  Com a ocorrência de erros em alguma componente do sistema, ele deve estar ciente deles. Com informações de monitores do sistema e de testes, requisitados autonomicamente pelo sistema, feitos nas componenentes defeituosas, o sistema deve ser capaz de traçar o problema, suas causas e soluções. Depois disto, as possíveis soluções devem ser aplicadas e testadas, autonomicamente e em conjunto com a auto otimização, a fim de solucionar o problema sem qualquer interferência externa.

- **Auto proteção**

  O sistema deve, a despeito da existência de programas de proteção contra ataques maliciosos ou efeitos de erro em cascata incorrigidos pela auto reparação, decidir como e quando defender-se de tais ataques. Além da simples defesa, ele deve também ser capaz de, baseado no aprendizado de ataques e problemas anteriores, prever possíveis e insipientes problemas na rede, protegendo-se antes de eles ocorrerem, e não depois.

Assim, segundo [Kephart and Chess 2003, Dobson et al. 2006], estas são as quatro características principais do auto-gerenciamento, o centro de qualquer sistema autonômico. O outro elemento principal do sistema são os próprios elementos. Afinal, um sistema autonômico é, simplesmente, componentes comunicantes. Sendo-se capaz de atingir estas *self-* properties*, ou seja, comunicando-se elementos autonômicos, será possível criar um sistema autonômico funcional e independente de interferências externas.

Em [Dobson et al. 2006], o conceito de comunicação autonômica é explicitado. Segundo o autor, a diferença entre esta e a computação autonômica é o foco da autonomia. Nesta, como já visto, o foco são os sistemas e suas componentes, assim como o gerenciamento de sua capacidade computacional. Naquela, o foco torna-se o gerenciamento dos recursos de rede nos níveis de infraestrutura e de usuário. A despeito desta diferença, as características supracitadas e descritas são as mesmas para ambas as abordagens, indiferentes às demais características.

Tendo-se feito a diferenciação, convém exemplificá-la. Na descrição do auto-gerenciamento, foram utilizados exemplos em sistemas autonômicos. Segue um exemplo utilizando uma rede de computadores básica, a fim de mostrar como tais conceitos poderiam ser utilizados em comunicações autonômicas.

Tomemos uma rede de computadores simples. Cada um de seus nós, ou seja, computadores, será uma componente, e a função de monitoramento da rede será, exatamente, o auto-gerenciamento citado e explicado acima. Sua auto configuração entraria em ação no cálculo de sua tabela de roteamento, além de quando qualquer novo nó fosse agregado à rede, como um computador novo comprado para uma empresa. Sua auto otimização incluiria a configuração dos parâmetros dos programas monitorando e gerenciando a rede. Além disso, com a descoberta de algum melhoramento nestes programas ou alguma reconfiguração capaz de melhorar a performance da rede, tais modificações devem ser efetuadas autonomicamente. Em seguida, deve ocorrer um teste, cujo objetivo é confirmar a eficiência da modificação. Sendo confirmada, a modificação é mantida, caso contrário, ela é desfeita. Se algum nó deixar de participar da rede, por ter perdido sua conectividade ou ter ocorrido algum erro em seu funcionamento, deverá haver uma resposta autonômica. Esta é a auto reparação, para refazer roteamento a fim de a rede continuar completamente conectada. Além disto, quaisquer funções desempenhadas pelo nó devem ser repassadas a outro nó, a fim de manter a funcionalidade completa da rede. A auto proteção de uma rede consiste em *firewalls* e outros métodos de impedimento de intrusões. Além destes métodos mais básicos, o aprendizado deve ser incluído a fim de possibilitar o acesso restrito a informações, comunicação ou componentes da rede.

3.2 Passado e Presente da Comunicação Autonômica

Obviamente, redes de computadores não é o único uso possível do conceito autonômico. Quando falamos em comunicação autonômica, no entanto, este é o primeiro conceito à mente. Afinal, comparar a comunicação com uma rede e as componentes com os computadores ligados por ela é bem intuitivo. De fato, por isto os conceitos foram exemplificados por este meio.

Vários estudos em comunicações autonômicas podem ser encontrados na literatura. Assim, a fim de apresentá-los, deve-se pensar em um método de classificá-los em grandes grupos de desenvolvimento e pesquisa. Em [Dobson et al. 2006], esta divisão é feita em 5 grandes grupos de estudo e pesquisa em comunicações autonômicas. Seguindo esta divisão, a seguir, serão apresentados, de modo geral, os estudos e avanços da área. Levando-se em consideração o fato de a abordagem a seguir basear-se no artigo supra-

citado, não serão repetidas as citações de artigos presentes nele.

3.2.1 Análise e Projeto de Algoritmos Autonômicos

Comunicações autonômicas, assim como os sistemas autonômicos, além de outras propriedades, tendem a ser distribuídos e intensamente mutáveis. Tais aspectos da autonomicidade não são difíceis de entender. Como já explicitado, pode-se utilizar a Internet como uma grande rede a fim de ligar o sistema inteiro. Assim, seria impraticável ter um só ponto central coordenando milhões de dispositivos. A crescente mobilidade e conectibilidade dos dispositivos computacionais, como celulares, PDAs, dentre outros, os permitem sairem e entrarem em redes rápida e constantemente. Assim, a mutabilidade topológica da rede é clara.

Os algoritmos de gerenciamento de redes clássicos não prevêm tais características cada vez mais comuns. Em sua maioria, eles utilizam um conhecimento prévio da rede, algo impraticável em redes com tamanha mobilidade. Além disso, costumam utilizar um algoritmo centralizado de gerenciamento. Um gerente da rede, localizado, comumente, em um dispositivo dedicado exclusivamente a isto, geria a rede como um todo. Tais redes não costumam ser grandes o suficiente para requerer mais do que um dispositivo com alto poder computacional, e qualquer expansão poderia ser acompanhada por uma equivalente da computabilidade do gerente. Em redes com mihões de usuários, ou com restrições energéticas, um só gerente causa, além de um único ponto de falha da rede como um todo, um gargalo cuja expansão computacional nunca seria suficientemente grande.

Pelas razões supracitadas, os algoritmos, além dos sistemas, devem ser modificados. Algoritmos distribuídos e capazes de lidar com mudanças inesperadas na rede são indispensáveis para as comunicações autonômicas. Muitos algoritmos antigos devem ser e foram repensados, enquanto outros novos surgiram, e ainda muitos surgirão ou serão reformulados.

Os sistemas para comunicação em grupo, por exemplo, devido a sua pouco desenvolvida capacidade de lidar com muitos nós, foram reformulados em um contexto de redes ad hoc. Outros algoritmos são extensões de soluções clássicas a fim de proverem um algoritmo com as novas características requeridas. Entre eles estão roteamento multisalto em redes *overlay*, tabelas *hash* distribuídas para gerenciar redes ad hoc, além de conceitos de Inteligência Artificial aplicados, como redes neurais.

Alguns novos sistemas, como comunicação em grupo e *multicast* usando probabilidade, também surgem. Uma iniciativa recente é a de utilizar uma abordagem baseada em comportamentos biológicos como uma solução para diversos problemas. Mais adiante,

mais será explicitado sobre esta alternativa em particular. Outra abordagem é utilizar topologias auto-adaptativas, visto haverem estudos mostrando uma adaptação da topologia ao estado do sistema.

Outra importante característica para um algoritmo autonômico é a colaboração. Tal colaboração já foi descrita anteriormente, quando os sistemas multi-agente foram apresentados. Para obter-se um algoritmo autonômico descentralizado e distribuído, algum grau de cooperação entre os agentes, ou seja, os componentes, faz-se necessário. A troca de informações úteis e interesses pode auxiliar a tomada de decisões distribuída, visto não disporem, os nós, de informação global da rede. Assim, qualquer daqueles algoritmos poderia ilustrar esta discussão, assim como o BOINC, um projeto da universidade de Berkeley, além de [Dobson et al. 2006], projetos com teoria dos jogos aplicada, assim como utilizando modelos de economia.

Outra característica fundamental, assim como de qualquer sistema de comunicação, é a confiabilidade a estabilidade da comunicação. A principal função de um destes sistemas é comunicar, como o nome diz. Assim, é uma característica primordial a troca de informações ocorrer a qualquer momento, sem interferências ou mudanças no conteúdo das mensagens. Em [Dobson et al. 2006], é utilizado um método de meio termo entre a otimalidade de tempo de transmissão e a corretude dos dados transmitidos.

3.2.2 Sensibilidade ao Contexto e Semântica na Comunicação

O conceito de contexto envolve tanto sistema quanto usuários. O contexto de um determinado sistema de comunicação envolve seus usuários, suas preferências, as entidades envolvidas no processo, dentre outros. A fim de melhorar a resposta do sistema, as informações de contexto podem ser utilizadas. Assim, dependendo do comportamento de um usuário em particular, o sistema pode inferir a utilização de determinadas entidades e serviços, pré- disponibilizando-as e reservando recursos da rede, por exemplo.

Como mostrado em [Dobson et al. 2006], informação de contexto são dados brutos identificando as características de uma entidade. Uma entidade pode ser um usuário, uma entidade, um local, dentre outros. O contexto, então, é inferido a partir de uma sequência de informações de diversas entidades em um tempo. Tais informações não são somente sobre as entidades, mas também sobre seus relacionamentos.

Pode-se implementar a sensibilidade a contexto de duas formas diferenciadas. Na primeira, passiva, a informação de contexto é constantemente recolhida pelos integrantes do sistema. Assim, sempre há informação quando ela é necessária, embora ela não receba nenhum tipo de tratamento posterior. Assim, tal informação não serve ao sistema. Na

ativa, a informação é repassada a quem ela interessa. Um protocolo pode ser determinado a fim de processar a informação de contexto, compará-la com sua tabela de contexto e encaminhá-la à entitade à qual ela será útil. Assim, a informação é útil ao sistema como um todo, auxiliando na autonomicidade.

Outros desafios para a sensibilidade a contexto incluem a própria representação da informação de contexto. Por si só, este é um desafio. Além disso, a rede deve prover, baseada em sua informação captada, serviços, sem necessitar nem mesmo do conhecimento do usuário. Gerenciar as estruturas de contexto [Coutaz et al. 2005] também é um desafio considerável, pois deve-se combinar informações de várias camadas da rede, das mais baixas, com informaes de tráfego de dados e vazão, até as mais altas, com informações sobre os usuários e seus comportamentos.

Com a popularização desta técnica, ela passou a ser aplicada a mais áreas. Pesquisa de sensibilidade ao contexto para aplicações na Internet surgiram, além de outras áreas de comunicação, como o P2P (*Peer to peer*, ou ponto a ponto), computação pervasiva, dentre outros. Para a computação pervasiva, cujo objetivo é integrar diferentes elementos, fazendo-os trabalhar em conjunto, ter a informação de contexto é fundamental. Trabalhar em conjunto significa trocar informações para melhorar o desempenho geral. Se cada elemento conhece o contexto da rede, eles podem maximizar seu desempenho otimizando as funções da rede e minimizando a troca de informações, visto terem ciência do papel por eles desempenhado.

A sensibilidade ao contexto é uma ferramenta poderosa, auxiliando em muito a autonomicidade de um sistema. Ela deve, no entanto, ser mantida com cuidado. Uma arquitetura confusa de contexto pode gerar informações intelegíveis, inviabilizando seu uso em alguma aplicação útil. Assim, ela deve ser aplicada com cautela, a fim de tornar-se útil ao sistema, e não um estorvo.

3.2.3 Novos Modelos de Programa para Coordenação e Comunicações

Assim como acontece com os algoritmos, os modelos de programação tradicionais também tornam-se ultrapassados em vista das exigências e particularidades da autonomicidade. Nesta sub-seção, fala-se sobre os paradigmas tradicionais de coordenação, como cliente-servidor ou memória compartilhada.

Geralmente, tais métodos baseiam-se em conhecimento prévio da topologia da rede, ou seja, os componentes conhecem-se e sabem da existência uns dos outros. Como já explicado anteriormente, tal conhecimento prévio é algo inexistente em autonomicidade.

A constante mutabilidade e escalabilidade negam tal conhecimento à rede, tornando-o demasiado complicado e inútil frente a mudanças. Além disto, os métodos tradicionais não trazem o conceito de sensibilidade ao contexto. Como já dito, a informação contextual tem um papel importante na autonomicidade, dada a necessidade de a rede trabalhar em prol do usuário. Ela não deve aborrecer-lhe com questões internas, e sim auxiliar-lhe em suas tarefas, dado seu comportamento durante o tempo ou seu perfil. Sem o conhecimento do contexto, torna-se difícil a aplicação de sistemas distribuídos, uma das bases da autonomicidade. Por fim, os métodos tradicionais são estritamente divididos em camadas. Esta forma, apesar de ser a mais adotada e uma boa estratégia, impede a aplicação de obter informações sobre a rede, e vice-versa. Por exemplo, se uma aplicação tivesse acesso à vazão da rede, poderia reconfigurar-se a fim de enviar somente a informação possível de passagem por ela, aumentando ou diminuindo seu envio de tráfego dependendo da disponibilidade de largura de banda. A rede, no caminho inverso, poderia adaptar-se à aplicação, modificando seu protocolo entre UDP ou TCP, por exemplo, dependendo da aplicação a ser suportada. Assim, métodos tradicionais como estes não aplicam funcionalidades desejáveis à autonomicidade.

Por estas razões, novos métodos são necessários para os novos paradigmas a serem aplicados. São necessários cenários de elementos comunicantes, com suas comunicações e divisões longe da vista do usuário, trabalhando como um só.

Alguns métodos antigos, no entanto, podem ser usados a fim de alcançar a autonomicidade de um sistema. Pode-se citar dois exemplos de estratégias passíveis de utilização. A primeira são os modelos baseados em eventos. O exemplo mais clássico destes modelos é o *publish/subscribe*. Nele, os elementos comunicam-se entre si por meio de eventos gerados por eles e pelas respostas dadas aos eventos pelos outros elementos. Cada elemento tem suas respostas pré-determinadas. A desvantagem deste método é a clara necessidade de mapeamento dos possíveis eventos e das respostas requeridas. Dada a mutabilidade dos sistemas autonômicos, seria improvável conseguir mapear todos as eventualidades de um sistema com milhares de componentes, por exemplo. A segunda estratégia consiste nos modelos de espaço de tupla. Nele, são designados elementos específicos nos quais serão guardadas informações deixadas por diversos agentes. Tais elementos serão utilizados por mais de um agente, e as informações deixadas podem servir tanto para comunicação entre eles como de contexto, informações sobre o sistema. A desvantagem deste método é a escalabilidade. Quanto maior o sistema, mais desses pontos de encontro serão necessários. Além disto, se há uma falha nestes pontos, perde-se informação, talvez irrecuperável, além de inviabilizar a comunicação entre os agentes. Um gerenciador de serviços Web pode ser configurado de tal forma a trabalhar em cenários auto-gerenciáveis [Kaminski

et al. 2006].

Tendo em mente as desvantagens dos métodos antigos, fica clara a necessidade do desenvolvimento de novos modelos para sistemas autonômicos. Os modelos baseados em campos são como *frameworks*. A partir dos próprios componentes da rede ou de estruturas especializadas, eles dividem a rede em campos, cuja função é coordenar as ações das componentes. Este campo pode ser comparado a uma estrutura de dados distribuída, com funções tanto de armazenagem quanto de passagem de eventos. Parecidos com os modelos baseados em campos são os baseados em gradientes morfogênicos. Estes, além de estruturas semelhantes aos campos, utilizam gradientes de interesse, como na difusão direcionada já discutida, a fim de filtrarem as ações requeridas.

Uma categoria diferente de métodos são os baseados em biologia. Tais métodos, previamente citados e posteriormente melhor explicados, são baseados em sistemas biológicos existentes. Em sua maioria, são baseados em algum aspecto biológico do ser humano ou de criaturas cuja organização social é sua característica fundamental. Comumente, tais métodos são totalmente descentralizados, uma característica importante e desejável para a autonomicidade. Como um exemplo clássico, a inteligência de enxame [Bonabeau et al. 1998, Dorigo et al. 1991] pode ser citada. Algoritmos de formigas baseiam-se em um tipo de aprendizado por reforço. Na natureza, cada formiga é capaz de deixar feromônios por onde passa. Se ela sai e encontra comida, os feromônios criam um caminho para uma rota de alimentação, e outras formigas tendem a segui-lo. Assim, elas comunicam-se de um modo passivo e trabalham de modo descentralizado, sendo um bom método a se copiar. Um exemplo de aplicação da inteligência de enxames auto-adaptativa está em [Fórster et al. 2007]. Há, também, diversos métodos baseados em sistemas do ser humano. O sistema imunológico humano, por exemplo, trabalha de modo descentralizado, pois cada pequena célula trabalha por si só. Este tema será tratado em mais detalhes mais adiante.

Alguns métodos não almejam solucionar o problema, mas antes melhorar o desempenho dos métodos supra-citados. Eles, por meio de teorias epidemiológicas, almejam melhorar e diminuir a carga da manutenção de estruturas de dados distribuídas, como as utilizadas em campos e feromônios. Assim, eles almejam dar garantias probabilísticas a estes algoritmos. Alguns outros tentam garantir a presença, no sistema, de determinadas caracteríticas desejáveis. Para este fim, pode-se utilizar de redes *overlay*.

Outros métodos desenvolvidos têm em vista as redes P2P. Em [Gurun et al. 2006], o autor porta um método antigo para um novo patamar, alcançando, inclusive, melhoramento do gasto energético para a solução. Alguns são desenvolvidos para mais fins específicos, como coordenar milhares de pequenos coletores de informações em missões espaciais [Hinchey et al. 2007]. Eles devem, longe do controle humano, demonstrar

as propriedades de auto-* e trocar informações, pois cada um somente capta um tipo de informação. Trocando-as, podem chegar a alguma conclusão geral sobre o estudado. Também pode ser utilizada nas redes móveis de próxima geração, como uma maneira de aplicar a autonomicidade a estas redes [Kuhne et al. 2007]. Ainda, há a coordenação autonômica de redes sem fio híbridas [Shen et al. 2005b], desenvolvendo um *framework* autonômico para conseguir coordenar as funções da rede, como o gerenciamento de recursos de rádio, ou RRM, as rotas, dentre outras funções necessárias.

Assim, seja utilizando novos ou antigos modelos de coordenação, vê-se a sua necessidade inerente à autonomicidade. A descentralização é, aqui, papel fundamental dos métodos, dada a elevada escalabilidade de redes de comunicação autonômica. Seja para coordenar as atividades de pequenos sensores de calor ou para rotear informações por uma rede de escala mundial, a distribuição da tarefa pela rede, ao contrário da centralização da atividade, é uma caracterítica fundamental ao alcance da autonomicidade da comunicação.

3.2.4 Segurança e Confiabilidade

Estas sempre foram preocupações das redes, desde o seu início. Redes de comunicações devem ser seguras, pois seus usuários, seja um órgão governamental de segurança nacional ou pessoas tranquilamente conversando em um bate-papo, não gostariam se seus dados fossem lidos ou modificados. Dados secretos devem ser mantidos secretos, e nenhum dado deve ser maliciosamente colocado no sistema. A segurança da rede e a confiabilidade dos dados são, e sempre serão, assuntos primordiais a serem tratados em qualquer sistema de comunicações.

De início, esta segurança foi feita por *firewalls*, a fim de impedir a entrada de usuários estranhos, e protocolos de confiabilidade nas comunicações, a fim de impedir usuários externos de escutar ou modificar conversações em andamento. Tais mecanismos são a mais básica proteção de qualquer sistema de comunicações. Obviamente, com o aumento da complexidade dos sistemas, há também o aumento das falhas de segurança. Concomitante ao aumento da escalabilidade, cresce a complexidade em desenvolver e configurar tais programas e sistemas de defesa. Assim, vê-se a necessidade do surgimento de novos métodos de implementar a segurança nas comunicações.

Em um sistema autonômico, as políticas de segurança serão implementadas pelas componentes de gerenciamento. Elas têm regras e políticas para gerenciar as outras componentes do sistema. Entre estas regras, deve constar a segurança. As soluções de confiabilidade clássicas abundam no mercado. Entre elas, pode-se citar algumas. A troca de chaves é a mais simples. Ela consiste em cada elemento ter uma chave e conhece-

rem as chaves uns dos outros. Assim, ao enviar uma mensagem a outro elemento, ela a criptografa com a chave daquele elemento, de forma a somente ele estar possibilitado de entendê-la. Alguns outros métodos derivam deste, como terem chaves públicas e privadas, ou as chaves serem distribuídas por uma entidade central ao invés de trocadas entre os elementos. Para a confiabilidade, um método pode ser trocar mensagens criptografadas de modo a somente o receptor poder decifrar. Assim, pode-se ter certeza do contato com o elemento correto. Além destes, círculos de confiança, identificadores únicos para os elementos, como o SSO e o Microsoft .Net, ou ainda busca ativa de credenciais foram desenvolvidos.

Tais métodos de obtenção de confiabilidade, no entanto, não encaixam-se nos desígnios da autonomicidade. Em sua maioria, são falhos e centralizados, ou carecem de escalabilidade. A criptografia, por outro lado, é um bom método de segurança de mensagens, embora possa ser sempre aprimorado, pois alguns são centralizados. Com isto em vista, novos métodos de confiabilidade e segurança necessitavam ser desenvolvidos.

Assim, começou-se a caminhar na direção de métodos de segurança e confiabilidade autonômicas. Um método seria utilizar um *framework* distribuído de gerenciamento de credenciais. Com um método distribuído de gerar e distribuir as credenciais, o problema de escalabilidade da rede pode ser resolvido. Assim, este método é muito utilizado, como pode-se ver nos artigos citados acima. Almejando o melhoramento do método e sua maior adaptação à comunicação autonômica, a negociação de confiança foi desenvolvida. Nela, as credenciais podem ser "discutidas" pelos componentes do sistema até haver um acordo em um nível de segurança determinado. O futuro, no entanto, parece encaminhar-se para os métodos de reputação digital. Tais métodos irão disseminar, na ocorrência de uma troca de mensagens para aquisição de confiança entre dois elementos, pela rede, informações sobre esta aquisição. Assim, os outros elementos poderão, autonomicamente, construir conhecimento sobre esta troca, facilitando uma futura aquisição de confiança.

Depois de muito estudo, notou-se a necessidade de métodos realmente autonômicos a fim de conseguir uma segurança e confiabilidade verdadeiramente autonômicas. Assim, diversos métodos foram desenvolvidos. Alguns deles começaram a utilizar sistemas multi-agentes e agentes inteligentes. Pode-se utilizar, por exemplo, métodos autonômicos baseados no paradigma da Rede de Pacotes Cognitiva [Dobson et al. 2006]. Outra solução, já abordada em sessões anteriores, é a biologicamente inspirada. Inspirando-se no sistema imunológico do ser humano, é possível desenvolver sistemas de segurança capazes de aprender com ameaças anteriores, proteger-se contra elas por meio de um método inteligente e prever futuras ameaças. Este assunto será melhor trabalhado adiante.

A segurança deve ser feita a nível de serviços do sistema. Assim, tal segurança cai além da de uma rede comum. A capacidade de segurança autonômica deve garantir a utilidade dos serviços sendo providos pelos elementos, além de uma baixa utilização de recursos por parte deles. Além disto, deve certificar-se da confiabilidade e segurança no momento do oferecimento do serviço, a fim de garantir seu oferecimento por um elemento válido, assim como em qualquer requisição por parte do elemento provedor, ou comportamentos estranhos de elementos. Tais medidas devem ser tomadas com fim de certificar-se da autenticidade e validade dos elementos e serviços.

3.2.5 Teste e Validação

Esta área é de importância fundamental ao desenvolvimento da autonomicidade. Os testes, como já dito em seções anteriores, são essenciais às auto otimização e reparação. Depois de detectado um problema ou instalada alguma funcionalidade ou componente nova, um teste é necessário a fim de mensurar a melhora ou piora do sistema. Além disto, cada validação tem um teste diluído em seu contexto.

As validações são de igual importância. Como dito pouco acima, a segurança utiliza-se muito desta valiosa técnica a fim de certificar-se da confiabilidade de serviços e componentes, dentre outras coisas. Além disto, a sensibilidade ao contexto precisa ser constantemente validada. Suas informações não podem, sob hipótese alguma, transmitir uma visão errônea do sistema autonômico como um todo.

O teste de uma rede, no entanto, pode-se extender à rede como um todo. Se a simulação não consegue suportar um teste apropriado, *testbeds* são necessários. Um *testbed* é um ambiente semelhante ao real montado para testes reais em redes. Enquanto algumas redes não podem ser testadas por simulação, outras requerem um teste real a fim de ver como elas realmente funcionam, ou para conseguir informações reais.

Os testes de dentro da rede ou suas validações são um assunto deveras específico para serem tratados com maior profundidade. Cada aplicação necessitará deles, no entanto, transformando-os em parte indispensável de qualquer sistema autonômico.

3.3 Futuro das Redes Autonômicas

Na última sessão, foi extensamente explicado como e onde as comunicações autonômicas surgiram e operam. Foram descritas as principais características dos sistemas autonômicos. Posteriormente, sendo-se mais específico, foi explicada a diferença entre os sistemas autonômicos e as comunicações autonômicas. Foi dada uma opção de

classificação de suas soluções por cinco aspectos fundamentais das comunicações autonômicas.

Assim divididas, foram dados exemplos e explicações nesta divisão. Os exemplos utilizados, em sua grande maioria, foram de sistemas desenvolvidos e consolidados, ou em estágio de desenvolvimento. Dentre estas soluções, mais de uma vez foram citados os sistemas biologicamente inspirados. Seja na forma de algoritmos, soluções para modelos de organização e coordenação de elementos ou segurança do sistema, esta abordagem foi utilizada, amplamente, com sucesso nas comunicações autonômicas.

Tendo em vista este fato, deve-se, agora, dar uma opção de futuro. Muitos dos conceitos e ideias apresentados podem continuar a ser trabalhados a fim de conseguir mais e melhores soluções para comunicações autonômicas. Os sistemas biologicamente inspirados destacam-se por sua abrangência dentro da área. Eles podem ser utilizados, e o são, em várias “áreas” do problema, como visto. Tais sistemas são uma área de grande estudo, e com muitas soluções ainda a serem apontadas.

Assim, o próximo capítulo tratará destes sistemas, colocando-os em maiores detalhes.

Capítulo 4

Sistemas Biologicamente Inspirados

Sistemas Biologicamente Inspirados são qualquer tipo de sistema cuja solução seja baseada em algum tipo de sistema existente na Natureza. Tais soluções para problemas são bem antigas, datando, no mínimo, de 1991. Notável, também, a versatilidade de sua utilização. Nos mais diversos campos eles podem ser utilizados, e de várias maneiras. Visão artificial [Enke 1997], controle motor [Zadel 1996], sistemas baseados em comportamentos [Brooks 1991], aprendizado em robótica [Agah and Bekey 1997, Beer et al. 1997], comunicação [Mortara 1997, Bartfai 1998], resolução de algoritmos complexos como o problema do caixeiro viajante [Dorigo et al. 1991, Dorigo and Gambardella 1996, Colorni et al. 1992] e segurança de redes [Hofmeyr et al. 1998, Somayaji 2007], dentre outros.

Com o relato, em 2001, do aumento de complexidade dos sistemas, pela IBM, a computação autonômica foi proposta. O nome, em inglês, dado a ela é *autonomic computing*, pensado por causa do *autonomic nervous system*, ou sistema nervoso autônomo. Tal sistema faz parte de nosso sistema nervoso. Sua função é controlar atividades insconscientes, tal como o batimento cardíaco, controle da temperatura e da digestão. A computação autonômica tem exatamente este objetivo, utilizar o próprio sistema para controlar partes de si mesmo, deixando somente as decisões de mais alto nível para o gerente.

Como pôde ser visto nos exemplos citados acima, a utlização de Sistemas Biologicamente Inspirados data de antes do manifesto da IBM. Eles, no entanto, começaram a ser bem mais utlizados a partir desta data, pois é óbvia sua utilização para a computação autonômica, visto ser seu próprio nome baseado em um sistema biológico. Assim, com a crescente tendência à automação, o uso de Sistemas Biologicamente Inspirados acompanhou este crescimento. Alguns tipos de soluções biológicas, no entanto, tornaram-se mais populares que outras, sendo mais utilizadas.

Abaixo há uma divisão arbitrária, baseada em [Somayaji 2007], das aplicações nas

quais conceitos biológicos foram e podem ser utilizados a fim de melhorar o desempenho dos sistemas ou adequá-los à autonomicidade. Como um motivador para este estudo, apresenta-se, a seguir, um pensamento ao qual pode-se chegar sem muito esforço. A Natureza, há bilhões de anos, aprimora suas criações. Assim, por que não aprender com ela?

4.1 Robôs

Os robôs podem utilizar-se muito da associação com a Natureza [Beer et al. 1997]. Afinal, um robô tende a duplicar algum ser da Natureza, seja um pequeno inseto, como nos robôs de investigação, ou o próprio ser humano. Sendo assim, torna-se óbvia o fato de os projetistas basearem-se no estudo da Natureza a fim de aprimorar seus mecanismos.

Cada robô individual pode possuir diversas semelhanças biológicas em sua construção. É mais natural basear-se no sistema de locomoção de insetos, por exemplo, se comparado ao uso de rodas. Afinal, não existe criatura na Natureza utilizando rodas para sua locomoção, mas pernas articuladas são um bom ponto evolutivo. De fato, a locomoção de um inseto é um sistema inteligente, pois ele consegue mover-se com patas faltando, simplesmente refazendo a estratégia de quando mover cada pata.

Em [Beer et al. 1997], o autor desenvolve um método distribuído e biologicamente baseado para a caminhada de robôs assemelhados a insetos. Este é método, baseado no comportamento de caminhada dos insetos, permite ao robô reconfigurar sua passada em terrenos ruins, permitindo certa autonomia necessária a robôs cujas missões os impeçam de contato humano. Tais missões incluem locais inacessíveis, hostis ou até missões espaciais. Os robôs devem enfrentar e vencer seus obstáculos, auto-configurando sua locomoção, autonômicamente, de acordo com o terreno enfrentado.

Além da inteligência de enxames, outra estratégias utilizam os feromônios para a solução de problemas. Tal estratégia pôde ser comprovada na seção das redes de sensores. Da mesma forma, em [Brooks 1991], o autor utiliza a comunicação feromônica a fim de coordenar não formigas, mas robôs. Esta, no entanto, não parece ser a única

4.2 Defesa

Abordagens biológicas para a segurança de computadores vêm sendo usados há muito tempo [Somayaji 2007, Hofmeyr et al. 1998]. O termo “vírus de computador” é, atualmente, utilizado tão comumente a ponto de, às vezes, esquecermo-nos de sua origem

biológica. Assim o é com vários termos, como *firewalls*, detectores de intrusão, dentre outros, arraigados no vocabulário computacional.

Com isto em mente, a seguir será mostrada um visão geral sobre a área. Primeiramente, será apresentado, superficialmente, o sistema imunológico humano, a fim de formar uma base para o entendimento do sistemas posteriores. A seguir, serão apresentados alguns sistemas de defesa baseados em biológia já desenvolvidos. Por último, alguns rabiscos sobre o futuro desta área.

4.2.1 Sistema Imunológico Humano

Nosso corpo dispõe de várias barreiras contra microorganismos. A primeira delas é a pele, um cobertor sobre quase toda a extensão de nosso corpo. Mesmo em locais desprotegidos por ela, algum outro fator atua como proteção, como unhas ou pêlos. Se o invasor consegue passar essa primeira defesa, adentrando o corpo, outro esqudrão deve entrar em ação. Esta defesa é o sistema imunológico.

O sistema imunológico humano é composto por diversos tipos de células cuja função é o reconhecimento e extermínio de corpos estranhos. Tais células comunicam-se, gerenciando suas buscas e patrulhas, por meio de elementos químicos. Assim, o conjunto, como um todo, parece um sistema distribuído, pois cada célula é um elemento, com características autonômicas, pois elas gerenciam a si mesmas em suas buscas e no combate aos microorganismos. De certa forma, parecem agentes, pois elas comunicam-se a fim de obter melhor resultado se comparado ao trabalho individual de cada célula.

Nosso corpo tem dois tipos de células de defesa contra microorganismos estranhos, o de reconhecimento inato e o adaptativo. O primeiro é o de reconhecimento de padrões. Se o corpo, previamente, houvera-se defendido contra este tipo de microorganismo, terá produzido células cuja função é procurar por este padrão e destruí-lo, de modo muito parecido ao funcionamento de um anti-vírus para computadores. O outro modo, de predição, utiliza um método diferenciado. Ele guarda registro de todas as células e dos sinais emitidos por elas. Quando ele encontra algum sinal diferente de seus registros, acusa como invasor e destrói. Este método é denominado de seleção negativa.

Um dos desafios das células imunológicas é justamente o de acusar como invasor a célula correta. Se o sistema encara uma célula corporal, por exemplo, a qual seus registros não marcam como natural do corpo, ele a destruirá, mesmo não sendo um invasor. Outro problema é quando alguns microorganismos, de menor tamanho se comparados às células, escondem-se dentro delas. Neste caso, uma técnica ligeiramente diferente é necessária para sua detecção, utilizando complexos peptídicos.

4.2.2 Sistemas Imunológicos Artificiais

Dada a complexidade do sistema imunológico do ser humano, escondida por trás de tão simplória explanação, seria inviável traduzi-lo por completo para processos computacionais. Assim, as estratégias são utilizar algumas ideias gerais a fim de melhorar os métodos atuais de segurança, ou desenvolver novas abordagens, possivelmente multidisciplinares [Kim et al. 2007].

Alguns destes sistemas são baseados na seleção negativa. Assim, eles devem dispor de componentes autonômicas capazes de detectarem, utilizando sua base de dados, todo e qualquer elemento estranho no sistema. Algumas estratégias foram desenvolvidas, como apresentado em [Somayaji 2007]. A primeira utilizava esta estratégia a fim de melhorar um anti-vírus, prevendo possíveis novos vírus, ao invés de somente combater os já catalogados. A segunda foi desenvolvida para a detecção de intrusão. Ambas as abordagens sãoo insipientes e têm falhas e limitações, sendo necessário algum aprimoramento no sentido de uma verdadeira autonomicidade. Muito trabalho está sendo desenvolvido na área [Somayaji 2007], inclusive com aplicações para outras áreas, como bancos de dados [Somayaji 2007].

Semelhante à seleção negativa, o REALGO (*REtrovirus ALGOrithm*) [Edge et al. 2006] utiliza uma estratégia semelhante ao RNA de transcrição inversa biológico [Edge et al. 2006]. Este processo consiste em transformar o RNA em DNA. O processo criado é semelhante à seleção negativa, pois também acumula conhecimento a fim de detectar vírus novos por comparação. A diferença está em como esta informação é acumulada. Na seleção reversa, assim como em nosso corpo, acumula-se informação a partir de vírus já combatidos e destruídos. Nesta nova abordagem, os vírus podem ser destruídos antes mesmo de terem tempo de se disseminar.

Outros sistemas são baseados no MHC. O MHC é um peptídeo cuja função é localizar microorganismos escondidos dentro de uma célula. Poder-se-ia comparar o sistema biológico com um algoritmo de teste de comportamento do sistema. Assim, caso houvesse um “vírus” escondido em uma célula, ou seja, elemento do sistema, o algoritmo seria capaz de detectá-lo pelo funcionamento anormal da componente. Foram propostos diversos métodos para simular o MHC. Os mais básicos consistem em fazer diversas chamadas ao sistema, em uma situação na qual ele esteja desinfectado, armazená-las e compará-las com chamadas esporádicas [Somayaji 2007]. Houveram várias evoluções deste sistema, como em [Somayaji 2007]. Em face destes métodos, ataques mimetizantes [Somayaji 2007] foram desenvolvidos. Estes ataques inviabilizavam o funcionamento das técnicas desenvolvidas. Soluções foram desenvolvidas [Somayaji 2007], mas ainda

são insipientes e pouco abrangentes.

O texto apresentado tem o objetivo de introduzir e exemplificar, de maneira reduzida, os sistemas imunológicos artificiais, assim como todas os assuntos desta sessão. Isto assemelha-se mais a um resumo, se comparado a um tratado especializado e pormenorizado. Mais informações podem ser encontradas na literatura como um todo. Uma boa fonte de busca, descrevendo bem melhor o tema, encontra-se em [Kim et al. 2007].

4.2.3 Diversidade

Diversas soluções foram colocadas na subseção anterior. Nem todas as soluções, no entanto, advêm da observação do corpo do ser humano. A Natureza é ampla e repleta de boas ideias e soluções para problemas. Deve-se saber trabalhar com ela a fim de obter os melhores resultados. Assim notaram, há muito, os fazendeiros. Eles perceberam uma diminuição considerável da qualidade do solo com o plantio, em diversos anos seguidos, de uma mesma cultura no espaço. Cada tipo de planta consome, proeminentemente, um tipo de nutriente do solo, e, ano após ano, este nutriente tende a diminuir. Assim, a rotatividade de culturas foi desenvolvida, dando tempo ao solo de recompor certo nutriente plantando algum outro grão, consumidor primário não deste nutriente, mas de outro.

Um problema semelhante ao da monocultura foi notado com a disseminação dos programas pelo globo. Utilizar, em larga escala, os mesmos programas tem uma desvantagem crucial, uma falha de segurança. Se é descoberto um método de burlar a segurança do programa, milhões de computadores estarão abertos à intrusão. Assim, uma diversidade de programas seria desejável, porém, impraticável. A “monocultura de software” tem suas vantagens, como o aprendizado em larga escala da utilização do programa. Assim, métodos teriam de ser desenvolvidos a fim de diminuir a vulnerabilidade dos programas.

Comumente, o ataque a este tipo de sistema consiste em saber como o código do programa funciona a fim de atacar suas bases programacionais. Assim, métodos foram desenvolvidos com o fim de modificar as bases destes programas [Somayaji 2007]. Tais bases constituem-se de *buffer*, memória, estruturas algorítmicas simples, como filas ou listas, conjuntos de instruções, dentre outros.

Apesar da sua efetividade, estas soluções não detêm componente biológica alguma. A única comparação plausível é com a rotação de culturas, comparada a uma rotação de bases programacionais. A reprodução e a morte são exemplos de métodos biológicos de diversidade. Poderiam ser utilizados nas soluções para este problema. A reprodução gera indivíduos únicos a partir de outros indivíduos, enquanto a morte permite a ascensão de outros indivíduos, gerando maior diversificação. Assim, falta uma solução mais biológica

para um problema biológico. Afinal, na Natureza, quem não se adapta é extinto.

4.2.4 Homeostase

Homeostase é o nome dado à auto-regulação corporal. Em todo ser vivo, há certos parâmetros cuja regulação é essencial. Os répteis são animais de sangue frio. A temperatura de seu corpo é controlada pela quantidade de raios solares recebidos por eles. Assim, pela manhã, eles tomam Sol a fim de aumentarem sua temperatura, pois seu corpo é incapaz de regulá-la. No ser humano, ao contrário, a temperatura é regulável. Quando estamos com frio, os poros fecham-se e trememos para gerar calor. Quando estamos com calor, os poros se abrem e transpiramos. Outra regulação importante ao ser humano é a de fluídos. A transpiração e a urina fazem o papel desta regulação.

O computador tem regulações semelhantes. Ele também deve regular sua temperatura, a fim de não super aquecer, e regulações elétricas, com fim de não queimar circuitos. Os programas devem ter regulações semelhantes. Afinal, o sistema imunológico é como uma última defesa da homeostase. Os patógenos simplesmente afetam o equilíbrio de um sistema, assim como o frio o ou calor. A diferença é a intensidade do desequilíbrio. Para pequenos desequilíbrios, somente tremer funciona, para outros, o sistema imunológico deverá ser ativado. Assim, sistemas foram desenvolvidos [Somayaji 2007] tendo por objetivo esta homeostase. O citado é uma agregação ao *kernel* do Linux cujo objetivo é priorizar as tarefas de programas com comportamento normal, deixando de lado aqueles com problemas. Uma homeostase simples, se comparada à contida em [Somayaji 2007]. Baseada na anterior, esta sufoca o acesso dos programas à rede, diminuindo o acesso de programas perniciosos.

4.2.5 Futuro

Diversos trabalhos baseados em Sistemas Biologicamente Inspirados foram desenvolvidos. Se revistos em detalhes, é possível notar que tais iniciativas, apesar de basearem seus problemas e resoluções em aspectos biológicos, não conseguem mimetizar as soluções naturais. O conhecimento científico das células do nosso sistema imunológico ainda é precário. Assim, os projetistas “improvisam” em suas soluções, enquanto o ideal seria mimetizar as soluções de modo perfeito. Afinal, como já dito, são bilhões de anos de evolução, não de ser humano, mas da Natureza.

A despeito dos diversos trabalhos existentes na área, muito poucas soluções naturais foram realmente exploradas. A Natureza é uma infinidade de inspiração, basta saber

ver. Ótimas soluções para os mais variados problemas existem, bastando alguém com conhecimento para vê-los, discerni-los e aplicá-los.

A maior barreira para isto é o próprio conhecimento humano, assim como a falta de tempo para se apreciar um “simples” fenômeno natural, coisa tão pouco importante nos dias de hoje. Afinal, o homem considera-se acima da Natureza. A maior barreira do ser humano é, assim como sempre foi, o prórpio ser humano.

4.3 Inteligência de Enxames

Uma outra inspiração da Natureza muito utilizada na Computação e em Redes são as colônias de insetos. Utilizando a inteligência de enxames, tal solução implementa a comunicação feromônica entre pequenos elementos, as formigas da rede. Alguns trabalhos não utilizam a terminologia do formigueiro, mas sim abelhas ou outros insetos sociais. As formigas, no entanto, ocupam lugar de destaque em meio a estes algoritmos. No próximo capítulo, esta estratégia será ampla e minunciosamente abordada.

Pesquisas em várias áreas da computação acabam necessitando de outras áreas. Com o crescimento da complexidade em encontrar soluções para problemas em grafos utilizando métodos mais convencionais, táticas de Inteligência Artificial começaram a ser utilizadas.

No início, foram citadas soluções de Inteligência Artificial para resolver problemas com grafos. Algumas destas soluções, mais clássicas, baseiam-se em heurísticas para resolver seus problemas. Um heurística é um método de solução de um problema de otimização. Ou seja, ela é uma fórmula a ser calculada a fim de saber se a solução está ou não convergindo para o ótimo.

Pode-se citar diversos exemplos de algoritmos de IA com heurística, a fim de exemplificar a explicação. Algoritmos genéticos utilizam uma população de amostras. Elas serão misturadas obedecendo a uma métrica de porcentagem de pai e mãe e quantos cruzamentos devem ser feitos, ou seja, a heurística. Assim, algum objetivo, relativamente aleatório, será alcançado. O A*, lê-se ”a estrela”, utiliza uma heurística de encontrar caminhos mais curtos, mas leva em consideração todos os caminhos obtidos, e não só o melhor já encontrado. Outro exemplo a têmpera simulada, que também revisita as soluções anteriores, mas cuja heurística é diferente da do A*. Existem muitos outros, estes são somente alguns exemplos.

Alguns problemas, no entanto, não puderam ser bem resolvidos com estes algoritmos. O problema do caixeiro viajante [Lawler et al. 1985], por exemplo, é um problema de otimização em que o caixeiro deve visitar todos os nós da rede, apenas uma vez cada, da melhor maneira possível, ou seja, utiliando o menor caminho entre todos. Os algoritmos

supra-citados, assim como outros da mesma época, não conseguiam resolver bem este problema. Então, em 1991, Marco Dorigo [Dorigo et al. 1991] utilizou, pela primeira vez, inteligência de enxame para resolver este problema.

A inteligência de enxame é uma variação de sistemas multi-agentes. Ela parte do pressuposto do funcionamento de um enxame. Várias criaturas com inteligências rudimentares conseguem realizar tarefas complexas pela união de seus esforços. Assim acontece com todos os insetos sociais, como formigas, cupins, abelhas, dentre outros. Na resolução do problema do caixeiro viajante, uma colônia de formigas foi utilizada.

Uma colônia de formigas gira em torno, assim como a maior parte do reino animal, de sobrevivência e continuação da espécie. No âmbito da utilização de tal comportamento computacionalmente, leva-se em conta somente a captura de alimento. Para tal, as formigas, quando saem a procura de alimento, deixam trilhas de feromônios, substâncias químicas captadas por suas antenas, para guiar as outras formigas pelo caminho. Quanto mais formigas andarem pelo caminho, ou seja, quanto mais comida houver ali, mais forte a trilha deixada, pois cada formiga aumenta-a com seus próprios feromônios. As trilhas pouco usadas, no entanto, serão dissipadas pelo vento e, consequentemente, deixarão de existir se não forem reforçadas mais que dissipadas.

O algoritmo de otimização baseado em formigas utiliza uma heurística, dependente do problema abordado, em conjunção com os feromônios. Assim, consegue-se uma solução mais robusta e versátil, conseguindo resolver problemas antes insolúveis.

O sistemas de colônias de formigas, no entanto, não foram criados para solucionar o problema do caixeiro viajante. Tal sistema biológico já existia, pelo menos, desde 1983 [J. L. Deneubourg and Verhaeghe 1983], uma data anterior à do problema do caixeiro viajante, 1985. Anteriormente ao surgimento desta analogia a um sistema biológico, alguns artigos já tratavam problemas de modo semelhante, visto ser a estratégia de sistemas multi-agentes bem anterior. A abordagem tratada na maioria dos artigos seguintes, no entanto, foi desenvolvida por Marcos Dorigo e seus colegas, em 1991, com seu ACS (*Ant Colony System*) [Dorigo et al. 1991].

4.3.1 Utilizações de sistemas de formigas

Antes de ser aplicado ao problema do caixeiro viajante, esta abordagem parece ter sido pouco utilizada. Até o ano de 1990, poucos foram os artigos com esta abordagem para a solução de algum problema. A partir desse ano, houveram muitas publicações. De fato, depois de aplicada a este problema por Dorigo, ela foi aplicada em muitos outros problemas, como descrito anteriormente. Além disso, surgiram várias aplicações diferentes para

o caixeiro viajante [Dorigo and Gambardella 1996, Dorigo and Caro 1999, Colorni et al. 1992, Shtovba 2005, Stutzle and Hoos 1997, Gambardella and Dorigo 1996].

Além do problema do caixeiro viajante, ele foi aplicado a outros problemas de otimização, como o problema de ordenação sequencial [Gambardella and Dorigo 2000], o problema dos missionários [Parunak and Brueckner 2000], problemas de subconjuntos [Leguizamon and Michalewicz 1999], caminho hamiltoniano [Wagner and Bruckstein 1999], roteamento veicular, coloração em grafos, busca de espaços contínuos [Bonabeau et al. 1998], dentre muitos outros. Assim, pode-se notar o sucesso desta abordagem, aplicada, eficientemente, a vários problemas diferenciados.

Os problemas supra-citados restringem-se ao escopo de grafos. A técnica dos sistemas de formigas foi levada mais além, a outras áreas. As colônias de formigas foram aplicadas às redes, tanto para o roteamento [Bonabeau et al. 1998, Caro and Dorigo 1997, Subramanian et al. 1997b] quanto para balanceamento de carga [Schoonderwoerd et al. 1997]. A utilização de colônias de formigas em redes consiste em fazer delas, pacotes. Elas, então, trafegam pela rede a fim de cumprir com sua função. Nos casos citados acima, estas funções seriam o balanceamento de carga ou o roteamento dos pacotes pela rede. À medida que transitam pela rede, as formigas deixam seu feromônio pelo caminho percorrido, auxiliando, como descrito abaixo, no roteamento dos pacotes, ou na função para a qual a colônia fora designada.

Em [Schoonderwoerd et al. 1997], além de balanceamento de carga, também há o roteamento dos pacotes pela rede. Além disso, este foi o primeiro algoritmo a explorar completamente a vantagem do descentralização da colônia de formigas. Nele, o roteamento é feito por meio de tabelas de roteamento probabilísticas, construídas a partir da deposição feromônica na rede. Quanto mais feromônios são colocados em um certo caminho, maior a probabilidade impressa na tabela de roteamento para a passagem do pacote por este caminho. A probabilidade impressa na tabela pelos feromônios é calculada de acordo com o congestionamento do caminho e a distância entre o nó atual e o nó originador da formiga. Assim, quanto mais distante do nó do qual fora enviada, menos feromônio a formiga deixará. Em [Bonabeau et al. 1998], uma modificação do algoritmo anterior é feita. Utiliza-se conceitos de programação dinâmica a fim de calcular esta distância da formiga por meio de uma "idade". O AntNet [Caro and Dorigo 1997] usa estratégia semelhante ao primeiro algoritmo descrito. Suas diferenças consistem em três pontos. A depeito da utilização da tabela de roteamento probabilística, há uma probabilidade de desconsiderar a tabela e atribuir a mesma probabilidade a todos os vizinhos, repesentando a aleatoriedade de escolha de caminhos. Esta característica permite a fuga de mínimos locais, ou seja, concentrar-se em um único caminho por causa dos crescentes feromônios e

esquecer-se dos outros. Outra diferença está na atualização das tabelas, efetuada por uma formiga mandada de volta pelo caminho, e não pela original. A última diferença consiste na introdução de uma nova estrutura, uma lista cuja função é guardar a memória da formiga, utilizada para o retorno até o nó inicial. Em [Subramanian et al. 1997b], o autor também fala sobre tabelas de probabilidade igualitárias e sobre os algoritmos utilizados no AntNet para a manutenção das tabelas e das listas.

Todas as propostas anteriores tratavam de roteamento, tratando, também, o balanceamento de carga. Algumas outras propostas, no entanto, objetivam tratar os parâmetros de QoS da rede em conjunção com o roteamento. QoS (*Quality of Service*, ou qualidade de serviço) trata parâmetros da rede como atraso, *jitter* (variação do atraso), custo, dentre outros, a fim de poder garantir certas características, como tempo de entrega dos pacotes, taxa de perda, vazão, dentre outras. Em [Subing and Zemin 2001], o roteamento levando em conta os parâmetros de QoS é utilizado. Assim, diferentemente dos anteriores, ao invés de coletar informações somente sobre o balanceamento de carga, as formigas o fazem com o estado da vazão, atraso, custo, distância, perda, dentre outras. Tais informações sobre o comportamento da rede também serão levadas em consideração no momento do roteamento, se um tipo de tráfego necessitar delas. Por exemplo, se uma aplicação de vídeo for ser transmitida pela rede, ela deve passar pelo canal com menor atraso e *jitter* possíveis, visto a necessidade do vídeo de ser transmitido em tempo real, sem paradas.

As redes de sensores, como anteriormente explicado, são uma especializações das redes ad-hoc. Por elas terem certas particularidades, dá-se a elas uma nomenclatura diferenciada. Por sua limitada capacidade energética e computacional, as soluções de redes de sensores são, em sua maioria, diferentes das de redes ad-hoc. Assim, estas soluções não foram incluídas na sessão de redes de sensores, e algumas delas serão tratadas aqui.

Há diversas soluções para a problemática energética em redes ad-hoc. Apesar de estas redes, diferentemente das de sensores, não terem a energia como uma preocupação inerente, algumas redes a têm. Dentre estas soluções está o ARAMA (*Ant Routing Algorithm for Mobile Ad-hoc Networks*) [Hussein and Saadawi 2003, Hussein et al. 2005], é um algoritmo de roteamento para redes ad-hoc baseado em formigas. Assim como os anteriormente discutidos, ele utiliza tabelas de roteamento probabilísticas para o roteamento, sendo uma ponte entre os algoritmos para redes previamente apresentados e os de redes ad-hoc. No [Saha and Pathak 2006], a diferença para os anteriores está somente em um peso atribuído aos caminhos encontrados. Em [Srisathapornphat and Shen 2003], a conservação energética é realizada por meio do desligamento esporádico da interface de rádio dos nós. Como já explicado, a interface de rádio de um nó é o dispositivo mais

energeticamente custoso do nó. Tal desligamento é feito utilizando as informações das formigas, sem atrapalhar o roteamento ou seccionar a rede.

Como uma variação desta estratégia, em [Shuang et al. 2007], faz-se roteamento aumentando o tempo de vida da rede ad-hoc. Para conseguir isto, as formigas recolhem informações sobre a bateria restante em cada nó, assim como sobre a capacidade de envio e contagem de saltos. Utilizando estas métricas, é possível construir uma tabela de roteamento probabilística invertida a fim de saber qual o melhor próximo passo para o pacote roteado. A tabela é invertida, pois, quanto mais formigas tiverem andado por um caminho, menos energia os seus nós tendem a dispor. Além disto, neste algoritmo, as formigas não precisam guardar o caminho percorrido, economizando energia no seu envio, recepção e processamento.

No AEADMRA (*Ant-based Energy Aware Disjoint Multipath Routing Algorithm*) [Wu et al. 2007], a estratégia é bem diferente. Como especificação de destino, ao invés da tabela de roteamento probabilística, temos uma localização espacial exata, obtida por um GPS ou similar. A rede é dividida em um grades, ou *grids*, semelhante a um grupo, ou *cluster*. Somente o cabeça da grade tem as informações de roteamento. As formigas são utilizadas para criar os melhores caminhos minimalmente disjuntos, ou seja, com o menor número possível de nós em comum, entre os cabeças das grades.

4.3.2 Por que Utilizar as Colônias de Formigas?

Tendo-se mostrado tais utilizações dos formigueiros, pode aparecer uma pergunta. Por que utilizar as formigas? Há vários bons motivos pelos quais este método deve ser utilizado, alguns encaixando-se melhor em certas áreas se comparados a outros.

Primeiramente, como explicado no começo desta sessão, este é um método inovador. Para a Inteligência Artificial, cujo foco sempre fora a heurística em seus algoritmos, a introdução de um fator extra, o feromônio, certamente fora decisivo. A prova desta afirmação está nas soluções de problemas antes insolúveis por sua complexidade, como o problema do caixeiro viajante, além de outros já citados. Com a introdução dos feromônios, um novo fator probabilístico e de aprendizado somava-se à heurística existente, aumentando a capacidade de resolução de um algoritmo.

Além disto, os algoritmos baseados nesta abordagem apresentam, por natureza, duas características importantes e fundamentais a certas aplicações. São descentralizados, pois cada formiga, assim como na Natureza, cria uma pequena parte da solução, utilizando seus feromônios. Assim, quando todas estas pequenas partes são postas juntas, uma solução inviável de ser alcançada por uma só formiga, com sua pequena capacidade de

processamento, é obtida. Assim, nenhum comando central é necessário. As formigas, por si mesmas, comunicam-se e obtêm a solução do problema. A segunda característica é a dinamicidade. A mudança é uma constante na Natureza, é a provedora do esquilíbrio. Assim, uma colônia está sempre em mudança, seja da formigas mortas e nascidas, dos caminhos até a comida ou da própria localização do ninho, se for necessário. Assim, se os caminhos são impossíveis de serem atravessados por algum motivo, como um desastre natural, novos caminhos são abertos. Se a comida acaba em certo local, as formigas param de frequentá-lo, deixando o vento dispersar o caminho feromônico até sua inexistência.

As características supra-citadas são as responsáveis pela aplicação deste modelo algorítmico às redes. A descentralização é uma boa característica para um algoritmo de roteamento em uma rede, visto a escalabilidade inerente às redes. Utilizando um método de roteamento centralizado, o elemento responsável torna-se, além de concentrador do método, um ponto de falha da rede. Além de aumentar o tráfego de mensagens de controle na rede, pois será este ponto a controlá-la, ele impede seu crescimento, pois nós muito distantes deste elemento não seriam bem controlados. Além disto, se esse elemento, por algum motivo, sair da rede, ela, por inteiro, parará. Além da descentralização, a dinamicidade também é importante. Se a rede não for capaz de reconfigurar suas rotas, a perda de um dos nós da rede irá fazê-la colapsar por inteiro. Assim, é necessário um algoritmo independente da topologia da rede, tanto no início de sua utilização quanto no decorrer de seu tempo de vida.

Pelo exposto até o momento, é possível notar a adequação das características dos algoritmos baseados em colônias de formigas às da autonomicidade.

4.3.3 Formigas em Autonomicidade

Com o manifesto da IBM em 2001, as portas abriram-se para o desenvolvimento de toda sorte de soluções para os problemas francamente apresentados. Em pouco tempo, a inteligência de enxame passou a figurar entre as fileiras de recentes e inovadoras soluções para os problemas criados.

Muitas destas soluções já forma citadas e explicadas sob o crivo das redes de sensores. Muitas soluções autonômicas foram desenvolvidas, utilizando as colônias de formigas, a fim de solucionar os problemas de redes de sensores, em especial o energético. Tendo-se em vista a extensa explicação desta tema em particular na primeira sessão deste documento, ele não será reapresentado aqui.

Ainda mais profundamente tratado foi o assunto dos sistemas multi-agentes. Como já visto, a formiga é um agente, e um sistema de colônia de formigas é, de fato, um sistema

multi-agentes. Além disto, diversas propostas com o fim de soluções para a problemática energética de redes de sensores foram mostradas, assim como as de formigueiros. Assim, como para o assunto anterior, seria desperdício reapresentá-las aqui.

Observando-se tais fatores, pode-se crer no desenvolvimento de uma solução para o problema energético de redes de sensores baseado em sistemas de colônias de formigas. Algumas destas soluções já existem, mas nem todas as possibilidades foram exploradas ainda. Com este pensamento, o protocolo BiO4SeL foi desenvolvido. Ele será apresentado no próximo capítulo.

Capítulo 5

O Protocolo BiO4SeL

Neste capítulo, serão descritos os aspectos pertinentes à proposta da dissertação. Será explicado, primeiramente, o problema como um todo e o foco específico a ser tratado. Posteriormente, serão tratados os aspectos de resolução a serem tratados ou não, o porquê e a forma de escolha do tratamento do problema. A seguir, a proposta de resolução será relatada, seguida por alguns dos trabalhos relacionados disponíveis na área, juntamente com o método de comparação.

5.1 Descrição do Problema

Uma Rede de Sensores é formada, comumente, por vários sensores gerenciados por uma Estação Base. Cada sensor dispõe de uma bateria de pequena capacidade, sendo impraticável substituí-la, dado o preço destes dispositivos. Quando ela acaba, o próprio sensor é substituído. Quanto ao funcionamento, eles captam informação ambiente e encaminham estas informações para a Estação Base. Ela processa-as, e consegue os dados desejados.

Para enviar informação à Estação Base, o método mais clássico é todos os sensores alcançarem, com sua cobertura, a Estação Base. Este método é muito dispendioso, e inviabiliza Redes de Sensores duradouras em larga escala. O próximo algoritmo mais simples consiste em repassar os pacotes pela rede até alcancá-la. Utilizando-se tal estratégia, será calculado um algoritmo de menor caminho entre os nós da rede e a Estação Base, levando-se em consideração a quantidade de saltos. Um punhado de nós, depois da aplicação do menor caminho, estarão a um salto da Estação Base.

Assim, todos os pacotes passarão por estes nós, enquanto somente um pacote passará pelo nó mais longínquo da Estação Base. Quanto mais pacotes passarem por um nó, mais bateria ele gastará. Fácil ver a proporcionalidade inversa entre a distância de um nó à

Estação Base e a bateria gasta por ele.

Quando estes nós tiverem sua bateria totalmente gasta, haverá uma ruptura da rede. Tais nós não mais enviarão capturas ambientais e nem repassarão pacotes vindos de outros nós. Assim, esta deve ser uma situação a ser evitada. De fato, este é um dos mais tratados problemas de Redes de Sensores.

Obviamente, este é apenas um exemplo básico, utilizando caminho mínimo como a estratégia de roteamento. Como já descrito, outras estratégias existem, e também apresentam seus problemas.

Tendo em mente a existência deste grave problema energético nas Redes de Sensores, os pesquisadores começaram a preocupar-se em resolvê-los. Diversos trabalhos já foram publicados com as mais diversas soluções. Inicialmente, a solução proposta foi diminuir o gasto energético dos sensores. Se os sensores com maior gasto passassem a consumir menos suas baterias, teriam maior vida útil. Como comprovado posteriormente, somente esta estratégia mostrava-se insuficiente, pois algum nó da rede continuava a acabar com sua energia antes do restante. Tal situação continuava a preocupar os estudiosos, pois a cobertura da rede, a partir do momento da queda do primeiro nó, tornava-se incompleta. Assim, seriam necessários novos métodos a fim de aumentar o tempo de vida da rede como um todo.

Há soluções tratando este problema em específico. A elas foi dado o nome específico de “aumento do tempo de vida”, ou seja, aumentar o tempo até o primeiro nó ficar sem energia. Esta estratégia pode ser, por exemplo, variar as rotas de envio, a fim de não gastar sempre a bateria dos mesmos nós no caminho.

A maioria destas soluções, entretanto, carece de autonomia. Apesar de algumas utilizarem informações locais em detrimento das globais, os aspectos da autonomia são deixados de lado. Assim, elas não conseguem auto configurar suas rotas nem auto otimizá-las, melhorando-as ou as modificando. Algumas geram rotas estáticas, não conseguindo repará-las frente a acontecimentos inesperados.

Algumas dessas soluções, entretanto, aplicam tais conceitos. Nenhum dos algoritmos autônomos, entretanto, utiliza as Colônias de Formigas, apesar de existirem soluções utilizando este algoritmo. Esta solução, proveniente do estudo e observação da Natureza, traz em si muitos dos conceitos autônomos. As Colônias melhoram constantemente suas rotas, e são capazes de se auto organizar após falhas.

Tais conceitos, no entanto, não são todos os necessários para um protocolo de roteamento autônomo. Algumas outras características, como a auto configuração completa das rotas, são necessárias. A proposta deste documento é criar um protocolo autônomo de roteamento baseado em Colônias de Formigas como objetivo de otimizar o tempo de

vida da rede. Esta clase de algoritmos será utilizada por sua já comprovada eficiência em aprimorar o tempo de vida. Além disso, será aplicada a autonomia, com o objetivo de melhorar a solução, além de deixá-la mais robusta e aplicável.

5.2 Aspectos do Problema

Tendo em vista a extensão da bibliografia tratada no artigo, tem-se diversas opções a tratar para construir uma nova solução para o problema energético das Redes de Sensores. A fim de justificar a escolha feita, aqui serão apresentadas, de modo resumido e comparativo, as diversas opções em cada âmbito do problema. Com este fim, a seção foi subdividida. Assim, as várias partes do problema podem ser abordadas separadamente, de modo mais claro e objetivo.

Alguns outros aspectos comparados estão no Apêndice B. Aqueles julgados mais relevantes para o trabalho estão dispostos a seguir.

5.2.1 Aspectos Concernentes Diretamente às Redes e à sua Composição

- **Tipo de Rede de Sensores**: Homogêneas ou heterogêneas?

  As redes homogêneas têm, por principal característica, ser composta por sensores de mesma capacidade de bateria, funcionalidade e restrições de *hardware*. Ao contrário, nas heterogêneas, os nós possuem estas funcionalidades diferentes.

  Geralmente, este tipo de estratégia é utilizada em soluções necessitando de consumos de energia e capacidades diferenciadas [Kim et al. 2006,Mann et al. 2005]. Assim, geralmente é utilizada como um artifício de manutenção energética no agrupamento. Sendo utilizados os nós de maiores pontecialidades como nós cabeça, pode-se aumentar seu tempo de vida, haja vista o rápido decrescimento de sua energia pelo encaminhamento das mensagens do grupo inteiro. Além disto, os nós cabeça deteriam as funcionalidades de gerenciamento de grupo implementadas, dado a seu maior capacidade computacional.

  Nas redes homogêneas, por outro lado, todos os nós podem ser cabeça e devem dispor das funcionalidades. Assim, temos um alargamento de opções, pois pode-se, a fim de aumentar o tempo de vida da rede, revezar o cabeça pela rede inteira, mudando os nós consumidos [Akyildiz et al. 2002b].

Como comprovado pelos autores dos artigos previamente citados, há melhoria no desempenho devido às redes heterogêneas. Apesar disso, a maioria dos trabalhos ainda utiliza as homogêneas em suas soluções. O maior argumento a este favor é, justamente, o ganho de generalidade. Nem todas as redes disporão destes sensores de alto nível. Uma das maiores vantagens da Rede de Sensores é justamente seu baixo custo, impossível se o custo de cada sensor for alto, dada sua relativamente rápida necessidade de reposição. Sua reposição poderá tornar-se demasiado custosa, tendo-se em vista a impossibilidade da mudança de bateria dos sensores. Assim, esta proposta basear-se-á em redes homogêneas, embora possa ser utilizada, sem garantia de melhoria extra de desempenho, em redes heterogêneas.

- **Funções de sensoriamento**: Levar em consideração o gasto energético?

  Como visto em [Slama et al. 2006], o gasto energético da recepção dos pacotes deve ser levado em consideração nas simulações, assim como o gasto básico do sensor, o de captura de informação, de processamento de pacote e informação e o de envio. Assim, todos esses gastos devem ser utilizados a fim de termos uma acurada medição do gasto energético da rede.

  No NS-2, estão implementados os gastos de envio e recebimento de pacotes, assim como o gasto relativo ao funcionamento comum e em modo dormente do sensor. Os gastos de sensoriamento e processamento inexistem nesta implementação.

  Para fins comparativos, no entanto, o gasto com sensoriamento é desnecessário. Para comprovarmos essa afirmação, basta lembrarmos a função principal de uma Rede de Sensores, obter informações do meio. Assim, independentemente do protocolo, quando informação for enviada, ela terá sido sensoriada do meio, e haverá um gasto energético, sempre igual para sensores e ambientes iguais, associado a este envio. Concluindo, pode-se imaginar este gasto unido ao gasto de transmissão, tornando sua descrição redundante para comparações.

  Também para comparação, o gasto de processamento pode ser ignorado sem muitos problemas. Assim como ocorre ao sensoriamento, a aplicação a utilizar o protocolo gastará a mesma quantidade de energia com processamento independente do roteamento. Além disso, de acordo com os maiores *surveys* da área, o gasto de processamento é milésimamente pequeno se comparado ao gasto de envio e recepção de pacotes. Assim, poucas instruções a mais ou a menos no código de roteamento influem muito pouco no nível final da bateria do sensor.

  Assim, para este trabalho, assim como para tantos outros, somente serão levados

em consideração os gastos energéticos de recepção, transmissão e o básico de um sensor ligado.

- **Quantidade de estações base**: Levar em consideração somente uma Estação Base, ou *sink*?

  A quase totalidade dos artigos sobre o tema vistos e estudados não trata este tema. Sua importância para redes de alta escalabilidade é negligenciada. Em tais redes, a possibilidade de ter-se mais de uma Estação Base, comunicantes, a fim de trocarem as informações recebidas, melhorará o rendimento do algoritmo, pois ele poderá escolher a Estação Base mais próxima.

  Como em outros trabalhos da área, este protocolo tratará o aumento do quantidade de estações base como possível. Testes não foram efetuados com mais de uma Estação Base, mas, pelo método de escolha de rota do algoritmo, é possível acrescentá-las, respeitando as fases devidas. Estes testes estão previstos como trabalhos futuros.

- **Dinamicidade da rede**: Deve ou não ser levada em consideração?

  Quando fala-se sobre dinamicidade em uma Rede de Sensores, isto significa nós movendo-se, saindo e entrando na rede. Muitos artigos tratam estes fatores, enquanto outros simplesmente ignoram-nos ou tratam-nos de maneira incompleta.

  Quanto à mobilidade dos nós, ela não é tão importante neste contexto. Redes adhoc, por sua própria definição, tendem a ser móveis. Seus dispositivos componentes tendem a mover-se pela extensão do sinal enviado pela central da rede, pois, em muitos casos, eles são pequenos dispositivos móveis. Celulares, PDA's, dentre outros, tendem a formar estas redes, movendo-se dentro do alcance. Em Redes de Sensores clássicas, no entanto, o nós tendem a ficar parados e desempenharem uma função no local onde estiverem. Assim, essa tendência à estática, na maioria das Redes de Sensores, isenta-as de tratamentos mais específicos quanto à mobilidade, apesar de Redes de Sensores móveis existirem.

  Entrada e saída de nós da rede, no entanto, é algo comum às duas redes supracitadas. Ambas devem lidar com os casos de mudança da topologia da rede, diferentes da mobilidade por sua pontualidade. Os nós saindo da ou entrando na rede, o fazem uma vez, ao contrário da mobilidade, pois ela permite rápidas entradas e saídas, acarretando mais sobrecarga de identificação na rede.

  Deve-se lembrar da proposta da solução. Aqui, propõe-se um método autonômico, utilizando formigas, para aumentar o tempo de vida de uma Rede de Sensores.

Sendo assim, não se pode ignorar a dinamicidade da rede, visto ser a auto-configuração uma das quatro características fundamentais de um sistema autonômico. Quanto à entrada e saída dos nós da rede, é mais simples de ser levada a cabo. Como já citado, elas não tendem a ocorrer com frequência, acarretando uma sobrecarga com a qual as formigas podem lidar, haja vista a quantidade de soluções na literatura sem eficiência comprometida por este aspecto. Quanto à mobilidade, ela não será tratada, como especificado em vários artigos. A constante mudança de vizinhança, ocasionada por uma alta mobilidade dos nós, pode ser um problema para a atualização das tabelas de roteamento.

Assim, serão tratadas as entradas e saídas dos nós da rede, mas nenhuma mobilidade. Ainda assim, a proposta inclui os elementos necessários para tratá-la, mas sem teste algum.

- **Topologia da rede**: Plana, hierárquica ou baseada em localização?

  Na topologia plana, inexiste qualquer tipo de hierarquia. Todos os nós da rede são iguais, desempenhando exatamente as mesmas funções, menos as estações base. Assim, possibilita-se a formação de melhores caminhos únicos dos nós às estações base e de caminhos múltiplos, abrindo um enorme leque de soluções para as problemáticas de Redes de Sensores. Estes caminhos são formados em tempo de execução do algoritmo, e podem ser modificados a cada envio de informação, tornando-a extremamente dinâmica, apesar do gasto extra para se manter as tabelas de roteamento. Em redes dinâmicas, com nós movendo-se constantemente, esta não é uma estratégia recomendável, dado o gasto extra de mantunenção das tabelas.

  A topologia hierárquica consiste, basicamente, de grupos e árvores. Ela utiliza-se de uma infra estrutura para a concentração de dados em um único nó de cada grupo, a fim de acabar com redundâncias e economizar energia. As estratégias baseadas nesta topologia levam vantagem da estrutura para otimizar seu roteamento, em troca de uma configuração inicial custosa. Assim, métodos simples de agregação ou complexos de sensibilidade energética dos grupos, ao invés de nós separados, conseguem economizar a energia da rede como um todo.

  Para a baseada em localização, os nós conhecem sua localização geográfica por meio de GPS. Nas abordagens mais antigas, pequenos receptores são colocados em cada sensor, a fim de eles poderem receber sua localização via satélite. Nas mais recentes, utiliza-se protocolos de disseminação da localização. Apesar do gasto extra pela recepção e atualização da localização, a informação de distância entre

os nós permite certas funcionalidades ao protocolo de roteamento. De maneira simples, permite minimizar a abrangência da área de sensoriamento, promovendo uma economia da energia dos nós.

Dadas as características de cada uma destas topologias, é claro notar a inexistência de melhor ou pior, no geral. Cada uma delas adapta-se a um caso específico melhor, se comparada a outra. Todas têm suas vantagens e desvantagens. Para uma aplicação de formigas, a mais conveniente, na visão do autor deste documento, assim como na visão dos autores de diversos trabalhos semelhantes, é a topologia plana com uma hierarquia de quantidade de saltos até a Estação Base. Esta quantidade de saltos, como utilizada em outros protocolos, é localmente mantida depois de conseguida, não consome muita energia da rede e é uma forma segura e testada de conseguir rotas na rede.

5.2.2 Aspectos do Algoritmo de Roteamento

- **Conhecimento prévio da rede**: Assumir ou não?

  Como descrito desde o início, esta é uma abordagem autonômica. Ela deve ser dinâmica, e sem qualquer conhecimento da rede como um todo, para aprimorar sua escalabilidade.

- **Conhecimento das estações base**: Prévia ou em tempo de execução?

  No algoritmo ARAMA [Hussein and Saadawi 2003, Hussein et al. 2005], parte-se do pressuposto de os nós já conhecerem os possíveis destinos, pois somente podem encaminhar formigas com este conhecimento prévio. Diferentemente do ARAMA, o ARA [Gunes et al. 2002] utiliza formigas em inundação. Ainda assim, somente a Estação Base previamente conhecida responde com uma formiga de retorno.

  Este conhecimento funciona bem em simuladores, como o NS, e utilizando-se somente uma Estação Base. Ao criar-se um fluxo de dados, neste simulador, você designa fonte e destino, simulando um conhecimento prévio, por parte do nó, das estações base da rede. Assim, tendo-se somente um destino possível, basta criar o fluxo e simular a ação da camada de roteamento. Tendo-se várias estações base, seria necessária uma abordagem diferenciada.

  A ideia, para a proposta, é torná-la completamente autonômica. Assim, ela não deveria, previamente, conhecer as estações base da rede. Ela deveria, pelo contrário, ter algum método de descoberta das estações base da rede sem um prévio conhecimento. Como já mostrado, qualquer tipo de informação sobre a rede limita a

sua expansibilidade, e não faria sentido dispor de várias estações base para poucos sensores.

Muitos artigos trazem soluções para este contexto. Uma solução possível é as estações base enviarem formigas por *broadcast* para os sensores, avisando de sua localização. Outra possível solução seria os nós, ao serem ligados, enviarem formigas em *broadcast* e as estações base responderem, enviando suas localizações. As duas se parecem muito, podendo ser utilizadas igualmente. Em ambas, o TTL dos pacotes iria impedir os nós de conectar-se a estações base muito distantes.

- **Quantidade de caminhos**: Único ou múltiplos?

  Muitos algoritmos de roteamento utilizam-se de estratégias cujo foco é calcular os melhores caminhos e utilizá-los durante o funcionamento da rede. Como previamente observado, tal estratégia não cabe no contexto de uma Rede de Sensores, pois tais caminhos estáticos iriam desgastar a bateria dos nós rapidamente.

  Assim, soluções alternativas surgiriam. Uma destas é a inteligência de enxame. Por sua própria natureza, esta estratégia cria diversos caminhos possíveis, escolhendo entre eles por simples probabilidade. Assim, se um caminho cessar de existir, por término da bateria ou problemas no sensor, a rede é capaz de se auto-adaptar e remodelar as rotas.

  Para a proposta aqui apresentada, estes diversos caminhos serão implementados. Utilizando-se a tabela de roteamento probabilística, haverá sempre uma possibilidade de alimentar qualquer caminho. Ao cair um desses caminhos, outros podem ser utilizados. Além disto, eles irão crescer seus feromônios, dado a inexistência de formigas passando pelo nó agora inexistente.

- **Especificação do destino**: Qual utilizar?

  Como já previamente descrito, há três elementos básicos em todo protocolo de roteamento. São eles a especificação de destino, os objetivos de roteamento e a estratégia de roteamento. A especificação do destino trata de como os nós da rede são especificados um para o outro. Objetivos de roteamento são o parâmetro de escolha da rota. A estratégia de roteamento é como a informação chegará de seu remetente até seu destino.

  Os tipos de especificação de destino são diversos. Tabela de vizinhança, árvores, especificação geográfica, *mesh* [Yang and Ho 2006], dentre outros métodos, podem ser utilizados a fim de informar onde encontram-se os nós da rede. Os dois

primeiros, no entanto, foram os mais usados na literatura abrangida, com suas ramificações.

Algumas soluções utilizando formigas, como em [Selvakennedy et al. 2007, Zhang and Huang 2006b], utilizam-se de árvores. Algumas outras, em redes ad-hoc [Hussein and Saadawi 2003, Hussein et al. 2005, Saha and Pathak 2006, Srisathapornphat and Shen 2003], herdando a solução das redes comuns [Bonabeau et al. 1998, Caro and Dorigo 1997, Subramanian et al. 1997b, Schoonderwoerd et al. 1997], utilizam tabelas de roteamento probabilísticas. Tais tabelas são produzidas pelos feromônios deixados pelas formigas, indicando os melhores caminhos, ou seja, os mais utilizados. Além dos feromônios, estas soluções utilizam informações de carga no cálculo das probabilidades a fim de melhorar o desemprenho da rede. Tais soluções provaram sua eficiência.

Uma solução mais recente utiliza tabelas de roteamento probabilísticas inversas [Shuang et al. 2007]. Também designada para redes ad-hoc, seu objetivo é utilizar, além dos feromônios e balanceamento de carga, informações energéticas sobre a rede, a fim de melhorar o seu tempo de vida. Assim, a ideia é impedir fluxos por onde a energia esteja baixa na rede, preservando os nós com menor energia.

Para esta proposta, tabelas de roteamento probabilísticas inversas serão utilizadas. A razão da escolha é a sua adequação às redes ad-hoc e às de sensores. Nas Redes de Sensores, estratégias como essas costumam ser vistas com maus olhos pelo fato de haver sobrecarga para a manutenção da tabela de roteamento. No protocolo apresentado, a troca de pacotes de atualização é praticamente desnecessária. Assim, acredita-se ser possível utilizar tais tabelas em Redes de Sensores, utilizando uma solução inovadora e de desempenho comprovado por outros artigos. Devido ao sucesso de utilização de formigas nas Redes de Sensores, tal proposta deve render bons resultados.

- **Objetivo de roteamento e estratégia de roteamento**: Qual utilizar?

  Como previamente constatado, estas duas informações costumam confundir-se. A maioria dos trabalhos na área tratam-nas com uma única solução, sem fazer distinção entre o objetivo e a estratégia de roteamento. De fato, o objetivo é o modo de fazer a escolha para o próximo nó do roteamento, ou seja, os elementos utilizados para escolher entre um ou outro nó. A estratégia é como a informação viajará do remetente até o destino.

  Por serem muito próximos, os conceitos anteriores, em alguns casos, não são dis-

tintos. Os algoritmos de agrupamento, por exemplo, mesclam-nos. Eles decidem quem será o próximo pela mesma métrica do roteamento, ou seja, enviar para os cabeças e, deles, para a Estação Base.

Em alguns artigos, esta separação é bem clara. Nos artigos sobre formigas citados no tópico acima, o objetivo de roteamento são as informações, diferentes em cada artigo, formadoras da probabilidade nas tabelas de roteamento. A solução é uma espécie de estratégia gulosa probabilística, como o caminho mínimo, a cada salto, escolhendo, probabilisticamente, o melhor caminho, indicado pela maior probabilidade. Outro exemplo é o roteamento baseado em mapa energético [Al-Karaki and Al-Mashaqbeh 2007]. O mapa de energia da rede é somente o objetivo de roteamento, utilizado para a decisão de roteamento pelo algoritmo.

Na solução proposta, entretanto, elas não se confundem. A divisão é bem clara [Zhang and Huang 2006b], assim como nos algoritmos de formiga supra-citados. O objetivo de roteamento é dado pela probabilidade contida na tabela de roteamento. A estratégia, assim como acima, é uma versão de estratégia gulosa probabilística, mas escolhendo as maiores probabilidades ao invés do menor número de saltos. Este sistema será semelhante ao adotado em [Shuang et al. 2007].

- **Fases do algoritmo**: Quais e quantas utilizar?

  Os algoritmos clássicos de otimização utilizam duas fases. A primeira, de inicialização, é composta por troca de informações no intuito de formular um estado inicial para a solução. Por exemplo, se a solução é menor caminho, o algoritmo cria o menor caminho entre os nós nesta fase inicial. A segunda fase é a de manutenção, na qual a rota criada é mantida por algum método.

  Boa parte dos algoritmos de roteamento estudado seguem esse paradigma. Aqueles baseados em formigas utilizam-nas para criar e manter as rotas, ativamente. Nestes algoritmos, as formigas atualizam a tabela de roteamento, somente.

  Na solução proposta nesta dissertação serão utilizadas três fases. A primeira, de reconhecimento, será uma troca de *hellos*, a fim de todos os nós conhecerem seus vizinhos. Esta fase é, muitas vezes, ignorada pelos protocolos, mas é necessária. A segunda, de inicialização, tem um papel semelhante ao descrito acima. A diferença deste protocolo é o fato de esta fase ter, como protagonista, a Estação Base. Ela envia as formigas, e não os nós. Assim, a Estação Base encontra e cria caminhos para os nós, e não o contrário. Esta escolha foi feita baseada na vulnerabilidade percebida no ARAMA [Hussein et al. 2005]. Em um cenário com muitos nós, as

formigas deste protocolo tendem a perder-se sem encontrar a Estação Base, pois fazem uma busca cega por ela. A terceira e última fase, de manutenção, é a execução do algoritmo propriamente dita. Criadas as rotas, as tabelas de roteamento são mantidas nesta fase, e os dados, enviados e recebidos.

5.2.3 Aspectos das formigas

- **Cálculo dos feromônios**: Como fazê-lo?

  Classicamente, nas soluções, o cálculo do feromônio a ser acrescido ou retirado dos caminhos é efetuado no nó de destino, utilizando informações globais. Tais dados seriam coletados nos dados durante o caminho da formiga ou contidos em cada nó.

  Tais cálculos levam em conta muitos fatores. Muitas soluções utilizam-se somente da quantidade de saltos da formiga até atingir seu objetivo. Outras levam em conta outros parâmetros de QoS, como o tempo de vida do pacote, o atraso, dentre outros. Algumas ainda utilizam a carga de bateria dos sensores pelos quais passou. Poucas estratégias, no entanto, preocupam-se com a manutenção dos sensores funcionando, ou seja, uma distribuição igualitária dos níveis energéticos pela rede como um todo.

  Esta solução pretende manter os níveis energéticos em todos os nós da rede em um padrão mais igualitário possível durante toda a sua utilização. Esta solução não é completamente inovadora [Hussein et al. 2005, Shuang et al. 2007], mas a estratégia utilizada será calculada localmente, e sem necessidade do custo do caminho. Assim, as formigas de retorno serão desconsideradas para este uso, pois é seu principal papel, nos protocolos anteriores, atualizar a quantidade feromônica dos nós. Tenta-se, assim, reduzir o gasto energético da rede.

- **Formigas voltando**: Quanto às formigas, utilizar ou não formigas voltando para atualizar os feromônios?

  Façamos uma breve retrospectiva do aspecto biológico da coleta de alimentos de uma colônia de formigas. Cada operário, ao sair à "caça", verifica os feromônios deixados no ar por seus companheiros durante suas viagens. Ela escolhe e segue um caminho, reforçando sua quantidade de feromônios enquanto percorre o caminho. Caso não consiga encontrar comida e retornar, os feromônios são dissipados com o vento, sem serem reforçados. Assim, este caminho morre. Quanto mais comida em certo local, mais feromônios são depositados, e mais formigas o percorrem, perfazendo o processo autocatalítico.

Como previamente descrito, a inteligência de enxame foi amplamente utilizada em problemas de otimização. Nestes problemas, somente a formiga de retorno deposita feromônios no caminho ótimo encontrado. Assim, carregar a formiga de avanço com o caminho percorrido e depois enviar uma formiga de retorno por este caminho é uma solução bem comum neste contexto. Suas desvantagens, no entanto, são notórias. Além do tamanho crescente do pacote, há o congestionamento na rede, causado pelo constante trafegar de formigas pela rede.

Em Redes de Sensores, estas desvantagens tornam-se mais restritivas ainda. O tamanho do pacote é um problema, pois encaminhá-lo por vários nós da rede pode ser muito custoso, em termos energéticos da bateria, se o pacote for grande demais. Além disso, o congestionamento é um problema, tanto de bateria quanto de controle de congestionamento. O padrão do IEEE 802.15.4 [IEEE 2006] não prevê este controle. Assim, quanto menos pacotes trafegando, melhor.

Aqui, nota-se uma clara permuta de perda e ganho para a rede envolvendo os pacotes a serem enviados e a bateria dos nós. Por um lado, elas são necessárias para a atualização dos caminhos, dada a informação desta atualização ser, geralmente, global. Assim, elas precisariam obter informação em cada nó do caminho para calcular seu índice de atualização, ou seja, o quão bom ele é [Hussein et al. 2005]. Por outro lado, os gastos energéticos de envio são, aproximadamente, 1000 vezes maiores se comparados àqueles de processamento. Assim, é sempre preferível operações dentro do sensor a troca de mensagens.

A solução a ser desenvolvida aqui, no entanto, é completamente independente de qualquer índice global. Ainda assim, algumas poucas formigas de retorno ainda serão necessárias. As formigas de retorno serão necessárias somente para uma possível redução feromônica em um caminho a ser evitado, como um ciclo, por exemplo.

- **Tipo de pacote**: Formigas embarcadas nos pacotes (*piggybacking*) ou como pacotes separados?

  Na grande maioria das soluções vistas, as formigas, assim como o é nos algoritmos de roteamento clássicos, são pacotes separados. Chamados de pacotes de controle, eles têm a função de criar e manter rotas na rede. Eles podem ou não carregar informações. O mais importante, no entanto, é sua chegada e envio dos nós da rede.

  As formigas, de acordo com os artigos vistos e estudados, geralmente carregam informação em quantidade razoável. Elas carregam, pelo menos, o histórico dos

nós por onde passaram, possivelmente muitos em uma rede densamente povoada, e um índice de melhoramento do caminho. Assim, colocá-la junto ao pacote, como um outro cabeçalho ou na parte de dados, poderia ser muito custoso para o envio, sobrecarregando demais cada pacote.

Com a praticamente total abolição da necessidade constante de formigas de retorno, a informação do caminho percorrido pode ser retirada, tornando-a bem menor. Além disso, como iremos tomar como principal métrica a energia dos nós da rede, esta energia será medida em termos de chegadas e saídas de qualquer pacote, pois qualquer pacote diminui a carga da bateria do sensor. Assim, qualquer pacote terá a possibilidade de ser uma formiga, confundindo-se estes dois conceitos em um só.

Ainda assim, algumas formigas serão necessárias, como na fase inicial e para uma eventual sincronização de tabela de roteamento. Elas, no entanto, serão em número bem reduzido se comparado à quantidade de formigas enviadas em outros algoritmos.

Serão utilizadas, então, formigas como pacotes separados e embarcadas em pacotes de dados, cada uma com uma finalidade específica.

- **Descoberta de caminhos**: Usando inundação ou as formigas?

  Nos algoritmos mais tradicionais, a descoberta de caminhos na rede é feita utilizando-se a inundação. No Ant-Net e no ARA, ela é feita deste modo. Já no ARAMA, ela passa a ser feita por meio das formigas, controlando a sobrecarga de pacotes na rede. Mesmo sem enviar muitos pacotes para a rede, é possível, e mais econômico, encontrar os caminhos desejados.

  Esta solução é bem adaptada para a situação na qual os nós de destino da rede, as estações base, estão declaradas. Quando elas ainda são desconhecidas pelos nós, para onde enviar as formigas? Sendo este o cenário tratado, as formigas em inundação serão necessárias na fase de inicialização da rede, a fim de “descobrir” as estações base da rede.

  De fato, as estações base são as verdadeiras descobridoras, pois as formigas de inicialização partem delas. Assim, não se precisa de uma formiga de retorno para demonstrar a localização e caminho do nó até a Estação Base. Além disso, a estratégia do ARAMA, como provado pelos testes e comparações, não tem escalabilidade. Suas formigas tendem a perder-se antes de encontrar a Estação Base em cenários muito grandes.

Assim, resta somente saber quantas formigas a Estação Base precisa enviar para conseguir alcançar todos os nós da rede. Um número arbitrário será utilizado, baseado nas experimentações, por ser este o método comumente utilizado.

- **Tipos de formigas**: Quais utilizar?

  Os protocolos com formigas estudados até o momento as utilizam, geralmente, como criadores de rotas entre fonte e Estação Base. Neste caso, elas são de dois tipos, a de ida e a de volta, com parâmetros diferentes, dependendo do algoritmo. Além disso, geralmente utilizam `hellos`.

  Na solução apresentada neste texto, as formigas terão mais de dois tipos. Primeiramente, há dois tipos de `hello`, um para a fase de Reconhecimento (*ihello, initialization hello*) e outro para a de Troca de Dados. Ainda nesta fase, outras duas formigas serão utilizadas, com o objetivo de refazer vizinhanças perdidas. A formiga de ida, utilizada na fase de Inicialização de Rotas, estará presente, mas não estará a de retorno. Além dela, estarão presentes a formiga negativa, cuja função é diminuir a quantidade feromônica em um caminho ruim, e dois tipos de formigas embarcadas nos dados, a comum e um tipo especial de `hello`, utilizado com a função de diminuir a quantidade de pacotes trocados.

5.2.4 Comparação

A Tabela 5.2.4 mostra uma comparação entre os algoritmos baseados em formigas. Nela, ressalta-se as diferenciações estudadas para o problema e solução propostos. Assim, pode-se utilizá-la como um bom resumo para este tipo de solução em Redes de Sensores. A comparação com o algoritmo BiO4SeL é feita a partir dos parâmetros discutidos nesta seção. Ele será melhor descrito em seguida.

Tabela 5.1: *Comparação entre resoluções de roteamento em Redes de Sensores*

Métrica	ARA	ARAMA	BiO4SeL
Fase de incialização	Sim	Não	Sim
Descoberta e manutenção de caminhos	Inundação	Formigas	Misto
Suporte a diversas estações base	Sim	Sim	Sim
Implementação de diversas estações base	Não	Não	Sim
Formiga de retorno	Sempre	Sempre	Inicialização
Autonomicidade	Não	Não	Sim
Economia energética	Não	Sim	Sim
Suporte a nós movimentando-se	Inadequado	Inadequado	Inadequado (embora previsto)
Cálculo da tabela de roteamento	Formiga de retorno	Formiga de retorno	Qualquer pacote
Modelo de criação da tabela de roteamento	Reverso	Avanço	Avanço
Composição do feromônio	Saltos	Saltos e energia	Energia
Formiga de avanço	Inundação	*Unicast*	Inundação
Feromônio inicial de cada vizinho	Igual	Igual	Diferenciado

5.3 Protocolo BiO4SeL

A subseção anterior compõe-se de uma pormenorizada análise dos aspectos relevantes a uma solução de entrega de dados, na camada de roteamento, para Redes de Sensores. Foram explanados diversos pontos de interesse a serem comparados entre as propostas existentes e a desenvolvida nesta dissertação. Nesta subseção, será apresentado o *BiO4SeL* (*Biologically-inpired Optimization for Sensor network Lifetime*), uma proposta autonômica baseada em formigas para a resolução do problema de aumento do tempo de vida de uma Rede de Sensores.

Antes de prosseguir com o protocolo, deve-se mostrar como está disposta a tabela de roteamento probabilística, a ser utilizada neste trabalho. Ela tem o mesmo propósito das tabelas de roteamento convencionais, ou seja, guardar o próximo salto de um pacote transitando pela rede. Apesar disso, ela o faz de um modo diferente. Um exemplo é mostrado na Tabela 5.3.

Tabela 5.2: *Exemplo de tabela de roteamento probabilística para nó* `i`.

	v_1	v_2	...	v_{m_i}
d_1	$Q_i(v_1, d_1)$	$Q_i(v_2, d_1)$	...	$Q_i(v_{m_i}, d_1)$
d_2	$Q_i(v_1, d_2)$	$Q_i(v_2, d_2)$	...	$Q_i(v_{m_i}, d_2)$
:	:	:	:	:
d_n	$Q_i(v_1, d_n)$	$Q_i(v_2, d_n)$	...	$Q_i(v_{m_i}, d_n)$

Cada nó destino d é uma das estações base conhecidas do nó i. Cada vizinho v é um dos sensores dentro da área de alcance do raio de comunicação de i. Cada vizinho tem, em relação a cada EB, uma quantidade Q de feromônio, indicada nas interseções da tabela. A tabela baseia-se em uma probabilidade de escolha de saltos diretamente proporcional à quantidade de feromônio alocada nele. Assim, ela não deixa caminhos fixos, mas diversos caminhos com diferentes probabilidades de serem escolhidos.

O protocolo de roteamento para Redes de Sensores BiO4SeL divide-se em três fases distintas, de Reconhecimento (*bootstrap*), Sub-Seção 5.3.1, Descoberta Inicial de Rotas, Sub-Seção 5.3.2, e Troca de Dados, Sub-Seção 5.3.3.

5.3.1 Fase de Reconhecimento (*Bootstrap*)

O funcionamento desta fase pode ser visto no Algoritmo 1.

Esta primeira fase ocorre logo após os sensores serem ligados, ou seja, no começo de sua utilização. Cada um deles, por inundação, envia um pacote `ihello`. Tais pacotes têm

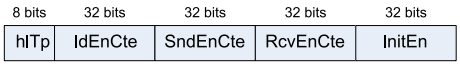
Algoritmo 1 Fase de reconhecimento do BiO4SeL.

1: nós enviam `ihello`
2: **if** nó recebe `ihello` **then**
3: guarda informação de bateria do vizinho
4: **end if**

tempo de vida de um salto, somente, e carregam, como visto na Figura 5.1, informações de bateria. Os nós receptores reconhecerão, então, o remetente como seu vizinho, assim como guardarão suas informações de bateria.

As informações contidas na figura são o tipo de pacote (`hlTp`), neste caso com o valor referente a `ihello`, e a sua informação de bateria. Esta informação pode ser diferente entre os sensores. É composta por energia inicial (`InitEn`) e pela taxa de consumo nos possíveis estados do sensor: ocioso (`IdEnCte`), envio (`SndEnCte`) e recepção (`RcvEnCte`). Esta implementação não leva em consideração o estado de sensoriamento, mas ele poderia ser facilmente adicionado com dois parâmetros, periodicidade e duração, se assim for conveniente. Esta informação foi propositalmente negligenciada por sua inutilização pelo simulador utilizado, o NS-2. A sobrecarga energética devido à adição destes parâmetros será ínfima na energia geral da rede, pois, nesta fase, cada sensor somente recebe uma mensagem de cada vizinho.

Figura 5.1: O pacote `ihello` (*Initialization Hello*) inicializa as tabelas de roteamento.



O tamanho deste pacote se deve a sua implementação no simulador NS-2. Neste simulador, os gastos energéticos e identificadores de pacotes são padronizados como tendo este tamanho. Este padrão foi mantido pelo fato de os protocolos implementados no simulador, assim como os outros implementados para comparação, terem estes tamanhos em seus próprios pacotes. Assim, faz-se uma comparação mais igualitária.

Além disto, o funcionamento desta fase pode ser afetado pela camada MAC. As perdas, por exemplo, poderiam ser um fator para desestabilizar esta troca de informações entre os sensores. Esta faceta da camada MAC não foi tratada ou avaliada neste protocolo. Apesar disto, tornou-se um trabalho futuro, como será possível ver no Capítulo 7.

5.3.2 Fase de Descoberta Inicial de Rotas

O comportamento geral desta fase é mostrado no Algoritmo 2.

Algoritmo 2 Fase de descoberta de rotas iniciais do BiO4SeL.

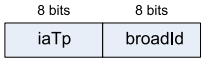
```
1: nós criam suas tabelas de roteamento
2: EB envia pacotes iant
3: if nó recebe iant then
4:     diminui a estimativa energética dos vizinhos
5:     if já recebida (mesmo identificador) then
6:         descarta iant
7:     else
8:         if saltos do pacote < distância atual à EB then
9:             atualiza distância à EB
10:        else
11:            decrease feromônio
12:        end if
13:    end if
14: end if
```

Esta fase consiste em enviar um certo número de formigas de inicialização (`iant`) para os nós da rede, a partir da Estação Base. Cada uma destas formigas será enviada por inundação, com tempo de vida grande, objetivando o alcance de toda a rede com sucessivos saltos.

Um só pacote enviado por inundação deveria ser suficiente para alcançar a rede inteira. O protocolo padrão para Redes de Sensores, entretanto, é o IEEE 802.15.4 [IEEE 2006]. Este padrão não prevê detecção de colisões. Assim, os reencaminhamentos dos pacotes podem não chegar a seus destinos por colisões, caso dois nós enviem seus pacotes ao mesmo tempo. Este tempo de reenvio é probabilístico. Portanto, não há certeza sobre a colisão dos pacotes.

A constituição deste pacote é mostrada na Figura 5.2. Ele é composto por tipo do pacote (`iaTp`) – neste caso tem o valor referente a `iant` – e identificador de inundação (`broadId`). Este identificador, juntamente com o identificador da fonte do pacote, é guardado em cada salto ao longo do caminho. Se um nó recebe a mesma `iant`, todas, além da primeira, serão descartadas. Assim, evita-se uma sobrecarga de pacotes repetidos na rede.

Figura 5.2: O pacote `iant` estabelece as rotas iniciais.

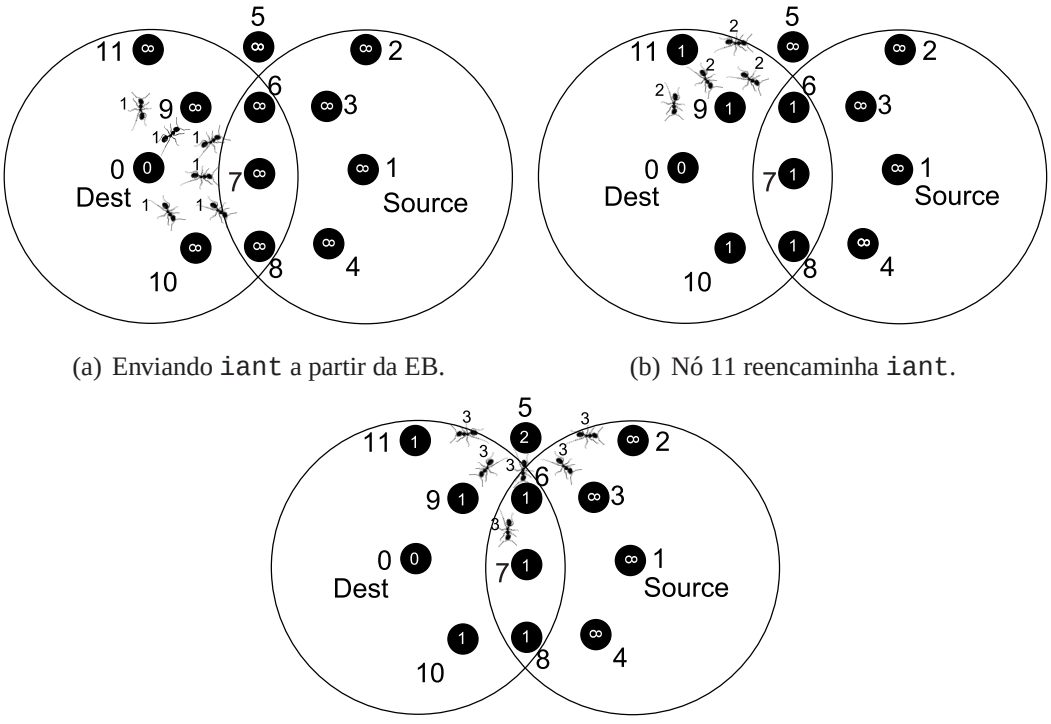


Este pacote tem por objetivo estabelecer rotas iniciais para a rede. Elas são de fun-

damental importância para o aspecto autônomo do algoritmo. De início, os nós da rede não detêm informação alguma sobre quem são seus vizinhos ou o destino da informação por eles coletada. Assim, as EB deve encontrar os nós, e não o contrário, garantido a formação destas rotas iniciais da rede.

Ao alcançar cada nó, a *iant* salva, naquele nó, sua distância até a Estação Base. Esta distância é dada na quantidade de saltos desde a EB. As formigas nesta etapa não aumentam os feromônios no caminho por onde vêm. Pelo contrário, se ela alcança algum nó no sentido contrário ao feito entre o nó e a EB, ela diminui o feromônio naquele salto, em relação a um destino, a EB de onde a formiga provém. A formiga sabe quando isto acontece por chegar a um nó contendo uma distância à EB previamente salva, de valor menor se comparada à quantidade de saltos da referida formiga. Assim, ela está fazendo um caminho mais longo e, consequentemente, pior, pelo menos nesta fase inicial.

Figura 5.3: Exemplo de envio de `iants` durante a fase de inicialização.



(a) Enviando `iant` a partir da EB.

(b) Nó 11 reencaminha `iant`.

(c) Nó 5 reencaminha `iant`.

A Figura 5.3 ilustra estas interações. A numeração externa aos nós representa seus identificadores, e a interna, a distância, em saltos, até a EB. Aquela fora das formigas indica a contagem de saltos do pacote. Na Figura 5.4(a), o destino, nó 0, envia uma `iant`. Ao alcançar os vizinhos, ela incrementa seus medidores de distância até a EB. Na Figura 5.4(b), após as mudanças nos nós, o nó 11 encaminha a formiga, escolha feita de

modo probabilístico e independete em cada nó. Os nós 9 e 6, após receberem uma `iant` com mesmo destino e identificador da última recebida, ignoram-na. Além disto, como a contagem de saltos da formiga é maior se comparada à distância miníma contida no nó, o feromônio destes nós para o 11 é reduzido, com relação ao destino 0. O mesmo ocorre na Figura 5.4(c), mas relacionado ao nó 5.

Há uma fórmula para o cálculo da diminuição feromônica para os chamados caminhos ruins. Tais caminhos são como os explicados acima, além de ciclos (*loops*), por exemplo. Assim, ela deve calcular uma quantidade de feromônio a ser diminuída do salto em questão, baseado na quantidade feromônica restante naquele salto ($F_d^v - F_{min}$). É uma quantidade restante, pois há mínimo e máximo feromônico, como citado acima, previstos no algoritmo, para cada salto na rede. Se eles não existissem, o feromônio em certo caminho poderia decrescer a ponto de ele não ter mais chance alguma de ser escolhido, ou crescer a ponto de sobrepujar completamente os demais.

Assim, basta calcular o parâmetro pelo qual esse feromônio restante será reduzido. Ele é calculado utilizando-se um coeficiente de redução feromônica K, um número arbitrário entre 0 e 1, cujo valor utilizado neste trabalho foi obtido através de experimentos. Este número não é exatamente a porcentagem de diminuição do feromônio, mas, utilizado em conjunto com um parâmetro de distância até a EB, vai gerá-la. Este parâmetro é uma estimativa de quão próximo o caminho está do melhor, em número de saltos. Como pode-se ver na fórmula abaixo, seria possível que esse parâmetro ultrapassasse o valor 1, gerando uma diminuição para além do mínimo estipulado. Para evitar isto, utiliza-se o valor mínimo entre o valor calculado e 0,9, ou seja, no máximo, 90% do feromônio será retirado. O valor de 0,9 evita que um nó tenha todo o seu feromônio retirado.

A fórmula citada é apresentada a seguir:

$$F_d^v = F_d^v - (min((K \times (2 - R_d^v)); 0,9)) \times (F_d^v - F_{min}), \tag{5.1}$$

na qual F_d^v é o feromônio para o destino d através do vizinho v, K é o coeficiente para diminuição feromônica, R_d^v é a relação de salto para o vizinho v com destino d, e F_{min} é o feromônio mínimo para um salto qualquer.

$$R_d^v = S_d^{min}/S^f, \tag{5.2}$$

na qual S^f é a contagem atual de saltos do pacote formiga e S_d^{min} é a mímina contagem de saltos em todos os vizinhos para o destino d, dada por:

$$S_d^{min} = Min\{S_d^v\}, \forall_{\text{vizinho}} v, \tag{5.3}$$

na qual S_d^v é a contagem de saltos a partir do vizinho v para o destino d.

Assim formam-se os caminhos iniciais da rede. Com a diminuição feromônica, a cada formiga enviada, de algum caminho ruim nos vizinhos de um nó, ao final do envio de diversas formigas, haverão caminhos com diferentes quantidades feromônicas. Os melhores (menores) caminhos da rede terão mais feromônios, tendo, consequentenmente, maior chance de serem escolhidos no envio de dados.

Além disto, a cada envio ou recebimento de qualquer pacote após a fase de reconhecimento, é possível estimar a bateria gasta pelo tráfego de pacotes nos vizinhos. Assim, é possível utilizar este dado decisivo para o roteamento sem necessidade de troca de pacotes entre os nós. Como almeja-se uma melhor distribuição do gasto energético pela rede, saber a energia dos vizinhos é de crucial importância.

Quando uma mensagem é enviada para um dado vizinho v, a estimativa energética para este vizinho (E^v) é atualizada com a fórmula:

$$E^v = E^v - G_r^v \times FC_r \times L_c \tag{5.4}$$

na qual G_r^v é o coeficiente de consumo energético para o estado de recepção do vizinho v (informado nos pacotes `ihello`), FC_r é um fator de correção de estimativa para recepção (a ser descrito na próxima fase), cujo valor inicial é 1, e L_c é o tamanho do pacote c, em bits.

Assim, a energia é atualizada dependendo do quanto foi gasto de energia pelo nó e dos bits enviados, ou recebidos. Assim, consegue-se uma porcentagem estimada de quanta energia foi gasta com envio, ou com recepção, dependendo do caso. Além disso, é utilizado o fator de correção, um desvio de cálculo aprimorado durante toda a utilização do algoritmo. Sua função é corrigir o cálculo da estimativa energética dos vizinhos a partir da energia real, descoberta via pacotes.

Quando uma mensagem é recebida de um vizinho, uma fórmula análoga à anterior é utilizada, subtituindo-se o coeficiente de consumo e o fator de correção do estado de recepção pelos do estado de envio. Assim, a fórmula utilizada é:

$$E^v = E^v - G_s^v \times FC_e \times L_c \tag{5.5}$$

sendo G_s^v o coeficiente de consumo energético para o estado de envio do vizinho v (também informado no `ihello`), e FC_e o fator de correção para envio, também atualizado na próxima fase, cujo valor inicial é 1.

Além de efetuar um destes dois cálculos, ainda é feita a atualização pelo tempo de vida decorrido desde a última mensagem recebida, onde supostamente o nó vizinho estaria em

estado inativo. Para calcular este gasto do tempo inativo, utiliza-se a fórmula:

$$E^v = E^v - G_i^v \times (T_c - T_l) \tag{5.6}$$

sendo G_i^v o coeficiente de consumo energético para o estado inativo do vizinho v (igualmente contido no `ihello`), e T_c e T_l os marcadores de tempo (*timestamps*) de chegada para o pacote anterior e o pacote atual, respectivamente.

Há, ainda, um sistema de reenvio de pacotes. Em um ambiente densamente populoso, reencaminhamentos de `iants` eventualmente colidirão e se perderão. Isto acontece pelo fato de cada nó reencaminhar todas as `iants` nova recebidas, como mostrado na Figura 5.4(b), da seção anterior. Estes reencaminhamentos ocorrem em tempos probabilisticamente definidos, dentro de um espaço de tempo predeterminado. Como é escolhido probabilisticamente, e não se pode deixar o espaço muito grande, dois reencaminhamentos podem ocorrer ao mesmo tempo, colidindo e perdendo-se os dados.

Para garantir a chegada a todos os nós de todas as `iants` enviadas, implementa-se este sistema de reenvio de pacotes. Durante a chegada de cada `iant` a cada nó, é disparado um contador com tempo T. Então, ele reencaminha-a, dentro de um tempo calculado, a formiga para seus vizinhos. Seus vizinhos, por sua vez, devem reencaminhar a formiga. Se o nó receber algum dos reencaminhamentos de seus vizinhos, dentro do tempo T, o temporizador é desconsiderado. Se não receber, a formiga é reenviada, pois deve ter-se perdido em colisão, visto nenhum dos vizinhos tê-la recebido e, consequentemente, reencaminhado. Este reenvio somente é feito uma vez, a fim de evitar uma indesejável inundação da rede com pacotes.

5.3.3 Fase de Troca de Dados

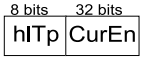
O comportamento geral desta fase é mostrado no Algoritmo 3.

Esta fase compreende o restante do funcionamento da rede. Nela, terão lugar os envios de dados dos nós à Estação Base. Com um conjunto inicial de caminhos formados, os dados são encaminhados à EB com escolhas probabilísticas a cada passo, baseadas nos feromônios depositados nos caminhos. A cada chegada do pacote a um nó, um novo salto é escolhido, o pacote é encaminhado, as energias são estimadas e o nó aguarda o próximo pacote.

A intervalos de tempo regulares, $\delta\tau$, formigas de anúncio, ou `hellos`, são enviadas a partir de cada nó. Semelhantes às `ihellos`, seu tempo de vida resume-se a um salto. Diferente delas, as formigas `hello` têm como objetivo apenas avisar aos vizinhos sobre a permanência do nó fonte vivo e próximo, além de informar o valor real de sua energia

remanescente. Com estas informações, os vizinhos poderão atualizar suas estimativas de disponibilidade de energia com relação ao fonte, registradas em suas memórias. O formato dessas formigas é ilustrado na Figura 5.4).

Figura 5.4: Pacote `hello`, a formiga de anúncio de vizinhança



Se um nó recebe esta mensagem de um vizinho previamente em sua tabela de roteamento, ele deve atualizar-se com um novo tempo de expiração para o vizinho e sua real energia. Este tempo de expiração será utilizado para retirar um possível nó morto, movido ou defeituoso de sua tabela de vizinhança. Em qualquer dos casos, não recebendo nenhuma notícia do vizinho dentro deste tempo, ele é retirado da tabela. Além disto, tendo a energia real do vizinho, é possível calcular o fator de erro na estimativa da energia gasta. O conjunto de fórmulas 5.7 é utilizado pelo nó para este cálculo.

$$\begin{aligned} E_r^v &= G_r^v * Q_r^v, \\ E_e^v &= G_e^v * Q_e^v, \\ E_t^v &= E_r^v + E_e^v, \\ FC_r &= FC_r + (((E_a^v - E^v) \times (E_r^v / E_t^v))/Q_r^v), \\ FC_e &= FC_e + (((E_a^v - E^v) \times (E_e^v / E_t^v))/Q_e^v), \end{aligned}$$

sendo E_r^v a energia gasta com recepção, pelo vizinho v, desde o último `hello`, Q_r^v o contador de pacotes enviados ao vizinho v desde o último `hello`, ou seja, os supostamente recebidos por ele, E_e^v a energia gasta com envio, pelo vizinho v, desde o último `hello`, Q_e^v o contador de pacotes recebidos do vizinho v desde o último `hello`, ou seja, os supostamente enviados por ele, E_t^v o total de energia gasta, pelo vizinho v, desde o último `hello`, FC_r o fator de correção para a recepção, inicialmente 1, E_a^v a energia atual do vizinho v (informado no pacote `hello`), E^v a estimativa da energia do vizinho v, calculada neste nó, e FC_e o fator de correção para o envio, inicialmente 1.

Para calcular o fator de erro, primeiro deve-se calcular o quanto de energia foi gasta em cada um dos modos de operação do sensor. Assim, baseado na quantidade Q de pacotes enviados e recebidos pelo sensor, e no coeficiente G de gasto energético em um estado, sabe-se o quanto deste erro "pertence" a cada um deles. Não há erro para o tempo, pois

ele é calculado localmente com o próprio relógio do sensor (apesar de alguns trabalhos relatarem um atraso deste relógio, esta variação está sendo desconsiderada).

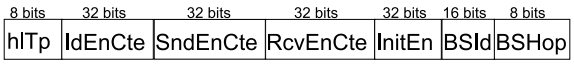
Calculado o gasto energético, teremos o fator de correção *FC* para cada estado. Ele é composto do fator de correção anterior mais a porcentagem de energia gasta por aquele estado em relação ao total gasto. Se essa porcentagem for positiva, o fator de correção será aumentado, a fim de aumentar o cálculo energético no momento da estimativa. Da mesma forma ocorrerá se a porcentagem for negativa.

Assim, cada uma destas equações calcula o fator de correção para a estimativa de um dos estados de algum vizinho. Este fator é importante, pois, mesmo com os dados sobre o cálculo energético enviados pelo próprio sensor, podem ocorrer erros de cálculo. Um destes erros baseia-se no fato de um nó não saber de todos os pacotes chegando ou saindo de seus vizinhos. Assim, com cada `hello` enviado, calcula-se um fator de erro, a fim de aproximar a estimativa energética o máximo possível do gasto real.

Um nó pode receber uma mensagem de `hello` de um vizinho *v* não constando em sua tabela de roteamento. Há mais de um motivo para isto acontecer. Um nó pode ter-se movido, ou ter sido deslocado até um certo ponto. Assim, ele ainda manterá sua tabela de roteamento antiga, mas alguns ou todos os seus vizinhos podem ter mudado. Ele pode, também, ter sido "adicionado" depois, colocado para substituir um sensor antigo sem energia, por exemplo. Ele estará com tabela de roteamento ainda por inicializar. Para ambos os casos, além de derivações ou cenários semelhantes, a troca de mensagens a seguir é suficiente para uma inserção adequada do nó no roteamento.

Se o descrito acima acontece, o nó recebendo o `hello` do vizinho desconhecido requisita as suas informações por meio do pacote `requestIHello` (*request initialization hello*), Figura 5.5. Este pacote é enviado ao desconhecido, somente com um salto, com o objetivo de fazê-lo identificar-se.

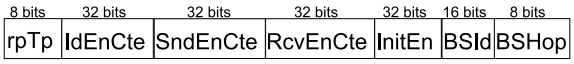
Figura 5.5: Pacote `requestIHello`, a formiga de requisição de anúncio



A Figura 5.5 demonstra os campos do pacote descrito. Assim como o `hello`, com o qual partilha muito de sua função, ele carrega informação de gasto energético. Além disto, o principal objetivo desta mensagem é fazer o sensor incorporar-se ao roteamento. Assim, ela conta com o identificador da EB (Estação Base, ou BS em inglês, *Base Station*) associada ao nó e a quantidade de saltos até ela. Se houver mais de uma EB, seria necessário enviar mais de uma mensagem, uma para cada.

Ao receber esta requisição, o nó envia uma resposta, o pacote *respRqIHello*, Figura 5.5. Este pacote é enviado, por inundação, para todos os vizinhos, com somente um salto de vida. Este pacote é igual ao `requestIHello`, somente diferindo dele por sua utilização. Ele carrega as informações de gasto energético do nó. Assim, os vizinhos podem guardá-la para o cálculo de seu decréscimo energético.

Figura 5.6: Pacote *respRqIHello*, a resposta ao pedido de anúncio de inicialização



O motivo desta mensagem em inundação é explicado a seguir. Deve-se considerar o fato de todos os vizinhos do novo nó terem recebido seu `hello` original, gerador do `requestIHello` por parte de um de seus vizinhos. Assim, todos os vizinhos cuja tabela de roteamento não contiver o novo nó enviarão seus próprios `requestIHello`. O novo nó envia sua resposta por inundação, já se adicionando à tabela de roteamento de todos os vizinhos ao mesmo tempo. Assim, evita-se a transmissão desnecessária de várias respostas, uma para cada requisição.

A outra forma de ocorrer esta troca de mensagens é o novo nó receber um `hello` de um nó antigo. Neste caso, o novo enviará sua requisição, e o antigo, a resposta. Neste caso, o fato de a requisição ser por inundação facilitará, pois todos receberão os dados do novo nó, salvando-os e enviando suas respostas.

Esta fase inteira, entretanto, tem por principal característica a transmissão de dados. Cada pacote de dados enviado leva consigo uma formiga de carona (*piggyback*), a `pbAnt`. Esta formiga é composta somente por um campo, o `pb_hop`. Quando criado o pacote de dados, este campo é preenchido com a distância mínima, em saltos, do nó atual até a Estação Base. Seu objetivo calcular o quão bom o caminho utilizado é, levando-se em conta somente a quantidade de saltos. Este dado será utilizado no cálculo de deposição feromônica.

Todo nó, ao receber um dado e não sendo seu destino, deverá reencaminhá-lo. Ao enviar ou reencaminhar um pacote deste tipo, uma rota deve ser escolhida. Ela é composta por saltos escolhidos a cada nó visitado. A escolha é feita de modo probabilístico, utilizando-se o feromônio associado a cada salto. Cada próximo salto de um nó, com relação a um destino, tem uma certa quantidade feromônica, situada entre um máximo (*maxF*) e um mínimo (*minF*). Para escolher o próximo nó, gera-se uma porcentagem de escolha, o feromônio de um salto dividido pelo total de feromônios em todos os saltos para um determinado destino. Assim, escolhe-se, com maior probabilidade, o salto com

maior feromônio associado. O último nó visitado pelo pacote será sempre descartado como uma próxima escolha, com o objetivo de evitar ciclos.

Há, entretanto, uma probabilidade (pE) de o salto ser escolhido não pela probabilidade feromônica, mas pela maior energia restante dentre os vizinhos. Esta escolha somente é feita se a energia restante no caminho com maior quantidade feromônica for inferior ou igual à metade da energia restante no vizinho com maior energia. Com esta tática, um caminho com pouca energia acaba por ser, com o tempo, preterido em função de outros a serem escolhidos. Assim, também, coloca-se uma forma de criar caminhos secundários para o caso de o primário não mais estar disponível, por algum motivo. Estes motivos podem ser o próprio decréscimo da energia no caminho ou falha em algum (ns) do (s) nó (s). Novamente, o último nó visitado é descartado para esta escolha.

Escolhido o próximo nó, o feromônio do caminho é atualizado. As equações a seguir modelam este processo. Estas fórmulas foram baseadas nas utilizadas em outros protocolos de roteamento utilizando formigas, como o Arama [Hussein and Saadawi 2003, Hussein et al. 2005]. Todas as constantes arbitrárias e os pesos citadas foram obtidas durante os experimentos, com o objetivo de otimizar o protocolo.

$$F_d^v = F_d^v + ((maxF - F_d^v) \times (c + e) \times fc), \tag{5.7}$$

sendo F_d^v o feromônio do vizinho v com relação ao destino d, $maxF$ o máximo feromônio possível em um salto, c a constante de caminho no cálculo feromônico, e a constante energética e fc a constante arbitrária associada ao cálculo.

Assim, o feromônio é adicionado de uma porcentagem do restante a ser colocado, ou seja, da subtração entre o máximo de feromônio e o atual. Esta porcentagem é calculada a partir de um balanço entre o quão bom o caminho é, ou seja, o menor possível (em saltos), e quanta energia o próximo salto possui. Utiliza-se, ainda, uma constante arbitrária para diminuir ou aumentar a mudança no feromônio.

A constante de caminho associada ao cálculo feromônico é calculada da seguinte forma.

$$c = cc \times pc, cc = minH/(cH + cnH),$$

sendo cc a variável de caminho, pc o peso do caminho no cálculo, $minH$ a mínima quantidade de saltos até o destino a partir da fonte, calculado na fase de reconhecimento e contido na formiga de carona em cada pacote de dados, cH a quantidade de saltos contada pelo pacote, cnH o mínimo de saltos até o destino a partir do nó corrente, calculado na fase de reconhecimento e salvo no prórpio nó.

Assim, a variável de caminho (*cc*) é dada pela relação entre as distâncias mínima e percorrida até o destino. Consequentemente, a constante de caminho (*c*) é obtida pela arbitrária e pelo peso do caminho.

A constante energética associada ao cálculo feromônico é calculada da seguinte forma.

$$e = ce \times pe, pe = 1 - pc, ce = E^v / EI^v,$$

sendo *ce* a variável energética, *pe* o peso da energia no cálculo, E^v a estimativa atual da energia do vizinho *v*, EI^v a energia inicial do vizinho *v*. Assim, a constate arbitrária da energia (*ce*) é dada pelo restante de energia do vizinho, e o peso da energia (*pe*) é sempre o complemento do peso do caminho (*pc*).

O peso energético no cálculo é o complemento do peso do caminho. Assim, os dois pesos completam uma unidade feromônica. A variável energética é dada pela porcentagem de energia restante no vizinho. Assim, quanto mais energia restar, maior será a constante energética para este caminho, aumentando seu feromônio.

Como consequência da atualização do feromônio em um salto, o feromônio total até aquele destino deve ser recalculado, como na equação a seguir.

$$F_d = F_d - (Fa_d^v - F_d^v), \tag{5.8}$$

sendo F_d o feromônio total para o destino *d*, F_d^v o feromônio ao destino *d* passando pelo vizinho *v*, e Fa_d^v o feromônio anterior deste salto, antes da atualização.

Ainda durante o reencaminhamento, a cada certa quantidade (*EvC*, variável obtida durante os testes de otimização do protocolo) de pacotes de dados recebidos de um vizinho *v* e indo para uma EB *d*, o feromônio somente naquele salto é evaporado. Para isto, utiliza-se a seguinte equação.

$$F_d^v = minF + (Ev^{EvI} \times (F_d^v - minF)), Ev = E^v / EI^v,$$

sendo F_d^v o feromônio ao destino *d* passando pelo vizinho *v*, *minF* o mínimo feromônio possível, *Ev* a porcentagem de evaporação, *EvI* o índice de evaporação, E^v a estimativa atual de energia do vizinho *v*, EI^v a energia inicial do vizinho *v*.

Nesta fórmula, a porcentagem de evaporação feromônica é diretamente proporcional à energia restante no vizinho. Esta decisão foi tomada com o objetivo de fazer as rotas variarem, e diminuir a probabilidade de uma rota com pouca energia ser utilizada. Deste modo, espera-se aumentar o tempo de vida da rede como um todo, dividindo-se

a carga pela rede inteira. Esta porcentagem é aplicada à diferença entre o feromônio atual e o mínimo. Além disto, é modificada pelo índice de evaporação, uma variável de configuração do protocolo, obtida por meio de testes. Tem o intuito de calibrar esta porcentagem para uma evaporação ideal. Este tipo de variável é utilizada em protocolos com este fim.

Novamente, como consequência da atualização (evaporação) do feromônio em um salto, o feromônio total até aquele destino deve ser recalculado, como na equação a seguir.

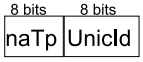
$$F_d = F_d - (Fa_d^v - F_d^v), \tag{5.9}$$

sendo F_d o feromônio total ao destino d, F_d^v o feromônio ao destino d passando pelo vizinho v, Fa_d^v o feromônio anterior, antes da atualização.

Passando agora à recepção de um pacote. O nó receptor guarda uma entrada para o pacote em uma fila. Esta entrada é semelhante à utilizada para `iants`, consistindo no identificador do nó fonte e da mensagem, tempo de expiração e último nó visitado. Ela é utilizada para evitar ciclos, ao receber um dado repetido. Há, no nó, um temporizador para retirar entradas muito antigas, pois elas só ocupam o pouco espaço disponível em um sensor.

Além de descartar um pacote repetido, o nó envia uma formiga negativa (`negAnt`, *negative ant*, Figura 5.7), pelo caminho inverso. Tal rota é conseguida percorrendo os últimos saltos guardados junto aos identificadores da mensagem, e é percorrida até reencontrar o ciclo, pois a própria formiga deixa entradas nos nós para detecção de ciclos. Esta formiga tem, por objetivo, não acrescentar feromônios, mas diminuir sua quantidade. A fórmula utilizada para esta diminuição é a mesma utilizada na fase 2 do algoritmo para caminhos ruins.

Figura 5.7: Pacote *negAnt*, a formiga de *desferomonização* de caminhos ruins.



A `negAnt` é constituída pelos campos mostrados na Figura 5.7. Há somente dois campos, os identificadores da formiga e do pacote causador do ciclo. Assim, ela pode percorrer o caminho inverso ao do pacote, guiando-se pelo identificador.

5.4 Avaliação em critérios de Inteligência Artificial

Este algoritmo foi baseado em um algoritmo originalmente desenvolvido no âmbito de IA. Assim, ele merece uma avaliação nos moldes desta disciplina.

A ordem de complexidade do algoritmo será calculada para cada nó. Como cada nó opera seu roteamento em separado, o BiO4SeL será executado em cada nó, separadamente. Como esperado, a complexidade é da ordem da quantidade de pacotes, ou seja, *O(n)*, sendo *n* a contagem de pacotes recebida em ou enviada por um nó.

Explicando a complexidade do algoritmo por etapas. Cada fase do protocolo opera de forma separada, e cada uma delas é *O(n)*. Na Fase de Reconhecimento 5.3.1, teremos cada nó recebendo *k* pacotes, sendo *k* a quantidade de vizinhos do nó. *k* sempre uma variável de escala menor que *n*, pois a quantidade de pacotes enviada e recebida durante a utilização de uma rede é maior que a quantidade de vizinhos de um nó. Na Fase de Descoberta Inicial de Rotas 5.3.2, cada nó recebe e reenvia uma quantidade de pacotes igual a $k \times l$, sendo *l* a quantidade de pacotes enviada pela Estação Base. Novamente, *l* é uma variável de pequeno porte, segundo será demonstrado nos testes, Capítulo 6. A Fase de Troca de Dados 5.3.3 tem complexidade *O(n)*, a maior de todas.

5.5 Trabalhos relacionados

Algoritmo 3 Fase de troca de dados do BiO4SeL

```
1: eventualmente, cada nó envia hello
2: eventualmente, cada nó envia dados
3: while nó recebe pacote do
4:     decrementa estimativa energética dos vizinhos
5:     if pacote = hello then
6:         if previamente na tabela de roteamento then
7:             conserta estimativa energética dos vizinhos
8:         else
9:             envia requestIHello
10:        end if
11:    end if
12:    if pacote = requestIHello then
13:        guarda informação de bateria
14:        envia respRqIHello
15:    end if
16:    if pacote = respRqIHello then
17:        guarda informação de bateria
18:    end if
19:    if pacote = data then
20:        if previamente recebido then
21:            descarta
22:            envia formiga negativa no caminho inverso
23:        else
24:            calcula próximo salto
25:            ajusta feromônio
26:            encaminha dados
27:        end if
28:        if EB then
29:            descarta
30:        end if
31:    end if
32:    if pacote = formiga negativa then
33:        decrementa feromônio
34:        if primeiro nó do laço then
35:            descarta
36:        else
37:            encaminha
38:        end if
39:    end if
40: end while
```

Tabela 5.3: *Resoluções de roteamento em Redes de Sensores, geral*

Protocolo	Infra-estrutura	Economia de energia	Roteamento	Formigas	Informação global	Vantagem única
[Akyildiz et al. 2002b]	Grupo	Rot. dos cabeças	Grupo	-	Não	Conh. dos viz.
[Al-Karaki and Al-Mashaqbeh 2007]	Grupo e Árvore	Rot. dos cabeças	Grupo e Árvore	-	Não	Menos mens. de atual.
[Iqbal et al. 2006]	Grupo	Rot. dos cabeças		-	Sim	Hist.
[Xu et al. 2005]	-	Balanc. de carga	Caminhos	-	Não	Garg. da rede
[Zongkai et al. 2005]	-	Balanc. de carga	Caminhos	-	Não	Nó ún. em cam.
MNLR [Slama et al. 2006]	-	Balanc. de carga	Diss. de inter.	-	Sim	
[Selvakennedy et al. 2007]	Grupo	Grupo + Form.	Tab. rot. prob.	Grupo	Não	Grupo + Form.
ARA	-	Form.	Tab. rot. prob.	Rotas	Não	-
ARAMA [Hussein et al. 2005]	-	Form.	Tab. rot. prob.	Rotas	Não	-
BiO4SeL	-	Form.	Tab. rot. prob.	Rotas	Não	Auton. do rot.

Legenda da Tabela 5.3. Grupo - agrupamento, Rotas - roteamento, Rot. dos cabeças - rotação dos cabeças do grupo, Tab. rot. prob. - tabela de roteamento probabilística, Balanc. de carga - balanceamento de carga, Form. - Colônias de Formigas, Conh. dos viz. - conhecimento dos vizinhos, Auton. do rot. - autonomicidade do roteamento, Menos mens. de atual. - menos mensagens de atualização, Hist. - histórico da rede, Garg. da rede - gargalo da rede nos dois últimos saltos para a EB, Nó ún. em cam. - nós em um só caminho, Diss. de inter. - disseminação de interesses

Tabela 5.4: *Resoluções de roteamento em Redes de Sensores, com formigas*

Protoclo	Tempo de vida	Bal. de carga	Roteamento	Avanço	Retorno	Atualização
ARA	Não	Não	Ativo	Inundação	Cam. único	Retorno
ARAMA [Hussein et al. 2005]	Sim	Não	Reativo	Cam. único	Cam. único	Retorno
AntHocNet [Caro et al. 2005]	Sim	Não	Híbrido	Inund. + cam. único	Não	Retorno
IAR [GhasemAghaei et al. 2007]	Não	Não	Reativo	Cam. único	Cam. único	Retorno
[Chen et al. 2007b]	Sim	Não	Reativo	Cam. único	Cam. único	Retorno
AOER [Shuang et al. 2007]	Sim	Sim	Híbrido	Inund.	Prob.	Avan. e ret.
BiO4SeL	Sim	Sim	Híbrido	Inund. + cam. único	Não	Qq. pacote

Boa parte dos trabalhos até agora tratados tem pouca relação com aquele apresentado aqui. A maioria destes trabalhos faz parte da pesquisa bibliográfica realizada a fim de conhecer, mais a fundo, o assunto estudado. Muitas soluções para o problema do roteamento em Redes de Sensores foram citadas, assim como diversas para diminuir o gasto energético nestas redes. A seguir, serão relembradas algumas cujo funcionamento ou finalidade assemelhe-se ao da proposta apresentada. Assim, todas elas são, de alguma forma, próximas da solução e parâmetros para sua eficiência. Aqui, será feita uma explicação e comparação mais objetiva dos pontos em comum e diversos entre eles e o BiO4SeL.

Uma característica importante desta proposta é o aumento do tempo de vida da rede. Muitos dos trabalhos apresentados apresentam soluções para este problema. Há diversos modos de solucioná-lo. Os mais interessantes para esta comparação, entretanto, são aqueles cujo foco é o mapa energético da rede e o balanceamento de carga na rede, além de soluções relacionadas a formigas e agentes.

A manutenção do mapa energético da rede consiste no conhecimento do nível de bateria dos nós na rede como um todo ou somente da vizinhança de cada sensor. Este conhecimento é utilizado para manter um nível médio de energia na rede como um todo. Ele pode ser alcançado por mais de um método.

Entre estes métodos, estão alguns baseados na infra estrutura da rede. Criando uma infra estrutura, torna-se mais simples manter a energia gasta em níveis semelhantes em todos os nós. Em [Akyildiz et al. 2002b, Al-Karaki and Al-Mashaqbeh 2007, Iqbal et al. 2006], utiliza-se o agrupamento para este fim. Geralmente, o agrupamento dos nós provê uma economia energética devido à fusão dos dados e à rotação do cabeça pelos componentes do grupo. Além disso, cada uma das soluções tem uma vantagem única, como a tentativa de descentralização por conhecimento dos vizinhos, menor quantidade de mensagens de atualização e utilização de histórico da rede, respectivemnte. A desvantagem destas abordagens é a sobrecarga de mensagens trocadas para a mudança deste cabeça, formação dos grupos e roteamento dentro de cada grupo. Em [Al-Karaki and Al-Mashaqbeh 2007], utiliza-se árvores. Elas diminuem a necessidade de reroteamentos, mas ainda assim sobrecarregam para a formação e sustentação da infra estrutura. O BiO4SeL somente cria uma infra estrutura para a criação do caminho inicial da rede, e sua manutenção é feita sem gasto extra, no mesmos pacotes da de vizinhanças.

Outras soluções baseiam-se não na infra estrutura para ditar os caminhos na rede, mas no roteamento em si. Em [Xu et al. 2005], utiliza-se a ideia de os nós a um salto da Estação Base serem os primeiros a acabar sua energia, tornando-se o gargalo energético da rede. Assim, maximizando estes, maximiza-se o tempo de vida da rede. Sua desvantagem é a criação de caminhos, pois eles são estáticos, requerendo manutenção, ou seja,

gastando bateria com envio de mensagens. Essa desvantagem é facilmente vencida por um algoritmo de formigas. [Zongkai et al. 2005] aprimora-o, evitando a concorrência de dois ou mais caminhos em um mesmo nó, ou dividindo a energia do nó entre eles. Ainda assim, as desvantagens apontadas continuam. Eles, no entanto, têm muito em comum com o algoritmo desenvolvido neste texto. Assim também é o MNLR [Slama et al. 2006], *Maximum Network Lifetime Routing*. Nele, a EB utiliza informações, como a bateria e a distância, de todos os nós da rede para configurar caminhos com o objetivo de gastar a energia igualmente pela rede como um todo. O BiO4SeL é dinâmico e autônomo.

Dentre as soluções de roteamento utilizando formigas, alguns métodos podem ser evidenciados. Pela versatilidade de sua solução, ela é utilizada de diversas formas em Redes de Sensores, com objetivos diferentes. Vejamos alguns deles a seguir.

As formigas podem ser utilizadas indiretamente para o aumento do tempo de vida da rede. Em [Selvakennedy et al. 2007], o autor utiliza formigas para mediar o agrupamento dos sensores. Assim, ele consegue melhorar o tempo de vida da rede. Além disto, minimiza o gasto energético e também alcança métricas de balanceamento de carga. Assim, alcança parte dos objetivos desejados, tornando-o uma boa métrica. A desvantagem, aqui, é o agrupamento, e os problemas trazidos com ele. Em [Shen et al. 2005a], aumenta-se o tempo de vida da rede por meio de controle da topologia da rede. Otimiza-se a área de abrangência de cada sensor para cada um gastar somente o mínimo possível de energia. O BiO4SeL não utiliza grupos ou outras camadas, apresentando roteamento plano.

Elas também podem ser utilizadas diretamente para a obtenção de caminhos ótimos e economia de bateria. No ARAMA [Hussein and Saadawi 2003, Hussein et al. 2005], elas criam caminhos ótimos entre os nós enviando informações e a EB, utilizando tabelas de roteamento probabilísticas. Estas não formulam somente um próximo passo para cada destino da rede, mas guardam as probabilidades de cada vizinho ser o próximo passo para cada destino da rede. Assim, forma-se, sempre, mais de um caminho para cada destino, uns com maior e outros com menor probabilidade, dependendo da heurística e dos feromônios depositados. Este algoritmo foi a base para o desenvolvimento do BiO4SeL, guardando dele muitas similaridades, mas diferenças cruciais, como a inversão de sentido da busca de caminhos. No ARAMA, fatalmente, com muitos nós no cenário, o algoritmo não encontrará rotas à EB.

O algoritmo acima, apesar da desvantagem apresentada, é utilizado como base para diversos outros. Apesar de ele conseguir aumentar o tempo de vida da rede, alguns outros algoritmos partem da base de seu raciocínio para tentar melhorá-lo. O ARAMA proclama melhorar o ARA, modificando a propagação das formigas de inundação para caminho único. Em [Caro et al. 2005], os caminhos são, inicialmente, criados por formigas em

inundação. Ao encontrarem um nó já com um caminho para a EB, a formiga torna-se de caminho único. Em [GhasemAghaei et al. 2007], utiliza-se uma estratégia muito semelhante ao já apresentado. Aqui, entretanto, utiliza-se informações globais, algo a ser evitado em prol da escalabilidade. Em [Chen et al. 2007b], utiliza-se o ARAMA em união ao ZRP, *Zone Routing Protocol*, para diminuir o tempo e a sobrecarga de descoberta de caminhos. Cria-se uma zona de c saltos ao redor de cada nó, ao invés de somente o vizinho, interconectados por uma infra estrutura. Assim, roteia-se para fora da zona, somente, e de uma zona para a próxima.

Muitos algoritmos, entretanto, encontram como solução para o problema energético na rede o balanceamento de carga pela rede como um todo. Alguns deles já foram apresentados, e alguns utilizando formigas serão dispostos. No AOER [Shuang et al. 2007], as formigas são enviadas por inundação. Se elas conseguirem melhorar o caminho, são reencaminhadas. Senão, são mortas. Assim, a(s) formiga(s) chegará(ão) ao destino pelo(s) melhor(es) caminho(s), aumentando o feromônio no caminho. Como a formiga de avanço não salva o caminho percorrido, ela retorna pelo melhor possível, diminuindo o feromônio pelo caminho. Assim, probabilisticamente, a próxima formiga toma um caminho diferente, gastando a energia de outros nós da rede e alcançando um balanceamento de carga na rede. O problema deste método é sua tabela de roteamento. Ela inclui todas as possíveis fontes da rede, e não só os próximos saltos, como as apresentadas, tornando-se menos escaláveis e autônomas. Em [Huang et al. 2006], utiliza-se um método matemático de otimização para a escolha do próximo salto, levando em consideração a energia dos nós vizinhos. Ela, entretanto, cria caminhos imutáveis até o fim da rede, uma desvantagem clara se comparada a qualquer paradigma com formigas. Semelhante a esta solução é a contida em [Liu et al. 2007], com a adição de ângulo de deflexão. Assim, há a necessidade de um GPS para a localização dos nós, não necessariamente disponível em qualquer sensor.

Tabela 5.5: *Resoluções de roteamento em Redes de Sensores, com agentes*

Algoritmo	Tempo de vida	Bal. de carga	Inicialização	Roteamento	Cálculo energético	Agregação	Inf. global	Cálc. da energia dos viz.
EAR [Shah and Rabaey 2002]	Sim	Sim	Inundação restrita à partir da EB	Probabilístico, custo e energia	Igual para recepção e envio	Retorno	Não	Não
[Gan et al. 2004]	Sim	Sim	Desconsidera	Probabilístico, custo e energia	Diferenciado para recepção e envio	Implícita	Tempo	Sim
MADSN [Qi et al. 2001]	Não	Não	Agrupamento	Salto único para o cabeça	Diferenciado	Sim	Nós do grupo, no chefe do grupo	Não
[Yongtao et al. 2006]	Não	Não	Agrupamento	Dentro dos grupos	Diferenciado	Sim	Não	Não
[Shakshuki et al. 2007]	Sim	Não	Inundação	Pela EB	Não	Sim	Sim	Não
MADD [Chen et al. 2007a]	Sim	Não	Difusão direcionada	Pela EB e local	Não	Sim	Sim	Não
EMA [Pan et al. 2007]	Sim	Não	Inundação	Probabilístico	Não	Sim	Não	Não
AbDD [Malik et al. 2007]	Sim	Sim	DD	Inund. + cam. único	Sim	Não	Não	Sim
BiO4SeL	Sim	Sim	Híbrido	Inund. + cam. único	Não	Qq. pacote	Não	Sim

Há, também, diversas soluções na área de multi agentes. A diferença entre um agente e uma formiga é o fato de um agente ser autônomo por si só, carregando consigo o código necessário à aplicação, a fim de ser executado onde vá. Assim, o código não precisa estar no sensor, mas trafega pela rede juntamente com o agente. A vantagem desta estratégia é a autonomicidade e a liberação de memória do nó, recurso este, escasso. A desvantagem é a clara sobrecarga de envio e recepção de informações na rede, diminuindo o tempo de vida dos sensores.

As soluções com agentes em Redes de Sensores geralmente vêm associadas à dis-se-mi-na-ção de interesses pela rede e à agregação de dados. Elas resumem-se a encontrar os nós fonte e utilizar os agentes para agregar os dados. Em [Gan et al. 2004, Shah and Rabaey 2002], ocorre um balanceamento de carga pela rede, e os dados são agregados implicitamente. Ao chegar um dado a um certo nó, ele testa se algum outro dado, anterior àquele e da mesma área de coleta, já passou pelo nó. Se sim, ignora este e não o repassa. Se não, repassa o dado. Falta a ele somente não tempo global, ou seja, um mesmo tempo em todos os nós, algo improvável em um cenário real. Já em [Shakshuki et al. 2007, Chen et al. 2007a], o agente vai em cada nó fonte agregando os dados antes de ir para a EB. Nelas, a desvantagem é o conhecimento global da rede, apesar de a difusão direcionada, nelas utilizada, ser bem semelhante à inteligência de enxames das formigas. Outra desvantagem é a inutilização da energia do caminho formado entre os fontes e a EB, diminuindo o tempo de vida da rede por não variá-lo.

Dado tal quantidade de protocolos semelhantes ao BiO4SeL, é possível notar o extenso trabalho realizado nesta área. Para comparar ao protocolo apresentado neste texto, foi escolhido, desta lista, somente o ARAMA. Esta escolha foi feita devido ao nível de detalhes de implementação encontrados no artigo deste algoritmo. Alguns outros protocolos, como o AOER [Shuang et al. 2007], são mais semelhantes ao BiO4SeL, mas falta-lhes detalhes para uma implementação e comparação seguras, sem chance de não fazer jus ao original. Além disto, será comparado ao AODV, por já ser trazido no simulador utilizado, como explicitado no próximo capítulo.

Capítulo 6

Resultados e Discussões

Neste capítulo, serão apresentados e discutidos os resultados mais representativos para discussão. Antes disto, os diversos parâmetros utilizados serão demonstrados, embasando e possibilitando a repetição dos experimentos.

6.1 Parâmetros da Simulação

6.1.1 Parâmetros do NS

Foram utilizados diferentes e diversas quantidades de nós e tamanhos de cenários. Todos os testes foram encenados utilizando-se os dados da Tabela 6.1.

Tabela 6.1: *Cenários utilizados nos testes*

Quantidade de nós	Tamanho do cenário (m^2)	Densidade (nos/tam)
10	20x20	0,0250
20	30x30	0,0222
40	50x50	0,0160
80	70x70	0,0163
160	100x100	0,0160
320	140x140	0,0163

Estes tamanhos foram obtidos a partir de experimentos para descobrir o maior tamanho possível, para cada quantidade de nós, com uma margem aceitável de cenários sem nós desconexos. A margem utilizada, nestes testes, foi de 90%. Assim, estes são os maiores cenários possíveis nos quais, em 90% dos testes de geração de cenários aleatórios, não houveram nós desconexos.

Este é um teste necessário, tendo-se em vista o fato de os cenários dos testes do protocolo serem gerados aleatoriamente, e estarem propensos a terem nós desconexos. Esta minimização teve o objetivo de economizar tempo de simulação, pois nenhum teste é realizado em um cenário com nós desconexos. No objetivo de medir tempo de vida e energia média de uma rede, é indesejável nós inalcançáveis, com bateria imutável do início ao fim da simulação.

Para descobrir se um cenário contém nós desconexos, um teste simples é realizado. Envia-se, de cada nó, uma mensagem por inundação, com tempo de vida de um salto. Estas mensagens devem ser enviadas a intervalos de tempo maiores se comparados ao tempo de envio e recebimento da mensagem por um vizinho. Assim, garante-se a chegada de todas as mensagens, sem colisões. Se algum nó, depois de todos terem enviado, não houver recebido mensagem alguma, ele estará desconexo na rede, ou seja, nenhum outro nó o alcança. Este é o exato procedimento da Fase de Descoberta Inicial de Rotas do BiO4SeL. Por isto, ele foi utilizado com o fim de testar a conectividade da rede.

Assim, estes tamanhos de cenários acabaram por gerar densidades diferentes. Apesar de este fato não contribuir para a comprovação de escalabilidade do algoritmo, os cenários com os tamanhos especificados são necessários para uma adequada avaliação do algoritmo. Mesmo assim, nos cenários a partir de 40 nós, a densidade é bem semlhante. Nestes cenários, a escalibilidade pode ser levada em consideração.

Os testes foram feitos da seguinte forma. Para cada categoria, ou seja, cada quantidade de nós, 100 cenários diferentes foram gerados, aleatoriamente. Em cada um destes cenários, foram feitos 50 testes, com o propósito de ter uma amostra significativa para média. Cada um destes cenários é uma geração aleatória da disposição da quantidade correspondente de nós. Para cada cenário, o AODV é testado somente uma vez, pois é determinístico. Assim, apresenta sempre o mesmo resultado para testes em um cenário e parâmetros específicos. O ARAMA e o BiO4SeL, entretanto, são probabilísticos. Assim, são necessários muitos testes para ter-se uma média confiável dos resultados. Para isto, foi utilizado um intervalo de confiança (α) de 0,95. Pelo mesmo motivo, diversos cenários são gerados, para obter-se uma média do desempenho do protocolo para uma quantidade específica de nós.

Para identificadores dos nós, utilizam-se números únicos. Eles são distribuídos, consecutivamente, do zero e até o $n-1$, sendo n o número total de nós da rede. Em todos estes cenários, há somente uma fonte e um destino. Eles são os nós $n-2$ e $n-1$, respectivamente, e estão localizados nos extremos da rede. Assim, tenta-se conseguir o máximo possível de saltos entre eles. Por exemplo, no cenário de 10 nós, fonte e destino serão os nós 8 e 9 e estarão localizados nos pontos (1,1) e (19,19), respectivamente.

O modelo de tráfego utilizado é o CBR (*Constant Bit Rate*). Foi escolhido pela facilidade de ser utilizado e mesurado, assim como pela maioria dos artigos lidos utilizarem-no também. É enviado, na fase de envio de pacotes, é um pacote de 70 kb a cada 0,4 s, ou seja, 82 KB/s.

Os pacotes começam a ser enviados a partir de 28 s de simulação. Este é um tempo estimado para o bom funcionamento das fases anteriores do BiO4SeL, e mantido em todos os testes, para uma melhor comparação dos protocolos. O total de tempo de simulação é 200 s. Isto simula uma utilização como a de envio de imagens, com um alto tráfego em pouco tempo.

Outras configurações do NS podem ser encontradas na Tabela 6.2. As configurações inexplicadas até o momento são padrão do simulador para este tipo de teste. A parte de energia também é padrão, copiada dos próprios arquivos internos do NS.

Tabela 6.2: *Configurações do NS*

Característica	Valor
Canal	Sem fio
Propagação	*TwoRayGround*
Camada física	IEEE 802.15.4
Fila de interface	*PriQueue*
Tipo de camada de enlace	*LL*
Antena	*OmniAntenna*
Tamanho máximo da fila de interface	50
Parâmetro energético para o envio	0.660
Parâmetro energético para a recepção	0.395
Parâmetro energético para ficar ligado	0
Alcance da antena	15m
Modelo energético	*EnergyModel*
Energia inicial dos nós	1
Energia inicial da EB	2

Estes parâmetros foram escolhidos, em sua maioria, por serem o padrão trazido no NS. A energia inicial dos nós foi escolhida como 1 para melhor visualização nos gráficos, e a da Estação Base como 2 para uma adaptação de uma energia inesgotável.

Para esta simulação, foram utilizadas somente uma fonte e uma EB. Isto foi feito pela simplicidade do teste, e por seguir um padrão nos textos da área.

6.1.2 Parâmetros do Protocolo

Diversos são os parâmetros necessários para o funcionamento do protocolo. Estes parâmetros foram obtidos mediante testes, objetivando a otimização do tempo de vida e do funcionamento do algoritmo. Eles estão na Tabela 6.3.

Tabela 6.3: *Parâmetros do protocolo*

Parâmetro	Valor
Tempo de envio de cada `ihello`	$t = 0,01 \times id$
Quantidade de formigas de inicialização (`iant`) enviadas	$iant_cnt = 5$
Intervalo entre cada envio de *iant*	0,5s
Intervalo entre cada envio de *hello*	7,5s a 12,5s
Tempo de expiração do contador de *hello*	13s
Tempo de expiração das entradas de *unicast*	0,5s
Tempo de expiração do contador de entradas de *unicast*	1s
Mínimo de feromônio ($minF$)	0.000001
Mínimo de feromônio ($maxF$)	0.01
Feromônio inicial (iF)	0.0001
Intervalo de envio de *hello* ($hlInt$)	10s
Expiração do contador de pacotes (EvC)	2 pacotes
Índice de evaporação (EvI)	3
Peso do caminho (pc)	0,1
Constante arbitrária associada ao cálculo feromônico (fc)	0,1
Coeficiente de diminuição feromônica (K)	0,6

Legenda. *id* - identificador único de cada nó.

Seguem explicações quanto aos parâmetros adotados.

Fase de Reconhecimento: O tempo de envio de cada *ihello* é calculado para nenhum colidir em outro. Assim, garante-se o conhecimento de cada nó sobre seus vizinhos.

Fase de Inicialização: A quantidade de formigas de inicialização enviadas foi calculada por meio de testes. 0,5 s é tempo suficiente para o envio de outra formiga sem colisão com a anteriormente emitida.

Quanto aos outros parâmetros, foram obtidos pelos testes. Esta configuração representa a melhor obtida, nos testes, para o algoritmo.

6.2 Resultados

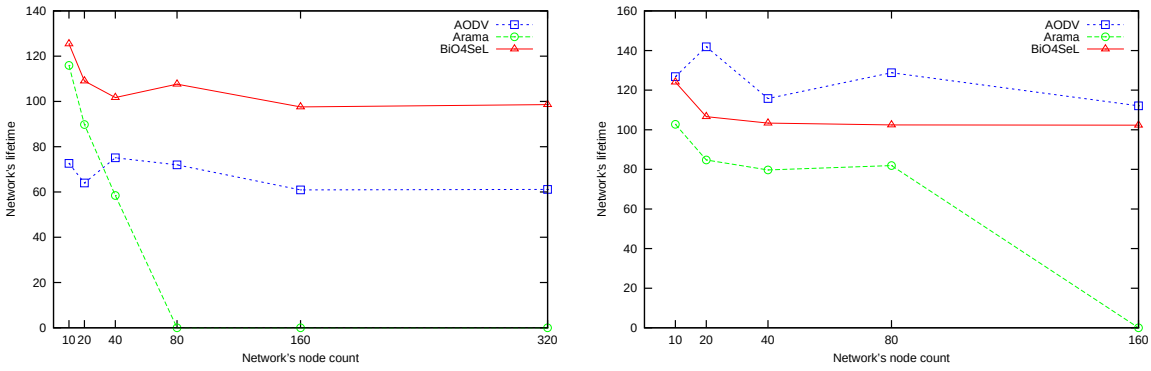
Nesta seção, serão mostrados os mais representativos gráficos obtidos como resultados. Eles serão explicados no intuito de demonstrar as melhorias alcançadas pelo protocolo BiO4SeL em comparação a seus antecessores. Ele foi comparado ao AODV, padrão para a área, e ao ARAMA, um algoritmo de formigas para o roteamento em redes de sensores. Algumas particularidades destes protocolos serão mostradas com maiores detalhes durante as comparações.

Alguns dos resultados obtidos não serão demonstrados nesta seção. Esta escolha foi feita pela sua grande quantidade. Aqueles ignorados aqui são colocados no Apêndice 2.

Uma observação quanto aos resultados. Em todos eles, o Intervalo de Confiança previamente descrito foi utilizado. Em muitos gráficos, ele é bem visível. Em alguns deles, entretanto, o seu valor foi pequeno o suficiente para ser ignorado nos gráficos.

6.2.1 Experimento 1: Tempo de Vida pela Quantidade de Nós

O tempo de vida da rede é o tempo até o primeiro nó da rede "zerar" a energia de sua bateria.



(a) Comparação do tempo de vida dos protocolos

(b) Comparação do tempo de vida dos protocolos, AODV sem `hellos`

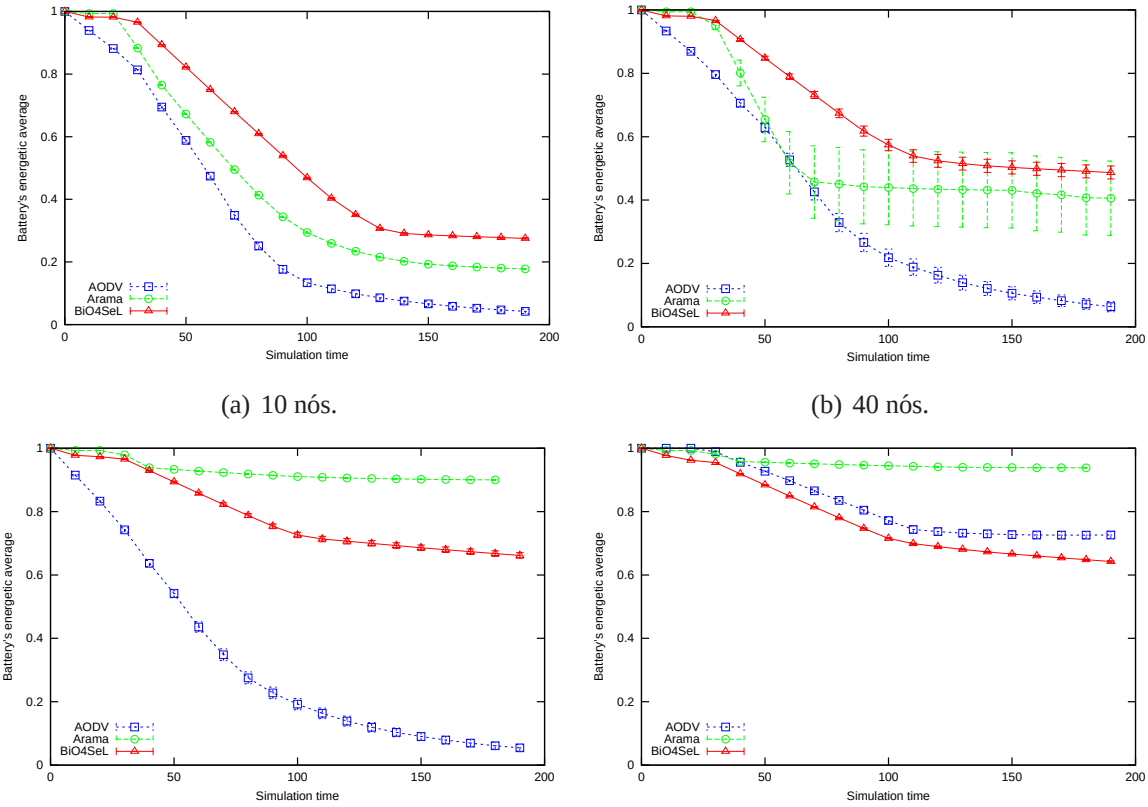
Figura 6.1: Tempo de vida pela quantidade de nós

O gráfico da Figura 6.1(a) torna o ganho em tempo de vida do BiO4SeL bem claro. Em todas as quantidades de nós, ele supera o ARAMA e o AODV em tempo de vida. O ARAMA leva seu tempo de vida a zero em certos pontos por não conseguir encontrar rota nesta quantidade de nós.

O gráfico da Figura 6.1(b) mostra o AODV sem `hellos`. Sem sua sinalização inteira, seu tempo de vida aumenta consideravelmente. Ele gasta bem menos energia, se

comparado à versão com os `hellos`. Comparado ao BiO4SeL, consegue um ganho, realmente esperado. O modo de cálculo energético feito pelo NS, no envio e recepção de nós, possibilita o acontecimento deste fato.

6.2.2 Experimento 2: Média de Bateria na Rede



(a) 10 nós.

(b) 40 nós.

(c) 160 nós.

(d) 160 nós, AODV sem “hello”.

Figura 6.2: Média de bateria na rede

Na Figura 6.2(a), vemos a melhor média de bateria alcançada pelo BiO4SeL. Em todos os momentos, ele supera os outros protocolos em média de bateria, exceto nos 20s iniciais. Nestes, ele perde para o ARAMA, pois este ainda não iniciou seu envio de dados. Como ele somente envia as formigas de reconhecimento de rotas após iniciar a troca de pacotes, ele não gasta bateria alguma até este momento.

O AODV gasta bem mais bateria por enviar muitos pacotes de controle. A fim de manter uma possível mobilidade na rede, o AODV envia diversos “hello”, a ponto de congestionar a rede. Assim, sua média de bateria cai mais rápido, se comparado aos outros protocolos. Vale lembrar a igual mobilidade do BiO4SeL, sem este gasto desnecessário de bateria.

Na Figura 6.2(b), o comportamento dos protocolos é bem semelhante ao da Figura 6.2(c). A partir desta quantidade de nós, o ARAMA não mais encontra caminhos na rede. Isto acontece devido a uma falha deste protocolo.

O ARAMA tem um problema quanto à escalabilidade. Inicialmente, ele ele busca por rotas com formigas de saltos “cegos”. Não há nenhum conhecimento prévio da rede, seja embutido nos nós ou com uma fase de inicialização. Em uma rede com muitos nós entre a EB e o nó gerando as formigas, há a possibilidade clara de ela perder-se sem encontrar seu destino. Por ter um tempo de vida fixo, ela acaba por vagar pela rede até formar um ciclo, detectado pela lista de saltos carregada por ela, ou até acabar seu tempo de vida. Assim, este protocolo, em geral, funciona bem somente para pequenos cenários.

Quando a primeiras formigas são enviadas, a tabela de roteamento é criada, para ser atualizada no retorno da formiga. Como este retorno não acontece, todos os próximos saltos de todos os nós continuam como de início, com uma quantidade feromônica igual em todos. No momento de envio dos dados, o pacote irá escolher, a cada salto, o nó com maior quantidade feromônica. Como todos têm a mesma, ele simplesmente escolhe o primeiro da lista. Assim, os dados irão trafegar sempre no mesmo caminho, perdidos ou em ciclo infinito. Como o protocolo não implementa uma formiga negativa para ciclos, os nós deste caminho serão sempre os escolhidos, até sua energia zerar. Neste ponto, a energia da rede mantém-se a mesma até o final da simulação.

Na Figura 6.2(c), também da média de bateria pelo tempo, foi colocada como uma comparação com a anterior. Com um cenário bem maior, ela guarda algumas das características da antecessora, mas revela algumas novas.

A partir do momento 50, o ARAMA passa a não gastar mais muita energia por não ter conseguido encontrar rota alguma. Ele, entretanto, continua a enviar dados, mas eles não seguem muito longe pela falta de um caminho até a EB. Assim, sua média energética sofre pouca queda, se comparada às dos outros protocolos. Além disto, há gasto com sinalização e tentativa de encontrar rotas.

A média energética do ARAMA, entretanto, é maior. Isto acontece somente pelo fato de ele ter o mesmo gasto médio de bateria, mas ter mais nós. Assim, a média geral aumenta.

Na Figura 6.2(d), também da média de bateria pelo tempo, foi colocada como uma comparação com a anterior. Esta, entretanto, traz o AODV sem enviar seus “hellos”. Isto implica em uma grande economia energética, mas também em uma completa ineficiência em tratar movimentação ou término de bateria dos nós. Ainda assim, o BiO4SeL consegue uma melhor eficiência.

6.2.3 Experimento 3: Distribuição Energética, no Tempo de Vida

As Figuras 6.3 e 6.4 representam a distribuição energética nos nós da rede no momento do tempo de vida. Assim, os seus pontos mais altos são os locais com mais energia, e os mais baixos, com menos. Além disto, elas passaram por um processo de interpolação no eixo y a fim de torná-los mais reais. A fonte de envios encontra-se no ponto $(1,1)$, e a estação base, no outro extremo da rede, ponto $(x-1,x-1)$, x sendo o tamanho do lado da rede.

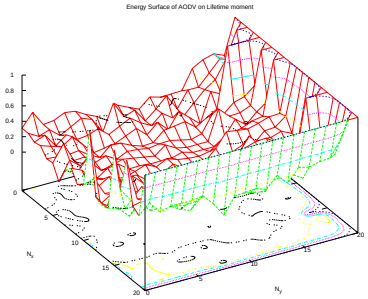
Na Figura 6.3(a), assim como a Figura A.2(a), pode-se ver uma queda relativamente igualitária na rede como um todo. Difícil tirar alguma conclusão, pois são poucos nós, tornando a rede muito espaçada e pouco utilizada. Além disso, essa "boa utilização" da rede é, em parte, causada pelos `hellos` enviados cosnstantemente. Como a rede é pequena, estes pacotes tendem a alcançar boa parte dos nós de uma só vez, assim como os próprios pacotes de dados. Conclui-se, então o fato de a energia ser mais utilizada pela rede como um todo, menos nos pontos próximos ao valor 20 do eixo y, nos quais nenhum nó deve ter sido alocado.

Na Figura 6.3(b), assim como na Figura A.2(b), a explicação do comportamento é bem semelhante à da Figura 6.3(a). Naquela, entretanto, não há tantos pacotes de sinalização, tornando a distribuição energética um pouco menos igualitária.

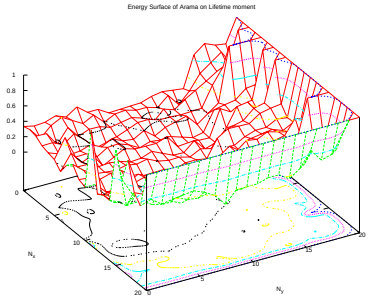
Na Figura 6.3(c), assim como na Figura A.2(c), a explicação do comportamento é bem semelhante à da Figura 6.3(a). Naquela, entretanto, não há tantos pacotes de sinalização, tornando a distribuição energética um pouco menos igualitária. Apesar disto, ainda há pouca, mas alguma, melhora em relação à Figura 6.3(b).

Na Figura 6.3(e), assim como nas Figuras 6.3(h), 6.4(c) e A.2(e), com o aumento da quantidade de nós, eles começam distribuir-se melhor pela rede. Assim, é possível fazer uma melhor explanação sobre o gasto energético. Ele torna-se visivelmente maior nos extremos da rede, pois é de onde vêm ou para onde vão, invariavelmente, todos os pacotes de dados da rede. Além disso, a sinalização ainda causa o mesmo fenômeno de gasto "igualitário" da energia na rede, na sua parte intermediária. Os níveis energéticos, nesta área, tornam-se mais iguais.

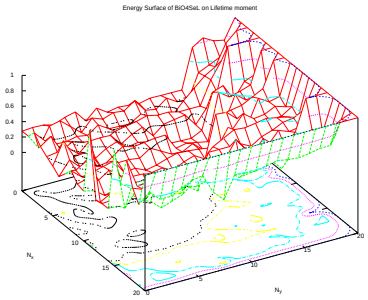
A Figura 6.3(f), assim como a Figura 6.4(d), mostra a diferença entre a presença e ausência de `hellos` na rede. Como não há a maior parte da sinalização, resta a troca de dados como grande causador da baixa energética. Tem-se, então, um gráfico com uma depressão visível entre os extremos da rede, a fonte e o destino. É possível notar a ausência de uma preocupação em variar os caminhos de envio de dados na rede. Assim, somente esta linha direta entre fonte e destino é utilizada, até sua exaustão.



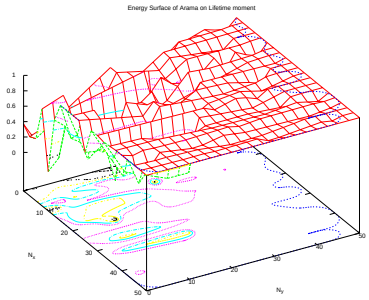
(a) AODV, 10 nós.



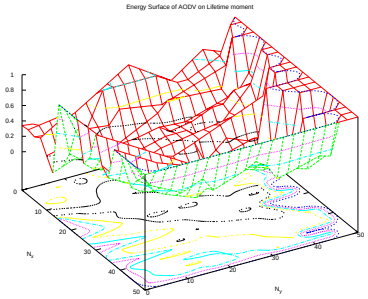
(b) ARAMA, 10 nós.



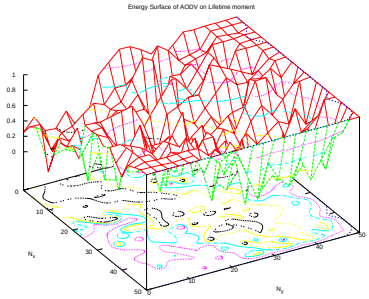
(c) BiO4Sel, 10 nós.



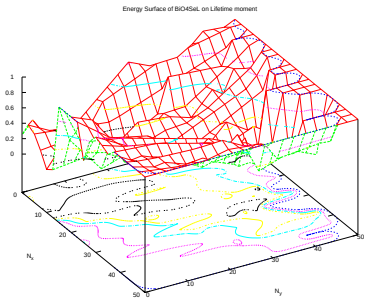
(d) ARAMA, 40 nós.



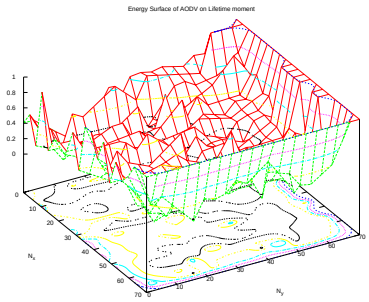
(e) AODV, 40 nós



(f) AODV sem `hellos`, 40 nós



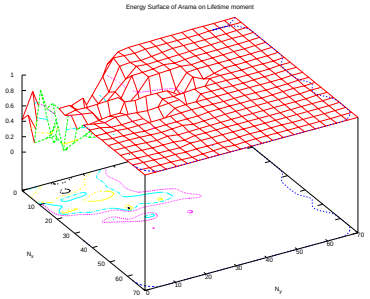
(g) BiO4Sel, 40 nós.



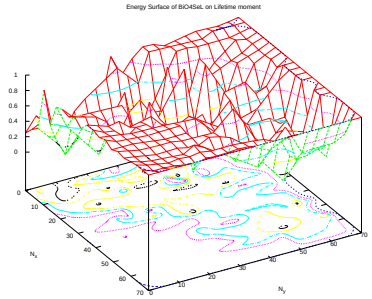
(h) AODV, 80 nós

Figura 6.3: Distribuição energética na rede, no tempo de vida, início

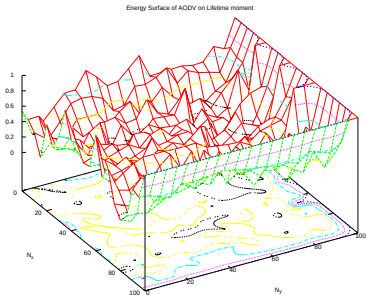
Na Figura 6.3(d), assim como nas Figuras 6.4(a), 6.4(e) e A.2(d), o ARAMA não mais consegue enviar dados. Assim, os pacotes gastam energia próximo à fonte, como



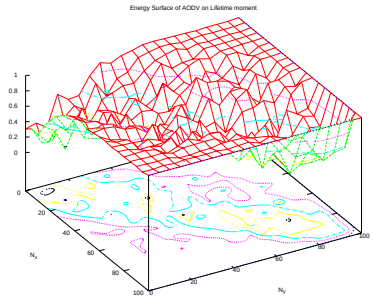
(a) ARAMA, 80 nós.



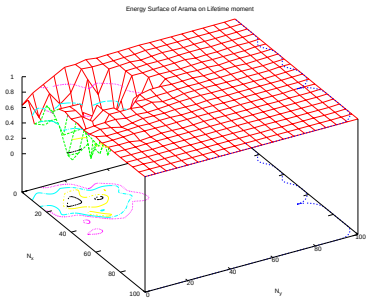
(b) BiO4Sel, 80 nós.



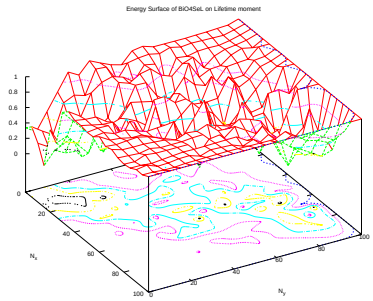
(c) AODV, 160 nós.



(d) AODV sem `hellos`, 160 nós



(e) ARAMA, 160 nós.



(f) BiO4Sel, 160 nós.

Figura 6.4: Distribuição energética na rede, no tempo de vida, restante

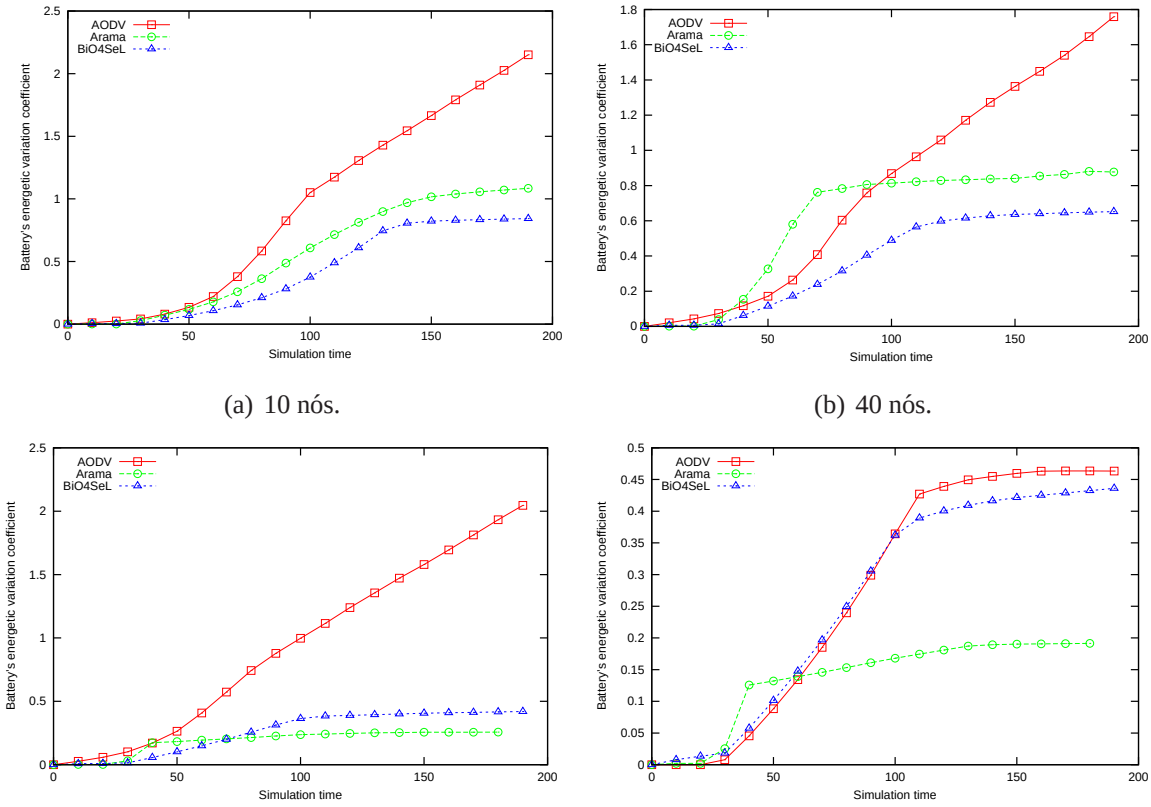
explicado anteriormete, mas não chegam à estação base. Assim, a energia longe da fonte fica próxima ao máximo.

Na Figura 6.3(g), assim como na Figura 6.4(b), nota-se um melhor desempenho se comparado ao AODV, com ou sem seus `hellos`. Ele consegue manter as energias da rede mais altas e mais igualitárias com sua distribuição de rotas. Os extremos ainda gastam mais energia, como esperado.

A Figura 6.4(f), assim como a Figura A.2(f), se comparada à Figura 6.4(d), retém pouca diferença, mas ela ainda existe. Assim como na versão de 40 nós, o BiO4SeL consegue uma melhor distribuição.

6.2.4 Experimento 4: Coeficiente de Variação da Média Energética

O coeficiente de variação é dado pela variância dividida pela média. Neste caso, da energia da rede no tempo de simulação. Esta é a métrica utilizada como parâmetro para uma comparação de variação geral entre as baterias dos protocolos apresentados.



(a) 10 nós.

(b) 40 nós.

(c) 160 nós.

(d) 160 nós, AODV sem “hello”.

Figura 6.5: Coeficiente de variação da média energética pelo tempo

Na Figura 6.5(a), assim como na Figura A.1(d), pode-se ver o maior coeficiente de variação como sendo o do AODV. Pelo fato de sua média ser bem baixa, o valor da métrica sobe. Com o BiO4SeL, por seu balanceamento de rotas, a energia da rede é consumida de um modo mais parcimonioso e igualitário, resultando em um menor coeficiente de variação.

Na Figura 6.5(b), como explicado anteriormente, o Arama pára de enviar dados. Assim, também estaciona seu coeficiente de variação. Como a energia mantém-se constante na maior parte dos nós, o coeficiente de variação praticamente inexiste.

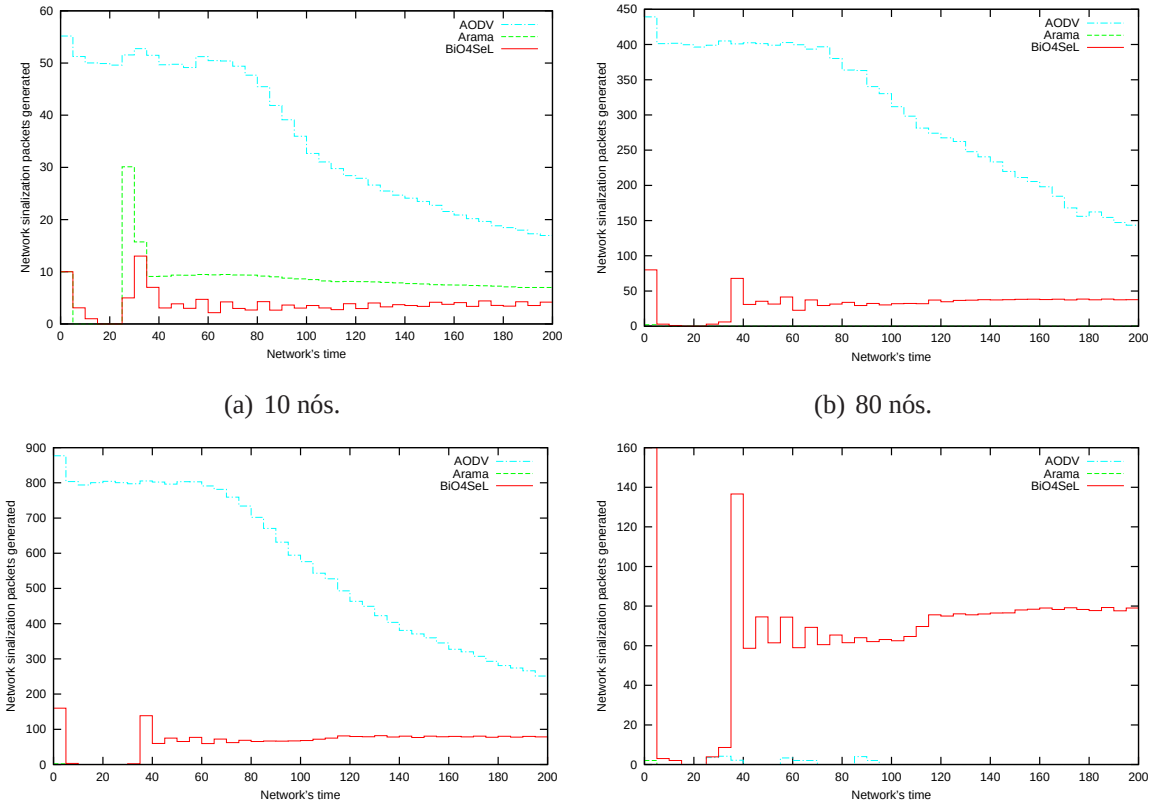
Na Figura 6.5(c), os algoritmos comportam-se como nas Figuras A.1(e) e A.1(f). Com 160 nós, pode-se notar, como na seção anterior, as consequências da pouca escalabilidade do ARAMA. Seu coeficiente de variação passa a tornar-se, a partir de certo ponto, cons-

tante. Isto ocorre pelo fato de ele não mais gastar a bateria dos nós a partir deste ponto, como explicado anteriormente.

Além disto, desde o começo, nota-se uma queda do coeficiente de variação do BiO4SeL. Ela é ocasionada pelo aumento da média das baterias, dado pelo aumento do número de nós. O AODV sofre o mesmo aumento. Apesar disto, ele continua a consumir toda a bateria da rede, resultando em um coeficiente cada vez maior.

Na Figura 6.5(d), também do coeficiente de variação da bateria pelo tempo, foi colocada como uma comparação com a anterior. Esta, entretanto, traz o AODV sem enviar seus "hellos". Sem o consumo por eles causado, o coeficiente de variação tende a diminuir. O melhor uso energético conseguido pelo BiO4SeL, entretanto, mantém o seu coeficiente ainda mais baixo, demonstrando sua superioridade.

6.2.5 Experimento 5: Pacotes de Sinalização pelo Tempo



(a) 10 nós.

(b) 80 nós.

(c) 160 nós.

(d) 160 nós, AODV sem "hello".

Figura 6.6: Histograma de envio de pacotes de sinalização

A Figura 6.6(a) traz a quantidade de pacotes de sinalização pelo tempo de simulação. Os protocolos comportam-se da seguinte maneira:

- **AODV** - Tem a maior quantidade de envio de sinalização dos protocolos. Mantém-na relativamente constante até o tempo de vida médio da rede. A partir deste momento, os nós começam a morrer e, consequentemente, a parar de enviar seus `hellos`. Assim, cada vez menos sinalização é enviada para a rede.

- **ARAMA** - De início, envia somente sinalização para reconhecimento dos vizinhos, assim como o BiO4SeL. Com o envio dos dados, envia-se as formigas de reconhecimento de rota, a fim de criá-la. Depois deste momento, os `hellos` mantêm a quantidade de sinalização relativamente fixa.

- **BiO4SeL** - Ele funciona de modo bem semelhante ao ARAMA. Envia, entretanto, mais formigas no começo e menos durante a troca de dados. No início, envia mais por ser a fase de inicialização das rotas, cuja realização, no ARAMA, é no início do envio de dados. Pelo mesmo motivo, envia menos na troca de dados. Neste caso, não leva-se em consideração os próprios dados como sinalização, mesmo eles levando formigas consigo.

Assim, o BiO4SeL consegue criar e manter as rotas com menos sinalização, se comparado aos outros dois protocolos. Consequentemente, gasta menos energia com ela, economizando-o para o envio dos dados, a principal atividade do protocolo.

A partir da Figura A.3(b), o Arama, por não encontrar mais rotas, somente envia uns poucos pacotes no início. Até o fim da simulação, a quantidade de sinalização enviada é pequena o suficiente, se comparada à dos outros protocolos, para não aparecer no gráfico.

A Figura 6.6(c) comporta-se de modo semelhante às Figuras 6.6(b) e A.3(c). Com o aumento da quantidade de nós, segue o aumento da quantidade de sinalização, mais especificamente formigas `hello`. Ainda assim, a escala de aumento do AODV supera a dos outros protocolos. Este aumento cresce com a quantidade de nós do cenário.

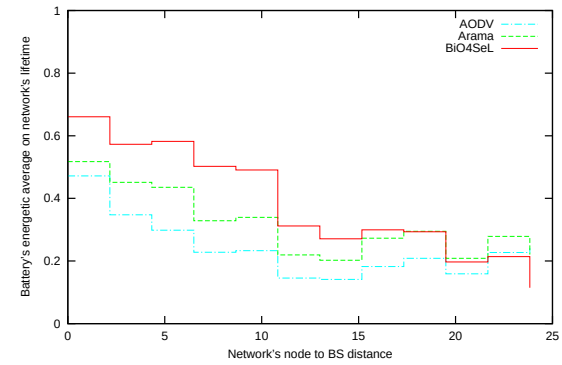
Além disto, o ARAMA, por não encontrar mais rotas, somente envia uns poucos pacotes no início. De resto, a quantidade é tão ínfima a ponto de não aparecer no gráfico.

Na Figura 6.6(d), nota-se a diminuição drástica do número de sinalização enviada pelo AODV sem os `hellos`. Quase nenhuma sinalização é enviada, mas ele perde a capacidade de mobilidade, adaptabilidade e reconhecimento da vizinhança, como explicado.

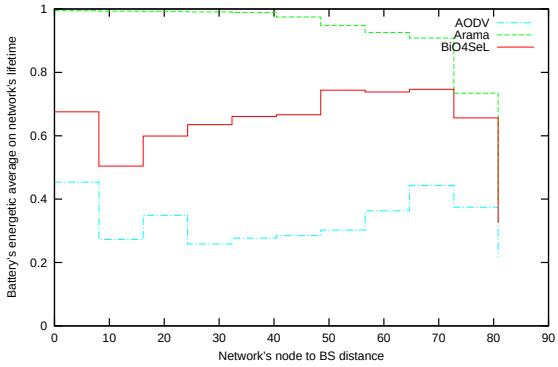
6.2.6 Experimento 6: Média de Bateria dos Nós pelos Saltos à EB

Apresenta-se a média de bateria dos nós pela distância, contada em saltos, até a estação base. Estes gráficos foram produzidos a partir de dados no momento do tempo de vida médio da rede. Este tempo foi escolhido pelo fato de dar uma ideia de como a rede

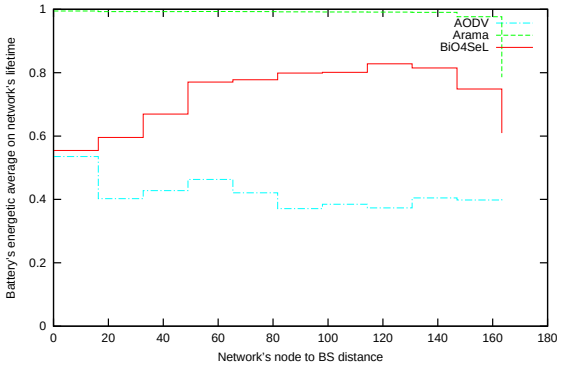
comporta-se até atingir o ponto da morte do primeiro nó. Neste caso, o tempo é importante, pois demonstra como se comporta a média de bateria próximo à EB, ao nó gerador e no meio do caminho. Isto demonstra como a bateria está sendo consumida durante a simulação.



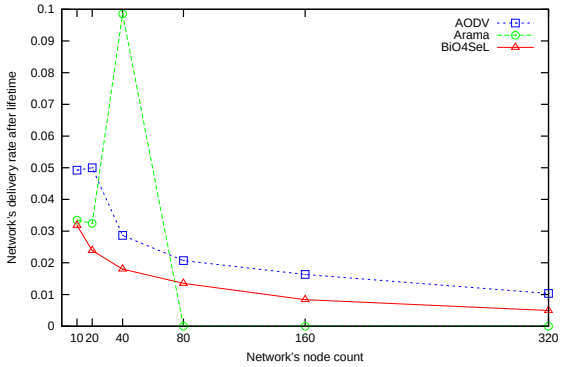
(a) 10 nós.



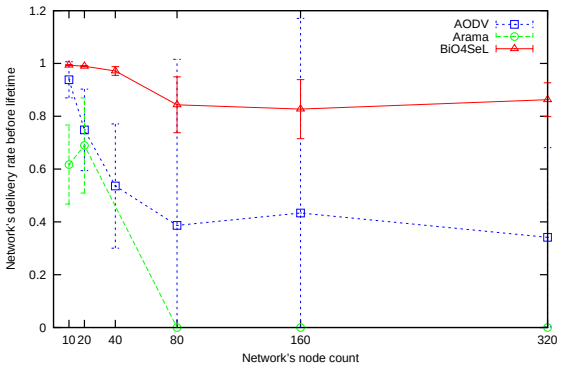
(b) 80 nós.



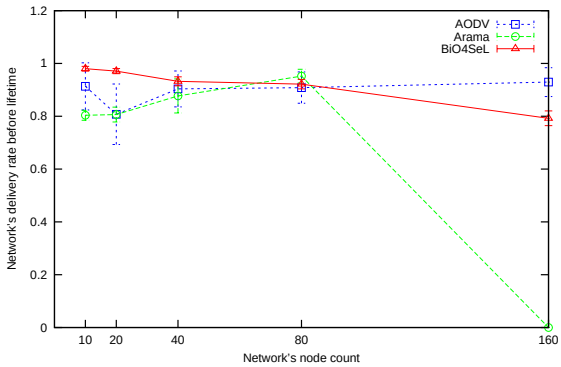
(c) 320 nós.



(d) Taxa de entrega com rede morta.



(e) Taxa de entrega com rede viva.



(f) Taxa de entrega com rede viva, AODV sem “hello”

Figura 6.7: Experimentos 6 e 7, média de bateria e taxa de entrega

Na Figura6.7(a), os três protocolos se comportam de maneira semelhante. As médias de bateria no tempo de vida são mais altas próximo da estação base e mais baixas próximas

à fonte dos dados, demonstrando uma provável queda energética nesta área. Esta queda acentuada é ocasionada, neste cenário, pelas suas dimensões. Pelo fato de o cenário ser bem pequeno, os nós mais próximos da fonte o recebem mais de uma vez, ao encaminhá-lo pela primeira vez e ao seu vizinho reencaminhá-lo. Assim, sua energia tende a cair mais rapidamente, se comparada ao restante da rede. Pelo mesmo motivo, os nós próximos à EB mantêm-se com mais energia, pois os intermediários recebem os mesmos pacotes mais de uma vez.

Comparando as diferenças, o tempo de vida o BiO4SeL é o mais alto dos 3. Assim, é fato a economia energética da rede com sua utilização, dado a maior média de bateria geral. A menor média do AODV deve-se, em parte, à sinalização enviada por este protocolo.

A Figura 6.7(c), assim como as Figuras 6.7(b) e A.3(f), é bem diferente da Figura 6.7(a). Com um cenário bem maior, nota-se o uso mais bem distribuído da rede. As maiores médias do AODV e BiO4SeL são esperadas, pois há mais nós pelos quais passar os pacotes. O AODV tem média mais baixa, e com pouca variação, devido ao envio constante de sua sinalização. O ARAMA, como esperado, não consegue mais encontrar rotas, e somente gasta energia de uma maneira sensível bem próximo da fonte.

Para o BiO4SeL, sua superioridade fica clara. A energia gasta próximo à fonte e próximo à EB ficam bem mais equiparadas, se comparado ao gráfico anterior. Além disso, entre 50 e 140 saltos, a energia gasta é bem semelhante, implicando em uma boa distribuição de carga pela rede. A média mais alta também implicada em um melhor uso energético deste protocolo.

6.2.7 Experimento 7: Taxa de Entrega pela Quantidade de Nós

Traduz-se por “rede viva” ou “rede morta”, respectivamente, antes e depois do tempo de vida.

A Figura 6.7(e) demonstra a supremacia do BiO4SeL em questão de entrega de dados à Estação Base. Por enviar menos sinalização, seus dados têm maior taxa de entrega se comparado ao AODV. O ARAMA, apesar de chegar a ultrapassar o BiO4SeL em taxa de entrega, o faz por quase não enviar sinalização durante a execução do algoritmo. Assim, ele consegue este ganho, ao custo de não conseguir encontrar rotas em cenários grandes, nos quais sua taxa de entrega cai para zero.

A Figura 6.7(f) serve de comparação com a Figura 6.7(e). Ao não enviar sinalização, o AODV consegue uma taxa de entrega superior. Apesar disto, perde o conhecimento de vizinhança e a possibilidade de uma possível mudança de rotas.

Na Figura 6.7(d), pode-se ver a proximidade deste fator em todos os algoritmos, com uma variação bem pequena. Assim, as diferenças são muito pequenas para significarem muito. O AODV consegue taxas minimamente maiores por começar a enviar menos sinalização. Além disso, o BiO4SeL, dependendo do cenário, pode ainda tentar enviar dados por vizinhos mortos, pois conta com um temporizador para retirá-los da tabela de vizinhança.

6.2.8 Experimento 8: Nós Mortos pelo Tempo de Simulação

Estes resultados representam a quantidade de nós mortos durante a simulação. Este experimento é útil para ver o quanto a energia dos nós está sendo distribuída pelos nós da rede. Se muitos nós caem em pouco espaço de tempo, isto significa uma boa distribuição de carga na rede, pois vários nós estarão na iminência de cair ao mesmo tempo.

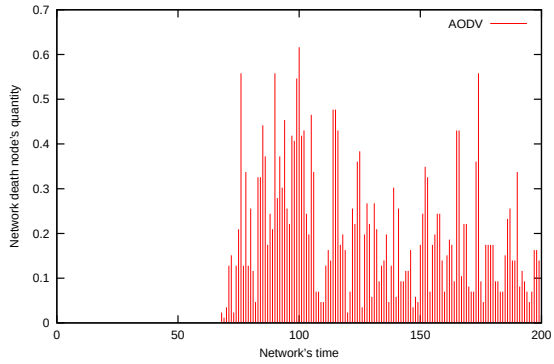
Nas Figuras 6.8(a) e A.4(a), há uma predominância de mortes espaçadas, denotadas pelas várias linhas altas, além de diversas delas seguidas. Além disso, há muitas mortes devido à utilização exagerada de sinalização, mesmo em estágios avançados da rede. Na Figura A.4(a), há com mais rajadas.

Nas Figuras 6.8(b) e A.4(b), as mortes vão acontecendo o tempo inteiro, e quase seguindo um padrão. Isto denota uma tendência do protocolo a comportar-se de modo semelhante em diversos cenários diferentes, pois o sumo dos testes, apresentado nas figuras, é bem regular. A Figura A.4(b) é ainda mais regular.

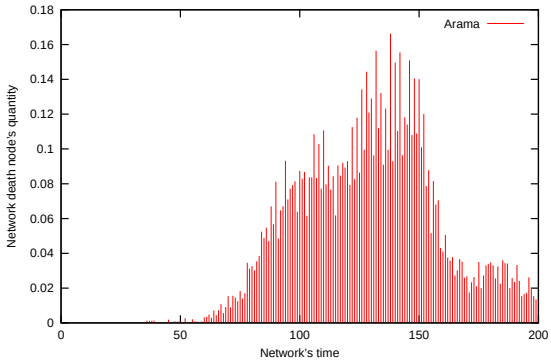
Nas Figuras 6.8(c) e A.4(c), a morte de nós acontece em rajadas. Isto pode ser observado pelos picos constantes ao longo do gráfico. Ou seja, as linhas não costumam ficar elevadas por muito espaço de tempo, subindo e descendo. Além disso, há um grande pico em dado momento, depois da morte do primeiro nó. Isto significa uma boa distribuição de carga pela rede, pois nós não vão morrendo aos poucos, mas vários em seguida. Com uma má distribuição de carga, vai-se acabando a energia em um caminho de cada vez, tornando a morte dos nós mais espaçada. Na Figura A.4(c), há menos rajadas.

A Figura 6.8(d) é como a Figura A.4(a), mas com mais rajadas. Estas rajadas, entretanto, são mais “grossas”, indicando mortes esparsas. Na Figura A.4(d), as rajadas são menos esparsas. Nas Figuras 6.9(a) e A.5(a), mantém-se alta a quantidade de mortes.

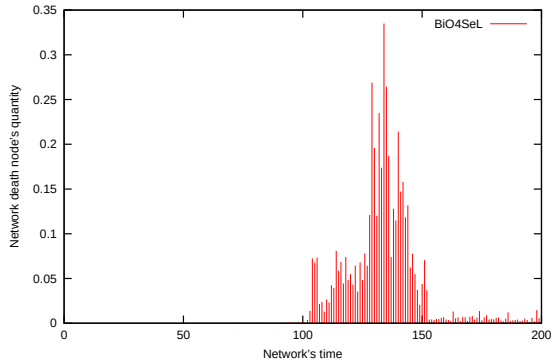
Nas Figuras 6.8(e), A.4(e), 6.9(c) e A.5(b), o ARAMA não encontra rotas. Como já explicado, ainda há gasto energético. Consequentemente, há mortes dos nós. Estas mortes ocorrem em grandes blocos, pois os caminhos percorridos pelos pacotes de dados perdidos são sempre os mesmos. Assim, os nós próximos à fonte tendem a gastar energia de modo igualitário, e morrer em tempos bem próximos.



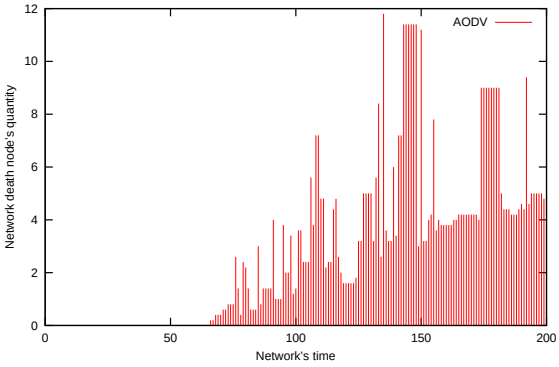
(a) AODV, 10 nós.



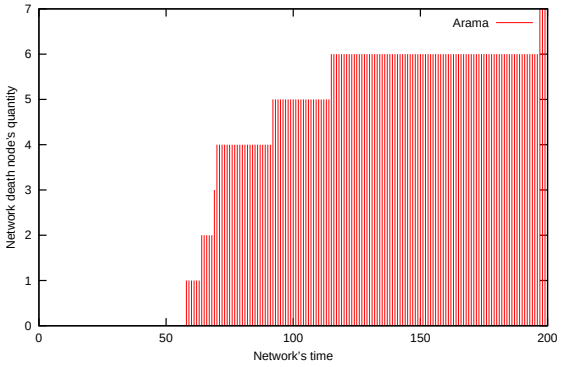
(b) ARAMA, 10 nós.



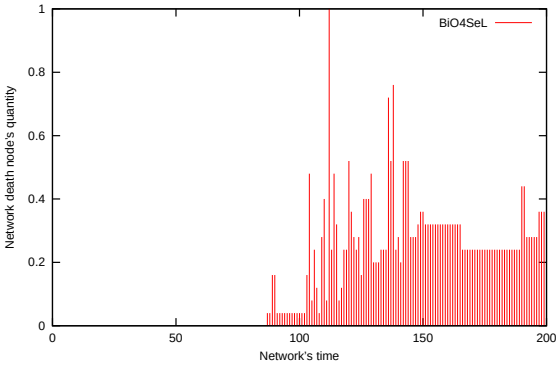
(c) BiO4Sel, 10 nós.



(d) AODV, 40 nós



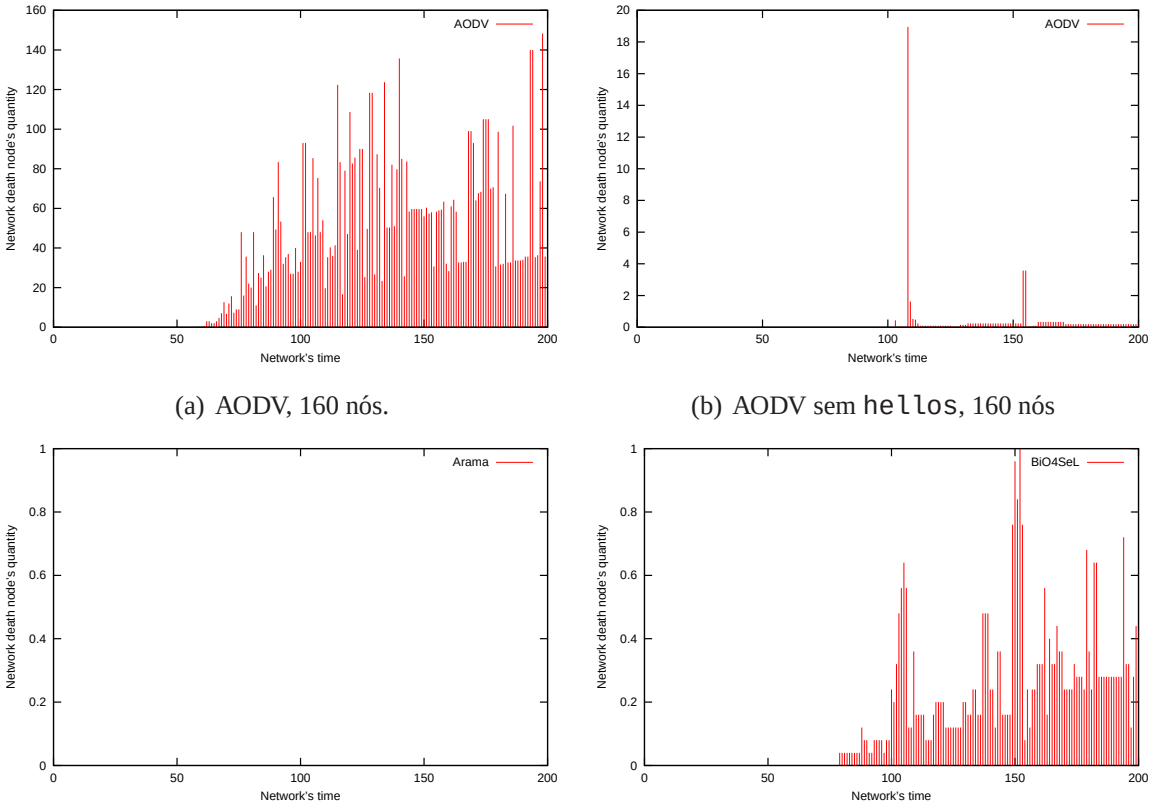
(e) ARAMA, 40 nós.



(f) BiO4Sel, 40 nós.

Figura 6.8: Histograma de morte dos nós, 10 e 40 nós

Nas Figuras 6.8(f), A.4(f), 6.9(d) e A.5(c), há várias fases. De início, as mortes ocorrem em tempos bem próximos, denotando uma morte consecutiva de nós com energia bem próxima. Depois, começam a ocorrer em rajadas. Mais para o fim da execução, os pacotes de dados não conseguem mais trafegar. As mortes finais, em blocos, ocorrerem pela sinalização enviada pela rede.



(a) AODV, 160 nós.

(b) AODV sem `hellos`, 160 nós

(c) ARAMA, 160 nós.

(d) BiO4Sel, 160 nós.

Figura 6.9: Histograma de morte dos nós, 160 nós

Capítulo 7

Conclusões e trabalhos futuros

Esta dissertação propôs o BiO4SeL, um protocolo de roteamento autônomo baseado em formigas para redes de sensores. Ele tem por objetivo otimizar o tempo de vida destas redes utilizando o quadro energético da rede, de modo autônomo. Assim, utilzando-se da propriedade auto catalítica da deposição feromônica, consegue-se diversas rotas para o envio dos dados. Utilizando-se estas rotas, o algoritmo consegue aumentar o tempo até a morte do primeiro nó da rede.

O restante deste capítulo mostra as contribuições do texto na Seção 7.1 e os trabalhos futuros na Seção 7.2.

7.1 Contribuições

Os algoritmos da área utilizam-se de algumas das seguintes características. Diversos utilizam o mapa energético da rede como uma base para o aprimoramento do tempo de vida, de modo estático ou não. Da mesma forma, os algoritmos de formigas e o balanceamento de carga foram extensamente utilizados. Estas soluções não somente foram idealizadas para a criação de rotas ótimas, como para o posicionamento dos sensores e modificação da área de abrangência do sinal. Todas as soluções, entretanto, têm o objetivo de aumentar o tempo de vida da rede.

Frente a este cenário, a primeira contribuição do BiO4SeL é no quesito autonomia. A maioria das soluções estudadas não trata este aspecto. Ainda assim, ele é muito importante, principalmente no âmbito das redes de sensores. Elas podem estar localizadas em ambientes adversos, impossibilitando interferência humana e requerindo autonomia em todas as suas funções. Além disso, a grande maioria dos protocolos trabalha com Estações Base conhecidas, algo não necessariamente verdadeiro. Tratar este aspecto dos protocolos é fundamental em muitos cenários, e praticamente não é abordado nos traba-

lhos lidos.

Outra contribuição desta dissertação é a dinamicidade alcançada na criação de rotas. Da forma como o algoritmo foi implementado, ele consegue uma grande robustez, se comparado até mesmo a outros algoritmos de formigas. As rotas criadas são diversas, trocadas dependendo-se da utilização energética da área, levando a um balanceamento da carga. Esta divisão de trafégo dos pacotes não chega a enviá-los por toda a rede, mas utiliza parte dela, ainda assim aumentando sua robustez e dinamicidade.

A inserção de nós é facilitada pela própria informação dos vizinhos. Da mesma forma como quando um nó move-se ou morre, a inserção utiliza somente informação local de seus vizinhos. Assim, bastam poucas mensagens a serem trocadas para a rede tornar-se completa após uma falha ou inserção de novo nó, sem a necessidade de envio de dados pela rede inteira.

A principal contribuição, entretanto, é para a otimização do tempo de vida. Uma pequena quantidade de pacotes de sinalização, aliada às vantagens descritas, gera um protocolo eficiente e robusto. Ele consegue melhorar o tempo de vida da rede, não somente otimizando a rota a ser seguida, mas dinamizando e modificando o roteamento com o tempo. Assim, consegue-se uma otimização do tempo de vida utilizando somente o roteamento, sem quaisquer outras técnicas.

Os resultados retirados dos experimentos confirmam as contribuições do protocolo. Foi comprovado o aumento do tempo de vida, como visto nos testes, com os experimentos de tempo de vida pelo tempo da rede e média energética pela distância. A melhor distribuição de carga foi mostrada nos histogramas de morte dos nós, pois eles tendem a morrer mais igualitariamente. Além disso, a capacidade de recuperar-se de falhas foi implementada com menor sobrecarga para a rede, como demostrado nos histogramas de sinalização.

7.2 Trabalhos futuros

Apesar das melhorias descritas acima, há diversos pontos a serem implementados. Alguns deles, apresentados a seguir, foram negligenciados durante o desenvolvimento ou teste do protocolo. Outros, como os próprios parâmetros e configuração do protocolo, podem ser aprimorados. O tempo de vida ainda pode ser melhorado mais, assim como a parte autonômica do protocolo.

Como trabalhos futuros, pode-se colocar uma série de pontos interessantes a serem trabalhados. Alguns deles são somente melhoramentos no algoritmo original ou análises nele. Outros são modificações para melhorá-lo.

- **Testar a mobilidade dos nós**

  Apesar de o suporte à mobilidade ser um dos pontos trabalhados no texto, nenhum teste foi feito tendo em vista nós movendo-se pela rede. Um possível trabalho futuro é testar a corretude do protocolo para este desafio e avaliar os impactos da movimentação nos protocolos. Para isto, avaliar diversos fatores, como perda de pacotes, velocidade de reabilitação (reconstrução das rotas após a mudança do nó), tempo de vida, dentre outras métricas. Além disso, buscar trabalhos com este fator em vista para comparação.

- **Criar mensagem para melhorar a reinserção dos nós na rede, por movimentação, ou inserção tardia** Colocar outra mensagem para avisar ao vizinho quando ele é antigo. Assim, se um vizinho antigo manda `hello`, o novo faz o pedido e ele envia a resposta, os outros vizinhos antigos receberão. Para evitar de mais vizinhos antigos mandarem seus `hellos` antes do novo mandar, um antigo manda uma mensagem, em resposta ao *respRqIHello*, para que o vizinho antigo que a mandou envie uma mensagem ao novo dizendo que ele é novo, e para ele mandar o `hello` logo. Assim, os outros vizinhos antigos recebem logo a informação, sem precisar ficarem mandando *respRqIHello*.

  Espera-se, com isto, reduzir a sobrecarga com a manutenção da rede. Apesar de este trabalho tratar objetivar a robustez, ela não é o foco, e sim a otimização do tempo de vida da rede. Assim, alguns detalhes foram ignorados até o ponto final do desenvolvimento do texto, como este ponto.

  Feita a mudança, analisar seu impacto sobre o protocolo como um todo. Verificar aumento ou diminuição de perda de pacotes, energia gasta, velocidade de reabilitação. Além disto, comparar com outros protocolos com estas métricas em consideração.

- **Implementar diversas Estações Base** Apesar de citado no texto, o protocolo não foi testado utilizando-se diversas Estações Base. Este aprimoramento não é tão simples quanto possa parecer. Somente guardar as rotas para cada EB é simples, e já está no protocolo. Um algoritmo autônomo, entretanto, deve escolher para qual EB enviar.

  Para este tipo de escolha, devem-se utilizar métricas, não necessariamente iguais às métricas utilizadas no algoritmo apresentado. Apesar disso, espera-se serem as mesmas métricas utilizadas. Além disso, deve-se efetuar todos os testes aqui presentes, além de outros, como inserção e remoção de EBs. Deve-se encontrar, ainda,

outros protocolos com os quais seja possível esta comparação, como outros baseados em Colônia de Formigas. Não foi encontrado, entretanto, nenhum trabalho com este foco, mudança autônoma de EBs durante a execução.

- **Implementar diversas fontes de dados** Apesar de não mostrado no texto, o protocolo foi testado utilizando-se todas os nós como fontes enviando dados. Os resultados, entretanto, não foram conclusivos, pois a energia da rede como um todo já descia igualmente. Assim, todos os protocolos testados apresentaram resultados semelhantes.

  Apesar disto, uma possibilidade seria testar os protocolos com diversos nós como fontes. Poderia ser utilizada uma distribuição probabilística para o envio dos dados a partir de diversas fontes diferentes, em tempos diferentes. Assim, seria uma nova análise de desempenho, colisão, tempo de vida, dentre outros fatores.

- **Rede heteregênea** Este protocolo foi designado para ser utilizado em uma rede homogênea, ou seja, com todos os nós simles e iguais. Apesar disto, soluções com heterogeneidade bem baixa, como 5% do total dos nós, apresentam bons resultados. Assim, como um trabalho futuro, pretende-se testar o protocolo neste tipo de redes, e avaliar os resultados.

- **Implementar em sensores** Os protocolos, utilizados no simulador, podem comportar-se diferentemente de como o farão em um dispositivo real. Além disso, o único objetivo do desenvolvimento de tecnologia tão específica é utilizá-la. Não existe protocolo de roteamento desenvolvido cujo objetivo não seja ser utilizado.

  Assim, tem-se como um trabalho futuro implementar o protocolo diretamente em sensores e testá-lo em ambiente real. Deve-se implementar, também, algum outro protocolo como parâmetro de comparação, e realizar os testes realizados no texto.

- **Camada MAC** A camada MAC é negligenciada para o propósito do protocolo. As perdas desta camada, por exemplo, não são levadas em consideração na primeira parte do protocolo, a Fase de Reconhecimento. Tais perdas podem ser um entrave ao correto funcionamento do algoritmo.

  Podem ser corrigidas, em seu ponto crítico, com um reenvio de pacotes semelhante ao descrito na Fase de Descoberta Inicial das Rotas. Apesar disto, o impacto desta modificação não foi avaliado no texto, sendo relegado a um futuro trabalho.

- **Segurança** O único dos aspectos das Comunicações Autonômicas não implementado no protocolo. É um aspecto importante para qualquer rede, pois todas elas

estão sucestíveis a ataques externos. Como o foco desta dissertação não foi a segurança, mas a melhoria do tempo de vida, a implementação deste aspecto das redes foi negligenciado.

Uma sugestão para sua implementação seria utilizar as próprias mensagens de controle da rede. Colocar poucos campos extras nestas mensagens seria pouco custoso em termos energéticos. Além disso, esta é uma necessidade em certos tipos de redes, com trocas de dados necessariamente privativas.

Apêndice A

Resultados Complementares

Neste capítulo, serão mostrados os resultados omitidos no Capítulo 6.

A.1 Resultados

A.1.1 Experimento 2: Média de bateria na rede

Na Figura A.1(a), o comportamento dos protocolos é bem semelhante ao da Figura 6.2(a). O Arama, entretanto, já começa a não encontrar caminhos na rede, como explicado no capítulo 6.

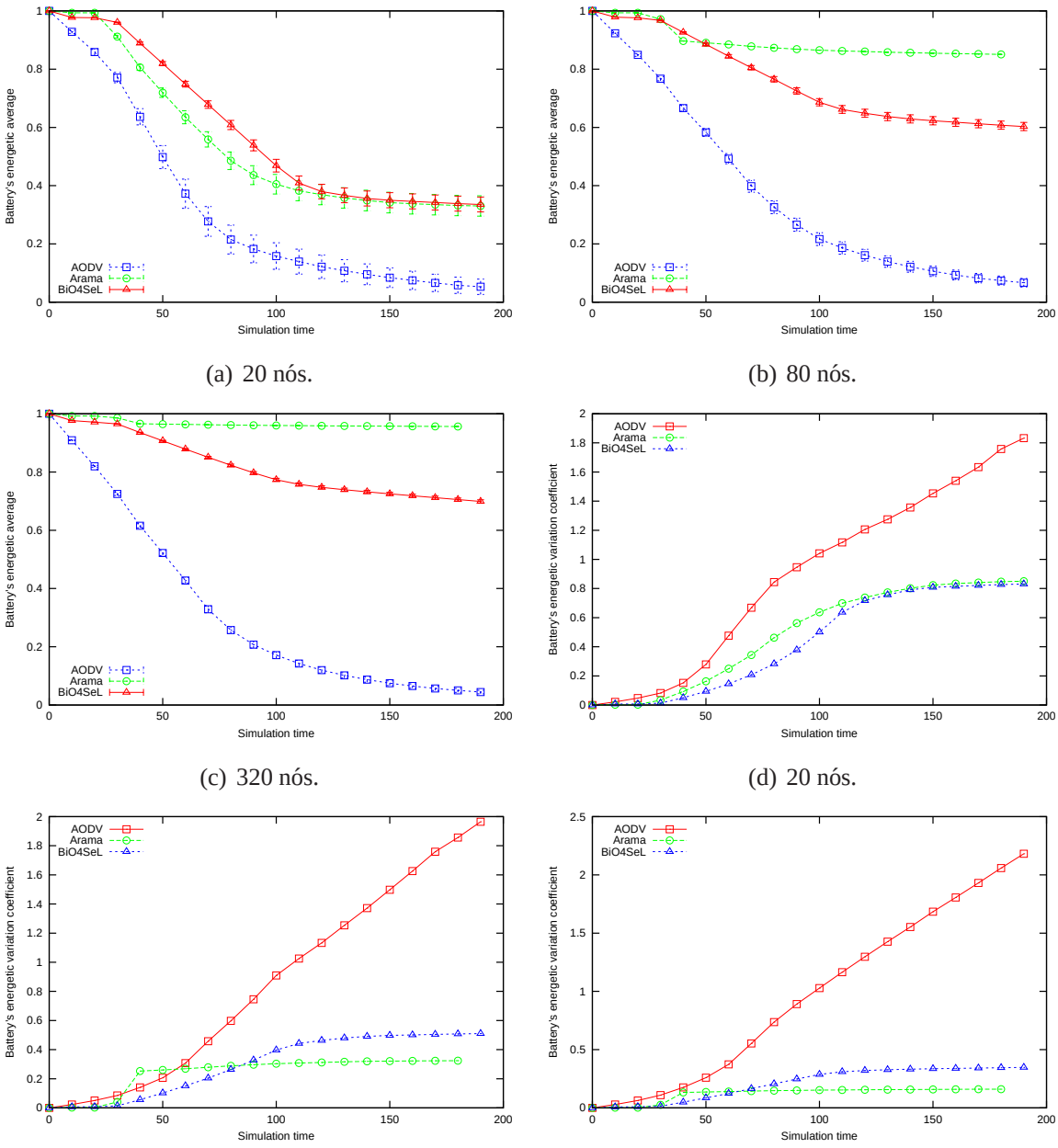
Na Figura A.1(b), assim como na Figura 6.2(c), o comportamente é semelhante ao da Figura 6.2(b). A média do Arama, entretanto, começa a aumentar. Isto acontece somente pelo fato de ele ter o mesmo gasto médio de bateria, mas ter mais nós. Assim, a média geral aumenta.

A.1.2 Experimento 3: Distribuição energética na rede, no tempo de vida

As explanações para as Figuras A.2(a), A.2(b), A.2(c), A.2(e), A.2(d) e A.2(f) estão no Capítulo 6.

A.1.3 Experimento 4: Coeficiente de Variação da Média Energética

As explanações para as Figuras A.1(d), A.1(e) e A.1(f) estão no Capítulo 6.



(a) 20 nós. (b) 80 nós.

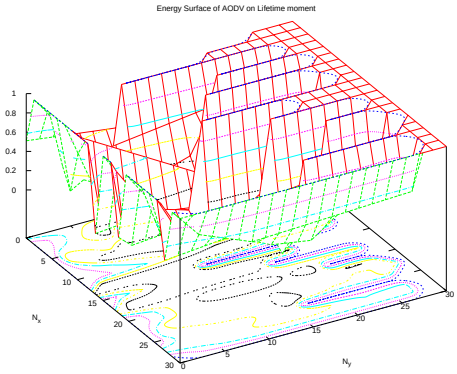
(c) 320 nós. (d) 20 nós.

(e) 80 nós. (f) 320 nós.

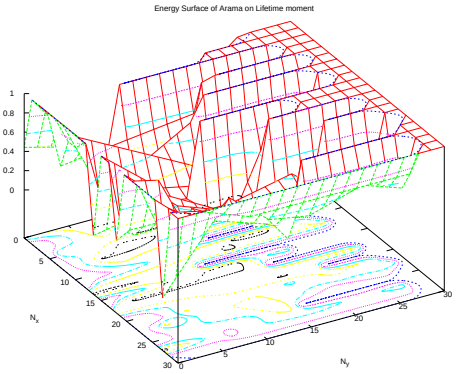
Figura A.1: Média de bateria e coeficiente de variação energética, apêndice

A.1.4 Experimento 5: Pacotes de Sinalização pelo Tempo

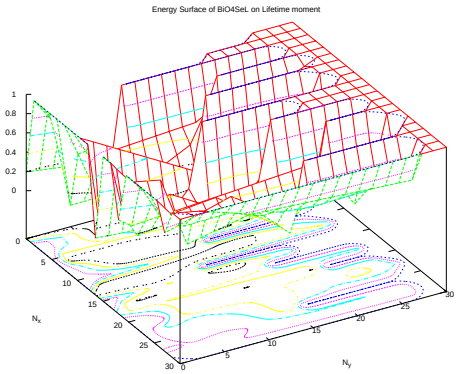
Na Figura A.3(a), comportamento dos protocolos é bem semelhante ao da Figura 6.6(a). Com o aumento da quantidade de nós vem o aumento da quantidade de sinalização, mais especificamente formigas `hello`. Ainda assim, a escala de aumento do AODV supera a dos outros protocolos. Este aumento continua nas outras instâncias.



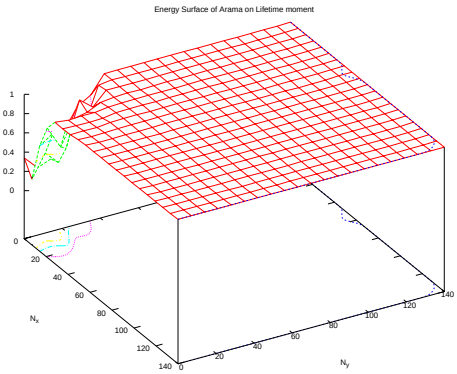
(a) AODV, 20 nós.



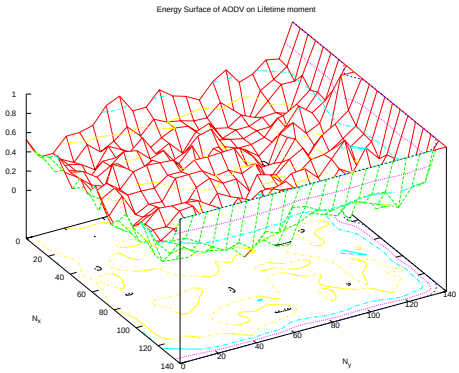
(b) Arama, 20 nós.



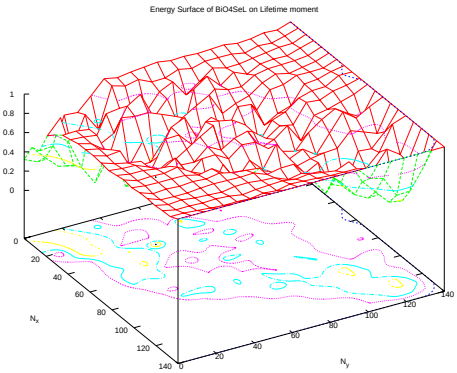
(c) BiO4Sel, 20 nós.



(d) Arama, 320 nós.



(e) AODV, 320 nós

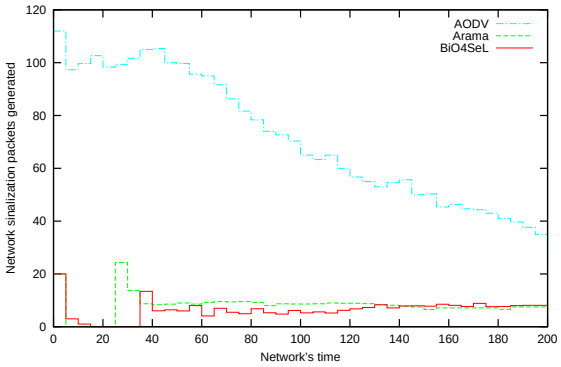


(f) AODV sem `hellos`, 320 nós

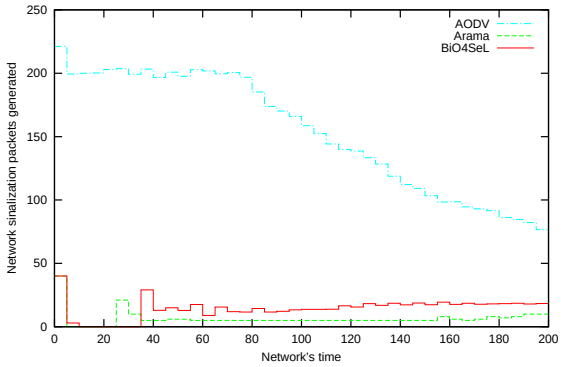
Figura A.2: Distribuição energética na rede, no tempo de vida

A.1.5 Experimento 6: Média de Bateria dos Nós pelos Saltos à EB

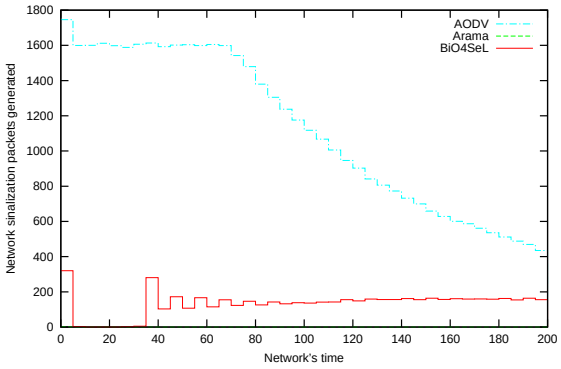
Na Figura A.3(d), comportamento dos protocolos é bem semelhante ao da Figura 6.7(a). Com um cenário relativamente maior, nota-se o uso mais bem distribuído da rede, mas ainda com sensível diferença próximo da fonte.



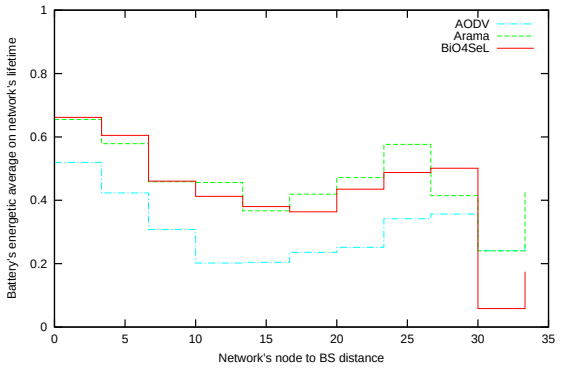
(a) 20 nós.



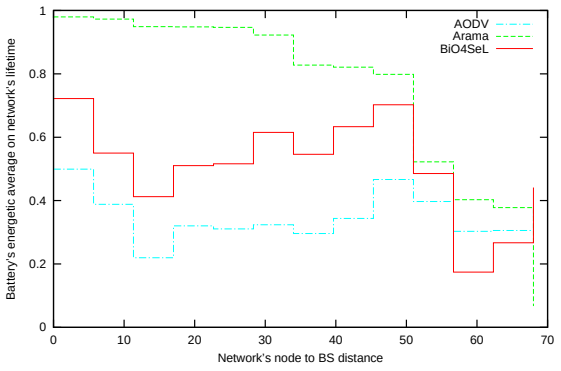
(b) 40 nós.



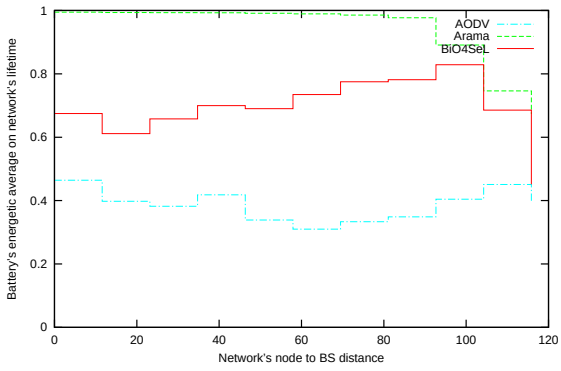
(c) 320 nós.



(d) 20 nós.



(e) 40 nós.



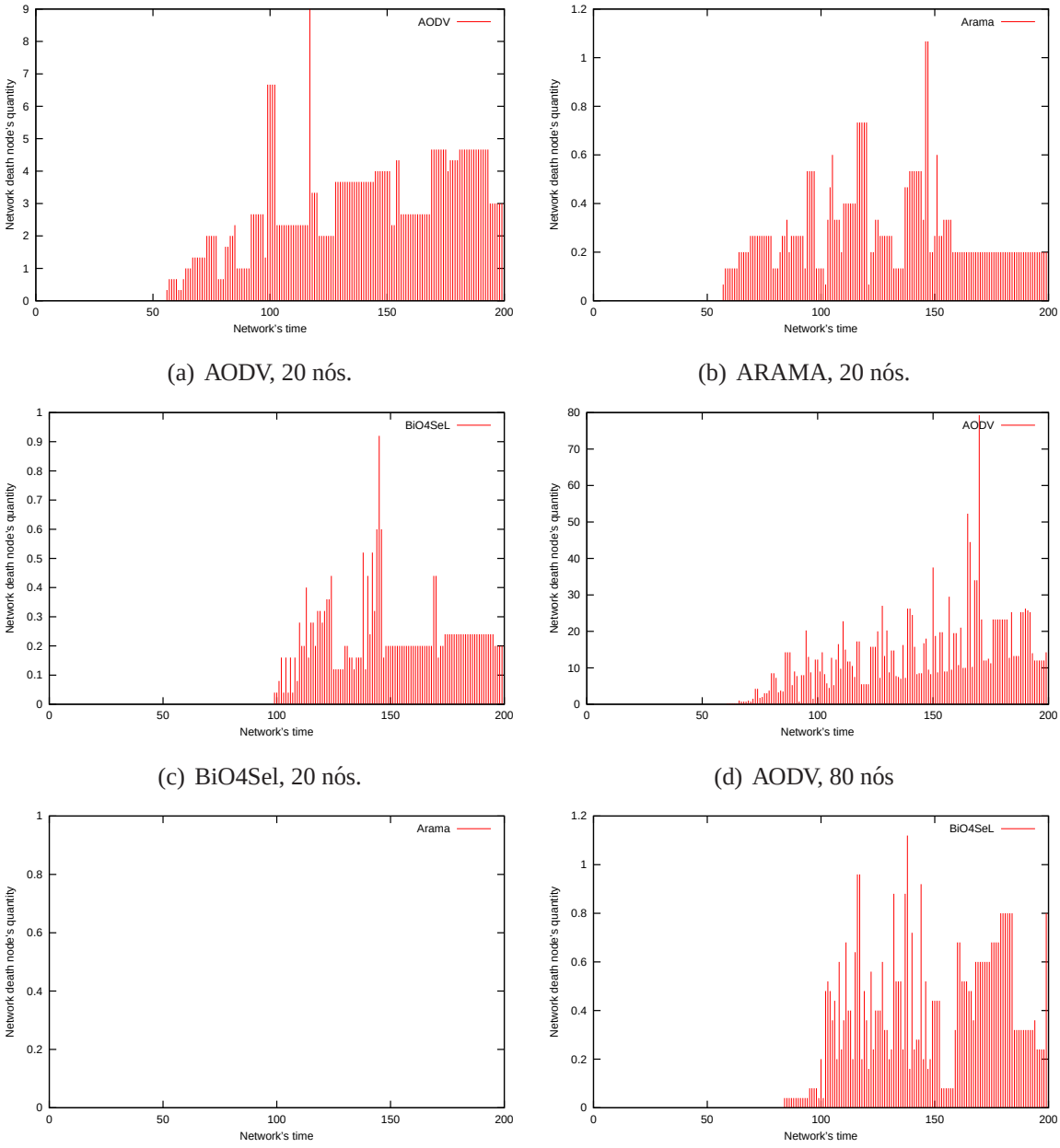
(f) 160 nós.

Figura A.3: Histograma de sinalização e média pelos saltos, apêndice

Nesta Figura, o Arama fica com média maior em certos pontos. Isto deve-se ao fato de ele enviar menos sinalização, além de começar a não encontrar caminhos. O BiO4SeL, entretanto, já começa a se destacar por conseguir manter um diferença menor entre a energia utilizada ao longo de toda a rede, entre 5 e 30 saltos.

Na Figura A.3(e), o Arama não consegue mais encontrar rotas, e somente gasta energia de uma maneira sensível bem próximo da fonte. Continua a tendência da Figura A.3(d).

A.1.6 Experimento 8: Nós Mortos pelo Tempo de Simulação



(a) AODV, 20 nós.

(b) ARAMA, 20 nós.

(c) BiO4Sel, 20 nós.

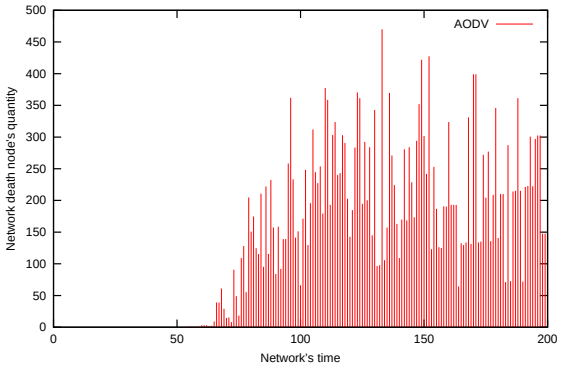
(d) AODV, 80 nós

(e) ARAMA, 80 nós.

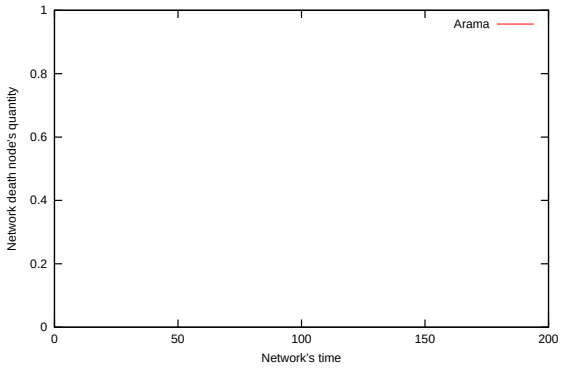
(f) BiO4Sel, 80 nós.

Figura A.4: Histograma de morte dos nós, 20 e 80 nós

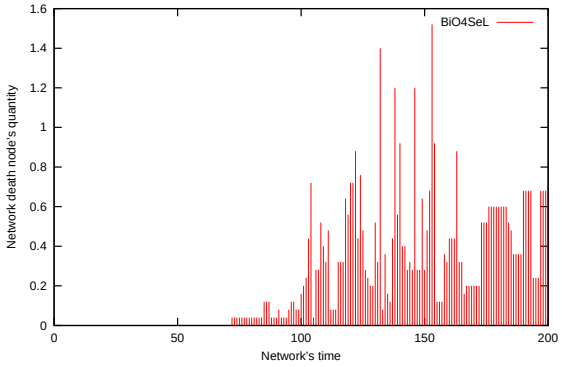
As explanações para as Figuras A.4(b), A.4(a), A.4(c), A.4(e), A.4(d), A.4(f), A.5(b), A.5(a) e A.5(c) estão no Capítulo 6.



(a) AODV, 320 nós.



(b) ARAMA, 320 nós.



(c) BiO4Sel, 320 nós.

Figura A.5: Histograma de morte dos nós, 320 nós

Apêndice B

Complemento da avaliação do protocolo

Neste apêndice, serão colocados alguns dos aspectos da comparação do início do Capítulo 5. A sua quantidade era grande demais para ficarem todos no corpo da Dissertação. Assim, por serem de menor importância, os aspectos abaixo foram colocados neste apêndice.

B.1 Aspectos do problema

B.1.1 Aspectos concernentes diretamente às redes e à sua composição

- **Tipo de sensores**: Simples ou complexos?

  Sensores simples, como explicado em [Liu 2006b], são incapazes de comprimir dados, modificar seu poder de transmissão ou conectar-se a grandes distâncias. Ou seja, eles restringem a quantidade de soluções possíveis, diminuindo o escopo da solução. Os complexos, como é imaginável, são os nós comuns da grande maioria dos artigos publicados, com pouca capacidade computacional, mas capazes dos processamentos supra-citados.

  Se utilizados, a solução somente seria capaz de ser utilizada neste tipo particular de rede, algo indesejável. Além disto, a estratégia não poderá ser comparada à maior parte das outras, ou seja, todas as utilizando qualquer das funções proibidas para sensores simples, pois estas utilizam nós diferentes daquela. Por outro lado, seria uma boa estratégia, visto que as formigas não necessitam de nós complexos. Basta, a elas, um processamento básico de reconhecimento.

  Assim, cada um dispões de vantagens e desvantagens. Obedecendo à idéia de deixar a solução mais genérica, além de poder contar com os algoritmos a comparar, a fim de demonstrar a eficiência da solução proposta, a escolha fica com os nós de tipo

complexo.

- **Modelo de bateria**: Qual utilizar?

  Os artigos, em geral, assim como para o tópico anterior, negligenciam este fator da rede. A maioria utiliza um modelo de bateria antigo e ultrapassado, tendo-se em vista as novas descobertas da área. Alguns nem mesmo chegam a citar o modelo utilizado, tornando difícil uma comparação simples entre as soluções.

  Os modelos antigos, segundo [Ma and Yang 2006], não competem com as novas descobertas no campo, possibilitando uma considerável economia de até 30% da energia dos sensores. Por outro lado, ao utilizarem um outro modelo, demonstram sua despreocupação com as camadas mais simples do roteamento, voltando-se ao algoritmo de roteamento.

  Neste trabalho, o foco estará no algoritmo, e não nas camadas mais baixas da rede. As baterias poderão ser diferentes em cada nó sem prejudicar a eficiência do algoritmo. Ou seja, o método de representação da bateria influirá somente no resultado final, e não nas comparações entre este e outros algoritmos, visto todos os algoritmos utilizarem a mesma representação.

  Assim, utilizar-se-á um modelo mais simples, entretanto, relativamente preciso. Tal modelo, como o utilizado em dezenas de artigos, servirá aos própositos desta proposta. Tal modelo, simples, é implementado pelas mais novas versões do simulador NS-2, sendo este o utilizado.

- **Controle de topologia da rede**: Qual utilizar?

  Assim como o tópico anterior, este refere-se às camadas mais baixas da rede. Enquanto o anterior trabalhava o físico do sensor, sua bateria, este tópico trata da topologia da rede, ou seja, a localização dos nós na rede.

  Em [Shen et al. 2005a], utiliza-se formigas para minimizar o gasto energético por interferências, dentre outros gastos, minimizando a abrangência do sinal de cada sensor. Assim, modifica-se a abrangência do sinal dos nós da rede. Isto, por si só, restringe a rede, pois seus nós devem ter a possibilidade de modificar, à sua vontade, a abrangência de seu sinal de rádio.

  Como discorrido em tópicos anteriores, isto limita a generalidade da solução, além de reduzir a possibilidade de comparatividade. A solução aqui proposta, como anteriormente dito, não almeja preocupar-se com as camadas mais baixas da rede, ignorando tal modificação.

- **Nós dormindo**: Utilizar esta estratégia ou não?

  Pelas definições do problema, os nós não poderiam ser postos para dormir sem uma perda de informação indispensável. Como um exemplo, imagine uma caldeira, e sensores dispostos nela captando as informações de temperatura, com pouca interseção entre eles. Se qualquer sensor for posto para dormir, alguma parte da caldeira ficará sem cobertura, podendo superaquecer e derreter. Mesmo não perdendo informação importante, pôr um nó para dormir diminuiria a quantidade de caminhos possíveis, possivelmente diminuindo a eficácia da estratégia de multi caminhos.

  Assim, levando-se em consideração um cenário deste tipo, colocar nós para dormir não é uma opção. Isto não retira a comprovada eficácia da técnica, mas ela pode ser inaplicável em certos contextos.

B.1.2 Aspectos do algoritmo de roteamento

- **Direção do roteamento**: Qual tratar?

  Quando fala-se desta direção, quer-se falar sobre a direção dos pacotes no roteamento. Há duas opções, da rede para a estação base ou dela para a rede. O método mais comum é da rede para a estação base, pois os sensores tem por principal função captar informação e a enviar à estação base. O caminho inverso ocorre quando é necessária a comunicação, pela estação base, aos sensores. Esta comunicação ocorre em alguns algoritmos, como na difusão direcionada [Zhang and Huang 2006b].

  A maioria dos artigos propõe-se a tratar somente a comunicação dos nós para a estação base, visto ser esta a mais utilizada. Somente alguns algoritmos tratam a comunicação inversa, pois esta é bem menos usual. A solução proposta também somente tratará a comunicação dos nós para a estação base, em primeira instância, visto não ter necessidade da inversa. Ao utilizar diversas estações base e autonomicidade em sua localização, pode ser utilizada esta estratégia para a fase inicial, ou seja, descoberta destes destinos da comunicação da rede.

- **Quantidade de alvos**: Inundação, *broadcast* ou *unicast*?

  Esta questão, de certa forma, inclui-se na anterior. Em uma comunicação da estação base aos sensores, a inundação é utilizada, por atingir todos os nós de uma vez. A comunicação inversa ocorre de um nó para a estação base, ou seja, em *unicast*. Não há necessidade de enviar a informação aos outros nós, visto ser a estação base a única capaz de tratá-los.

Como afirmado pelo autor em [Kang and Poovendran 2005], há diferenças entre o roteamento com restrição energética nestes dois paradigmas. Em *unicast*, se um nó está com pouca bateria, simplesmente passamos o fluxo a outro nó, poupando aquele. Na inundação, o fluxo sempre vai a todos os nós, sendo impossível tal solução. Assim, as soluções não são iguais, requerindo esforço e computação separada para cada uma.

Com base no exposto acima, para esta solução, a inundação será desconsiderada. A solução colocada aqui encaixa-se ao *unicast*, mas não à inundação.

- **Agregação de dados**: Utilizá-la ou não?

  A agregação de dados é um modelo bastante solidificado de economia energética na rede. Como já amplamente discutido, ela tem vantagens e desvantagens. A economia energética dada pelo envio somente de informações únicas é uma de suas vantagens. A rápida diminuição da bateria do nó cabeça ou a sobrecarga de pacotes de configuração para modificá-lo de tempos em tempos são algumas de suas desvantagens.

  Um método melhor seria o DSC [Wang et al. 2006], *Distributed Source Coding*. Nesta estratégia, como previamente apresentado, há a agregação de dados sem necessidade de formação de grupos. Os dados necessários à agregação são trafegados juntamente com os pacotes, exatamente como as informações das formigas da estratégia aqui descrita.

  Assim, seria possível, e plausível, incrementar esta opção como uma melhoria da aqui apresentada. Este, entretanto, não é o objetivo deste trabalho, tornando esta uma opção de prioridade secundária frente ao desafio apresentado.

B.1.3 Aspectos das formigas

- **Tratamento de falhas dos nós**: Utilizar qual método?

  Falhas nos nós são sempre passíveis de ocorrerem. Elas podem acontecer tanto pelo término da bateria como por falha no próprio sensor, em sua antena ou sistema internos. Pelo fato de serem redes sem fio, até o tempo pode atrapalhar a comunicação. O sensor pode, ainda, tendo sido comandado ou pelo ambiente, locomover-se ou ser locomovido. Dadas tantas possibilidades para um nó não poder mais ser alcançado, este deve ser um problema a ser tratado.

  Os algoritmos de formiga, dada a sua natureza de criação de diversos caminhos, são

bem adaptados a falhas em nós. Basta escolher e aprimorar um caminho alternativo ao reconhecer uma falha.

Este método, no entanto, pode ser implementado de mais de uma forma. Pode-se esperar as formigas, por si mesmas, aumentarem ou diminuirem a quantidade feromônica em um caminho, indo ou não por um caminho. Esta solução não é aconselhável no contexto da solução apresentada, dada a atualização da tabela de roteamento ser executada via as próprias trocas de pacotes, e não formigas de retorno. Outra solução seria utilizar formigas de retorno negativas ao encontrar uma dessas falhas. Tais formigas diminuiriam a quantidade feromônica no caminho do erro, e poderiam retirar aquele próximo salto da tabela de roteamento, para evitar futuros erros. As formigas de retorno, entretanto, podem entrar em colisão no caminho de volta, sendo um falso sinal de caminho quebrado. A solução mais clássica é simplesmente, de tempos em tempos, enviar pacotes de cada nó aos seus vizinhos avisando a presença daquele nó.

Assim, formigas de aviso, semelhantes a este método clássico, parecem ser o método melhor encaixável no contexto. Acredita-se ser este o método a menos consumir a bateria da rede como um todo, além de manter todos os nós com suas vizinhanças ativas. Isto encaixa-se na idéia do algoritmo.

- **Envio de formigas de `hello`**: Qual o espaço de tempo ideal entre cada envio?

  As formigas de `hello`, na estratégia proposta, serão enviadas a cada certo espaço de tempo. Este tempo, arbitrário, é assim utilizado na maioria dos protocolos estudados. Assim, é o mais indicado para ser utilizado. Além disso, nos caminhos sendo utilizados, as formigas de `hello` são enviadas como *piggyback* dos dados. Assim, evita-se mais envios e mais perdas de dados por colisão.

- **Método de criação da tabela de roteamento**: Avanço ou inverso?

  No ARA [Gunes et al. 2002], cada entrada da tabela de roteamento é composta por três campos. São eles o nó de destino da comunicação, o próximo salto e o feromônio associado. À medida que as formigas são enviadas, os nós começam a montar sua tabela de roteamento, de modo inverso ao caminho das formigas. Se uma formiga de avanço, *forward ant*, é enviada da fonte e recebida em um nó intermediário ou no destino, uma nova entrada da tabela é adicionada. O campo de destino da entrada será o nó fonte do pacote, o campo de próximo salto, o salto anterior do pacote, e o feromônio, a quantidade de saltos da formiga até o momento.

Assim, ele cria a tabela para o retorno à medida que a formiga de avanço caminha, e faz o contrário para a formiga de retorno, criando a tabela de avanço.

No ARAMA [Hussein and Saadawi 2003, Hussein et al. 2005], a tabela de roteamento é formada pelos mesmos três campos. Ao receber uma formiga de avanço, este algoritmo não adiciona somente uma entrada à tabela. Para cada entrada, ele adiciona o campo de destino da formiga ao campo de destino da entrada da tabela, e o endereço de um vizinho ao campo de próximo passo. O feromônio colocado em cada uma dessas entradas tem o mesmo valor básico. As formigas de retorno não modificam a tabela de roteamento, pois elas já contém os próximos nós em seu campo de memória.

Para esta solução, será utilizado um método semelhante ao utilizado no ARAMA. Além de ser mais simples, ele amplia as possibilidades para o encaminhamento dos pacotes.

- **Envio das formigas de inicialização**: Como fazer?

  No algoritmo ARAMA [Hussein and Saadawi 2003, Hussein et al. 2005], o autor, ao adicionar uma nova entrada à tabela de roteamento, coloca a mesma quantidade de feromônio inicial. Assim, uma formiga, ao passar por este nó indo ao destino especificado, teria a mesma probabilidade de ir para qualquer dos vizinhos, inicialmente.

  Esta é uma estratégia bem simples. Tamanha simplicidade faz o protocolo enviar formigas “cegas”, sem nenhuma idéia de qual caminho seguir para encontrar a estação base. Na solução apresentada, o envio de formigas a partir da estação base já irá criar, automaticamente, os caminhos até os nós. Não será necessário, para eles, encontrá-la. Basta garantir a chegada destas formigas a todos os nós, e eles terão caminho até lá. Além disso, ficará guardado o menor caminho, em número de saltos, até ela. Este será o caminho utilizado, inicialmente, pela fonte para encaminhar seus dados.

- **Atualização pelas formigas**: No momento de sua passagem ou posterior?

  Pode ser uma boa idéia ir guardando as atualizações da tabela em alguma variável, a fim de somente atualizar a tabela de roteamento a cada ciclo. Assim, gasta-se menos energia com o cálculo da probabilidade, menos simples do que atualizar a bateria.

  Pode ser que precise de uma tabela de baterias dos vizinhos, ai atualiza essa tabela

constantemente. vizinhos do nó, sua única informação, para o número de formigas a serem enviadas.

- **Manutenção das probabilidades de roteamento**: O tempo inteiro ou somente ao enviar?

  A probabiliadade da tabela de roteamento probabilística de um algoritmo de formiga não é, necessariamente, igual aos feromônios no caminho. Na tabela consta uma probabilidade. Esta probabilidade é calculada a cada modificação dos feromônios ou no momento do envio.

  O meu algoritmo irá receber muitas e frequentes modificações na tabela de roteamento, pois irá se modificar, quase o tempo todo, devido às modificações da bateria dos vizinhos. Ele irá enviar, provavelmente, bem menos do que modificar a tabela. Assim, será melhor calcular a probabilidade a cada envio.

Bibliografia

[Agah and Bekey 1997] Agah, A. and Bekey, G. A. (1997). Phylogenetic and ontogenetic learning in a colony of interacting robots. *Auton. Robots*, 4(1):85–100.

[Akkaya and Younis 2005] Akkaya, K. and Younis, M. (2005). Energy and qos aware routing in wireless sensor networks. *Cluster Computing*, 8(2-3):179–188.

[Akyildiz et al. 2002a] Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., and Cayirci, E. (2002a). Wireless sensor networks: a survey. *Computer Networks*, 38:393–422.

[Akyildiz et al. 2002b] Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., and Cayirci, E. (2002b). Wireless sensor networks: a survey. *Computer Networks*, 38:393–422.

[Al-Karaki and Al-Mashaqbeh 2007] Al-Karaki, J. N. and Al-Mashaqbeh, G. A. (2007). Energy-centric routing in wireless sensor networks. *Microprocess. Microsyst.*, 31(4):252–262.

[Anthony and Jannett 2005] Anthony, A. and Jannett, T. (2005). A framework for using agents in distributed sensor networks. *SoutheastCon, 2006. Proceedings of the IEEE*, pages 337–337.

[Balasubramaniam et al. 2006] Balasubramaniam, S., Botvich, D., Donnelly, W., Foghlú, M. O., and Strassner, J. (2006). Biologically inspired self-governance and self-organisation for autonomic networks. In *BIONETICS '06: Proceedings of the 1st international conference on Bio inspired models of network, information and computing systems*, page 30, New York, NY, USA. ACM.

[Bartfai 1998] Bartfai, G. A. (1998). Towards implementation of biologically inspired autonomous and adaptive information processing systems : Ph.d. dissertation. Technical report, Computer and Automation Research Institute of the Hungarian Academy of Sciences.

[Bash and Desnoyers 2007] Bash, B. A. and Desnoyers, P. J. (2007). Exact distributed voronoi cell computation in sensor networks. In *IPSN '07: 6th International Symposium on Information Processing in Sensor Networks*, pages 236–243, Washington, DC, USA. IEEE Computer Society.

[Beer et al. 1997] Beer, R. D., Quinn, R. D., Chiel, H. J., and Ritzmann, R. E. (1997). Biologically inspired approaches to robotics: what can we learn from insects? *Commun. ACM*, 40(3):30–38.

[Biswas 2005] Biswas, P. K. (April 18-21, 2005). Architecting multi-agent systems with distributed sensor networks. In *Integration of Knowledge Intensive Multi-Agent Systems, 2005. International Conference on*, pages 161–166.

[Biswas et al. 2006] Biswas, P. K., Qi, H., and Xu, Y. (Oct. 2006). A mobile-agent-based collaborative framework for sensor network applications. In *Mobile Adhoc and Sensor Systems (MASS), 2006 IEEE International Conference on*, pages 650–655.

[Bonabeau et al. 1998] Bonabeau, E., Henaux, F., Guérin, S., Snyers, D., Kuntz, P., and Theraulaz, G. (1998). Routing in telecommunications networks with ant-like agents. In *IATA '98: Proceedings of the second international workshop on Intelligent agents for telecommunication applications*, pages 60–71, London, UK. Springer-Verlag.

[Brooks 1991] Brooks, R. A. (1991). Integrated systems based on behaviors. *SIGART Bull.*, 2(4):46–50.

[Caro and Dorigo 1997] Caro, G. D. and Dorigo, M. (1997). AntNet: a mobile agents approach to adaptive routing. Technical Report IRIDIA/97-12, Université Libre de Bruxelles, Belgium.

[Caro et al. 2005] Caro, G. D., Ducatelle, F., and Gambardella, L. M. (8-10 June 2005). Swarm intelligence for routing in mobile ad hoc networks. *Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE*, pages 76–83.

[Chen et al. 2007a] Chen, M., Kwon, T., Yuan, Y., Choi, Y., and Leung, V. C. M. (2007a). Mobile agent-based directed diffusion in wireless sensor networks. *EURASIP J. Appl. Signal Process.*, 2007(1):219–219.

[Chen et al. 2007b] Chen, W.-M., Li, C.-S., Chiang, F.-Y., and Chao, H.-C. (2007b). Jumping ant routing algorithm for sensor networks. *Comput. Commun.*, 30(14-15):2892–2903.

[Cicirelli et al. 2007] Cicirelli, F., Furfaro, A., and Nigro, L. (2007). Exploiting agents for modelling and simulation of coverage control protocols in large sensor networks. *J. Syst. Softw.*, 80(11):1817–1832.

[Colorni et al. 1992] Colorni, A., Dorigo, M., and Maniezzo, V. (1992). An investigation of some properties of an "ant algorithm". In Manner, R. and Manderick, B., editors, *Proceedings of the Parallel Problem Solving from Nature Conference*, pages 509–520, Brussels, Belgiuum. Elsevier Publishing.

[Coutaz et al. 2005] Coutaz, J., Crowley, J. L., Dobson, S., and Garlan, D. (2005). Context is key. *Commun. ACM*, 48(3):49–53.

[De et al. 2003] De, S., Qiao, C., and Wu, H. (16-20 March 2003). Meshed multipath routing: an efficient strategy in sensor networks. *Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE*, 3:1912–1917 vol.3.

[Dobson et al. 2006] Dobson, S., Denazis, S., Fernández, A., Gaíti, D., Gelenbe, E., Massacci, F., Nixon, P., Saffre, F., Schmidt, N., and Zambonelli, F. (2006). A survey of autonomic communications. *ACM Trans. Auton. Adapt. Syst.*, 1(2):223–259.

[Dorigo and Caro 1999] Dorigo, M. and Caro, G. D. (1999). Ant colony optimization: A new meta-heuristic. In Angeline, P. J., Michalewicz, Z., Schoenauer, M., Yao, X., and Zalzala, A., editors, *Proceedings of the Congress on Evolutionary Computation*, volume 2, pages 1470–1477, Mayflower Hotel, Washington D.C., USA. IEEE Press.

[Dorigo and Gambardella 1996] Dorigo, M. and Gambardella, L. M. (1996). A study of some properties of ant-q. In *PPSN IV: Proceedings of the 4th International Conference on Parallel Problem Solving from Nature*, pages 656–665, London, UK. Springer-Verlag.

[Dorigo et al. 1991] Dorigo, M., Maniezzo, V., and Colorni, A. (1991). The ant system: An autocatalytic optimizing process. Technical Report 91-016 Revised, Dipartmento di Elettonica, Politecnico di Milano, Milano, Italy.

[Dulman et al. 2003] Dulman, S., Nieberg, T., Wu, J., and Havinga, P. (16-20 March 2003). Trade-off between traffic overhead and reliability in multipath routing for wireless sensor networks. *Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE*, 3:1918–1922 vol.3.

[Edge et al. 2006] Edge, K. S., Lamont, G. B., and Raines, R. A. (2006). A retrovirus inspired algorithm for virus detection & optimization. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 103–110, New York, NY, USA. ACM.

[Enke 1997] Enke, D. L. (1997). *Biologically inspired neural network connectionist models for use in artificial vision systems*. PhD thesis, University of Missouri - Rolla. Adviser-Cihan H. Dagli.

[Ergen and Varaiya 2007] Ergen, S. C. and Varaiya, P. (2007). Energy efficient routing with delay guarantee for sensor networks. *Wirel. Netw.*, 13(5):679–690.

[Estrin et al. 1999] Estrin, D., Govindan, R., Heidemann, J., and Kumar, S. (1999). Next century challenges: scalable coordination in sensor networks. In *MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 263–270, New York, NY, USA. ACM.

[Flammini et al. 2006] Flammini, M., Navarra, A., and Perennes, S. (2006). The real approximation factor of the mst heuristic for the minimum energy broadcasting. *J. Exp. Algorithmics*, 11:2.10.

[Fórster et al. 2007] Fórster, M., Bickel, B., Hardung, B., and Kókai, G. (2007). Self-adaptive ant colony optimisation applied to function allocation in vehicle networks. In *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 1991–1998, New York, NY, USA. ACM Press.

[Gambardella and Dorigo 1996] Gambardella, L. M. and Dorigo, M. (20-22 May 1996). Solving symmetric and asymmetric tsps by ant colonies. *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on*, pages 622–627.

[Gambardella and Dorigo 2000] Gambardella, L. M. and Dorigo, M. (2000). An ant colony system hybridized with a new local search for the sequential ordering problem. *INFORMS J. on Computing*, 12(3):237–255.

[Gan et al. 2004] Gan, L., Liu, J., and Jin, X. (2004). Agent-based, energy efficient routing in sensor networks. In *AAMAS '04: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 472–479, Washington, DC, USA. IEEE Computer Society.

[Ganesan et al. 2001] Ganesan, D., Govindan, R., Shenker, S., and Estrin, D. (2001). Highly-resilient, energy-efficient multipath routing in wireless sensor networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 5(4):11–25.

[GhasemAghaei et al. 2007] GhasemAghaei, R., Rahman, A., Gueaieb, W., and Saddik, A. E. (1-3 May 2007). Ant colony-based reinforcement learning algorithm for routing in wireless sensor networks. *Instrumentation and Measurement Technology Conference Proceedings, 2007 IEEE*, pages 1–6.

[Gunes et al. 2002] Gunes, M., Sorges, U., and Bouazizi, I. (2002). Ara-the ant-colony based routing algorithm for manets. *Parallel Processing Workshops, 2002. Proceedings. International Conference on*, pages 79–85.

[Guoying et al. 2000] Guoying, L., Subing, Z., and Zemin, L. (2000). Distributed dynamic routing using ant algorithm for telecommunication networks. In *WCC - ICCT '00 : International Conference on Communication Technology Proceedings*, volume 2, pages 1607–1612.

[Gurun et al. 2006] Gurun, S., Nagpurkar, P., and Zhao, B. Y. (2006). Energy consumption and conservation in mobile peer-to-peer systems. In *MobiShare '06: Proceedings of the 1st international workshop on Decentralized resource sharing in mobile computing and networking*, pages 18–23, New York, NY, USA. ACM.

[Haapola et al. 2005] Haapola, J., Shelby, Z., Pomalaza-Ráez, C., and Mähönen, P. (2005). Multihop medium access control for wsns: an energy analysis model. *EURASIP J. Wirel. Commun. Netw.*, 5(4):523–540.

[Hinchey et al. 2007] Hinchey, M., Dai, Y.-S., Rash, J. L., Truszkowski, W., and Madhusoodan, M. (2007). Bionic autonomic nervous system and self-healing for nasa ants-like missions. In *SAC '07: Proceedings of the 2007 ACM symposium on Applied computing*, pages 90–96, New York, NY, USA. ACM.

[Hofmeyr et al. 1998] Hofmeyr, S. A., Forrest, S., and Somayaji, A. (1998). Intrusion detection using sequences of system calls. *J. Comput. Secur.*, 6(3):151–180.

[Hong et al. 2002] Hong, X., Gerla, M., Wang, H., and Clare, L. (2002). Load balanced, energy-aware communications for mars sensor networks. *Aerospace Conference Proceedings, 2002. IEEE*, 3:3–1109–3–1115 vol.3.

[Horling et al. 2004] Horling, B., Mailler, R., and Lesser, V. (20-24 Sept. 2004). A case study of organizational effects in a distributed sensor network. In *Intelligent Agent Technology, 2004. (IAT 2004). Proceedings. IEEE/WIC/ACM International Conference on*, pages 51–57.

[Huang et al. 2006] Huang, R., Zhu, J., Yu, X., and Xu, G. (25-28 June 2006). The ant-based algorithm for the optimal many-to-one routing in sensor networks. *Communications, Circuits and Systems Proceedings, 2006 International Conference on*, 3:1532–1536.

[Huang and Jan 2004] Huang, S.-C. and Jan, R.-H. (2004). Energy-aware, load balanced routing schemes for sensor networks. In *ICPADS '04: Proceedings of Tenth International Conference on Parallel and Distributed Systems*, pages 419–425, Washington, DC, USA. IEEE Computer Society.

[Hussain and Matin 2006] Hussain, S. and Matin, E. S. A. W. (18-20 April 2006). Agent-based system architecture for wireless sensor networks. *Advanced Information Networking and Applications, 2006. AINA 2006. 20th International Conference on*, 2:5 pp.–.

[Hussein and Saadawi 2003] Hussein, O. and Saadawi, T. (2003). Ant routing algorithm for mobile ad-hoc networks (arama). In *Performance, Computing, and Communications Conference, Proceedings of the 2003 IEEE International*, pages 281–290, Washington, DC, USA. IEEE Computer Society.

[Hussein et al. 2005] Hussein, O., Saadawi, T., and Lee, M. J. (2005). Probability routing algorithm for mobile ad hoc networks' resources management. *IEEE Journal on Selected Areas in Communications*, 23(12):2248–2259.

[IEEE 2006] IEEE (2006). Ieee specific requirements part 15.4: Wireless medium access control (mac) and physical layer (phy) specifications for low-rate wireless personal area networks (wpans). *IEEE Std 802.15.4-2006 (Revision of IEEE Std 802.15.4-2003)*.

[Intanagonwiwat et al. 2003] Intanagonwiwat, C., Govindan, R., Estrin, D., Heidemann, J., and Silva, F. (2003). Directed diffusion for wireless sensor networking. *IEEE/ACM Trans. Netw.*, 11(1):2–16.

[Iqbal et al. 2006] Iqbal, M., Gondal, I., and Dooley, L. (2006). Online load balancing for energy-aware anycast routing. In *APCC '06: Asia-Pacific Conference on Communications*, pages 771–775, Washington, DC, USA. IEEE Computer Society.

[Islam and Hussain 2006] Islam, O. and Hussain, S. (2006). An intelligent multi-hop routing for wireless sensor networks. In *WI-IATW '06: Proceedings of the 2006 IEEE/WIC/ACM international conference on Web Intelligence and Intelligent Agent Technology*, pages 239–242, Washington, DC, USA. IEEE Computer Society.

[Iyengar et al. 2007] Iyengar, S. S., Wu, H. C., Balakrishnan, N., and Chang, S. Y. (2007). Biologically inspired cooperative routing for wireless mobile sensor networks. *IEEE Systems Journal*, 1(1):29–37.

[J. L. Deneubourg and Verhaeghe 1983] J. L. Deneubourg, J. M. P. and Verhaeghe, J. C. (1983). Probabilistic behaviour in ants: a strategy of errors? In *J. Theor. Biol.*, volume 105, pages 259–271.

[Jaikaeo et al. 2005] Jaikaeo, C., Sridhara, V., and Shen, C.-C. (7-10 Nov. 2005). Energy conserving multicast for manet with swarm intelligence. *Mobile Adhoc and Sensor Systems Conference, 2005. IEEE International Conference on*, pages 9 pp.–.

[Jain et al. 2006] Jain, S., Shah, R. C., Brunette, W., Borriello, G., and Roy, S. (2006). Exploiting mobility for energy efficient data collection in wireless sensor networks. *Mob. Netw. Appl.*, 11(3):327–339.

[Jamont and Occello 2003] Jamont, J. P. and Occello, M. (13-16 Oct. 2003). Using self-organization for functional integrity maintenance of wireless sensor networks. In *Intelligent Agent Technology, 2003. IAT 2003. IEEE/WIC International Conference on*, pages 535–538.

[Jayashree et al. 2007] Jayashree, L. S., Arumugam, S., and Rajathi, N. (2007). E/sup 2/lbc: an energy efficient load balanced clustering technique for heterogeneous wireless sensor networks. In *IFIP '06: International Conference on Wireless and Optical Communications Networks*, pages –, Washington, DC, USA. IEEE Computer Society.

[Kaminski et al. 2006] Kaminski, P., Litoiu, M., and Müller, H. (2006). A design technique for evolving web services. In *CASCON '06: Proceedings of the 2006 conference of the Center for Advanced Studies on Collaborative research*, page 23, New York, NY, USA. ACM Press.

[Kang and Poovendran 2005] Kang, I. and Poovendran, R. (2005). Maximizing network lifetime of broadcasting over wireless stationary ad hoc networks. *Mob. Netw. Appl.*, 10(6):879–896.

[Kephart and Chess 2003] Kephart, J. O. and Chess, D. M. (2003). The vision of autonomic computing. *Computer*, 36(1):41–50.

[Kim et al. 2007] Kim, J., Bentley, P. J., Aickelin, U., Greensmith, J., Tedesco, G., and Twycross, J. (2007). Immune system approaches to intrusion detection — a review. *Natural Computing: an international journal*, 6(4):413–466.

[Kim et al. 2006] Kim, J., Choi, J., and Lee, W. (2006). Energy-aware distributed topology control for coverage-time optimization in clustering-based heterogeneous sensor networks. *VTC '06 Vehicular Technology Conference*, 3(63):1033–1037.

[Ko and Huang 2005] Ko, R.-S. and Huang, C.-H. (2005). Using density-based d-hop connected d-dominating sets routing scheme to achieve load balancing for wireless sensor networks. In *WCNC '07: Wireless Communications and Networking Conference*, pages 4215–4220, New York, NY, United States. Hindawi Publishing Corp.

[Kuhne et al. 2007] Kuhne, R., Gormer, G., Schlager, M., and Carle, G. (2007). A mechanism for charging system self-configuration in next generation mobile networks. *Next Generation Internet Networks, 3rd EuroNGI Conference on*, pages 198–204.

[Lawler et al. 1985] Lawler, E. L., Lenstra, J. K., Kan, A. H. G. R., and Shmoys, D. B. (1985). *The traveling salesman problem: A Guided Tour of Combinatorial Optimization*. Wiley.

[Leguizamon and Michalewicz 1999] Leguizamon, G. and Michalewicz, Z. (1999). A new version of ant system for subset problems. *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, 2:–1464 Vol. 2.

[Lerman and Galstyan 2003] Lerman, K. and Galstyan, A. (2003). Agent memory and adaptation in multi-agent systems. In *AAMAS '03: Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, pages 797–803, New York, NY, USA. ACM.

[Liu 2006a] Liu, H.-I. L. C.-C. (2006a). An intelligent agent routing over wireless sensor networks. In *Mobile Adhoc and Sensor Systems (MASS), 2006 IEEE International Conference on*, pages 358–366.

[Liu and Hong 2006] Liu, J. and Hong, X. (2006). A traffic-aware energy efficient routing protocol for wireless sensor networks. In *CAMP '06: International Workshop on Computer Architecture for Machine Perception and Sensing*, pages 142–147, Washington, DC, USA. IEEE Computer Society.

[Liu 2006b] Liu, S.-C. (2006b). A lifetime-extending deployment strategy for multi-hop wireless sensor networks. In *CNSR 2006: Proceedings of the 4th Annual Communication Networks and Services Research Conference*, pages –, Washington, DC, USA. IEEE Computer Society.

[Liu et al. 2007] Liu, Y., Zhu, H., Xu, K., and Jia, Y. (24-27 Aug. 2007). A routing strategy based on ant algorithm for wsn. *Natural Computation, 2007. ICNC 2007. Third International Conference on*, 5:685–689.

[Ma and Yang 2006] Ma, C. and Yang, Y. (2006). Battery-aware routing for streaming data transmissions in wireless sensor networks. *Mob. Netw. Appl.*, 11(5):757–767.

[Madan et al. 2006] Madan, R., Cui, S., Lall, S., and Goldsmith, N. A. (2006). Cross-layer design for lifetime maximization in interference-limited wireless sensor networks. *IEEE Transactions on Wireless Communications*, 5(11):3142–3152.

[Malik et al. 2007] Malik, H., Shakshuki, E., Dewolf, T., and Denko, M. K. (21-23 May 2007). Multi-agent system for directed diffusion in wireless sensor networks. *Advanced Information Networking and Applications Workshops, 2007, AINAW '07. 21st International Conference on*, 2:635–640.

[Mandala et al. 2006] Mandala, D., Dai, F., Du, X., and You, C. (2006). Load balance and energy efficient data gathering in wireless sensor networks. In *MASS '06: IEEE International Conference on Mobile Adhoc and Sensor Systems*, pages 586–591, Washington, DC, USA. IEEE Computer Society.

[Mann et al. 2005] Mann, R. P., Namuduri, K. R., and Pendse, R. (2005). Energy-aware routing protocol for ad hoc wireless sensor networks. *EURASIP J. Wirel. Commun. Netw.*, 5(5):635–644.

[Misra et al. 2007] Misra, R., Mandal, C., and Guha, R. K. (2007). Coordinator rotation via domatic partition in self-organizing sensor networks. In *ISWPC '07: 2nd International Symposium on Wireless Pervasive Computing*, pages 5–7, Washington, DC, USA. IEEE Computer Society.

[Mortara 1997] Mortara, A. (1997). A pulsed communication/computation framework for analogvlsi perceptive systems. *Analog Integr. Circuits Signal Process.*, 13(1-2):93–101.

[Nasipuri and Das 1999] Nasipuri, A. and Das, S. R. (1999). On-demand multipath routing for mobile ad hoc networks. *Computer Communications and Networks, 1999. Proceedings. Eight International Conference on*, pages 64–70.

[Nurmi 2006] Nurmi, P. (2006). Modeling energy constrained routing in selfish ad hoc networks. In *GameNets '06: Proceeding from the 2006 workshop on Game theory for communications and networks*, page 6, New York, NY, USA. ACM.

[Obayashi et al. 2005] Obayashi, M., Nishiyama, H., and Mizoguchi, F. (2005). Secured cooperative multi-agent system in limited resources for intelligent sensor network. In *Industrial Electronics Society, 2005. IECON 2005. 31st Annual Conference of IEEE*, pages 6 pp.–.

[Ota et al. 2006] Ota, N., Ahrens, S., Redfern, A., Wright, P., and Yang, X. (Oct. 2006). An application-driven architecture for residential energy management with wireless sensor networks. In *Mobile Adhoc and Sensor Systems (MASS), 2006 IEEE International Conference on*, pages 639–644.

[Pan et al. 2007] Pan, Q., Wei, J., Liu, H., and Ma, K. (2007). High energy-efficient mobile agent based information aggregation in wireless sensor networks. *Wireless Communications, Networking and Mobile Computing, 2007. WiCom 2007. International Conference on*, pages 2654–2657.

[Park and Corson 1997] Park, V. D. and Corson, M. S. (1997). A highly adaptive distributed routing algorithm for mobile wireless networks. *infocom*, 00:1405.

[Parunak and Brueckner 2000] Parunak, H. V. D. and Brueckner, S. (2000). Ant-like missionaries and cannibals: synthetic pheromones for distributed motion control. In *AGENTS '00: Proceedings of the fourth international conference on Autonomous agents*, pages 467–474, New York, NY, USA. ACM.

[Perkins and Royer 1999] Perkins, C. E. and Royer, E. M. (1999). Ad-hoc on-demand distance vector routing. *wmcsa*, 0:90.

[Qi and Zhao 2007] Qi, B. and Zhao, C. (2007). Ant algorithm based load balancing for network sessions. In *ICNC '07: Proceedings of the Third International Conference on Natural Computation (ICNC 2007)*, pages 771–775, Washington, DC, USA. IEEE Computer Society.

[Qi et al. 2001] Qi, H., Iyengar, S., and Chakrabarty, K. (Aug 2001). Multiresolution data integration using mobile agents in distributed sensor networks. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 31(3):383–391.

[Qiao et al. 2006] Qiao, B., Liu, K., and Guy, C. (Dec. 2006). A multi-agent system for building control. In *Intelligent Agent Technology, 2006. IAT '06. IEEE/WIC/ACM International Conference on*, pages 653–659.

[Royer and Toh 1999] Royer, E. M. and Toh, C.-K. (1999). A review of current routing protocols for ad hoc mobile wireless networks. *Personal Communications, IEEE [see also IEEE Wireless Communications]*, 6(2):46–55.

[Rui et al. 2006] Rui, W., Yan, L., Gangqiang, Y., Chaoxia, L., and Quan, P. (16-21 July 2006). Swarm intelligence for the self-organization of wireless sensor network. *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 838–842.

[Saha and Pathak 2006] Saha, S. and Pathak, S. S. (2006). A novel swarm intelligence based routing scheme for manet using weighted pheromone paths. In *MILCOM '06: Military Communications Conference*, pages 1–7.

[Saoseng and Tham 2006] Saoseng, J.-Y. and Tham, C.-K. (Oct. 2006). Coordinated rate control in wireless sensor network. In *Communication systems, 2006. ICCS 2006. 10th IEEE Singapore International Conference on*, pages 1–5.

[Schoonderwoerd et al. 1997] Schoonderwoerd, R., Holland, O., and Bruten, J. (1997). Ant-like agents for load balancing in telecommunications networks. In *AGENTS '97: Proceedings of the first international conference on Autonomous agents*, pages 209–216, New York, NY, USA. ACM Press.

[Selvakennedy et al. 2007] Selvakennedy, S., Sinnappan, S., and Shang, Y. (2007). A biologically-inspired clustering protocol for wireless sensor networks. *Comput. Commun.*, 30(14-15):2786–2801.

[Shah and Rabaey 2002] Shah, R. C. and Rabaey, J. M. (2002). Energy aware routing for low energy ad hoc sensor networks. *Wireless Communications and Networking Conference, 2002. WCNC2002. 2002 IEEE*, 1:350–355 vol.1.

[Shakshuki et al. 2006] Shakshuki, E., Hussain, S., Matin, A. R., and Matin, A. W. (2006). P2p multi-agent data transfer and aggregation in wireless sensor networks. In *Mobile Adhoc and Sensor Systems (MASS), 2006 IEEE International Conference on*, pages 645–649.

[Shakshuki and Malik 2007] Shakshuki, E. and Malik, H. (2007). Agent based approach to minimize energy consumption for border nodes in wireless sensor network. *Advanced Information Networking and Applications, 2007. AINA '07. 21st International Conference on*, pages 134–141.

[Shakshuki et al. 2007] Shakshuki, E., Xing, X., and Malik, H. (2007). Mobile agent for efficient routing among source nodes in wireless sensor networks. *Autonomic and Autonomous Systems, 2007. ICAS07. Third International Conference on*, pages 39–39.

[Shen et al. 2005a] Shen, C.-C., Huang, Z., and Jaikaeo, C. (2005a). Ant-based distributed topology control algorithms for mobile ad hoc networks. *Wirel. Netw.*, 11(3):299–317.

[Shen et al. 2005b] Shen, C.-C., Huang, Z., and Jaikaeo, C. (2005b). Ant-based distributed topology control algorithms for mobile ad hoc networks. *Wirel. Netw.*, 11(3):299–317.

[Shtovba 2005] Shtovba, S. D. (2005). Ant algorithms: Theory and applications. *Program. Comput. Softw.*, 31(4):167–178.

[Shuang et al. 2007] Shuang, B., Li, Y., Li, Z., and Chen, J. (2007). An ant-based on-demand energy routing protocol for ad hoc wireless networks. In *WiCom '07 : Proceedings of the International Conference on Wireless Communications, Networking and Mobile Computing*, pages 1516–1519, Washington, DC, USA. IEEE Computer Society.

[Slama et al. 2006] Slama, I., Jouaber, B., and Zeghlache, D. (2006). Routing for wireless sensor networks lifetime maximisation under energy constraints. In *Mobility '06: Proceedings of the 3rd international conference on Mobile technology, applications & systems*, page 8, New York, NY, USA. ACM.

[Somayaji 2007] Somayaji, A. (2007). Immunology, diversity, and homeostasis: The past and future of biologically inspired computer defenses. *Inf. Secur. Tech. Rep.*, 12(4):228–234.

[Srisathapornphat and Shen 2003] Srisathapornphat, C. and Shen, C. (2003). Ant-based energy conservation for ad hoc networks. In *ICCCN '03 : Proceedings of the 12th International Conference on Computer Communications and Networks*, pages 22–37, Washington, DC, USA. IEEE Computer Society.

[Stutzle and Hoos 1997] Stutzle, T. and Hoos, H. (13-16 Apr 1997). Max-min ant system and local search for the traveling salesman problem. *Evolutionary Computation, 1997., IEEE International Conference on*, pages 309–314.

[Subing and Zemin 2001] Subing, Z. and Zemin, L. (2001). A qos routing algorithm based on ant algorithm. *ICC 2001 - IEEE International Conference on Communications*, 5(1):1587–1591.

[Subramanian et al. 1997a] Subramanian, D., Druschel, P., and Chen, J. (1997a). Ants and reinforcement learning: A case study in routing in dynamic networks. In *IJCAI (2)*, pages 832–839.

[Subramanian et al. 1997b] Subramanian, D., Druschel, P., and Chen, J. (1997b). Ants and reinforcement learning: A case study in routing in dynamic networks. In *International Joint Conferences on Artificial Intelligence 1997*, pages 832–839, Nagoya, Aichi, Japan.

[Sun et al. 2006] Sun, Y., Chen, G., Li, D., and Li, F. (4-6 Oct. 2006). A data fusion multiagent system of sensor network based on data fields. *Computational Engineering in Systems Applications, IMACS Multiconference on*, 2:2132–2136.

[Tamaki et al. 2007] Tamaki, H., ichi Fukui, K., Moriyama, K., Kurihara, S., and Numao, M. (6-8 June 2007). Automatic acquisition of sensor-network topology based on pheromone communication model. In *Networked Sensing Systems, 2007. INSS '07. Fourth International Conference on*, page 292.

[Taylor et al. 2004] Taylor, K., Ward, J., Gerasimov, V., and James, G. (16-18 Nov. 2004). Sensor/actuator networks supporting agents for distributed energy management. In *Local Computer Networks, 2004. 29th Annual IEEE International Conference on*, pages 463–470.

[Toh 1996] Toh, C.-K. (27-29 Mar 1996). A novel distributed routing protocol to support ad-hoc mobile computing. In *Computers and Communications, 1996., Conference Proceedings of the 1996 IEEE Fifteenth Annual International Phoenix Conference on*, pages 480–486.

[Tsirigos and Haas 2001] Tsirigos, A. and Haas, Z. J. (Nov 2001). Multipath routing in the presence of frequent topological changes. *Communications Magazine, IEEE*, 39(11):132–138.

[Tynan et al. 2007] Tynan, R., O'Harea, G., and Ruzzelli, A. (2007). Autonomic wireless sensor network topology control. In *ICNSC '07 : IEEE International Conference on Networking, Sensing and Control*, pages 7–13, Washington, DC, USA. IEEE Computer Society.

[Wagner and Bruckstein 1999] Wagner, I. A. and Bruckstein, A. M. (1999). Hamiltonian(t)-an ant-inspired heuristic for recognizing hamiltonian graphs. *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, 2:–1469 Vol. 2.

[Walla et al. 2007] Walla, J., Platt, G., James, G., and Valencia, P. (5-7 Feb. 2007). Wireless sensor networks as agents for intelligent control of distributed energy resources. In *Wireless Pervasive Computing, 2007. ISWPC '07. 2nd International Symposium on*, pages –.

[Wang et al. 2006] Wang, H., Peng, D., Wang, W., and Sharif, H. (2006). Optimal rate-oriented routing for distributed source coding in wireless sensor network. In *Q2SWinet '06: Proceedings of the 2nd ACM international workshop on Quality of service & security for wireless and mobile networks*, pages 55–58, New York, NY, USA. ACM.

[Wu et al. 2007] Wu, Z., Song, H., Jiang, S., and Xu, X. (2007). Ant-based energy aware disjoint multipath routing algorithm in manets. In *MUE '07: Proceedings of the 2007 International Conference on Multimedia and Ubiquitous Engineering*, pages 674–679, Washington, DC, USA. IEEE Computer Society.

[Xu et al. 2005] Xu, D., Linfeng, Y., Zongkai, Y., Wenqing, C., and Qifei, Z. (2005). An efficient energy distribution scheme in sensor networks. In *WCNM '05: Proceedings on International Conference on Wireless Communications, Networking and Mobile Computing*, pages 971–975, Washington, DC, USA. IEEE Computer Society.

[Yang and Ho 2006] Yang, K.-H. and Ho, J.-M. (2006). Proof: A dht-based peer-to-peer search engine. In *WI '06: Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 702–708, Washington, DC, USA. IEEE Computer Society.

[Yong et al. 2007] Yong, C. Y., Qiao, B., Wilson, D. J., Wu, M., Clements-Croome, D., Liu, K., Egan, R., and Guy, C. G. (23-27 June 2007). Co-ordinated management of intelligent pervasive spaces. *Industrial Informatics, 2007 5th IEEE International Conference on*, 1:529–534.

[Yongtao et al. 2006] Yongtao, C., Chen, H., Zhenyu, Z., and Haitao, L. (2006). Energy-efficient routing for mobile agents in wireless sensor networks. *Communications, Circuits and Systems Proceedings, 2006 International Conference on*, 3:1518–1522.

[Yuan et al. 2004] Yuan, P., Ji, C., Zhang, Y., and Wang, Y. (21-23 March 2004). Optimal multicast routing in wireless ad hoc sensor networks. *Networking, Sensing and Control, 2004 IEEE International Conference on*, 1:367–371 Vol.1.

[Zadel 1996] Zadel, S. (1996). An algorithm for bootstrapping the core of a biologically inspired motor control system. In *ICANN 96: Proceedings of the 1996 International Conference on Artificial Neural Networks*, pages 629–634, London, UK. Springer-Verlag.

[Zhang and Huang 2006a] Zhang, Y. and Huang, Q. (2006a). A learning-based adaptive routing tree for wireless sensor networks. *Journal of Communications*, 1(2):12–21.

[Zhang and Huang 2006b] Zhang, Y. and Huang, Q. (2006b). A learning-based adaptive routing tree for wireless sensor networks. *Journal of Communications*, 1(2):12–21.

[Zongkai et al. 2005] Zongkai, Y., Linfeng, Y., xu, D., and Qifei, Z. (2005). Multipath load balancing delivery based on decisive energy ratio in wireless sensor networks. In *RTCSA '05: Proceedings of 11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*, pages 277–280, Washington, DC, USA. IEEE Computer Society.

[Zussman 2003] Zussman, G.; Segall, A. (30 March-3 April 2003). Energy efficient routing in ad hoc disaster recovery networks. *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE*, 1:682–691.